Time Series Analysis

Assignment 1

Ozone Layer Thickness Time Series Analysis and Prediction

David Ben Gurion Dhanapal (S3859284)

Contents

List of Figures

1 Introduction

The ozone layer is a protective layer which wraps around the earth at an altitude of about 10km containing a high concentration of ozone, which absorbs most of the harmful ultraviolet radiation reaching the earth from the sun. It is depleting at a rather alarming rate and is for sure, an issue, scientists have warned, that needs to be addressed at the earliest.

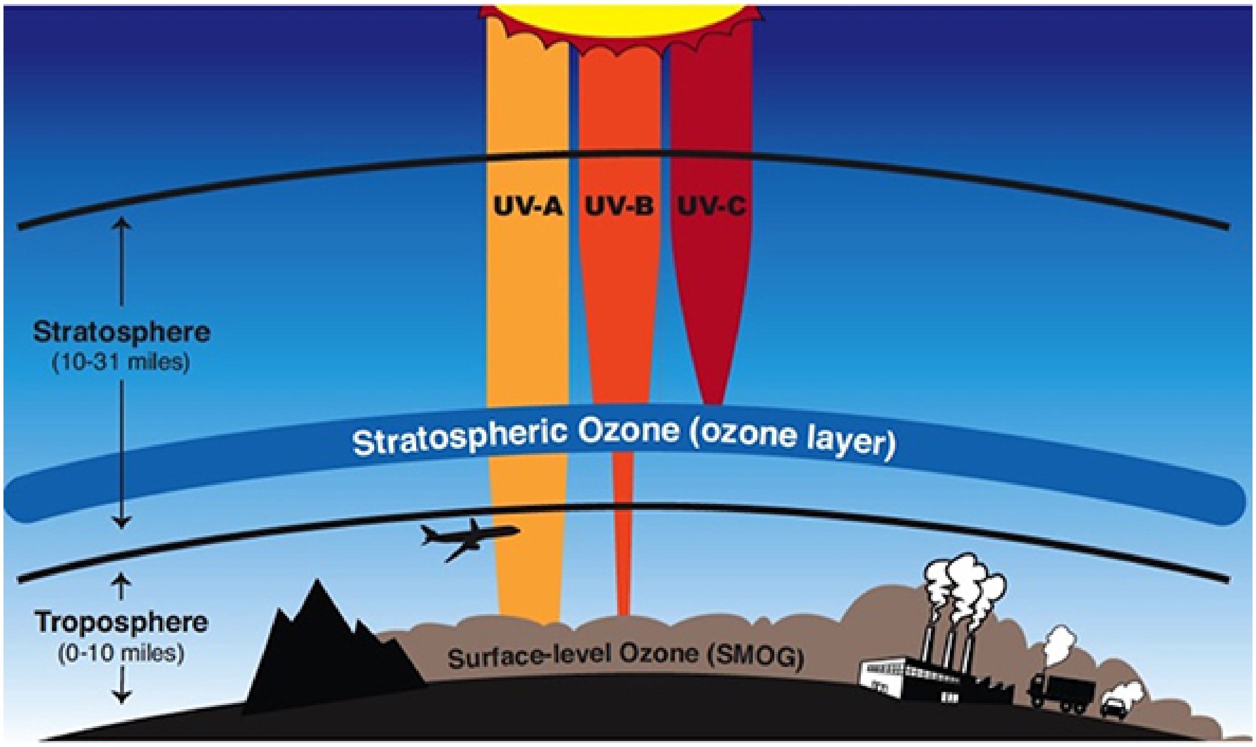


Figure 1: The Protective Ozone Layer (Credit:NASA)

In this detailed time series analysis, we use a dataset containing the yearly change in the ozone layer thickness values which are given in Dobson units - *'The Dobson Unit is the most common unit for measuring ozone concentration. One Dobson Unit is the number of molecules of ozone that would be required to create a layer of pure ozone 0.01 millimeters thick at a temperature of 0 degrees Celsius and a pressure of 1 atmosphere (the air pressure at the surface of the Earth)'*(NASA, 2018). They contain negative and positive values, where the negative values represent a decrease in the thickness and the positive values represent an increase in the ozone layer thickness. In this analysis, we use the data from the ozone dataset to create a corresponding time series and work on that.

1.1 Objectives:

Our main objectives for this time series analysis are:

- To find the best fitting model among the four main deterministic trend models - linear, quadratic, cosine or harmonic, cyclical or seasonal trend models for our ozone layer thickness time series.

- To give the predictions of yearly changes for the next 5 years using the best model that we find from the above four deterministic trend models.

- To propose a set of possible ARIMA(p, d, q) models using all suitable model specification tools we have at our disposal and also to include corresponding detailed comments and statements to back up and solidify our choices of the ARIMA model parameters.

The first two objectives are covered in detail in **Chapter 1** whereas, the third and the final objective is covered in **Chapter 2.**

2 Methods

To perform the whole time series analysis, model fitting and specification, we use **RStudio** with the R-language version **4.0.5** along with other necessary packages like tinytex, knitr, TSA and tseries and various other plotting techniques to visualize our data. The model fitting and prediction tasks from our Chapter 1 make use of the linear regression function **lm()** and the **predict()** function respectively. For model specification in Chapter 2, we use various transformation and differencing techniques along with various tools from the **tseries** package like unit root tests, EACF and BIC tables and ACF and PACF plots which serve as appropriate model specification tools to aid us in our task to propose a set of possible and appropriate ARIMA(p,d,q) models.

3 Prelude

In this section the analysis of the our time series is done, we:

- First, load the necessary libraries
- Read our data into a dataframe
- Create the required time series object from the data
- Comment on the **Five Valid Points**
- Analyze the impact of previous years on the current year and the correlations
- Plot the ACF and PACF graphs and make assumptions

3.0.1 Loading the essential packages:

Here, we load all the essential packages required for the analysis including the main **TSA** package which will be required for different plots and functions and will be helpful in choosing models and disgnostic checking.

```r
library(tinytex)
library(knitr)
library(TSA)
library(tseries)
```

3.0.2 Reading the data into a dataframe:

Now, we read the ozone layer thickness data into a dataframe using the **read.csv()** command after setting the working directory to the folder which contains the data file. Here, we use **header = FALSE** since the first row of the data file contains values and not column names.

```r
setwd("C:/Users/david/Desktop/TimeSeriesR")
ozone.thickness <- read.csv("data1.csv", header = FALSE)
```

We check the data-type of **ozone.thickness** and its dimensions along with its summary to verify if the correct data has been read into the dataframe.

```r
head(ozone.thickness)
```

```
##           V1
## 1  1.3511844
## 2  0.7605324
## 3 -1.2685573
## 4 -1.4636872
## 5 -0.9792030
## 6  1.5085675
```

```r
class(ozone.thickness)
```

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

```r
dim(ozone.thickness)
```

```
## [1] 90  1
```

```r
summary(ozone.thickness)
```

```
##        V1
##  Min.   :-11.5794
##  1st Qu.: -4.9281
##  Median : -2.6639
##  Mean   : -3.2023
##  3rd Qu.: -0.8746
##  Max.   :  3.4072
```

While we are at it, we also check if our new dataframe has any empty or **na** values:

```r
any(is.na(ozone.thickness))
```

```
## [1] FALSE
```

From the above information, it can be concluded that the 90 observations have been read into the dataframe without any errors.
To improve the readability, we rename the variable containing the ozone layer thickness to "Ozone Thickness" and we also label the rows with its corresponding year.

```r
names(ozone.thickness)[1] <- paste("Ozone Thickness")
rownames(ozone.thickness) <- seq(from=1927, to=2016)
head(ozone.thickness)
```

```
##      Ozone Thickness
## 1927       1.3511844
## 1928       0.7605324
## 1929      -1.2685573
## 1930      -1.4636872
## 1931      -0.9792030
## 1932       1.5085675
```

3.0.3 Creating a Time Series Object:

Now that our dataframe - **ozone.thickness** is ready, we convert it into a time series object on which we will work on, using the **ts()** function. Here, we use frequency = 1 as the data values are of each year, starting from 1927 to the year 2016.

```
ozone.thicknessTS <- ts(as.vector(ozone.thickness$‘Ozone Thickness‘), start = 1927, end = 2016,
                        frequency = 1)
ozone.thicknessTS
```

```
## Time Series:
## Start = 1927
## End = 2016
## Frequency = 1
##  [1]   1.35118436   0.76053242  -1.26855735  -1.46368717  -0.97920302
##  [6]   1.50856746   1.62991681   1.83242333  -0.83968364  -1.09611566
## [11]  -2.67457473  -2.78716606  -2.97317944  -0.23495548   0.97067000
## [16]   1.23652307   2.23062331   0.35671637  -2.12028099  -2.66812477
## [21]  -0.64702795   0.99608564   1.83817742   1.89922697  -0.55488121
## [26]  -1.40387419  -3.39178762   0.05777194   3.40717183   1.31488379
## [31]  -0.12882457  -2.51580137  -3.06205664  -3.33637179  -2.66332198
## [36]  -0.67958655  -2.11660422  -2.36318997  -5.36156537  -3.03103458
## [41]  -2.28838624  -1.06438684  -1.68813570  -3.16974819  -3.65647649
## [46]  -1.25151090  -1.08431732  -0.44863234  -0.17636387  -2.64954530
## [51]  -1.28317654  -4.29289634  -3.24282341  -3.60135297  -2.57288652
## [56]  -5.00586059  -6.68548244  -4.58870210  -4.32654629  -2.29370761
## [61]  -2.26456266  -2.27184846  -2.66440082  -3.79556478  -7.65843185
## [66] -10.17433972  -4.21230497  -2.82287161  -1.36776491  -4.43997062
## [71]  -3.78323838  -5.85304107  -8.55125744  -8.06501289  -7.75975806
## [76]  -6.65633206  -6.62708203  -7.83548356  -8.84424264  -7.67352209
## [81]  -7.05582939  -4.69497353  -7.12712128  -9.58954985 -10.19222042
## [86]  -9.33224686 -10.80567444 -10.86096923 -11.57941376  -9.30284452
```

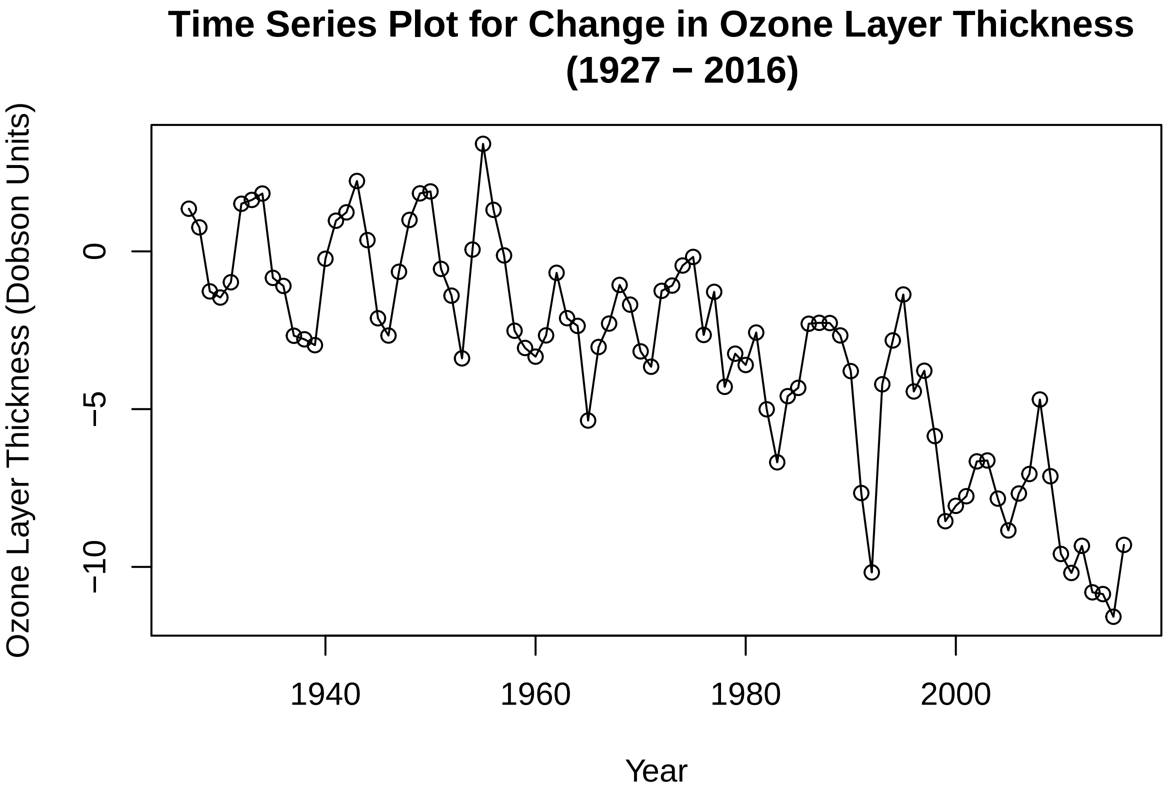
```
class(ozone.thicknessTS)
```

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

As we can see from the information displayed above, we have successfully created a time series object **ozone.thicknessTS** and we further check the datatype of the object using the **class()** function and we obtain “ts” as the result which indicates that it is a **time series**.

Now that we’ve created the time series object, we plot the time series which serves as one of the most useful descriptive tool in our analysis.

```
plot(ozone.thicknessTS, xlab=’Year’, ylab=’Ozone Layer Thickness (Dobson Units)’,
     main=’Time Series Plot for Change in Ozone Layer Thickness
     (1927 - 2016)’, type=’o’)
```



3.0.4 Five Valid Points:

Now that we've obtained a good plot of our time series, we can start with the descriptive analysis. We begin with analyzing and commenting on the five valid points:

- **1) Trend**:

From the time series plot, we can clearly infer that there is a significant **downward trend** which denotes a stable decrease in the ozone layer thickness over time. Since the time series has a trend, for now, we assume non-stationarity.

- **2) Seasonality**:

For a few years in the beginning there seems to be some kind of pattern but for all the years after, the pattern falls off and for a majority part of the plot, there is no seasonality. However, since we can visually observe a few patterns, we assume that this time series might have a **slight seasonality**.

- **3) Changing Variance**:

From the marked image **Figure 2**, we see that the time series has **slight changing variance** for some parts in the plot.

- **4) Change Point/Intervention**:

When we look at the time series plot, we can say with confidence that there is indeed **no significant change point** or intervention and no abrupt or sudden changes either.

- **5) Behaviour**:

From an overall perspective, we can infer that the series has a **dominant Moving Average (MA)** behaviour. However, we can also argue that the series has a **slight Autoregressive** behaviour as we can see

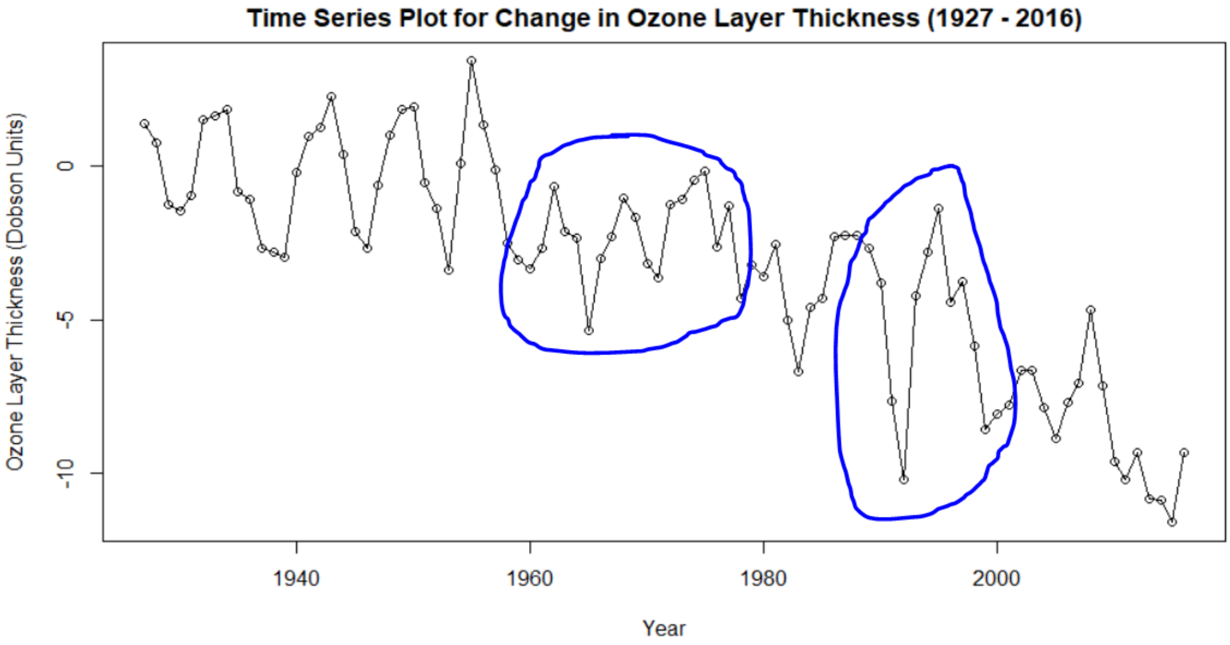


Figure 2: Changing Variance

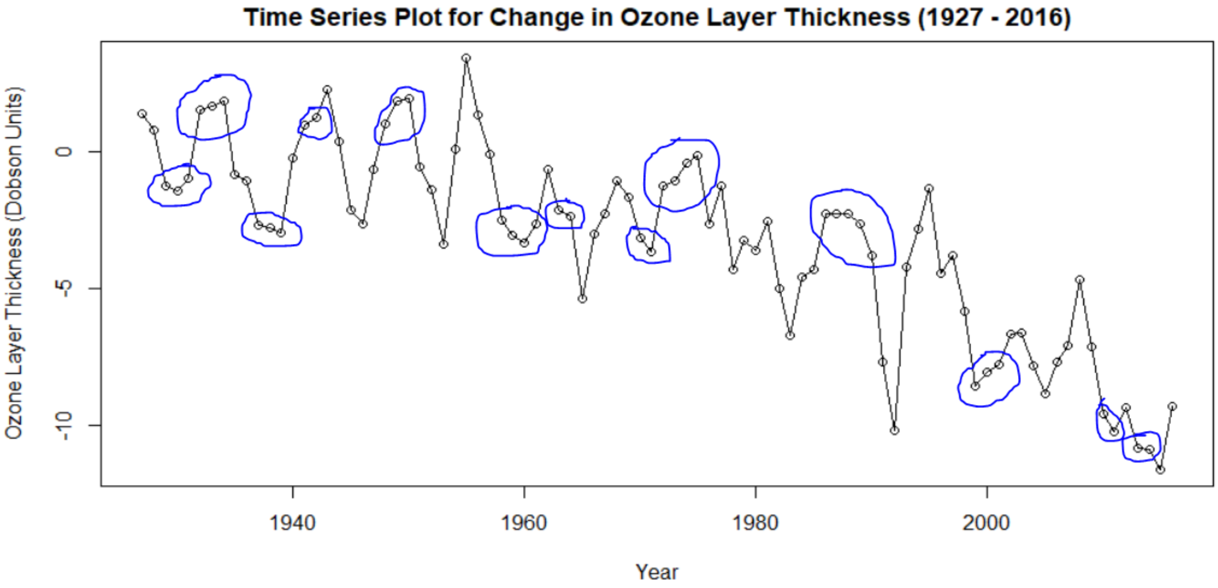


Figure 3: Behaviour

from the marked image **Figure 3**.
So, overall we can conclude that this time series has a **mixed/hybrid(MA/AR)** behaviour with Moving Average being the dominant one. **However, since we know that there might be slight seasonality, we cannot decide the behaviour without first removing/capturing the seasonality using a model**.

