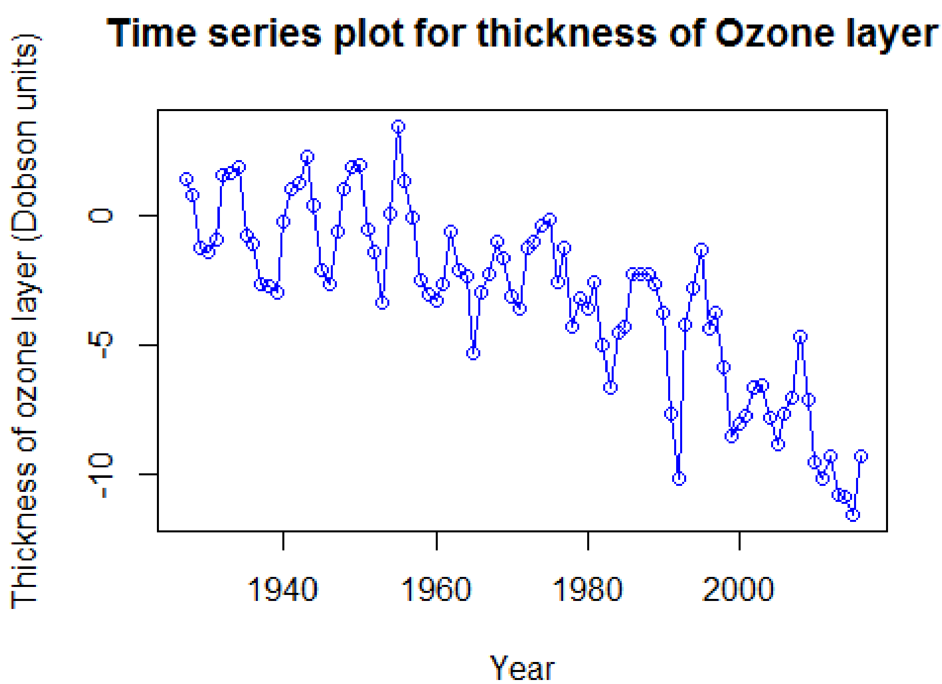
Ozone Depletion

Introduction

Ozone plays a different role in atmospheric chemistry at different heights in the Earth's atmosphere. Concentrations of ozone are higher in the stratosphere which plays a crucial role in absorbing potential dangerous UV radiation from the sun. Whereas, ozone concentrations are relatively low in the lower atmosphere i.e troposphere where it forms as an air pollutant and can have a negative impact over human health. The investigation involves analyzing the ozone layer thickness through yearly changes between year 1927 and 2016 in Dobson units. Post analysis, the goal is to forecast the yearly changes for the next 5 years. The dataset contains one variable where a negative value represents a decrease in the thickness and a positive value represents an increase in the thickness.

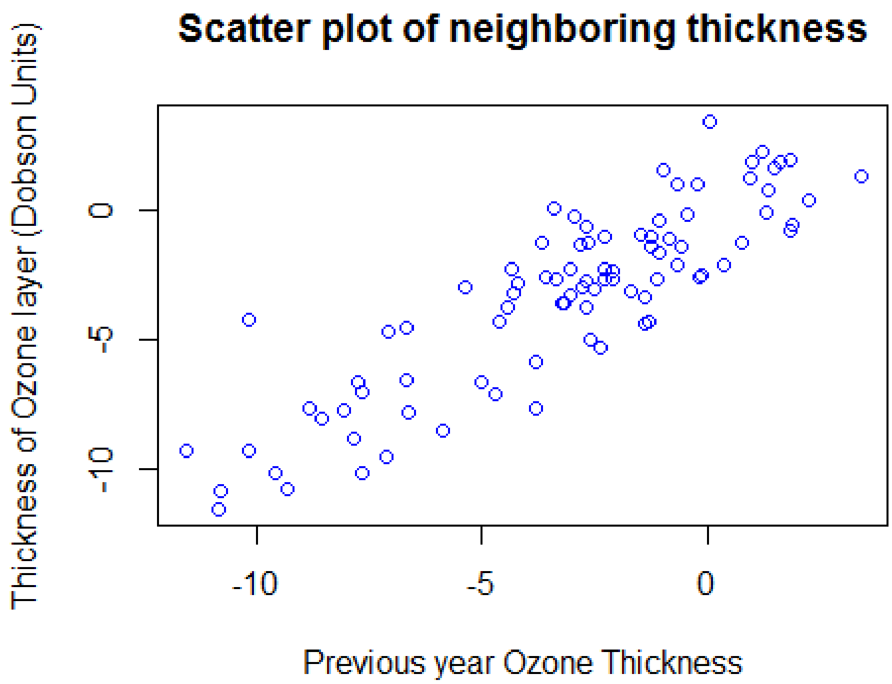
```
rm(list=ls())

# Loading the Libraries
library(TSA)

##
## Attaching package: 'TSA'

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

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

library(readr)

##
## Attaching package: 'readr'

## The following object is masked from 'package:TSA':
##
##     spec

library(tseries)

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

# Reading the dataset contents
ozone <-
read.csv("C:/Users/winuser/Downloads/Projects/Time_Series/data1.csv", header
```

```
= FALSE)

# Setting time series object to data frame and checking the class
ozone <- ts(as.vector(ozone), start = 1927, end = 2016, frequency = 1)
class(ozone)

## [1] "ts"

# Plotting time series data
plot(ozone, type = 'o', ylab = 'Thickness of ozone layer (Dobson units)',
xlab = 'Year', main = 'Time series plot for thickness of Ozone layer', col =
'blue')
```