3.0.5 Analyzing Impacts and Correlation:

Now that we've had a good look on the five valid points, we go ahead and try to find out the impact of a past year's value on the current year's. For this, we introduce the **zlag()** function from which we can generate the first and subsequent lags of the time series so that we can compare the time series with its first and subsequent lags.
We create the first lag of the series and store it in the variable **x** and to also make our working easier, we read our original time series into the variable **y**:

```
y = ozone.thicknessTS
head(y)
```

```
## [1]  1.3511844  0.7605324 -1.2685573 -1.4636872 -0.9792030  1.5085675
```

```
x = zlag(ozone.thicknessTS)
head(x)
```

```
## [1]         NA  1.3511844  0.7605324 -1.2685573 -1.4636872 -0.9792030
```

From the above outputs, we see that for the first lag of the time series, the first value is denoted as **NA** which makes sense as it is the first lag and the values for **x** are pushed one value to the right. Now that we have an insignificant value in **x**, we create a variable **z** to store the index and get rid of the first **NA** value in x using the **length()** function.

```
z = 2:length(x)
```

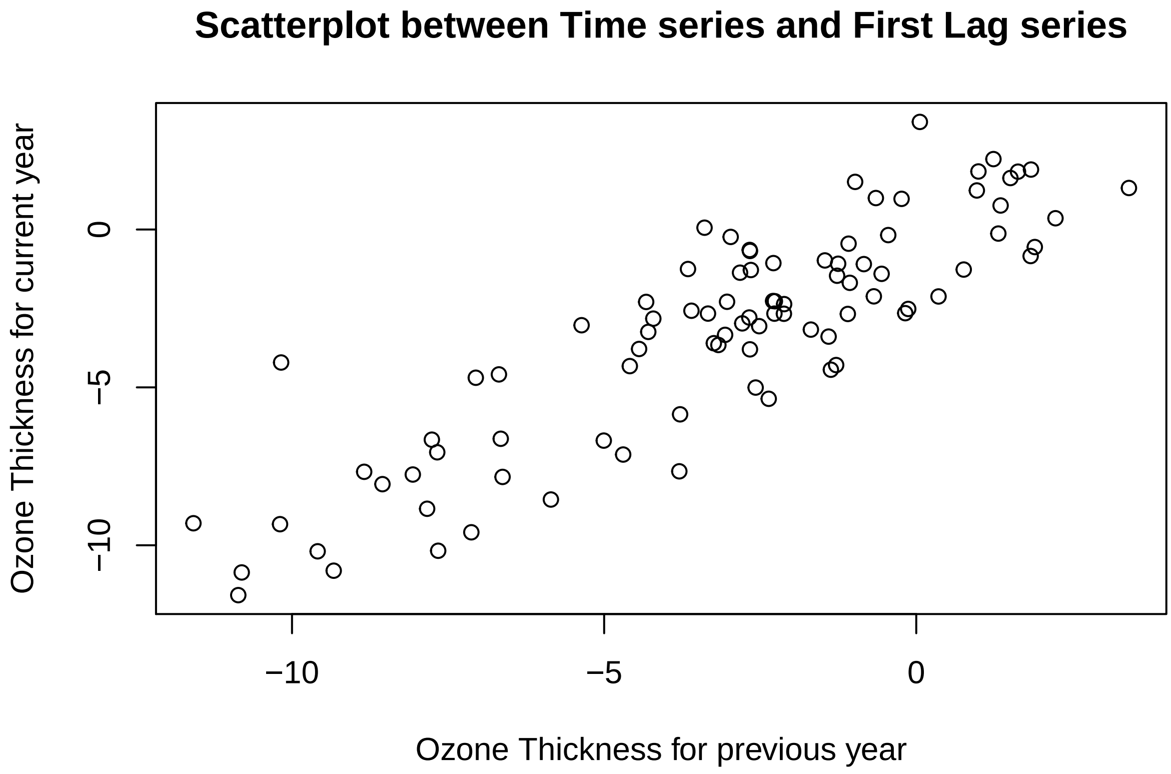
Now that we've removed the **NA** value from the lag series we use the **cor()** function to find out the correlation between the original series **y** and the first lag series **x** using the variable **z** for indexing:

```
cor(y[z], x[z])
```

```
## [1] 0.8700381
```

From the cor() function, we find that the value for the correlation between the original and the first lag series is about 0.87 which means that there is some sort of impact that the previous year has on the current year(good correlation). This significant correlation can backup the fact that the time series has an autoregressive(AR) behaviour as well.
Now, we obtain a scatterplot to visualize the correlation between the original series **y** and the first lag series **x**, where we plot the lag series on the x axis and the original time series on the y axis:

```
plot(y = y, x = x, xlab=’Ozone Thickness for previous year’, ylab=’Ozone Thickness for current year’,
     main=’Scatterplot between Time series and First Lag series’)
```



From the scatterplot, we infer that there is a **moderate positive correlation** between the original time series **y** and its first lag series **x**.

We don't stop here, we push forward to find out the impact on a current year by the year preceding its previous year(second lag). Now that we already have the first lag series stored in **x**, we again use the **zlag()** function on the variable **x** to create the **second lag series** and store it in the variable **v**.

```
v = zlag(x)
head(v)
```

```
## [1]         NA         NA  1.3511844  0.7605324 -1.2685573 -1.4636872
```

Since we have obtained the **second lag series**, we see that now there are two **NA** values for the first and second observations. We follow the same procedure we used for the first lag series. We create a variable **w** to store the index and get rid of the two **NA** values from the second lag series using the **length()** function.

```
w = 3:length(v)
```

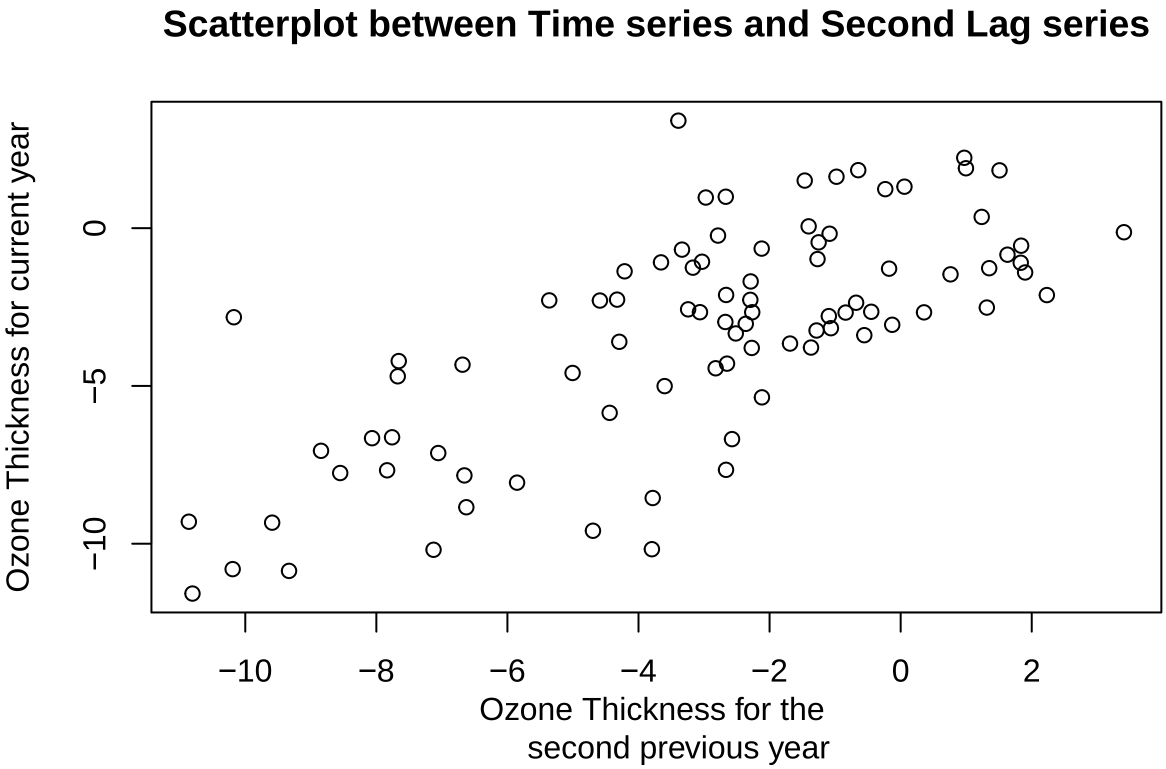
Now that we've removed the two **NA** values from the second lag series we use the **cor()** function to find out the correlation between the original series **y** and the second lag series **v** using the variable **w** for indexing:

```
cor(y[w], v[w])
```

```
## [1] 0.7198518
```

Now, we see that the correlation has somewhat reduced and it can be no longer classified as a strong correlation. Now, we again obtain a scatterplot to visualize the correlation between the original series **y** and the second lag series **v**, where we plot the second lag series on the x axis and the original time series on the y axis:

```
plot(y = y, x = v, ylab='Ozone Thickness for current year', xlab='Ozone Thickness for the
     second previous year',
     main='Scatterplot between Time series and Second Lag series')
```



The plot also tells us that the correlation between the original time series and the second lag series has surely decreased in comparison with the first lag series and it may now be considered as a very **slight positive correlation**.

As a last step, we find the correlation between the original series and its third lag series and store the third lag series in the variable **u**:

```
u = zlag(v)
head(u)
```

```
## [1]         NA         NA         NA  1.3511844  0.7605324 -1.2685573
```

As expected there are three **NA** values. We create a variable **i** to store the index and get rid of the three **NA** values using indexing with the **length()** function again:

```
i = 4:length(u)
```

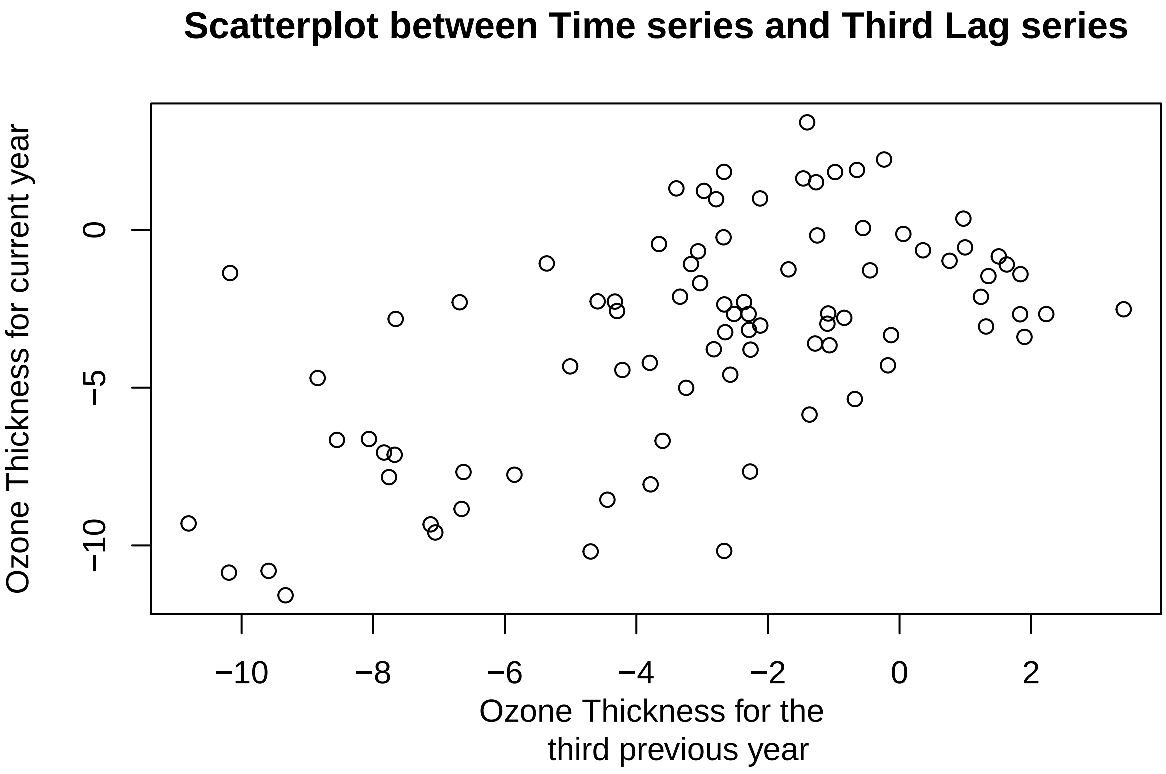
Now that we've removed the unwanted value indices, we use **cor()** function to get the correlation between the original time series **y** and its third lag series **u** using the created variable **i** for indexing:

```
cor(y[i], u[i])
```

```
## [1] 0.592762
```

Okay now the correlation between the series and its third lag series seems too weak with a value of about 0.59. We also obtain a scatterplot for these two series with the original time series **y** on the y axis and the third lag series **u** on the x axis.

```
plot(y = y, x = u, ylab='Ozone Thickness for current year', xlab= 'Ozone Thickness for the
     third previous year',
     main='Scatterplot between Time series and Third Lag series')
```



From the plot, we see that the correlation between the original time series and the third lag series has significantly decreased in comparison with the first and second lag series and it may now be considered as having **almost no correlation**.

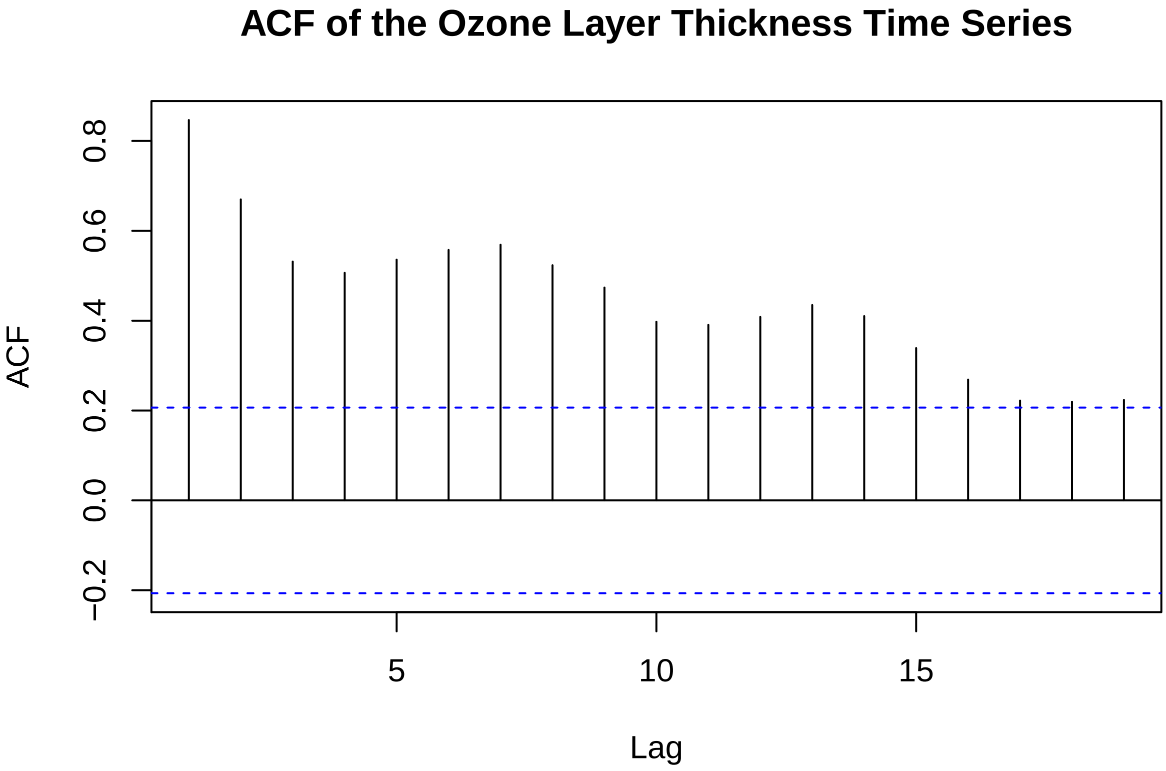
3.0.6 ACF and PACF:

ACF - Auto-Correlation Function

PACF - Partial Auto-Correlation Function

To further confirm the seasonality and the behaviour of the time series we use the **acf()** and **pacf()** plots to check for patterns and significance:

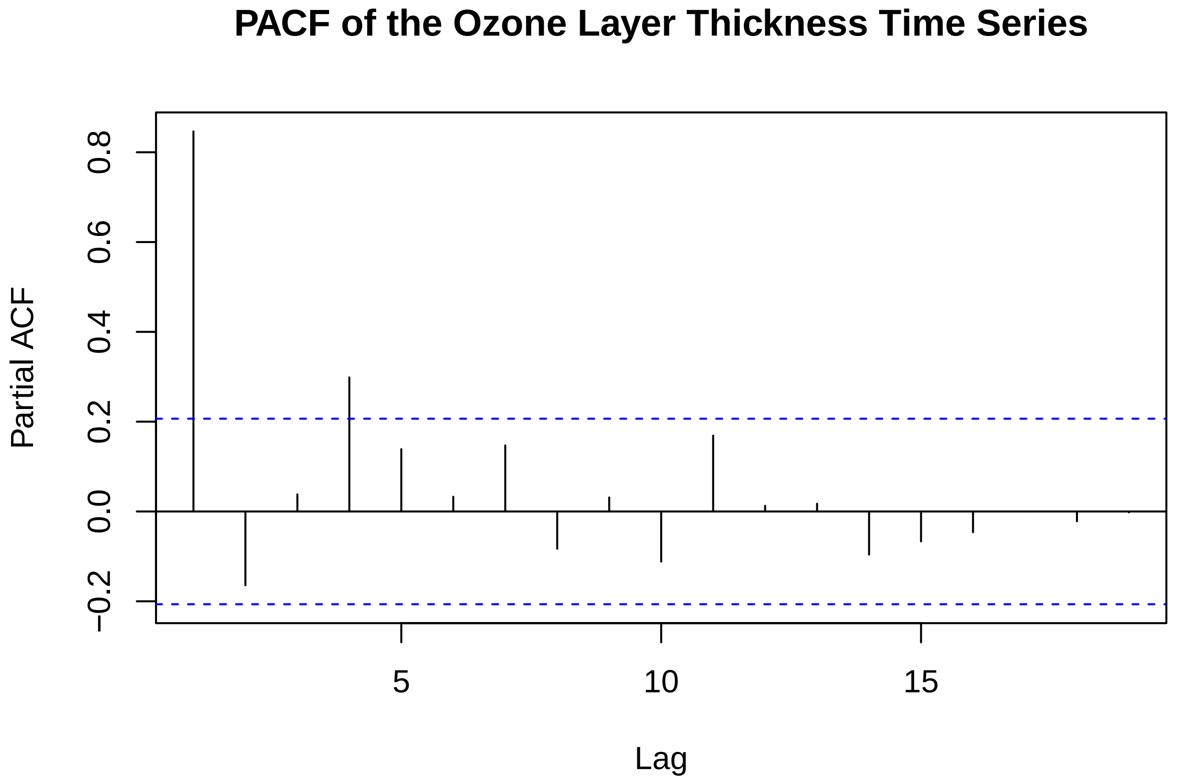
```
acf(ozone.thicknessTS, main='ACF of the Ozone Layer Thickness Time Series')
```



We definitely see a slowly decaying wave pattern in the Auto-Correlation Function plot of the time series suggesting that there's some seasonality at the least in the time series and also that there is non-stationarity. As there is significant autocorrelation for a large number of lags, we also infer the behaviour of the time series to be dominantly **Moving Average(MA)**. Alternatively, since there is an obvious downward trend in the series, the trend tends to mask out the MA characteristics from the ACF plot.