Trend: A downward trend can be observed in the mentioned plot. Seasonality: The seasonality is not present. Change in Variance: There seems to be no change in variance. Intervention: Intervention is no observed in the series. Behavior: The series looks auto regressive with many succeeding points and depicts Moving Average Behavior.

```
# Plotting scatter plot with respect to previous year
plot(y = ozone, x = zlag(ozone), ylab = "Thickness of Ozone layer (Dobson
Units)", xlab="Previous year Ozone Thickness", main = "Scatter plot of
neighboring thickness", col="blue")
```




```
# Correlation
y = ozone # Read the data in to y
x = zlag(ozone) #Generate first lag of the series
index = 2:length(x) # ignore the NA
cor(y[index],x[index]) #Calculate the correlation

## [1] 0.8700381
```

The scatter plot shows a correlation between neighboring points and the correlation function indicate a strong positive correlation of thickness with the previous year.

Model techniques

Task 1

Linear Model

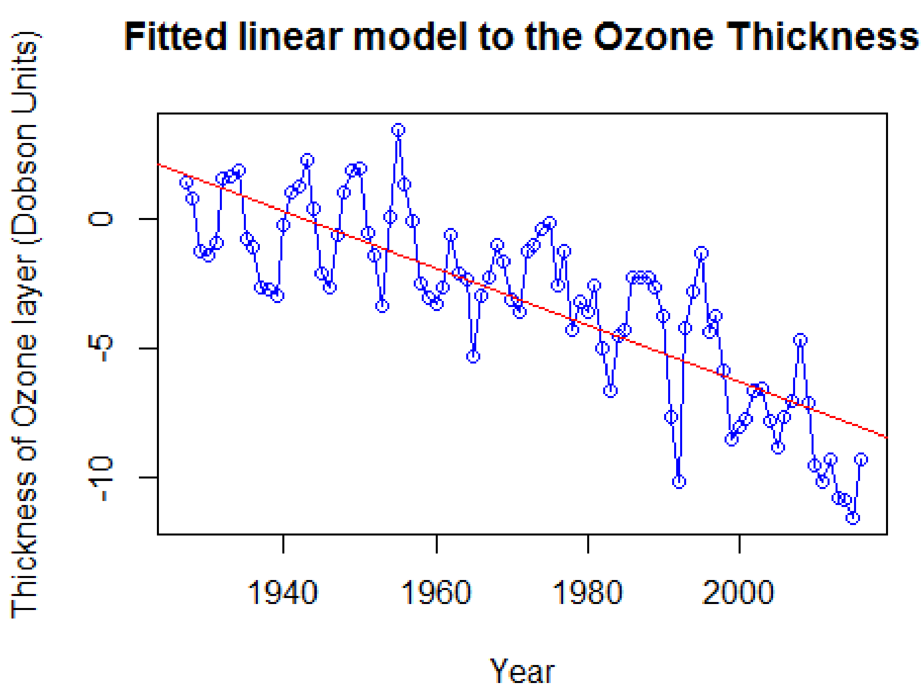
Fitting the series to Linear trend model.

```
# Linear trend model
ozone.model1 = lm(ozone~time(ozone)) # label the model as ozone.model1
summary(ozone.model1)

##
## Call:
## lm(formula = ozone ~ time(ozone))
##
```

```
## 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 ***
## time(ozone)  -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
```

Estimates of slope and intercept are β^1 = -0.110029 and β^0 = 213.720155, respectively. Slope of linear trend model is statistically significant at 5% significance level. According to multiple R2 (coefficient of determination), 66.55% of the variation in ozone data time series is explained by estimated Linear trend model.