Now, let us plot the Partial Auto-Correlation Function of the time series:

```
pacf(ozone.thicknessTS, main='PACF of the Ozone Layer Thickness Time Series')
```



From the PACF plot, we can see that about two of the lags have significant auto-correlation and this too follows a decreasing cosine wave-like pattern, from which we can also infer that the time series has **Auto-regressive(AR)** behaviour as well.

4 Chapter 1

4.1 Modelling:

In this chapter, we'll start by fitting the main four deterministic trend models and predicting the yearly changes for a given period of time using our best fitting model.

4.1.1 Model Fitting and Diagnostic Checking:

In this section, we aim to find the best-fitting model by trying out and analyzing various models. Assuming our series to have a **deterministic trend**, we try out the main four trend models for modelling deterministic trends:

1) Linear Model
2) Quadratic Model
3) Cosine or Harmonic Model
4) Seasonal or Cyclical Model

4.1.2 Linear Model:

First off, we begin with the most basic and simplest model which is the linear model.

4.1.3 Fitting:

To fit a linear trend model or any other model for that matter, we extract the time **t** from the time series object using the **time()** function.

```
t <- time(ozone.thicknessTS)
t
```

```
## Time Series:
## Start = 1927
## End = 2016
## Frequency = 1
##  [1] 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941
## [16] 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956
## [31] 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971
## [46] 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986
## [61] 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001
## [76] 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
```

Next, we use the **lm()** function which fits the linear model using our time series **ozone.thicknessTS** as the dependent data and time **t** as the independent data into our linear model variable **linear.model**.

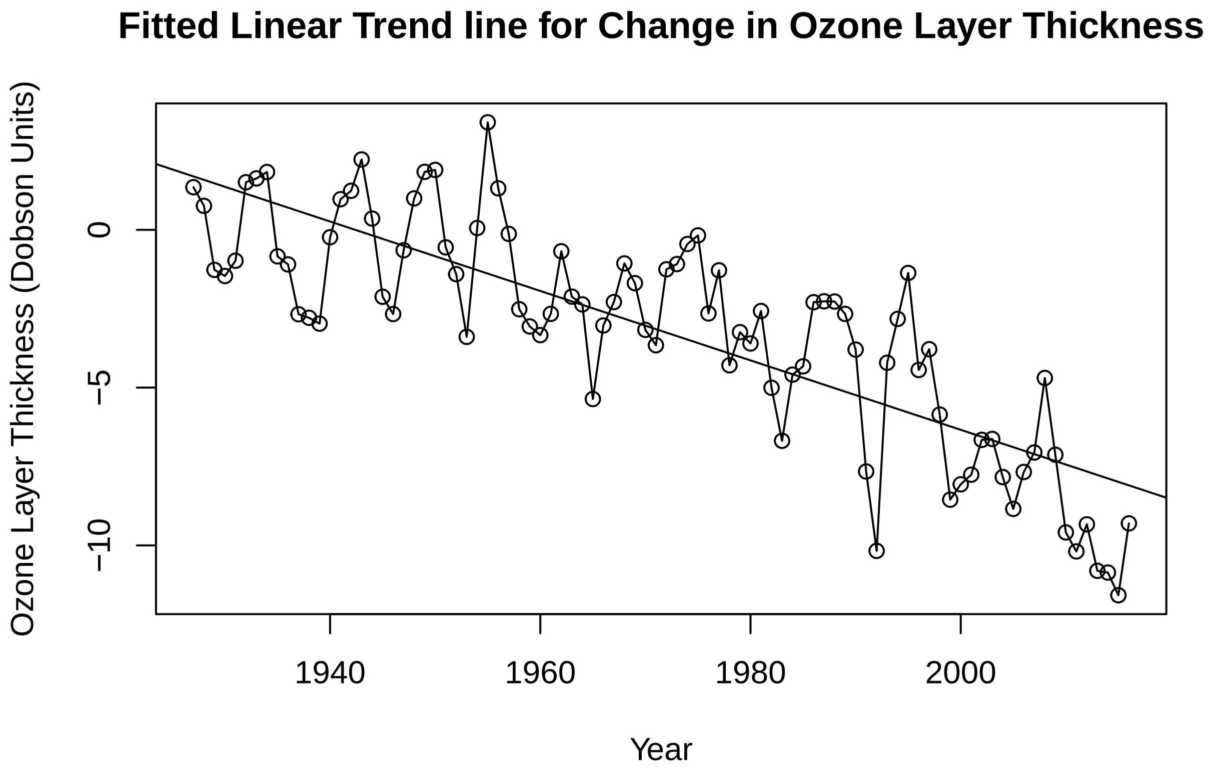
```
linear.model <- lm(ozone.thicknessTS ~ t)
summary(linear.model)
```

```
##
## Call:
## lm(formula = ozone.thicknessTS ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7165 -1.6687  0.0275  1.4726  4.7940
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 213.720155  16.257158   13.15   <2e-16 ***
## t            -0.110029   0.008245  -13.34   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 2.032 on 88 degrees of freedom
## Multiple R-squared:  0.6693, Adjusted R-squared:  0.6655
## F-statistic: 178.1 on 1 and 88 DF,  p-value: < 2.2e-16
```

From the **summary()** function, we get the residual details which we will analyze in detail at a later stage. We also see that the coefficients are both significant and also the overall model p-value is also significant which translates to our model as a whole being significant and effective for the most part. The estimate value for **t** is about **-0.11** which means that there will be that amount of increment(decrement in our case) per time point, year, in our case. The adjusted R-squared value is about 0.66 which is not considered to be a very ideal value, but also not that bad of a value.

Now let's add the fitted linear trend line to the original time series plot using the **abline()** function:

```r
plot(ozone.thicknessTS, ylab='Ozone Layer Thickness (Dobson Units)', xlab='Year',
     main='Fitted Linear Trend line for Change in Ozone Layer Thickness', type='o')
abline(linear.model)
```



The linear fitted trend line looks sufficient in estimating the model. However, let us delve deeper and analyze the residuals from the linear deterministic trend model and check whether the residuals contain any important information about the time series.

4.1.4 Residual Analysis/Diagnostic Checking:

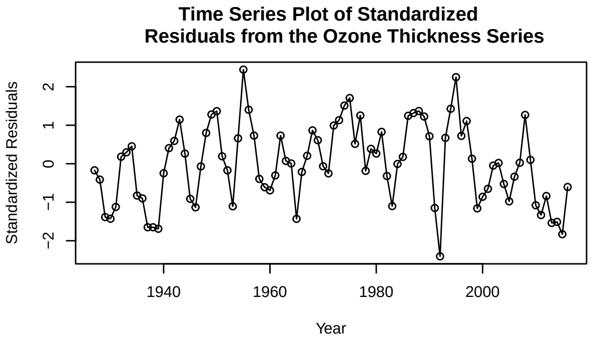
The following plots along with the Shapiro-Wilk Test are done for this linear trend model in order to analyze the effectiveness of the model and perform the diagnostic checking. A detailed summary of the residual analysis is given below after the plots. To begin with, we first use the **rstudent()** function to get the standardized residuals and store it in the variable **res.linearmodel** to make it easier for us to reuse the variable for different plots.

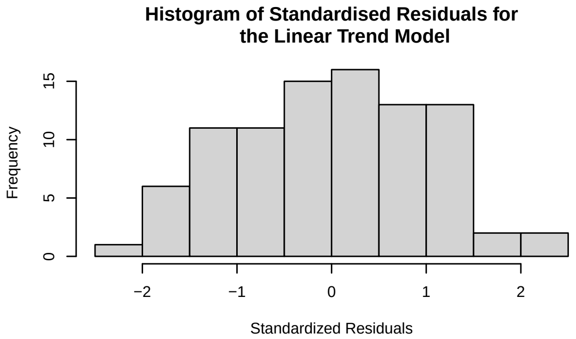
```r
par(mfrow=c(3,2))

res.linearmodel <- rstudent(linear.model)

# Time Series Plot of Standardized Residuals
```

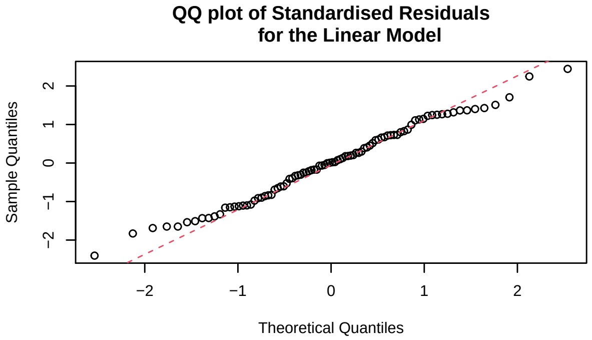
```
plot(y=res.linearmodel,x=as.vector(time(ozone.thicknessTS)), xlab=’Year’,
     ylab=’Standardized Residuals’,type=’o’, main = "Time Series Plot of Standardized
     Residuals from the Ozone Thickness Series")

# Histogram
hist(res.linearmodel,xlab=’Standardized Residuals’,
     main = "Histogram of Standardised Residuals for
     the Linear Trend Model")

# QQ plot
qqnorm(res.linearmodel, main = "QQ plot of Standardised Residuals
       for the Linear Model")
qqline(res.linearmodel, col = 2, lwd = 1, lty = 2)

# ACF
acf(res.linearmodel, main = "ACF of Standardized Residuals")

# PACF
pacf(res.linearmodel, main = "PACF of Standardized Residuals")

# Shapiro Wilk test
shapiro.test(res.linearmodel)
```

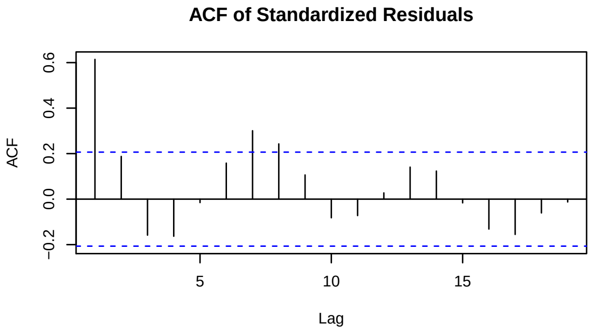
```
##
##  Shapiro-Wilk normality test
##
## data:  res.linearmodel
## W = 0.98733, p-value = 0.5372
```

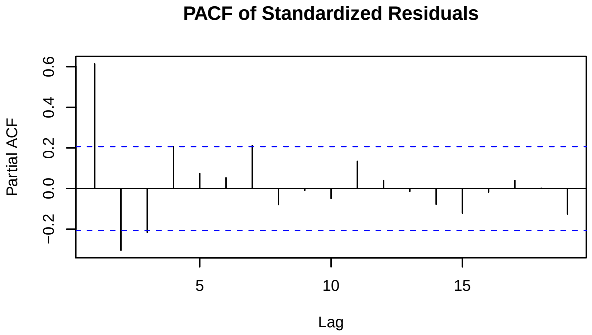
```
par(mfrow=c(1,1))
```











4.1.5 Summary of the Residual Analysis:

• The time series plot for the standardized residuals looks better in that it has an almost random pattern. However, there seems to be a slight trend which slowly rises up as a curve and then decreases. There still seems to be some kind of seasonality and changing variance for some parts. The behaviour is still a mix of MA/AR with MA being the dominant one.

• In the check for symmetricity, The histogram obtained is not exactly a symmetric one. It seems to be a slightly left-skewed distribution.

• For the QQ plot, most of the points are closer to the reference line, however, a few points at the beginning and at the end are further away. This seems somewhat appropriate but let us move on to the next diagnostic check.

• The ACF plot shows that there is autocorrelation for about 3 lags and the PACF plot also shows that there is still autocorrelation for about 4 lags. We are not exactly pleased with these results as there still exists some autocorrelation in the ACF and PACF plots and therefore our model has failed to capture some information from the time series. A wave-like pattern can also be seen in the ACF plot suggesting there might be some seasonality in the time series.

- Finally, we decide the normality of the standardized residuals using the **Shapiro-Wilk test**. This test calculates the correlation between the residuals and the corresponding normal quantiles. We see that the test gives us a p-value of about **0.54** which is definitely greater than **0.05**. Thus, we fail to reject the null hypothesis which states that there is normality. From this test, we can confirm normality for the standardized residuals.
- From the above summary, we can say that we are satisfied with this linear trend model. However, there are a lot of shortcomings for the model and there's also lags with significant autocorrelation in the ACF and PACF plots and therefore there must be a better model. So we try out the Quadratic Trend Model next.

4.1.6 Quadratic Model:

Next, keeping the characteristics of our linear model in mind, we proceed to the next one, which is the quadratic deterministic trend model.

4.1.7 Fitting:

Let us now try out the quadratic trend model. To use this model we use the already extracted time **t** from the time series, just like we did for the linear model using the **time()** function. But here, since it is a quadratic trend model we also make use of the square of time (t^2) for fitting the model. We use the linear regression function **lm()** and store our model into the variable **quad.model**.

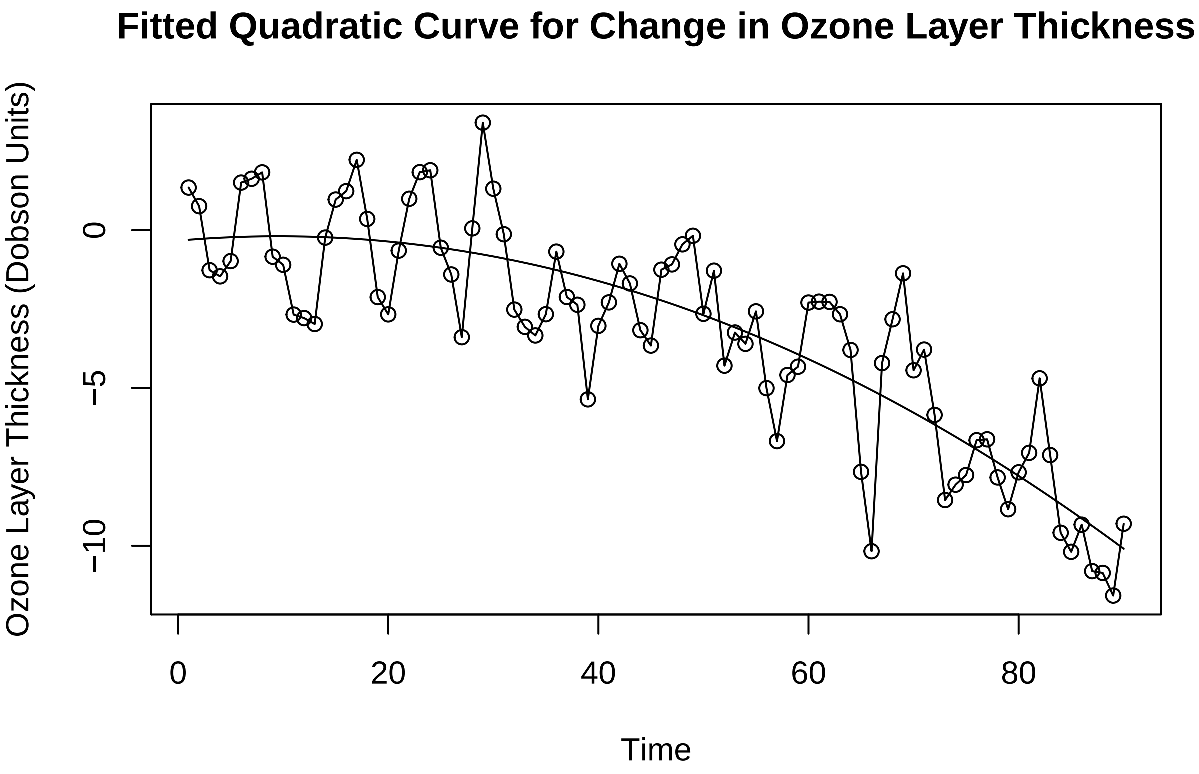
```
t2 <- t^2
quad.model <- lm(ozone.thicknessTS ~ t + t2)
summary(quad.model)
```

```
##
## Call:
## lm(formula = ozone.thicknessTS ~ t + t2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.1062 -1.2846 -0.0055  1.3379  4.2325
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.733e+03  1.232e+03  -4.654 1.16e-05 ***
## t            5.924e+00  1.250e+00   4.739 8.30e-06 ***
## t2          -1.530e-03  3.170e-04  -4.827 5.87e-06 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1.815 on 87 degrees of freedom
## Multiple R-squared:  0.7391, Adjusted R-squared:  0.7331
## F-statistic: 123.3 on 2 and 87 DF,  p-value: < 2.2e-16
```

From the model summary, we see that all the three coefficients are significant which indicates that our model might be significant in explaining the time series, which is also supported by the fact that the overall p-value is also significant(<0.05). The adjusted R-squared however, is the best indication that the quadratic model is a good and in fact, a better fit than the linear model. The value of about 0.73 is very close to being an ideal value for indicating a good fitting model.

Now, let us add the fitted quadratic trend line to the time series plot:

```
plot(ts(fitted(quad.model)),  xlab=’Time’, ylab=’Ozone Layer Thickness (Dobson Units)’,
     main=’Fitted Quadratic Curve for Change in Ozone Layer Thickness’, ylim=
     c(min(c(fitted(quad.model), as.vector(ozone.thicknessTS))), max(c(fitted(quad.model),
     as.vector(ozone.thicknessTS)))))
lines(as.vector(ozone.thicknessTS), type=’o’)
```



From the fitted visualization, we can see that visually it also looks like a better fit than the linear model. However, we cannot be sure just by visually observing the plot and hence we move on to the residual analysis/diagnostic checking.