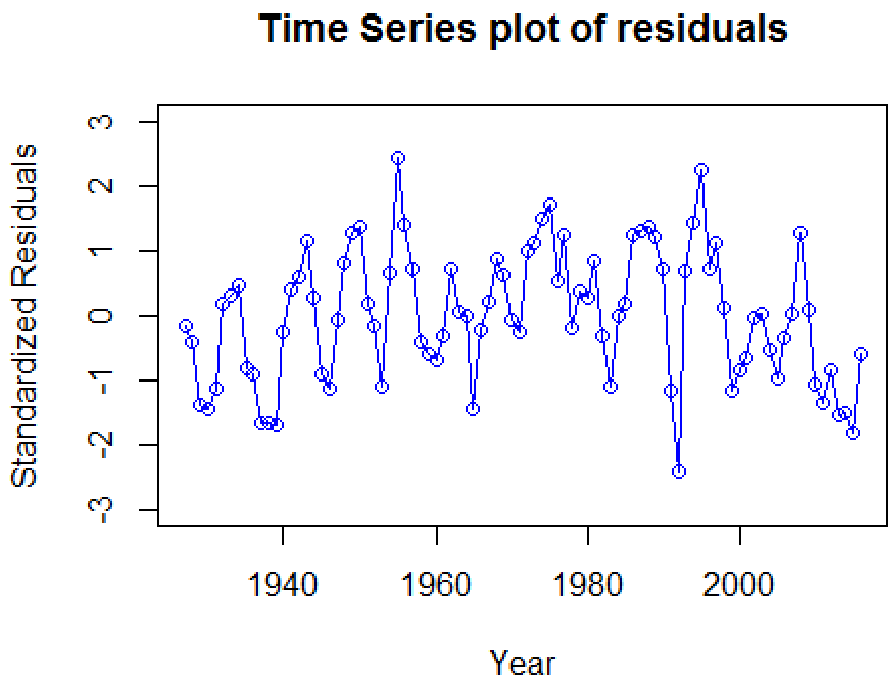
```
# Plotting linear model
plot(ozone, type = 'o', ylab = 'Thickness of Ozone layer (Dobson Units)',
xlab = 'Year', main = "Fitted linear model to the Ozone Thickness", col =
"blue")
abline(ozone.model1, col = "red") # add the fitted least squares line from
model1
```



Residual analysis

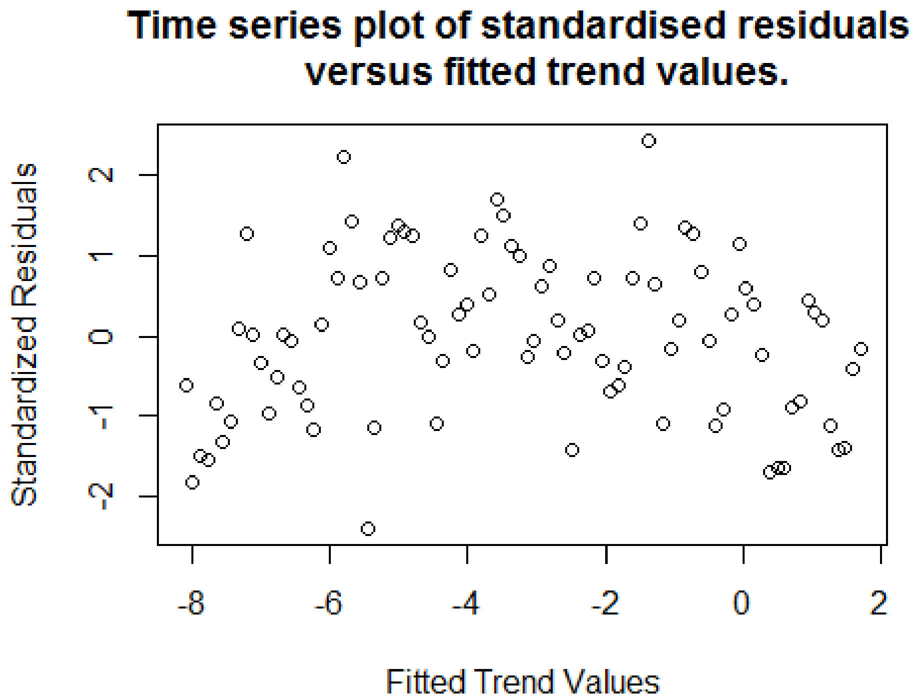
As mentioned earlier, if the trend model is reasonably correct, then the residuals should behave roughly like the true stochastic component, and various assumptions about the stochastic component can be assessed by looking at the residuals. Whereas, if the stochastic component is white noise, then the residuals should behave roughly like independent (normal) random variables with zero mean and standard deviation.

```
# Residual analysis
# Standardized Residuals
plot(y = rstudent(ozone.model1), x = as.vector(time(ozone)), type='o', ylab =
'Standardized Residuals', xlab = 'Year', main = "Time Series plot of
residuals", col="blue", ylim = c(-3,3))
```



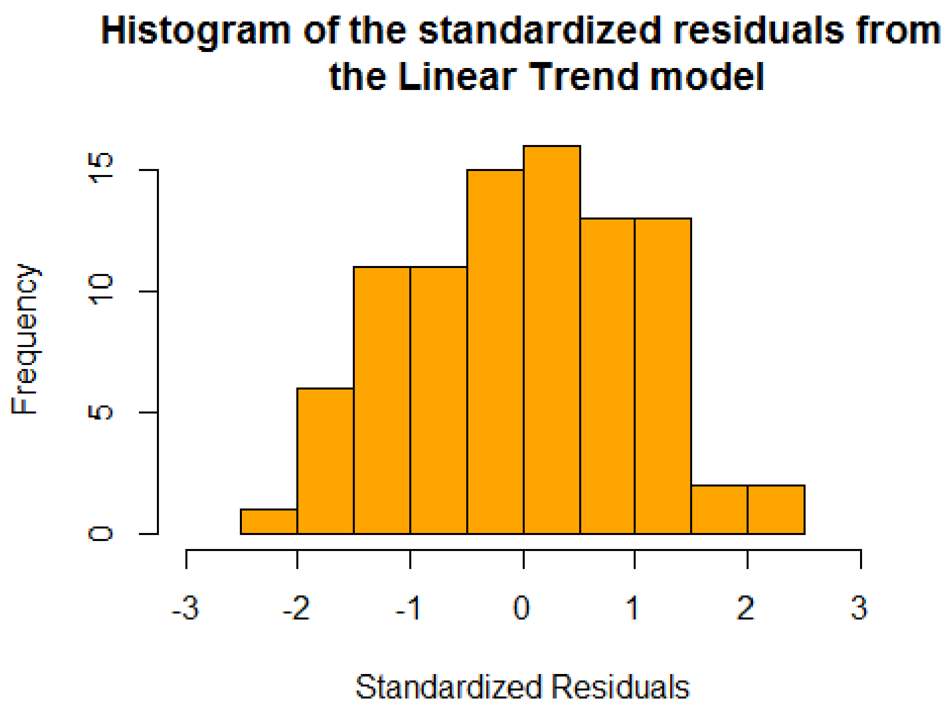
We can see in the residual plot that there are no departures from randomness.

```
plot(y = rstudent(ozone.model1), x = as.vector(fitted(ozone.model1)), type =
'n', ylab = 'Standardized Residuals', xlab = 'Fitted Trend Values', main =
"Time series plot of standardised residuals
     versus fitted trend values.", col = "blue")
points(y = rstudent(ozone.model1), x = as.vector(fitted(ozone.model1)))
```