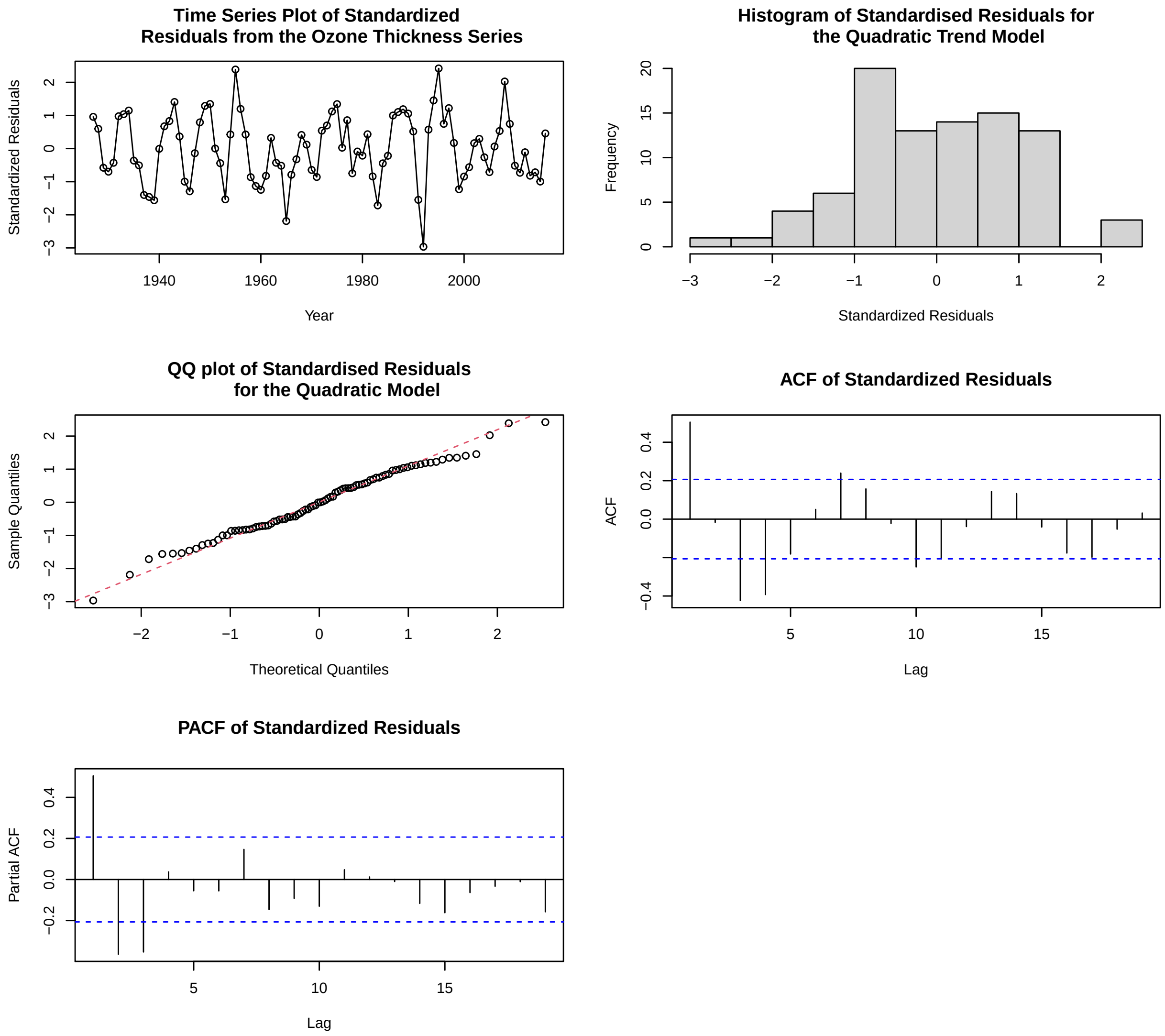
4.1.8 Residual Analysis/Diagnostic Checking:

The following plots along with the Shapiro-Wilk Test are done again for this quadratic trend model in order to analyze the effectiveness of the model and perform the diagnostic checking. A detailed summary of the residual analysis is given below after the plots. To begin with, we first use the **rstudent()** function to get the standardized residuals and store it in the variable **res.quad** to make it easier for us to reuse the variable for different plots.

```
par(mfrow=c(3,2))

res.quad <- rstudent(quad.model)

# Time Series Plot of Standardized Residuals
plot(y=res.quad,x=as.vector(time(ozone.thicknessTS)), xlab=’Year’,
     ylab=’Standardized Residuals’,type=’o’, main = "Time Series Plot of Standardized
     Residuals from the Ozone Thickness Series")

# Histogram
hist(res.quad,xlab='Standardized Residuals',
     main = "Histogram of Standardised Residuals for
     the Quadratic Trend Model")

# QQ plot
qqnorm(res.quad, main = "QQ plot of Standardised Residuals
       for the Quadratic Model")
qqline(res.quad, col = 2, lwd = 1, lty = 2)

# ACF
acf(res.quad, main = "ACF of Standardized Residuals")

# PACF
pacf(res.quad, main = "PACF of Standardized Residuals")

# Shapiro Wilk test
shapiro.test(res.quad)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.quad
## W = 0.98889, p-value = 0.6493
```

```
par(mfrow=c(1,1))
```



4.1.9 Summary of the Residual Analysis:

• The obtained plot of the standardized residuals looks a lot better than the one for the linear model. The trend seems to have been captured completely by this quadratic model and hence we observe that there is no significant trend in the residual plot. There might still be some seasonality from visually observing the plot and some parts of the plot still shows some change in variance. The behaviour, even though it may be screened by the seasonality, is a mix of MA/AR with MA being the dominant one.

• The obtained histogram looks like a more left-skewed distribution and not very symmetric. We cannot infer much from this distribution. We move on to the next check.

• We can see that the QQ plot still shows points at the ends drawing further away from the main reference line, however, in comparison, it is better than the QQ plot for the linear trend model.

• Both the ACF and PACF plots have about 3 or more lags with autocorrelation and thus this denotes that the residuals still contain a lot of information which the model has failed to capture. Further, the ACF plot shows a wave-like pattern which might indicate the presence of some seasonality in the time series which has not been captured by the model. Inspite of not being the most effective model, out of both the linear and the quadratic trend models, we would certainly go for the quadratic trend model as it is slightly more effective in comparison.

- As a final step, we decide the normality of the standardized residuals using the **Shapiro-Wilk test**. This test calculates the correlation between the residuals and the corresponding normal quantiles. This test confirms that the standardized residuals are indeed normally distributed as the p-value (0.6493) > 0.05(significance factor).

This quadratic deterministic trend model seems to be quite better than the linear trend model. Both have shortcomings and limitations, sure, but in comparison, supported by the results from the diagnostic checking and residual analysis in combination with the better R-Squared value and no trend in the time series plot of the standardized residuals, we can make a statement and say that we'd prefer the quadratic model to the linear model. However, since both the models failed to capture the seasonality as reflected by the patterns in the ACF plots, we go further and try out seasonal and harmonic models as well.

4.1.10 Time Series with Seasonality:

Since both the linear and the quadratic trend models have failed to capture the seasonality from the original time series, we move on to try and fit the cosine or harmonic model which we think might better capture the seasonality. This decision of ours is backed up by the fact that we observe a gradually decreasing wave-like pattern from the ACF plot of the ozone layer thickness time series which surely denotes some seasonality in the time series.

When we use **frequency = 1** in the time series object creation, we directly assume that there is no seasonality and thus our linear and quadratic models have failed to capture the seasonality. For this purpose, we now assume seasonality and use a frequency greater than 1 which can be calculated roughly from the ACF plot of the time series shown in **Figure 4**:

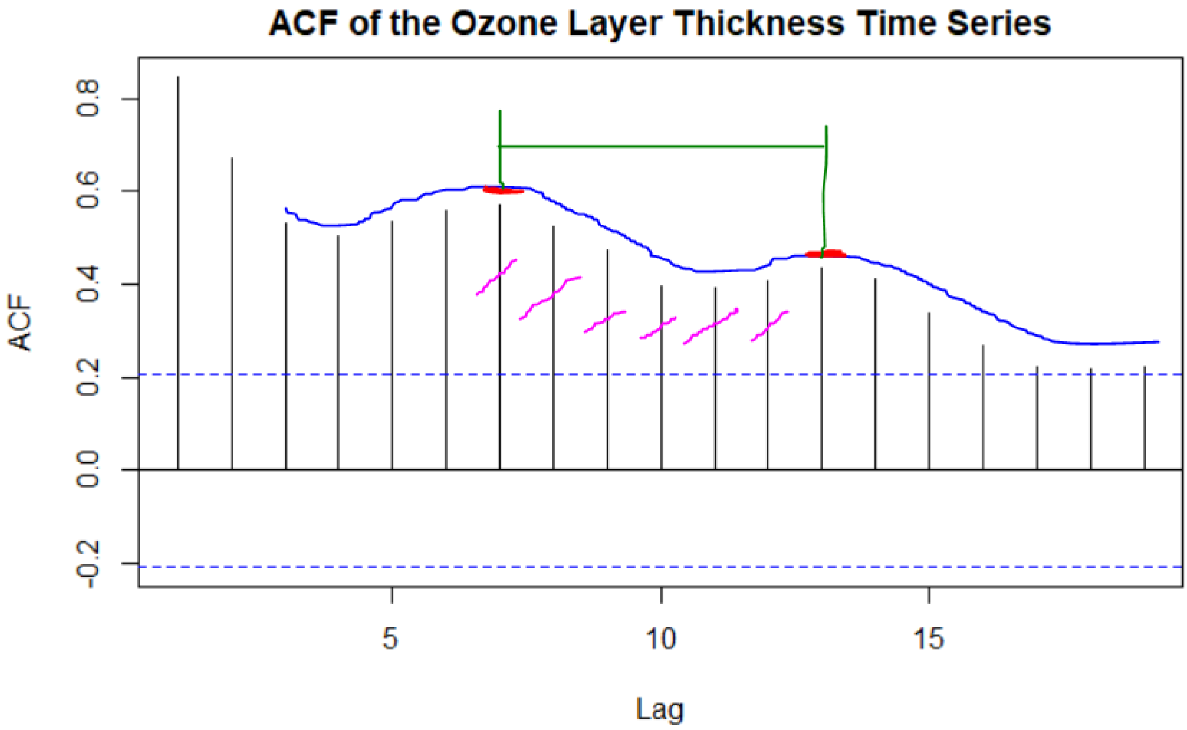


Figure 4: Selecting Frequency For Seasonality

From the ACF plot, using this technique to estimate the frequency of the seasonality, we obtain the frequency to be either **6** or **5**. On trying out both of these frequencies for the time series object creation and also other adjacent frequencies, we finalize on the frequency of **5** as it has the better statistical characteristics and diagnostics among its neighbors.

Now, we create a separate time series which we will be using just for the **cosine/harmonic** and **seasonal/cyclical** models where we will assume seasonality and input a frequency of **5**. We store the time series object in the variable **ozone.seasonalTS**. Here, we use the matrix() function inside the time series ts() function to create the time series and use **frequency = 5** as decided, where we assume the yearly data to repeat itself every 5 years denoting a **cyclic pattern**.

```
ozone.seasonalTS <- ts(as.vector(matrix(ozone.thickness$‘Ozone Thickness‘, nrow=18, ncol=5)),
                       frequency = 5)
ozone.seasonalTS
```

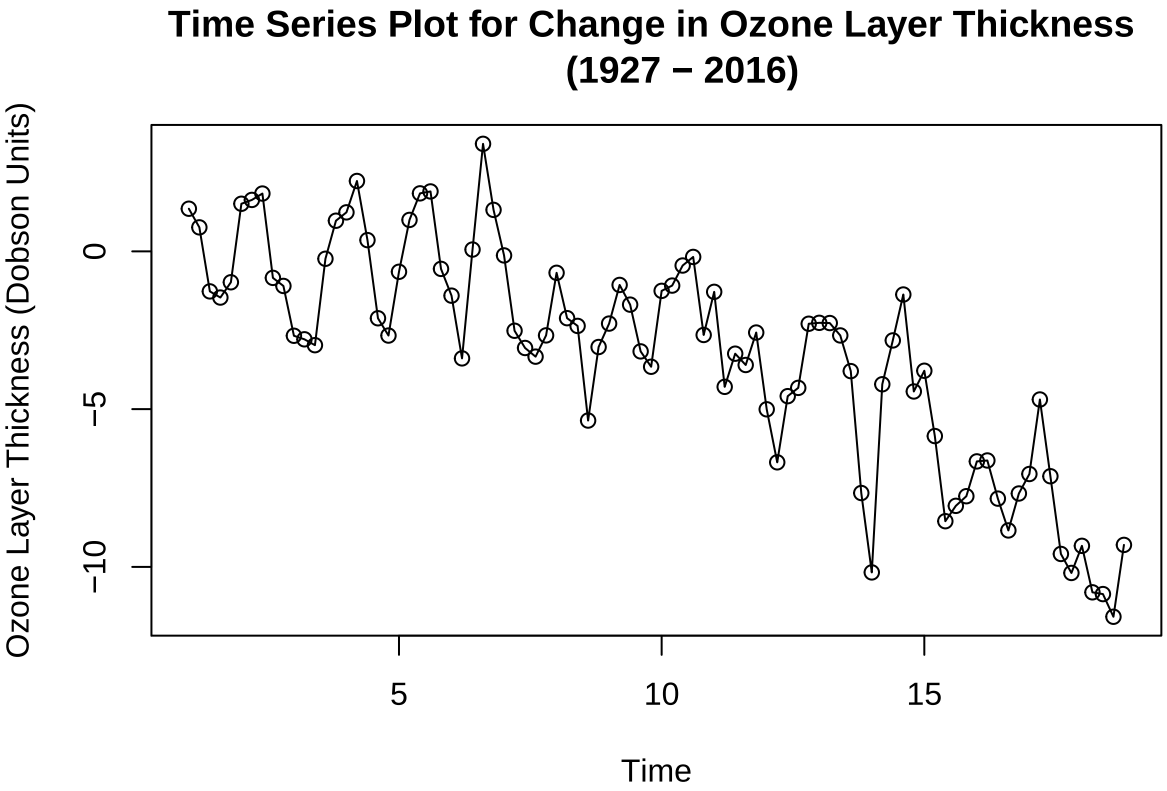
```
## Time Series:
## Start = c(1, 1)
## End = c(18, 5)
## Frequency = 5
##  [1]   1.35118436   0.76053242  -1.26855735  -1.46368717  -0.97920302
##  [6]   1.50856746   1.62991681   1.83242333  -0.83968364  -1.09611566
## [11]  -2.67457473  -2.78716606  -2.97317944  -0.23495548   0.97067000
## [16]   1.23652307   2.23062331   0.35671637  -2.12028099  -2.66812477
## [21]  -0.64702795   0.99608564   1.83817742   1.89922697  -0.55488121
## [26]  -1.40387419  -3.39178762   0.05777194   3.40717183   1.31488379
## [31]  -0.12882457  -2.51580137  -3.06205664  -3.33637179  -2.66332198
## [36]  -0.67958655  -2.11660422  -2.36318997  -5.36156537  -3.03103458
## [41]  -2.28838624  -1.06438684  -1.68813570  -3.16974819  -3.65647649
## [46]  -1.25151090  -1.08431732  -0.44863234  -0.17636387  -2.64954530
## [51]  -1.28317654  -4.29289634  -3.24282341  -3.60135297  -2.57288652
## [56]  -5.00586059  -6.68548244  -4.58870210  -4.32654629  -2.29370761
## [61]  -2.26456266  -2.27184846  -2.66440082  -3.79556478  -7.65843185
## [66] -10.17433972  -4.21230497  -2.82287161  -1.36776491  -4.43997062
## [71]  -3.78323838  -5.85304107  -8.55125744  -8.06501289  -7.75975806
## [76]  -6.65633206  -6.62708203  -7.83548356  -8.84424264  -7.67352209
## [81]  -7.05582939  -4.69497353  -7.12712128  -9.58954985 -10.19222042
## [86]  -9.33224686 -10.80567444 -10.86096923 -11.57941376  -9.30284452
```

Alternatively, we can also use the following code to create the same cyclic time series with **frequency = 5**. Note: The following code has not been run and is merely shown as an alternative/example.

```
exampleTS <- ts(data = as.vector(ozone.thickness$‘Ozone Thickness‘), start = 1927, frequency = 5)
```

Now we plot the created time series and check for the **five valid points**:

```
plot(ozone.seasonalTS, xlab=’Time’, ylab=’Ozone Layer Thickness (Dobson Units)’,
     main=’Time Series Plot for Change in Ozone Layer Thickness
     (1927 - 2016)’, type=’o’)
```



4.1.11 Five Valid Points:

Now that we've obtained a plot of our time series (this time with seasonality assumed), we can start with the descriptive analysis. We begin with analyzing and commenting on the five valid points:

- **1) Trend**:

Just by having a glance at the plot, we can say that there is the same slow downward trend as seen in the original time series which might denote non-stationarity.

- **2) Seasonality**:

Since we have assumed seasonality and set a frequency of **5** while creating the time series object, we say that the time series has, at the least, the slightest seasonality for sure.

- **3) Changing Variance**:

Yes, there is a changing variance for some parts of the time series plot, just the same as our original time series.

- **4) Change Point/Intervention**:

There is no obvious or abrupt/sudden change point or intervention.

- **5) Behaviour**:

From an overall perspective, we can see that the dominant behaviour is Moving Average(MA), however, on zooming in, we can observe that the series has a bit of Auto-regressive(AR) behaviour as well. On the other hand, an important thing to keep in mind here is that since our time series has obvious seasonality, it may mask or screen the overall behaviour of the time series and thus we cannot be too sure about its behaviour.

Now that we’ve got a good overlook on the five valid points, let us move on to the modelling and fitting stage.

4.1.12 Cosine or Harmonic Model:

We begin with the cosine/harmonic model first in our attempt to fit a model for the ozone layer thickness time series with assumed seasonality. Here we will be using the **ozone.seasonalTS** time series object for fitting and diagnostics.

4.1.13 Fitting:

First off, we create a variable **har.** which stores the result from the **harmonic()** function taking in the seasonal time series **ozone.seasonalTS** as an input parameter. Next, we create a variable **seasonal.data** which stores the results from both the seasonal time series and the **har.** variable which we have already created, as a dataframe.

After that is done, we now create our model into the variable **cosine.model** using the **lm()** function and using the formula for creating a harmonic model which include the **cos** and **sin** parameters along with the data given as the already created variable **seasonal.data**.

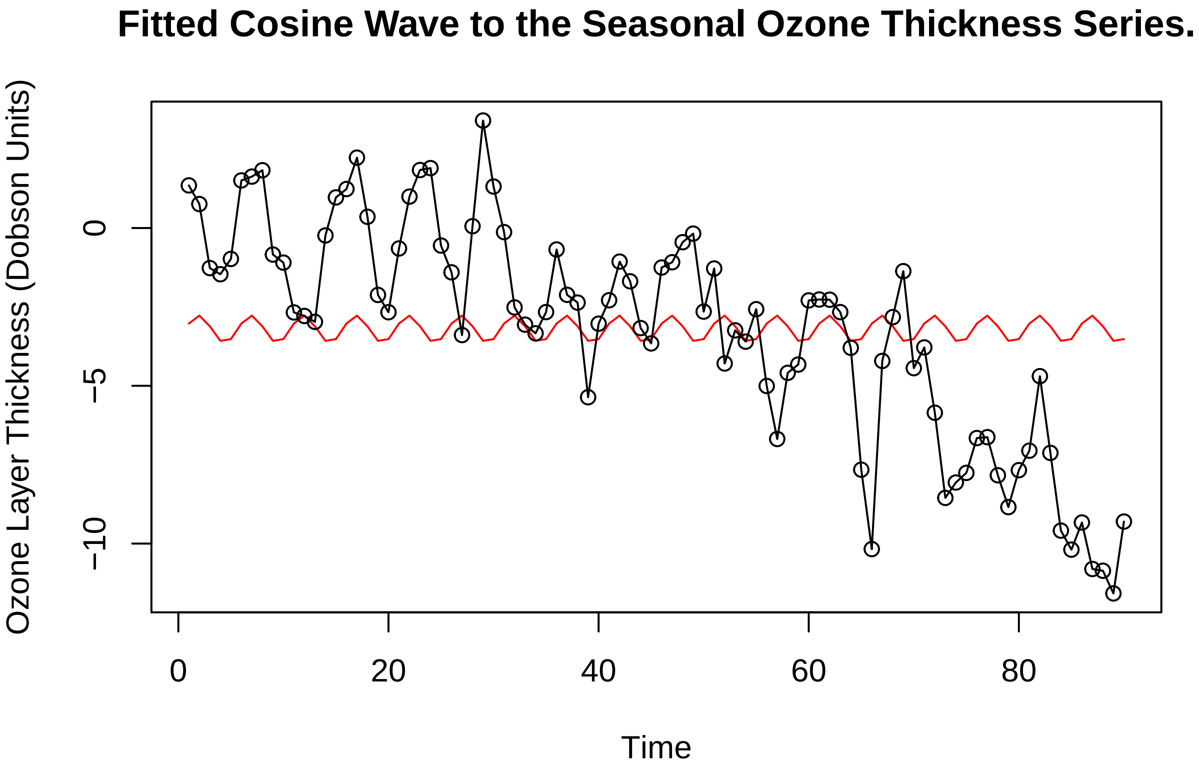
```r
har. <- harmonic(ozone.seasonalTS, 1)
seasonal.data <- data.frame(ozone.seasonalTS, har.)
cosine.model <- lm(ozone.seasonalTS ~ cos.2.pi.t. + sin.2.pi.t., data = seasonal.data)
summary(cosine.model)
```

```
##
## Call:
## lm(formula = ozone.seasonalTS ~ cos.2.pi.t. + sin.2.pi.t., data = seasonal.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0305 -1.8865  0.4964  2.4131  6.9823
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.2023     0.3732  -8.581 3.24e-13 ***
## cos.2.pi.t.   0.1761     0.5278   0.334    0.739
## sin.2.pi.t.   0.3919     0.5278   0.742    0.460
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 3.54 on 87 degrees of freedom
## Multiple R-squared:  0.007559,   Adjusted R-squared:  -0.01526
## F-statistic: 0.3313 on 2 and 87 DF,  p-value: 0.7189
```

From the **summary()** function on the cosine trend model, we can see really problematic and unwanted results such as the **cos** and **sin** coefficients being insignificant, the adjusted R-squared value being too low to be even considered for fitting. The adjusted R-squared value is about -0.015, the negative value denotes that the cosine trend model is highly insignificant. Another thing which can be observed from the summary is the overall **p-value**(0.7189) which is $>$ 0.05 and thus the created model as a whole is insignificant and might not capture majority of the information.

Anyway, we proceed and fit the cosine model to the seasonal time series using the following chunk of code:

```r
plot(ts(fitted(cosine.model)), ylab='Ozone Layer Thickness (Dobson Units)',
     main = "Fitted Cosine Wave to the Seasonal Ozone Thickness Series.",
     ylim = c(min(c(fitted(cosine.model), as.vector(ozone.seasonalTS))),
              max(c(fitted(cosine.model), as.vector(ozone.seasonalTS)))),
     col = "red")
lines(as.vector(ozone.seasonalTS),type="o")
```



From just a glance at the cosine trend fitted plot, we immediately confirm our suspicions on the efficiency/effectivity of the model. The fitted model does not fit the time series data and so there might be seasonal autocorrelation in the data but our cosine model could not capture it. We should also keep in mind that deterministic trend models cannot capture stochastic trends caused by autocorrelation. This seems to be about the worst model that we could be fitting to our seasonal time series and thus we decide that we will not be using this model. However, we still hope to find proof supporting our decision from the residual analysis.

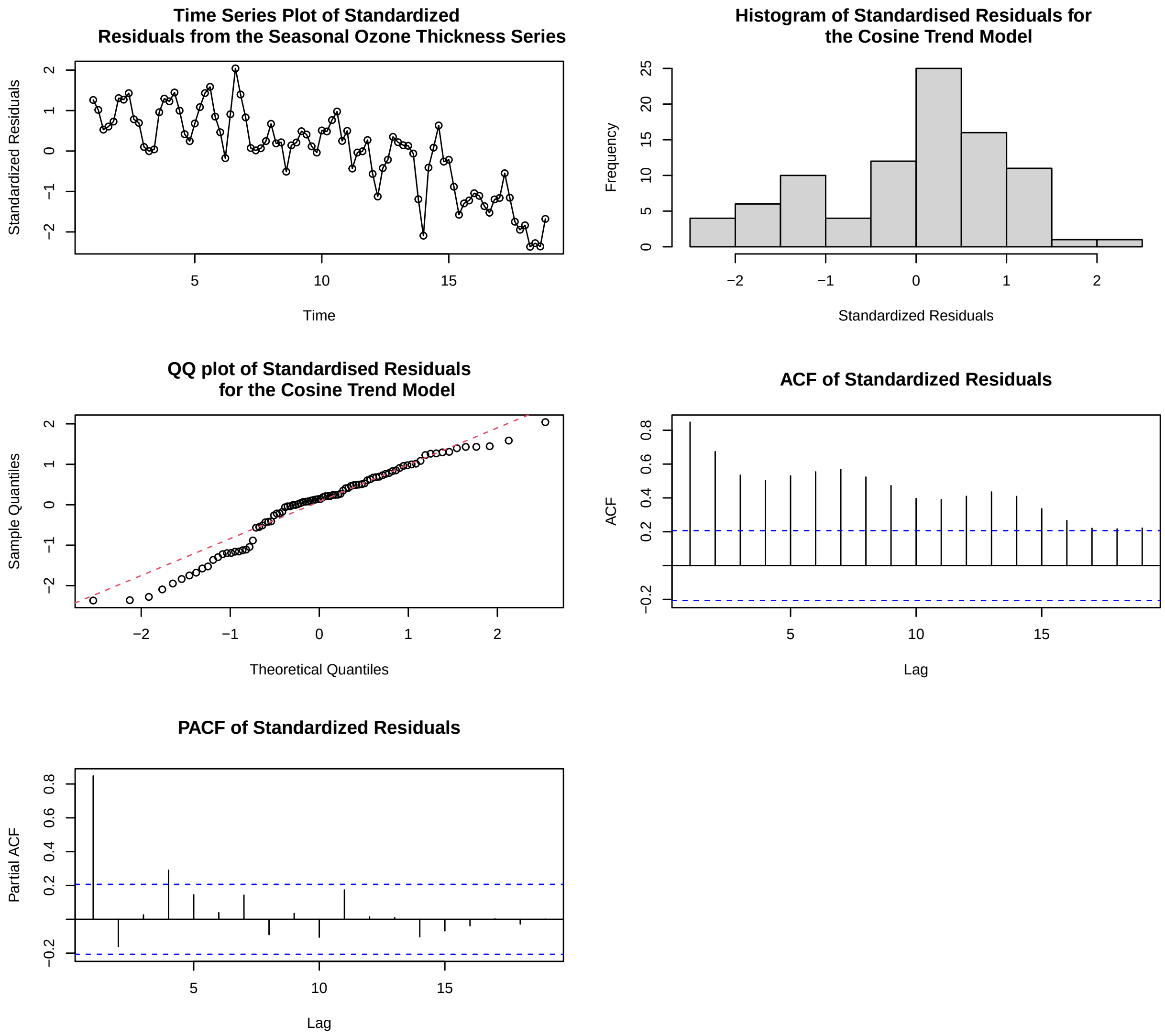
4.1.14 Residual Analysis/Diagnostic Checking:

The following plots along with the Shapiro-Wilk Test are done again for this cosine/harmonic trend model in order to analyze the effectiveness of the model and perform the diagnostic checking. A detailed summary of the residual analysis is given below after the plots. To begin with, we first use the **rstudent()** function to get the standardized residuals and store it in the variable **res.cosine** to make it easier for us to reuse the variable for different plots.

```r
par(mfrow=c(3,2))

res.cosine <- rstudent(cosine.model)

# Time Series Plot of Standardized Residuals
plot(y=res.cosine,x=as.vector(time(ozone.seasonalTS)), xlab=’Time’,
     ylab=’Standardized Residuals’,type=’o’, main = "Time Series Plot of Standardized
     Residuals from the Seasonal Ozone Thickness Series")

# Histogram
hist(res.cosine,xlab=’Standardized Residuals’,
     main = "Histogram of Standardised Residuals for
     the Cosine Trend Model")

# QQ plot
qqnorm(res.cosine, main = "QQ plot of Standardised Residuals
       for the Cosine Trend Model")
qqline(res.cosine, col = 2, lwd = 1, lty = 2)

# ACF
acf(res.cosine, main = "ACF of Standardized Residuals")

# PACF
pacf(res.cosine, main = "PACF of Standardized Residuals")

# Shapiro Wilk test
shapiro.test(res.cosine)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.cosine
## W = 0.95735, p-value = 0.004894
```

```r
par(mfrow=c(1,1))
```



4.1.15 Summary of the Residual Analysis:

- The obtained time series plot of the standardized residuals is almost the exact copy of our original seasonal time series meaning that our cosine model has captured almost nothing!

- The obtained histogram looks like a slight left-skewed distribution and not very symmetric. We cannot infer much from this distribution. We move on to the next check.

- In comparison with the linear and quadratic models, we see from the QQ plot that almost 50% of the time points are straying away from the reference line which is a further indicator of how inefficient the model must be in practice.

- Both the ACF and PACF plots have a lot of lags with significant autocorrelation and thus this denotes that the residuals still contain a lot of information which the model has failed to capture. Further, the ACF plot shows a wave-like pattern which indicates the presence of some seasonality in the time series which has not been captured by the model.

- As a final and hopeless step, we decide the normality of the standardized residuals using the **Shapiro-Wilk test**. This test calculates the correlation between the residuals and the corresponding normal quantiles. This test confirms that the distribution of the standardized residuals is surely not normal as we can see that the **p-value**(0.00489) < 0.05.

- Contrary to our expectations, this cosine trend model has also failed to capture the seasonality and is in fact, a bad fit in the overall sense. So we will definitely not be using this model which can be even justified from the bad results obtained from the residual analysis/diagnostic checking.

4.1.16 Seasonal or Cyclical Model:

Since we got really unexpected and bad results from the cosine deterministic trend model, we now try the cyclic model(as we have assumed the yearly data to repeat itself every 5 years) and see if we can capture the seasonality from the seasonal time series **ozone.seasonalTS**.

4.1.17 Fitting:

Here, we use the **season()** function to get the various seasons from the time series and create a variable **month.** to store them. Then we create our main cyclic model into the variable **seasonal.model** using the **lm()** function with the variable **month.** and we include the term **-1** as a parameter to remove the intercept term.

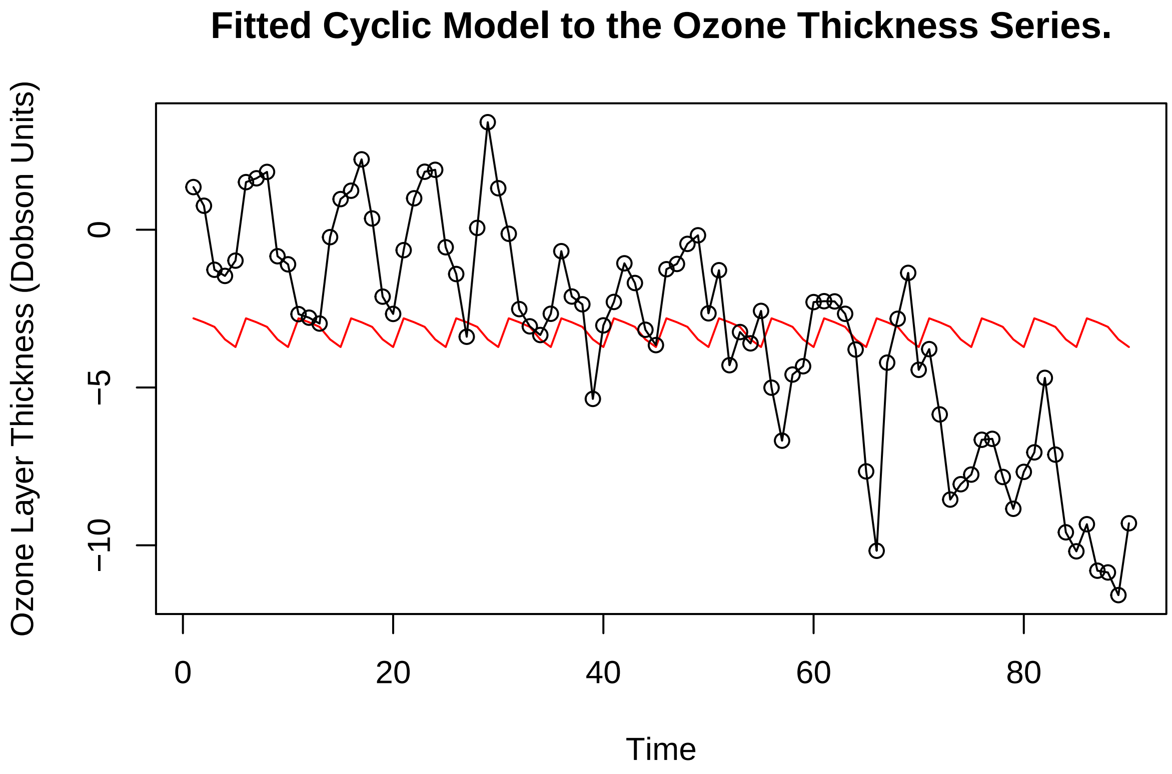
```
month.=season(ozone.seasonalTS)
seasonal.model=lm(ozone.seasonalTS ~ month. -1)
summary(seasonal.model)
```

```
##
## Call:
## lm(formula = ozone.seasonalTS ~ month. - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.1035 -1.8549  0.5309  2.5058  6.8830
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## month.Season-1  -2.8074     0.8434  -3.329 0.001291 **
## month.Season-2  -2.9326     0.8434  -3.477 0.000801 ***
## month.Season-3  -3.0785     0.8434  -3.650 0.000452 ***
## month.Season-4  -3.4759     0.8434  -4.121 8.72e-05 ***
## month.Season-5  -3.7170     0.8434  -4.407 3.04e-05 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 3.578 on 85 degrees of freedom
## Multiple R-squared:  0.4617, Adjusted R-squared:   0.43
## F-statistic: 14.58 on 5 and 85 DF,  p-value: 2.632e-10
```

From the **summary()** function, we can deduce that there seems to be huge improvement from the cosine model, because, here we can see that all the coefficients are significant along with the overall p-value for the model which is also(significantly) < 0.05 and thereby denoting significance. The adjusted R-squared value has also improved significantly and is now about 0.43 which is still not considered ideal.

Anyway, we proceed and fit the cyclic trend model to the time series using the following chunk of code:

```
plot(ts(fitted(seasonal.model)), ylab='Ozone Layer Thickness (Dobson Units)',
     main = "Fitted Cyclic Model to the Ozone Thickness Series.",
     ylim = c(min(c(fitted(seasonal.model), as.vector(ozone.seasonalTS))),
     max(c(fitted(seasonal.model), as.vector(ozone.seasonalTS)))), col = "red")
lines(as.vector(ozone.seasonalTS),type="o")
```



From just a glance at the cyclic trend fitted plot, we are disappointed as the fit looks really out of place and is almost the same as the cosine trend model. The fitted model does not fit the time series data and so there might be seasonal autocorrelation in the data but our seasonal model could also not capture it. Again, this may be because deterministic trend models cannot capture stochastic trends caused by autocorrelation. Anyway, let's move on and gather more information about the efficiency of the model from the residual analysis and diagnostic checking.

4.1.18 Residual Analysis/Diagnostic Checking:

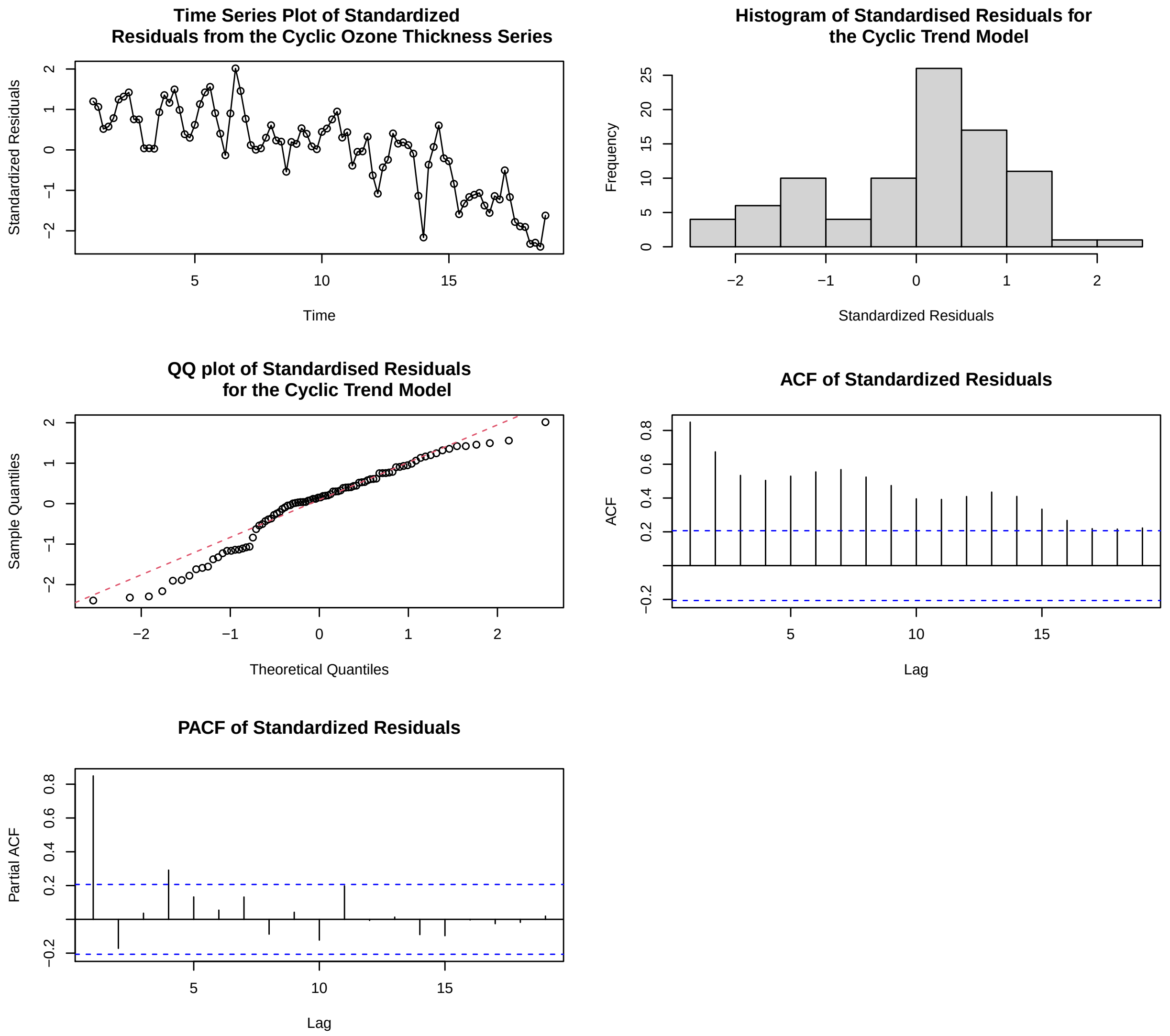
The following plots along with the Shapiro-Wilk Test are done again for this cyclic trend model in order to analyze the effectiveness of the model and perform the diagnostic checking. A detailed summary of the residual analysis is given below after the plots. To begin with, we first use the **rstudent()** function to get the standardized residuals and store it in the variable **res.seasonal** to make it easier for us to reuse the variable for different plots.

```
par(mfrow=c(3,2))

res.seasonal <- rstudent(seasonal.model)
```

```r
# Time Series Plot of Standardized Residuals
plot(y=res.seasonal,x=as.vector(time(ozone.seasonalTS)), xlab='Time',
     ylab='Standardized Residuals',type='o', main = "Time Series Plot of Standardized
     Residuals from the Cyclic Ozone Thickness Series")

# Histogram
hist(res.seasonal,xlab='Standardized Residuals',
     main = "Histogram of Standardised Residuals for
     the Cyclic Trend Model")

# QQ plot
qqnorm(res.seasonal, main = "QQ plot of Standardised Residuals
       for the Cyclic Trend Model")
qqline(res.seasonal, col = 2, lwd = 1, lty = 2)

# ACF
acf(res.seasonal, main = "ACF of Standardized Residuals")

# PACF
pacf(res.seasonal, main = "PACF of Standardized Residuals")

# Shapiro Wilk test
shapiro.test(res.seasonal)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.seasonal
## W = 0.95718, p-value = 0.004768
```

```r
par(mfrow=c(1,1))
```



4.1.19 Summary of the Residual Analysis:

- The obtained time series plot of the standardized residuals is almost the exact copy of our original seasonal time series meaning that our cyclic model as well has captured almost nothing!

- The obtained histogram looks like a slight left-skewed distribution and not very symmetric. We cannot infer much from this distribution. We move on to the next check.

- In comparison with the linear and quadratic models, we see from the QQ plot that almost 50% of the time points are straying away from the reference line which is a further indicator of how inefficient the model must be in practice.

- Both the ACF and PACF plots have a lot of lags with significant autocorrelation and thus this denotes that the residuals still contain a lot of information which the model has failed to capture. Further, the ACF plot shows a wave-like pattern which indicates the presence of some seasonality in the time series which has not been captured by the model.

- As a final step, we decide the normality of the standardized residuals using the **Shapiro-Wilk test**. This test calculates the correlation between the residuals and the corresponding normal quantiles. This test confirms that the distribution of the standardized residuals is surely not normal as we can see that the **p-value**(0.00476) < 0.05.

- Just like the cosine trend model, this cyclic model has also failed to capture the seasonality and is in fact, a bad fit in the overall sense. So we will definitely not be using this model as well which can be again justified from the bad results obtained from the residual analysis/diagnostic checking.

4.2 Predicting:

From the four deterministic trend models that have been fitted, after careful analysis and diagnostic checking, we finally decide to choose the **quadratic deterministic trend model** as the most effective among the four since it has better R-squared values and its diagnostic checking procedures proved more appropriate in comparison with the others. We will now use the said model to give the predictions of yearly changes for the **next 5 years**.

To begin our prediction, we first create a variable **yearsAhead** which stores the years in the future for which we will be predicting the changes in the ozone layer thickness as a dataframe under the variables **t** and **t^2** which we have already specified during the fitting of the quadratic model where **t** is **time(ozone.thicknessTS)**, where ozone.thicknessTS is our original time series. We also specify a variable **h** which stores the number of years for which we will be predicting.

```
h <- 5
yearsAhead <- data.frame(t = seq(2017, 2017+h-1, 1),
                         t2 = seq(2017, 2017+h-1, 1)^2)
```

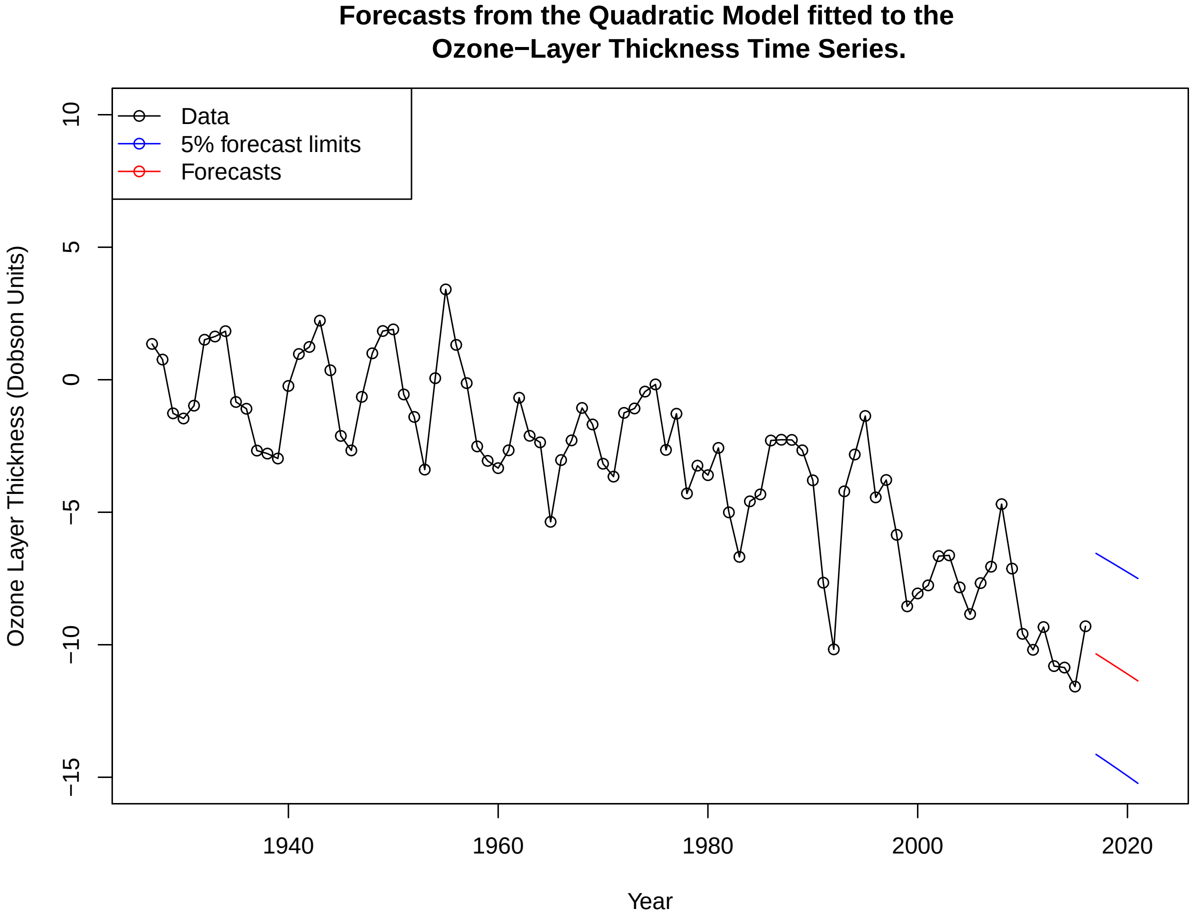
Next, we create the main prediction object using the **predict()** function and store it into the **quad.pred** variable as we will be using the quadratic trend model for predicting. We use the already created **yearsAhead** variable as the new data and also the interval parameter is set to **prediction**.

```
quad.pred <- predict(quad.model, newdata = yearsAhead, interval = "prediction")
quad.pred
```

```
##         fit       lwr       upr
## 1 -10.34387 -14.13556 -6.552180
## 2 -10.59469 -14.40282 -6.786548
## 3 -10.84856 -14.67434 -7.022786
## 4 -11.10550 -14.95015 -7.260851
## 5 -11.36550 -15.23030 -7.500701
```

Now that we’ve created the prediction object, we plot these predicted values next to the original times series in question by executing the following codes:

```
plot(ozone.thicknessTS, xlim= c(1927,2017+h), ylim = c(-15,10),
     ylab = "Ozone Layer Thickness (Dobson Units)",
     main = "Forecasts from the Quadratic Model fitted to the
     Ozone-Layer Thickness Time Series.", xlab = ’Year’, type=’o’)
lines(ts(as.vector(quad.pred[,3]), start = 2017), col="blue", type="l")
lines(ts(as.vector(quad.pred[,1]), start = 2017), col="red", type="l")
lines(ts(as.vector(quad.pred[,2]), start = 2017), col="blue", type="l")
legend("topleft", lty=1, pch=1, col=c("black","blue","red"),
       text.width = 21, c("Data","5% forecast limits", "Forecasts"))
```



From the plot we can see that our predictions look quite good and seem to follow the trend from the original time series. The red line represents the fitted values for the five years into the future, while the blue lines indicate the upper and lower confidence intervals for the fitted values. Even though the quadratic model had a few shortcomings and there was some information left over in the residuals along with significant auto-correlation for various lags in the ACF and PACF plots and seasonality which was not captured, this model seems to be the most appropriate from among the four deterministic trend models for fitting and predicting yearly change values from the time series.

5 Chapter 2

We don't just stop with the model fitting and the prediction. We would also like to specify/propose a set of possible and effective **ARIMA**(Auto-Regressive Integrated Moving Average) models using suitable model specification tools for our ozone layer thickness time series. Since the ARIMA models are specified with corresponding orders (p,d,q), our task here is to determine the possible orders and build up a set of appropriate ARIMA models using the orders. Also, our aim here is to just specify the models and their parameters and not fit them.

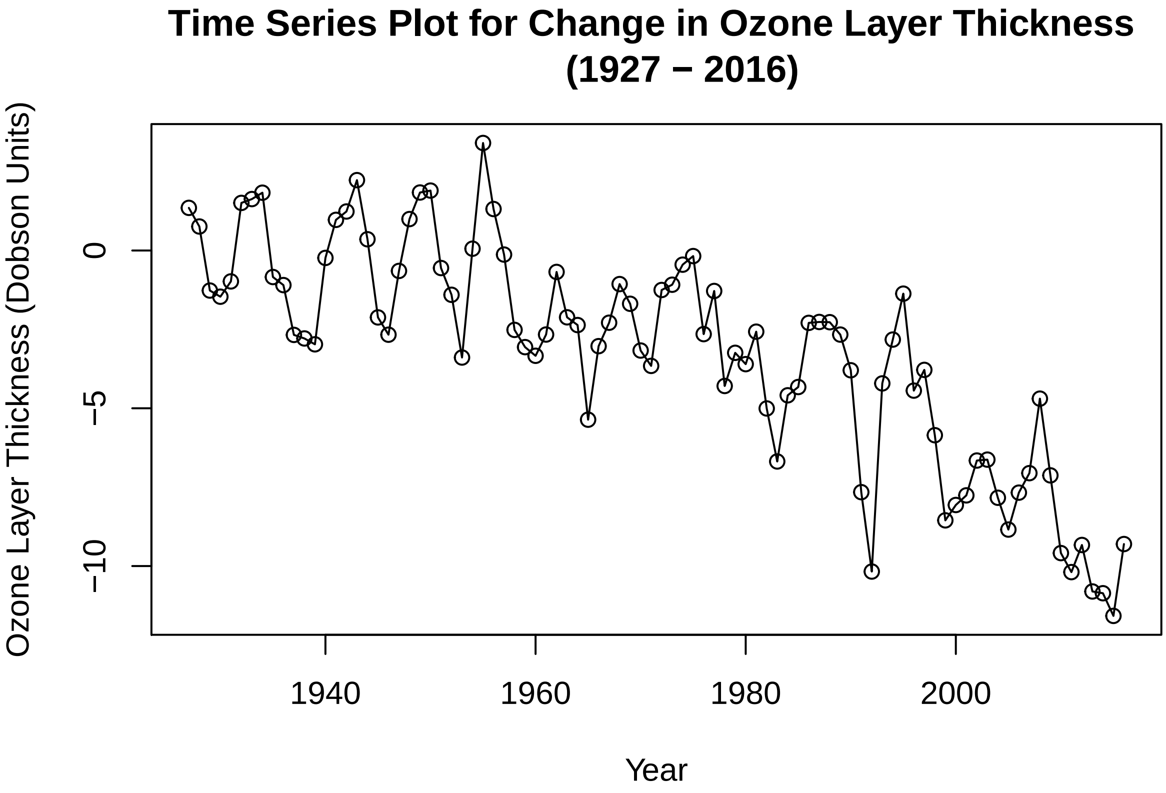
5.1 Preparation

Here, we summarize the characteristics of the time series, run tests on the series and perform techniques like transformation and differencing to prepare the time series for specifying the possible model parameters.

5.1.1 Five Valid Points:

The ozone layer thickness time series plot:

```
plot(ozone.thicknessTS, xlab='Year', ylab='Ozone Layer Thickness (Dobson Units)',
     main='Time Series Plot for Change in Ozone Layer Thickness
     (1927 - 2016)', type='o')
```

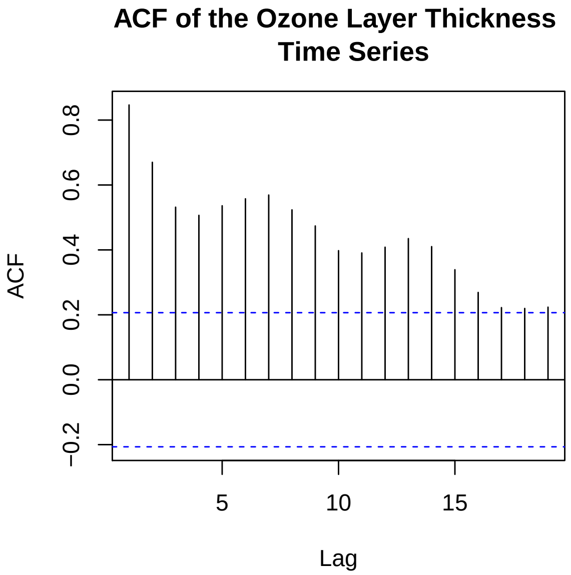


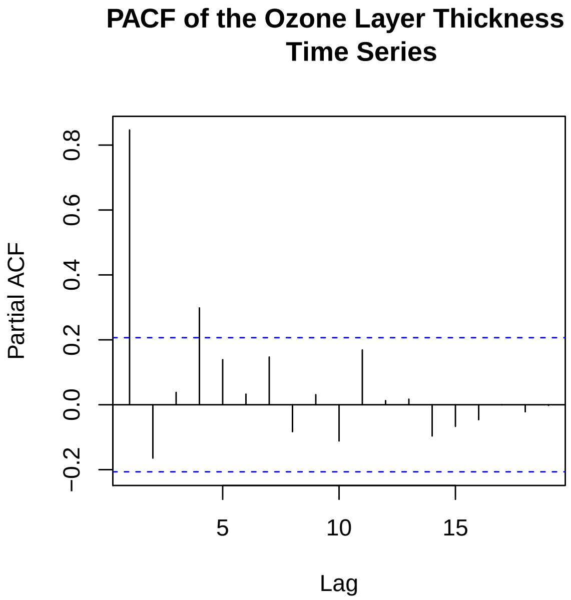
We have already summarized on the five valid points in the **Analyzing the Time Series** chapter which you can have a relook at by clicking **here.** We ultimately concluded that the series is indeed non-stationary with an obvious trend and exhibiting slight seasonality and changing variance. The behaviour can be categorized as dominant MA with AR characteristics in some parts with no obvious change point.

5.1.2 ACF and PACF plots:

We take a look at the ACF and PACF plots of our time series again here.

```
par(mfrow=c(1,2))
acf(ozone.thicknessTS, main ="ACF of the Ozone Layer Thickness
    Time Series")
pacf(ozone.thicknessTS, main ="PACF of the Ozone Layer Thickness
      Time Series")
```





```
par(mfrow=c(1,1))
```

From the ACF plot we can see a gradually decreasing wave-like pattern which confirms our suspicions on trend in the time series and consequently non-stationarity. Thus, surely we cannot determine any order parameters from these plots. However, alternatively, from the PACF plot, we can see that there are two lags - one with significant autocorrelation(lag 1) and another with slight autocorrelation(lag 4) and so we can keep the model **AR(2)** in mind for now.

5.1.3 Unit Root Tests:

To further solidify our deduction of non-stationarity, we apply the **Augmented Dickey-Fuller** test, where the null hypothesis is that the series in non-stationary. We use the default value of the parameter **k** which is 4.

```
adf.test(ozone.thicknessTS, alternative = c("stationary"))
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ozone.thicknessTS
## Dickey-Fuller = -3.2376, Lag order = 4, p-value = 0.0867
## alternative hypothesis: stationary
```

From the adf.test() function, we get a p-value of 0.0867 which is greater than 0.05 sure, but is somewhat of a borderline value. Let us try the same test again since we see a clear downward trend in the time series plot but this time with the parameter **k** set to a larger lag value.

```
adf.test(ozone.thicknessTS, k = 5, alternative = c("stationary"))
```

```
##
##  Augmented Dickey-Fuller Test
```

```
##
## data:  ozone.thicknessTS
## Dickey-Fuller = -2.5704, Lag order = 5, p-value = 0.3413
## alternative hypothesis: stationary
```

Now that we've set a larger lag value **k = 5**, we see that we get a good and strong indication of non-stationarity in the series from the p-value 0.3413 which is significantly greater than 0.05 and definitely not a borderline value. Thus we fail to reject the null hypothesis and confirm **non-stationarity**.

We also proceed to use the **Phillips-Perron**(pp.test) to confirm non-stationarity.

```
pp.test(ozone.thicknessTS, alternative = c("stationary"))
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  ozone.thicknessTS
## Dickey-Fuller Z(alpha) = -35.431, Truncation lag parameter = 3, p-value
## = 0.01
## alternative hypothesis: stationary
```

We are disappointed with this result from the pp.test() since we get a p-value denoting stationarity in the series but we are sure with our decision of non-stationarity which is also backed up by various factors incuding obvious trend in the plot, decaying wave pattern in the ACF plot and also the p-values from the **adf.test()**. Thus, we totally ignore the result from this test and do not take this into account.

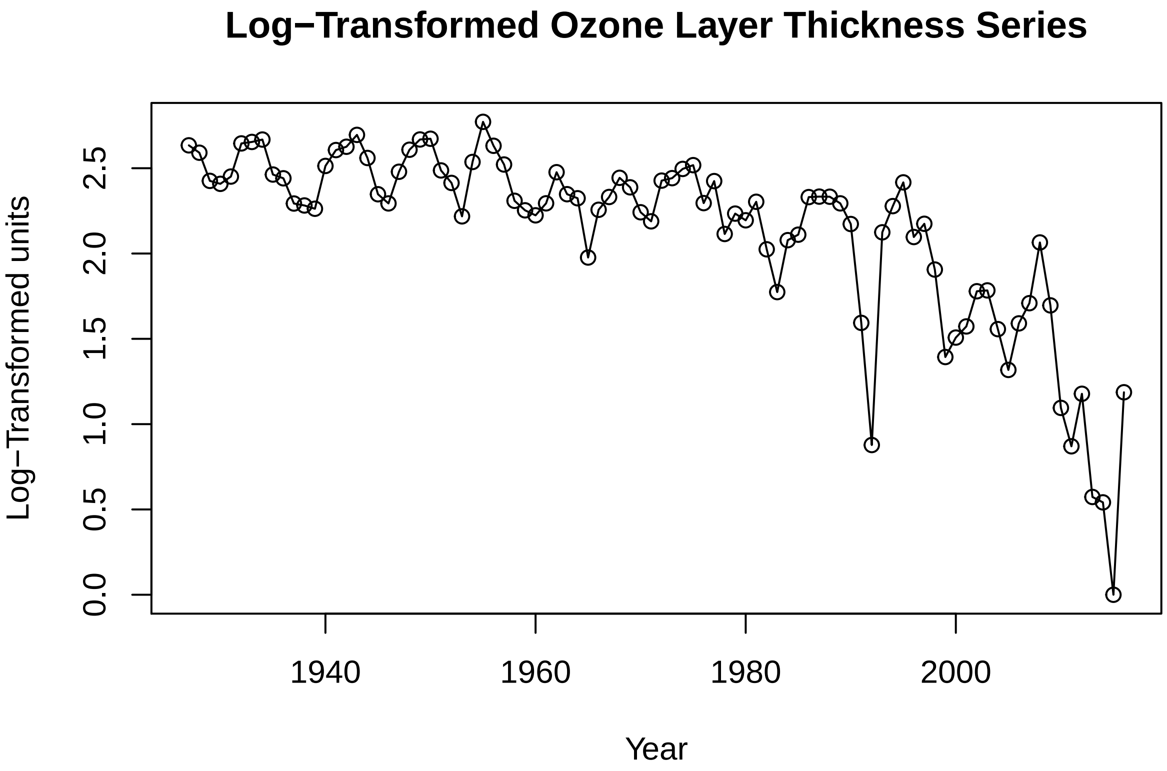
5.1.4 Transforming the Series:

Since we've decided that our series has changing variance in some parts, we need to transform the series using transformation techniques to stabilize the variance(making the series less variable) and improve the normality of the time series. We go ahead and perform two types of transformations:

5.1.5 Log Transformation:

The first transformation technique that we incorporate is the standard log transformation. Since our time series data includes negative values as well, we use the **log()** function along with the addition term and subtracting the minimum term as shown below. We store the log-transformed series into the variable **ozone.logTS**.

```
ozone.logTS <- log(ozone.thicknessTS + 1 - min(ozone.thicknessTS))
plot(ozone.logTS, type = 'o', main='Log-Transformed Ozone Layer Thickness Series',
     xlab='Year', ylab='Log-Transformed units')
```

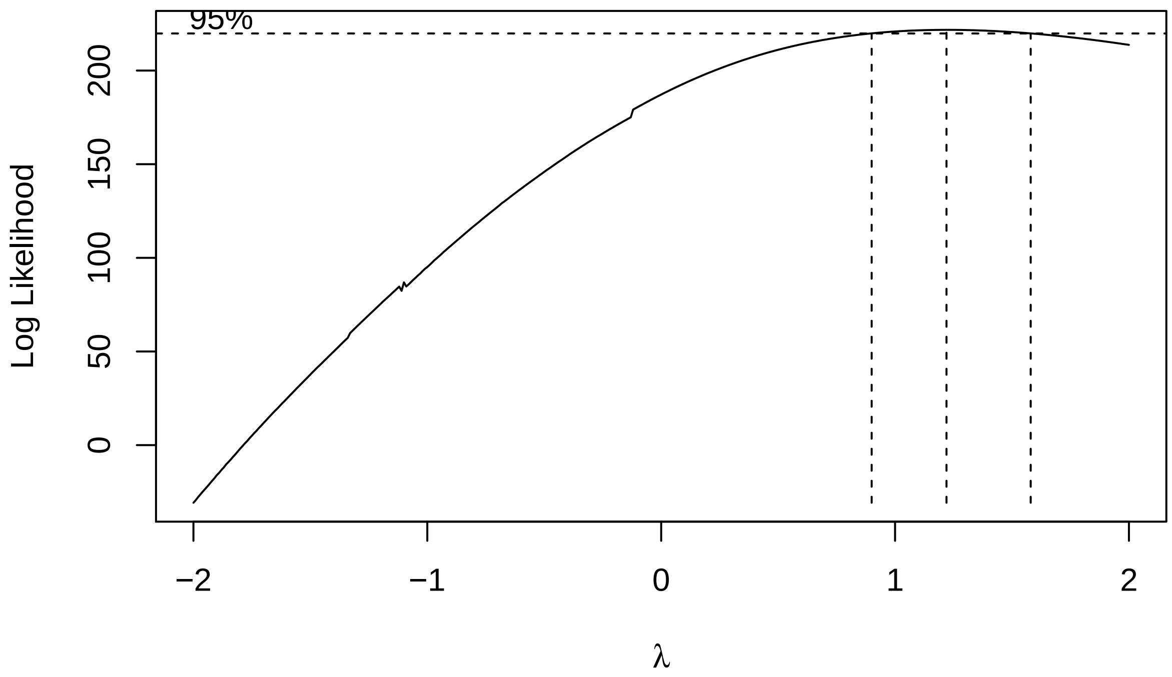


As seen from the plot of the log-transformed time series, we can conclude that this technique hasn't worked really well which is supported by the fact that there is still some parts with changing variance especially towards the end of the series. Thus we move on to the next transformation technique.

5.1.6 Box-Cox Transformation:

Now that our log-transformation hasn't worked out well, we try the Box-Cox transformation which uses the parameter **lambda** to transform the series. Again, since our series includes negative values as well, we use the **abs()** function to return the absolute value and work on that. We give the lambda search range to find the optimal value to be between **-2** and **2** by steps of **0.01** and check out the confidence interval **BC$ci**. We finally store the Box-Cox transformed series into the variable **ozone.BCTCS**.

```r
BC <- BoxCox.ar(ozone.thicknessTS + abs(min(ozone.thicknessTS)) + 1, lambda = seq(-2,2,0.01))
```



```
BC$ci
```

```
## [1] 0.90 1.58
```

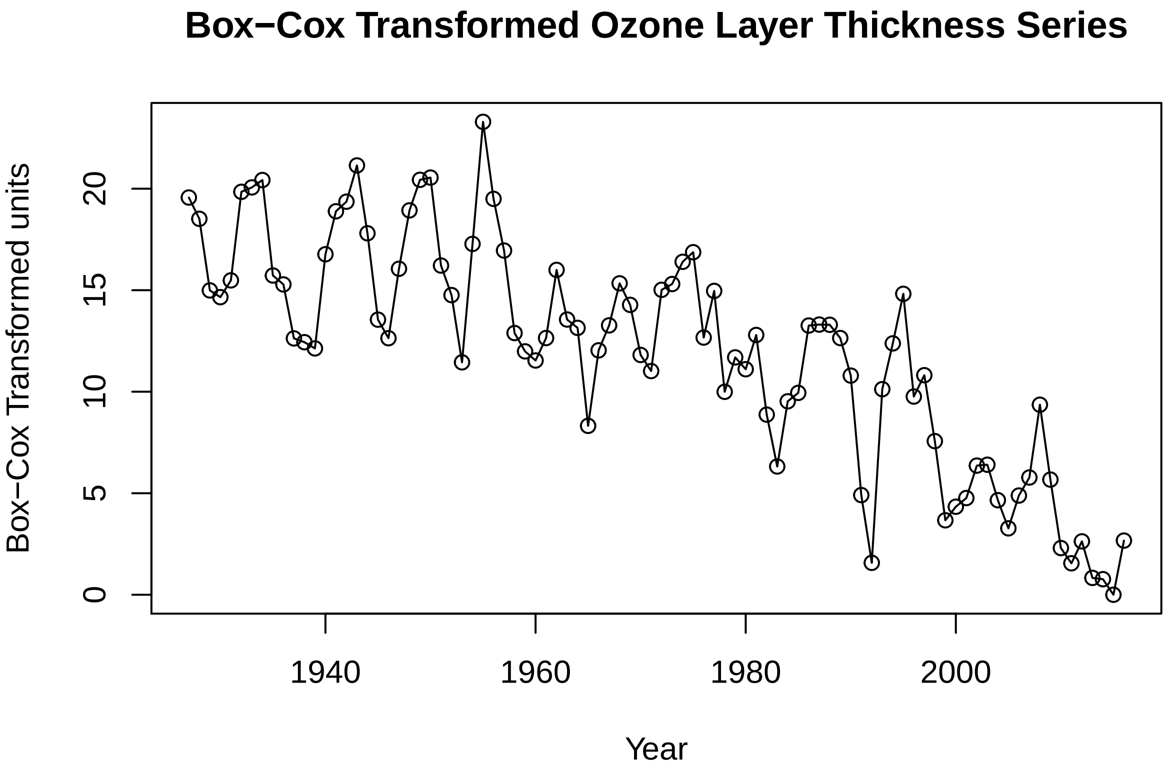
```
lambda <- BC$lambda[which(max(BC$loglike) == BC$loglike)]
lambda
```

```
## [1] 1.22
```

```
ozone.bcdata <- ozone.thicknessTS + abs(min(ozone.thicknessTS)) + 1
ozone.BCTS <- ((ozone.bcdata^lambda)-1)/lambda
```

From the results, we get the lambda value as 1.22 which is greater than and close to 1 meaning we do not expect a huge difference in the original series and the Box-Cox transformed series. Let us plot the transformed series and have a look at it ourselves.

```
plot(ozone.BCTS, type='o', main='Box-Cox Transformed Ozone Layer Thickness Series',
     xlab='Year', ylab='Box-Cox Transformed units')
```

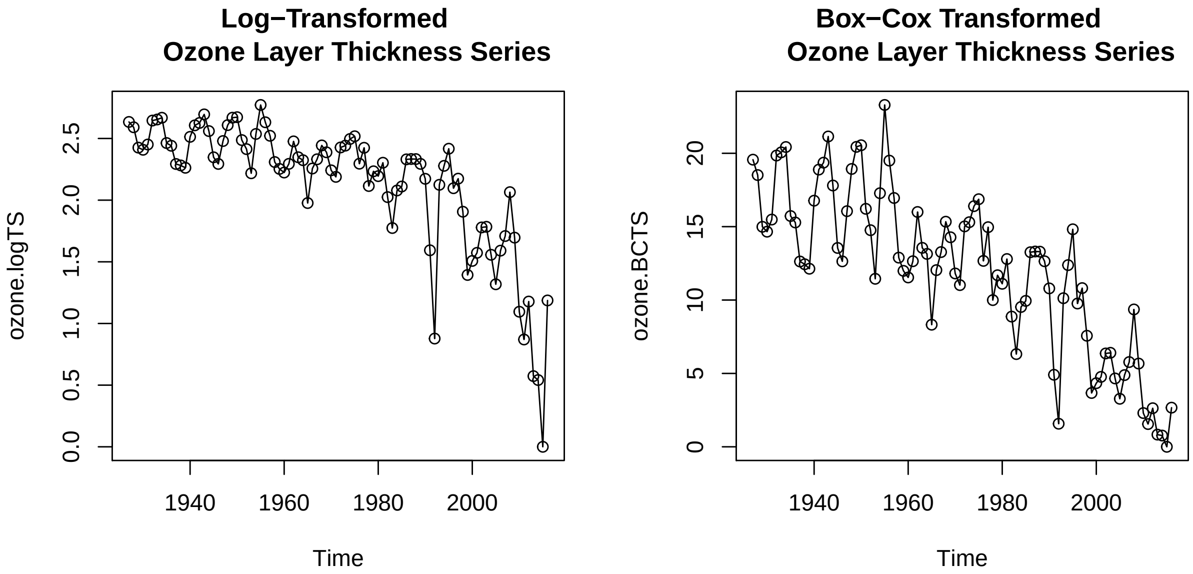


This plot still shows changing variance in some parts of the series and as expected, because of the lambda value being close to 1, we see that it looks almost the same the our original time series.

5.1.7 Comparison

Let us compare the log and Box-Cox transformed series and plotting them side by side:

```
par(mfrow=c(1,2))
plot(ozone.logTS, type='o', main='Log-Transformed
     Ozone Layer Thickness Series')
plot(ozone.BCTS, type='o', main='Box-Cox Transformed
     Ozone Layer Thickness Series')
```

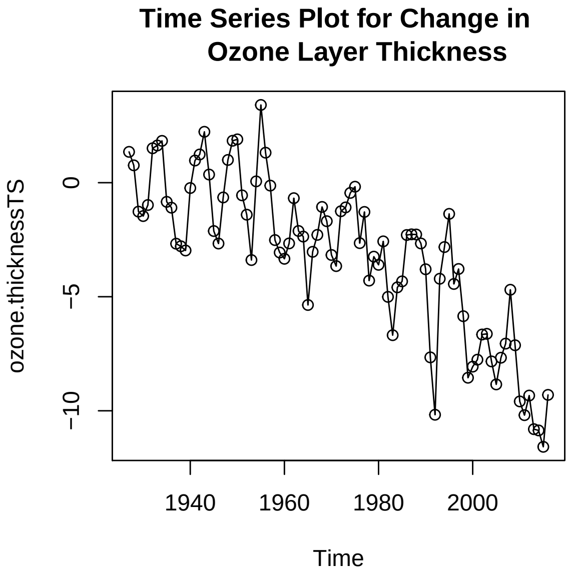


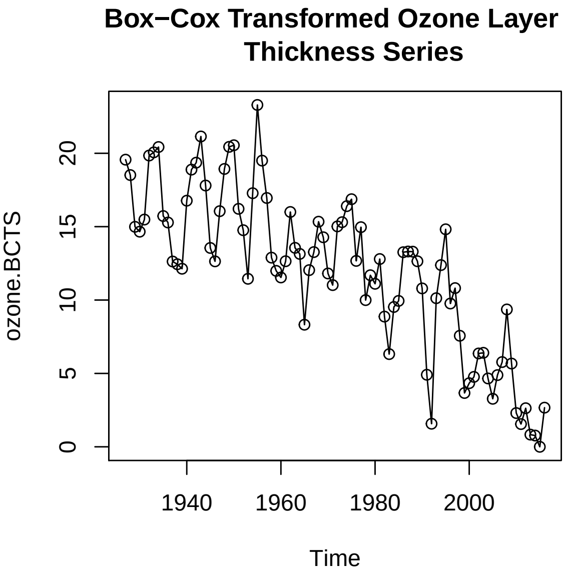
```
par(mfrow=c(1,1))
```

Out of the two, the Box-Cox transformed series definitely looks better in comparison. The log-transformed series seems to have huge changes in some parts, whereas the Box-Cox transformed series has relatively smaller changing variances.

Now that we've taken the Box-Cox transformation to be the better one among the two, let us compare it with our original time series by plotting them side by side.

```
par(mfrow=c(1,2))
plot(ozone.thicknessTS, type='o', main='Time Series Plot for Change in
     Ozone Layer Thickness')
plot(ozone.BCTS, type='o', main='Box-Cox Transformed Ozone Layer
     Thickness Series')
```





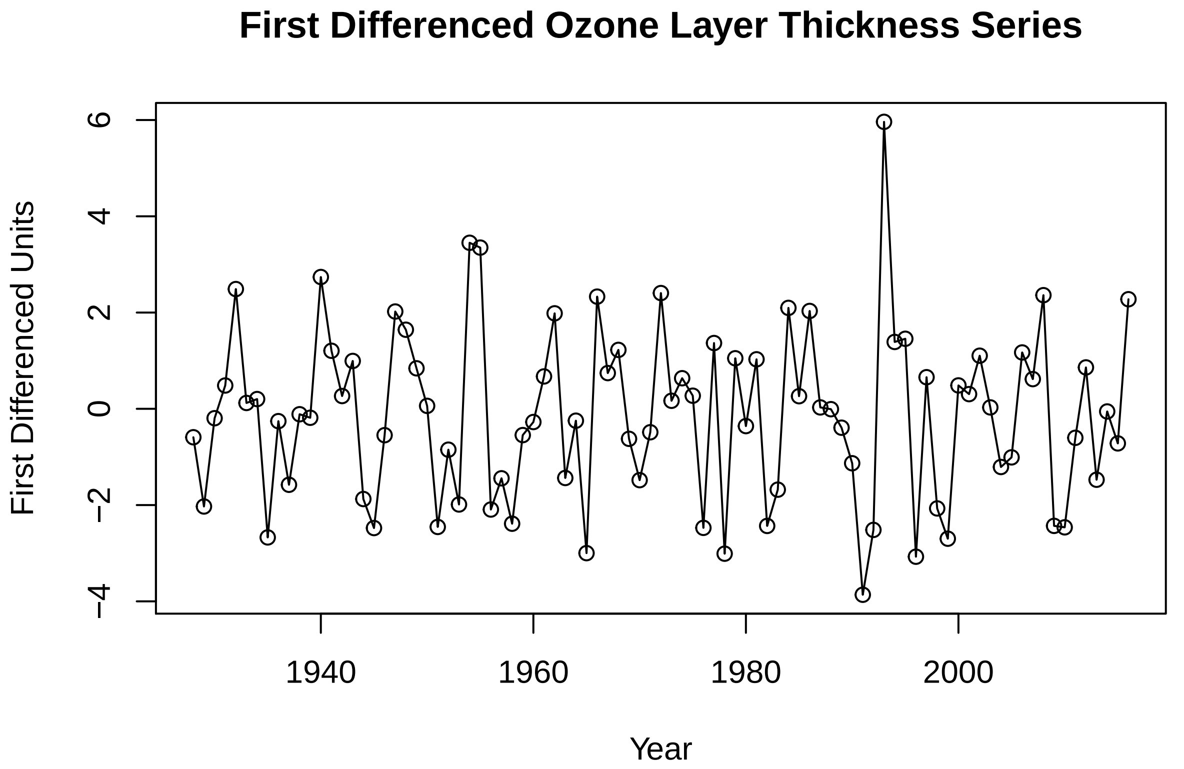
```
par(mfrow=c(1,1))
```

Voila! Just as expected, the Box-Cox transformation didn't make much of a difference and thus we choose to work with the raw data, which is our original time series.

5.1.8 Differencing:

We have already stated the non-stationarity of the time series, supported by observed results from the ACF plot and other unit-root tests. Now, we need to convert the non-stationary series into a stationary one. For this purpose we will be using the **diff()** function which will be differencing the series and try to make it stationary. The order of differencing accounts for the parameter **d** in the ARIMA(p,**d**,q) models. Differencing is done to a series until it becomes stationary. Thus, we start with the parameter **differences = 1**, and check if it makes the series stationary.

```
ozone.thicknessTSDiff <- diff(ozone.thicknessTS, differences = 1)
plot(ozone.thicknessTSDiff, type='o', xlab='Year', ylab='First Differenced Units',
     main='First Differenced Ozone Layer Thickness Series')
```



Now that we have our first differenced series plot, we go ahead and summarize the **five valid points**:

- **Trend** : From an overall perspective, we see that the trend has been removed by the differencing technique and this denotes stationarity for the most part.

- **Seasonality** : This is a difficult characteristic to assume as mainly there seems to be no seasonality but for some parts in the plot, there seems to be some kind of pattern. So we don't rule out seasonality completely.

- **Behvaiour** : Since we have assumed that there might be seasonality, it might be difficult to comment on the behaviour as well since seasonality might screen out the behaviourial characteristics. However, just by looking at the plot, we can see a dominant MA behaviour with slight traces of AR behaviour.

- **Changing Variance** : Our transformation techniques haven't worked quite well and thus we still see some parts of the series exhibiting changing variance.

- **Intervention** : From the plot, we can infer that inspite of having high and low points, there is no significant intervention/change point.

Apart from the valid points, mainly, we see that there is no longer a trend in the series which is a good sign regarding the stationarity. Alternatively, since we have removed the trend by differencing the series with order $\mathbf{d = 1}$, we do not go for higher-order differencing.

5.1.9 Unit Root Test:

To solidify our decision of stationarity even more, we use the **adf.test()** function again on the differenced time series to check for stationarity:

```
adf.test(ozone.thicknessTSDiff, alternative = c("stationary"))
```

```
## Warning in adf.test(ozone.thicknessTSDiff, alternative = c("stationary")): p-
## value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  ozone.thicknessTSDiff
## Dickey-Fuller = -7.1568, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

We get a message saying that the 'p-value is smaller than the printed p-value' where the printed p-value is 0.01 which is indeed lesser than 0.05 and so we reject the null hypothesis and thus take the alternative hypothesis which states that our differenced time series is, infact, a stationary series!

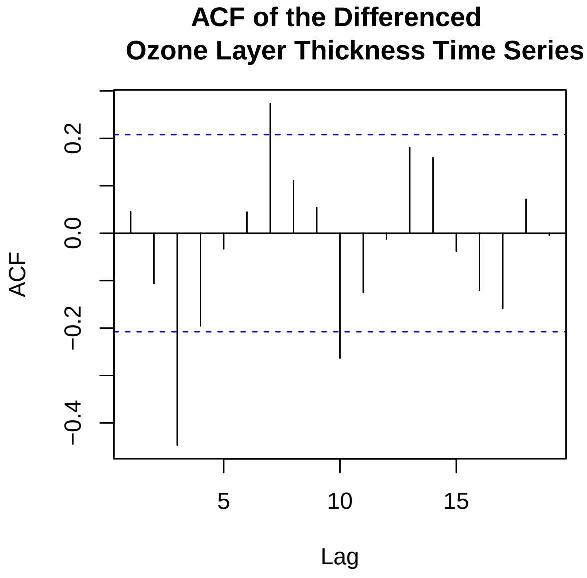
5.2 Model Specification:

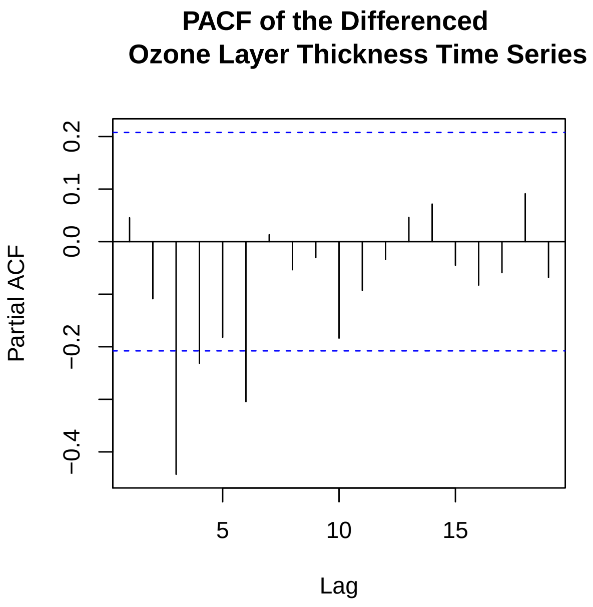
Now, that we've got a stationary series, we can now finally start with the model specfication, where we specify a set of possible/appropriate **ARIMA(p,d,q)** models for our differenced ozone layer thickness time series.

5.2.1 ACF and PACF plots:

As the first step in determining the possible models, we plot the ACF and PACF plots from the differenced time series and try to visually estimate the parameters **p** - from the PACF and **q** from the ACF plots.

```
par(mfrow=c(1,2))
acf(ozone.thicknessTSDiff, main ="ACF of the Differenced
    Ozone Layer Thickness Time Series")
pacf(ozone.thicknessTSDiff, main ="PACF of the Differenced
     Ozone Layer Thickness Time Series")
```





```r
par(mfrow=c(1,1))
```

First, we begin by analyzing the ACF plot of the series. The differencing seems to have worked quite well, we now have a stable ACF plot. From the ACF plot, we consider the following orders of **q** for the ARIMA(p,d,q) model:

- **1**, since we see a lag with a very significant autocorrelation, which is the lag 3.
- **2**, since we **also** consider the lag 4 which is on the borderline of the confidence interval and its autocorrelation is almost significant and thus it might be a good addition in our set of models.
- **0**, since we still observe somewhat of an up-down, cosine wave-like pattern which might be due to the seasonality present in the time series.

The ACF plot also shows lags 7 and 10 with significant autocorrelation but we do not consider them or include them in our set of possible **q** orders as they are quite far from the origin and are relatively bigger lags. Now, we move on to determining the possible orders of **p** for the ARIMA(p,d,q) model from the PACF plot. We consider the following orders for **p**:

- **3**, since there are clearly three lags (lags 3,4,6) with significant autocorrelation.
- **2**, since we also consider the lag 6 to be somewhat far from the origin and a higher lag and so we remove it from consideration.
- **1**, since the lag 4 has only slightly significant autocorrelation and is almost on the borderline of the confidence interval. So we also remove the lag 4 from consideration.

Thus, from the ACF and PACF plots, we have specified the following ARIMA(p,d,q) models where **d=1** since we used first differencing on the time series - **ARIMA(1,1,0)** | **ARIMA(1,1,1)** | **ARIMA(1,1,2)** | **ARIMA(2,1,0)** | **ARIMA(2,1,1)** | **ARIMA(2,1,2)** | **ARIMA(3,1,0)** | **ARIMA(3,1,1)** | **ARIMA(3,1,2)**.

5.2.2 EACF Table:

The next model specification tool that we'll use is the Extended Autocorrelation Function(EACF) table. Let's use the **eacf()** function on the differenced time series and examine our results.

```r
eacf(ozone.thicknessTSDiff)
```

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

From the Extended Autocorrelation Function(EACF) table, after choosing the 'top-left **o**', we can specify **three models** which include the immediate neighbors from the EACF table. Again, the order of **d** here is 1 since we are working on the first order differenced time series. The models we can specify from the EACF table are - **ARIMA(0,1,0)** | **ARIMA(0,1,1)** | **ARIMA(1,1,1)**. Here, we note that the specified model **ARIMA(1,1,1)** has also been chosen as a suitable specification from the ACF and PACF plots which is a good sign as it denotes consistency.

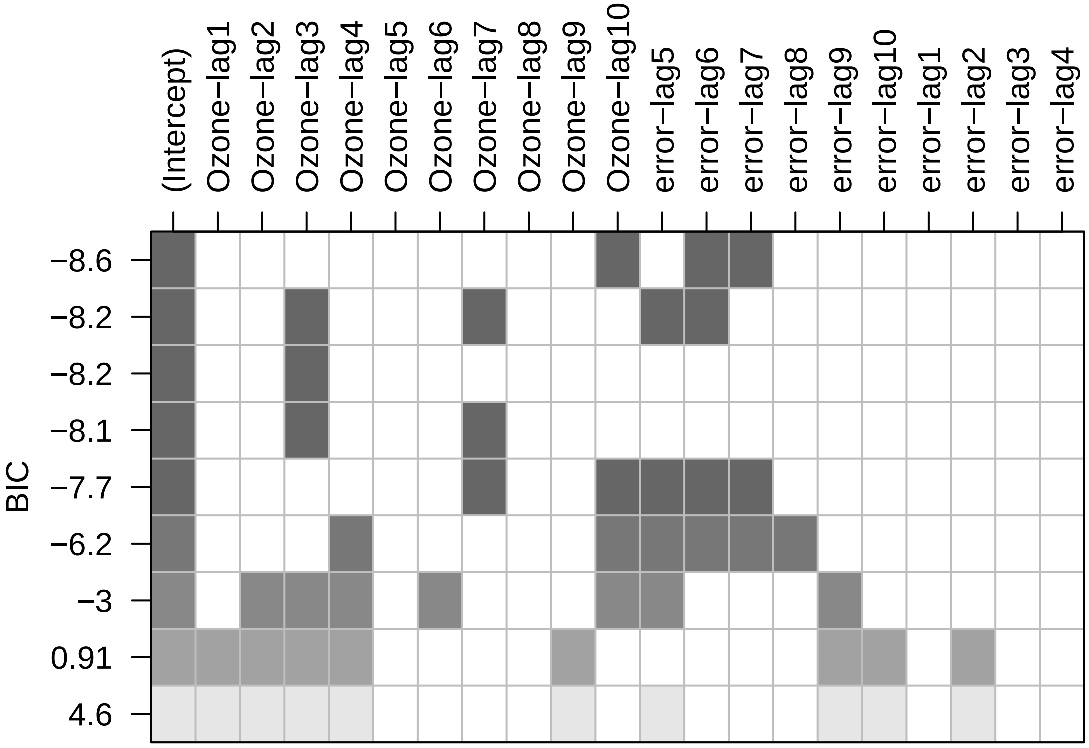
5.2.3 BIC Table:

We also use another tool called the Bayesian Information Criterion(BIC) table to help us in proposing a set of suitable ARIMA models. Using the **armasubsets()** function, when we give the **nar** and **nma** parameters as a high value(10), we get a BIC table with vague and non-supported or lightly supported higher orders(p = 10, q=6,7 etc) which seem very inconsistent with the differenced time series and also the other tools which we have used for helping us in specifying the ARIMA model orders such as the EACF table and the ACF and PACF plots.

```
ozoneBIC1 <- armasubsets(y=ozone.thicknessTSDiff, nar=10, nma=10,
                         y.name='Ozone', ar.method='ols')
```

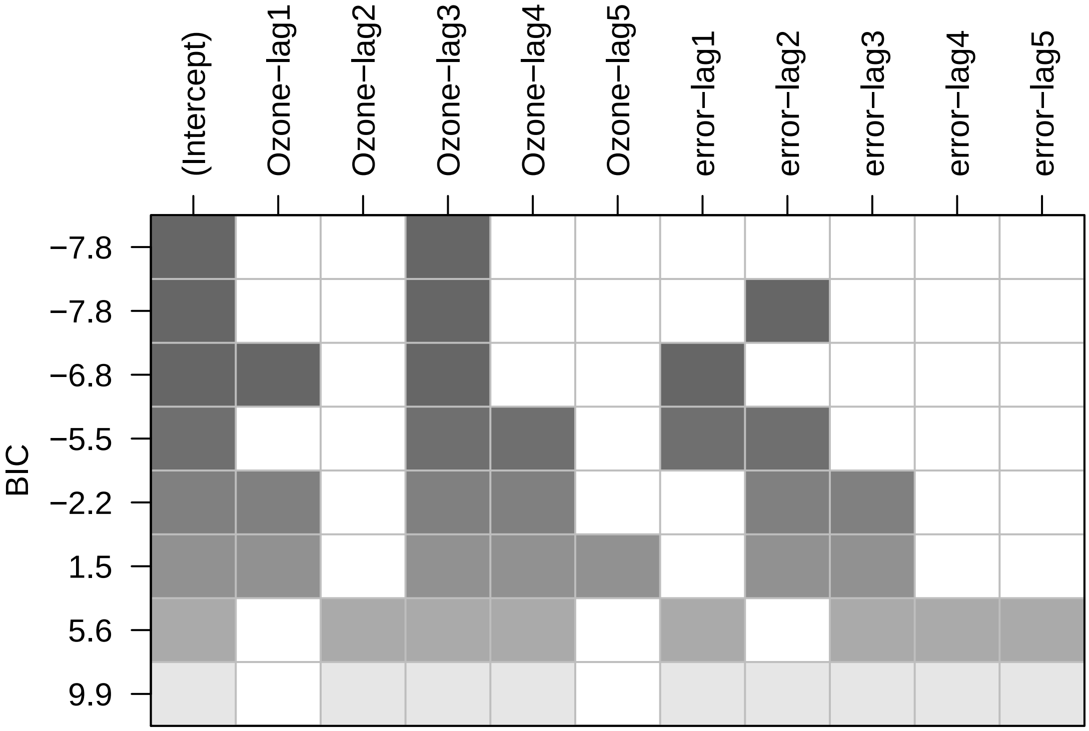
```
## Reordering variables and trying again:
```

```
plot(ozoneBIC1)
```



Because of the above issue, we now reduce the values of the **nar** and **nma** parameters to 5 and plot the BIC table to see that we get better results:

```
ozoneBIC2 <- armasubsets(y=ozone.thicknessTSDiff, nar=5, nma=5,
                         y.name='Ozone', ar.method='ols')
plot(ozoneBIC2)
```



Now, from the new BIC table, we include the orders of **p = 3** which is very strongly supported and **q = 2** which is also supported to a certain degree. We do not choose any other orders for p and q from the BIC table as the others seem very insignificant or are not strongly supported. Thus, from the BIC table we specify the following models - **ARIMA(3,1,2)**, where d = 1 as we are working on the first differenced time series. We again see that this model that we have specified with help from the BIC table has also been chosen from the ACF and PACF plots which again indicated that the specification tools that we have been using are consistent with each other.

5.3 Set of Possible Models:

From the ACF and PACF plots, the EACF table and the BIC table, we have finally specified a possible/appropriate set of ARIMA(p,d,q) models for the **first order** differenced ozone layer thickness time series and they are listed in the table shown below in the next page:

Number	Model Specification	Specification Tool
1	ARIMA(1,1,0)	ACF & PACF
2	ARIMA(1,1,1)	ACF & PACF and EACF
3	ARIMA(1,1,2)	ACF & PACF
4	ARIMA(2,1,0)	ACF & PACF
5	ARIMA(2,1,1)	ACF & PACF
6	ARIMA(2,1,2)	ACF & PACF
7	ARIMA(3,1,0)	ACF & PACF
8	ARIMA(3,1,1)	ACF & PACF
9	ARIMA(3,1,2)	ACF & PACF and BIC table
10	ARIMA(0,1,0)	EACF
11	ARIMA(0,1,1)	EACF

From the above table, we see that we have included **11** ARIMA(p,d,q) models in our set of possible and appropriate models. We also mention the specification tool which has helped us in inferring the corresponding model. We see that for some models there is consistency between different specification tools which seems like a strong indication of appropriate chosen models.

6 Conclusion

- To conclude, after visually summarizing on the characteristics of the ozone layer thickness time series and also calculating the correlation between the observations and its lags, we fitted the four main deterministic trend models namely, the linear, quadratic, cosine/harmonic and the seasonal/cyclic models. We completely ruled out the seasonal/cyclic model and the cosine/harmonic model since they proved to be largely insignificant models with bad results from the resiudal analysis and diagnostic checking. On comparing the linear and the quadratic model, inspite of both models having large limitation and shortcomings, we proceeded to choose the quadratic trend model as it displayed relatively better significance results and residual analysis.

- Since the quadratic trend model proved to be the most effective among the four, we used the same model to give the predictions of yearly changes for the next 5 years using the predict() function and we were satisfied with the results. However, the main drawback of our best model - the quadratic trend model was that it could not capture the seasonality from our time series which was reflected by the decaying pattern in the ACF plot.

- Since all our four deterministic trend models were very ineffective for fitting, we conclude that our time series has a **stochastic trend** and not a deterministic one as we had initially assumed. This statement is backed up by the fact that deterministic trend models cannot capture the stochastic trend caused by autocorrelation.

- While proposing a set of possible ARIMA(p,d,q) models for our first order differenced time series, we used a few model specification tools which helped us in arriving at a final set of 11 appropriate ARIMA models while keeping the **principle of parsimony** in mind:

Number	**Model Specification**	**Specification Tool**
1	ARIMA(1,1,0)	ACF & PACF
2	ARIMA(1,1,1)	ACF & PACF and EACF
3	ARIMA(1,1,2)	ACF & PACF
4	ARIMA(2,1,0)	ACF & PACF
5	ARIMA(2,1,1)	ACF & PACF
6	ARIMA(2,1,2)	ACF & PACF
7	ARIMA(3,1,0)	ACF & PACF
8	ARIMA(3,1,1)	ACF & PACF
9	ARIMA(3,1,2)	ACF & PACF and BIC table
10	ARIMA(0,1,0)	EACF
11	ARIMA(0,1,1)	EACF

- With careful deductions from the various model specification tools, the above parsimonious models from the table were specified for our ozone layer thickness time series. The highest number of parameters for a particular model among the specified ones was **5**(excluding the differencing parameter d = 1), which means that our set of models will be efficient and easy to analyze, work on and fit, if needed.

7 References

NASA. (2018, Oct 18). *NASA Ozone Watch*. Ozonewatch. https://ozonewatch.gsfc.nasa.gov/facts/dobson_SH.html

R Markdown. (n.d.). *Cheatsheets*. RStudio. https://rmarkdown.rstudio.com/lesson-15.HTML

Hyndman, Rob. (2011, Dec 14). *Cyclic and Seasonal Time Series*. Robjhyndman. https://robjhyndman.com/hyndsight/cyclicts/

Salvi, Jayesh. (2019, Mar 27). *Significance of ACF and PACF Plots*. Towards Data Science. https://towardsdatascience.com/significance-of-acf-and-pacf-plots-in-time-series-analysis-2fa11a5d10a8

Powell, Steve. (2011, Sep 7). *Cross-reference in Markdown*. Stackoverflow. https://stackoverflow.com/questions/5319754/cross-reference-named-anchor-in-markdown/

NASA. (n.d.). *The Ozone Layer*. UCAR.edu. https://scied.ucar.edu/learning-zone/atmosphere/ozone-layer

Demirhan, Haydar. (n.d.). *Modules 1 - 5*. RMIT University. Lecture Videos