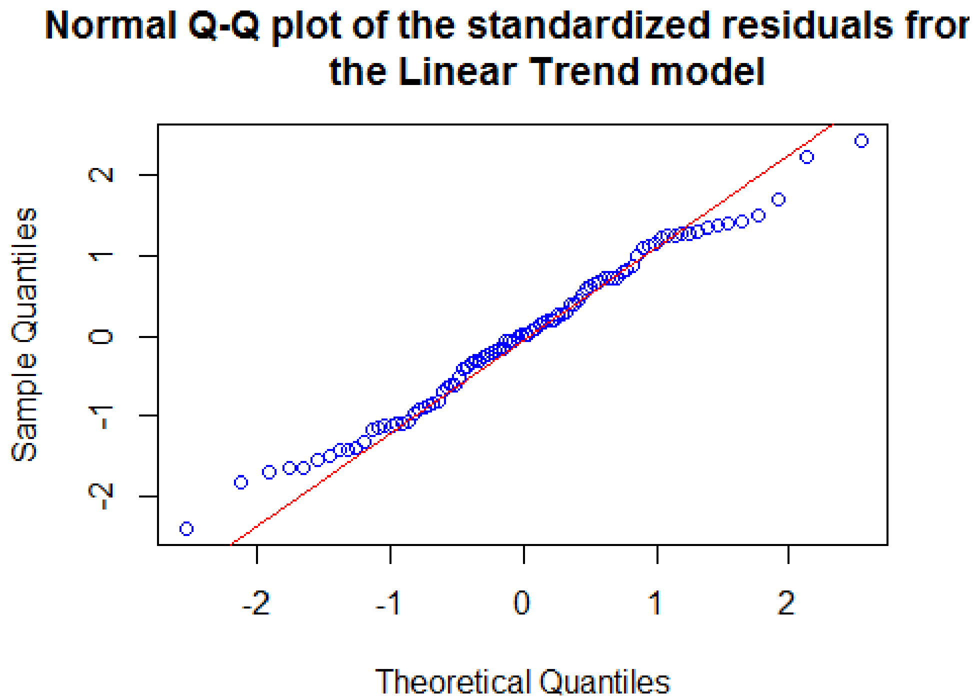
The scatter plot visualization shows random points however we don't see any rectangular pattern to confirm randomness.

```
# Histogram
hist(rstudent(ozone.model1), xlab = 'Standardized Residuals', main =
"Histogram of the standardized residuals from
     the Linear Trend model", col = "orange", xlim = c(-3,3))
```



Normality of Residuals can be checked using Histogram. The plot looks somewhat symmetrical and tails off at both high and low ends.

```
# Normal QQ plot
qqnorm(y = rstudent(ozone.model1), main = "Normal Q-Q plot of the 
standardized residuals from
     the Linear Trend model", col = "blue")
qqline(y = rstudent(ozone.model1), col = "red", lwd = 1)
```

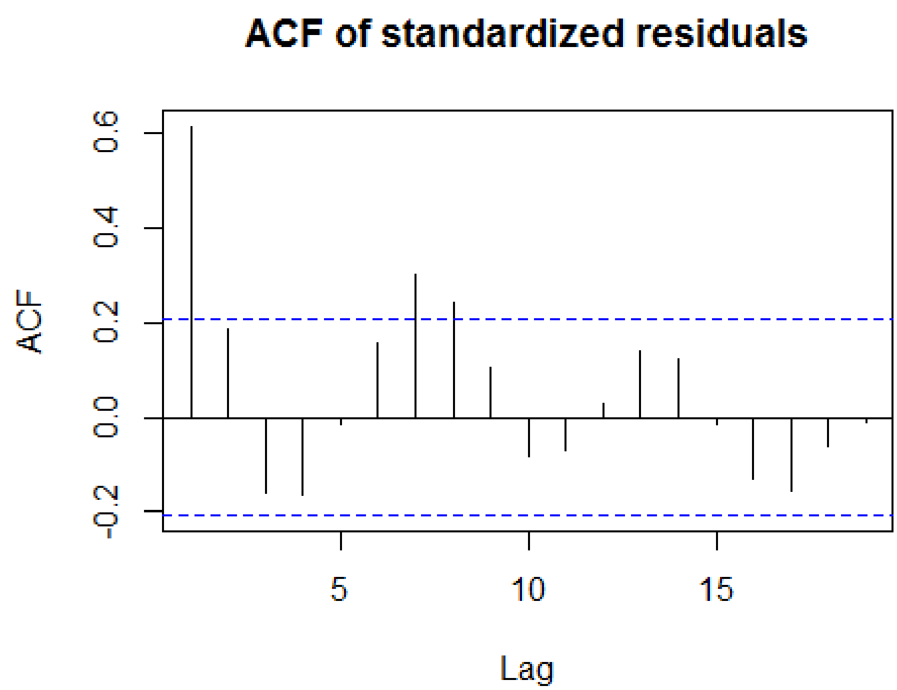


We can see both the end tails of distribution tailing away from straight.

```
# Shapiro-Wilk Normality test
shapiro.test(rstudent(ozone.model1))

##
##  Shapiro-Wilk normality test
##
## data:  rstudent(ozone.model1)
## W = 0.98733, p-value = 0.5372
```

Shapiro-Wilk test calculates the correlation between the residuals and the corresponding normal quantiles. High correlation corresponds to evidence of normality and vice versa. We get the p-value of 0.5372. Thus, we conclude not to reject the null hypothesis that the stochastic component of this model is normally distributed.

```
# Sample Auto-correlation Function
acf(rstudent(ozone.model1), main = "ACF of standardized residuals")
```



There are some lags above the horizontal dashed lines and thus we can infer that the stochastic component of the series is not white noise.

Quadratic model

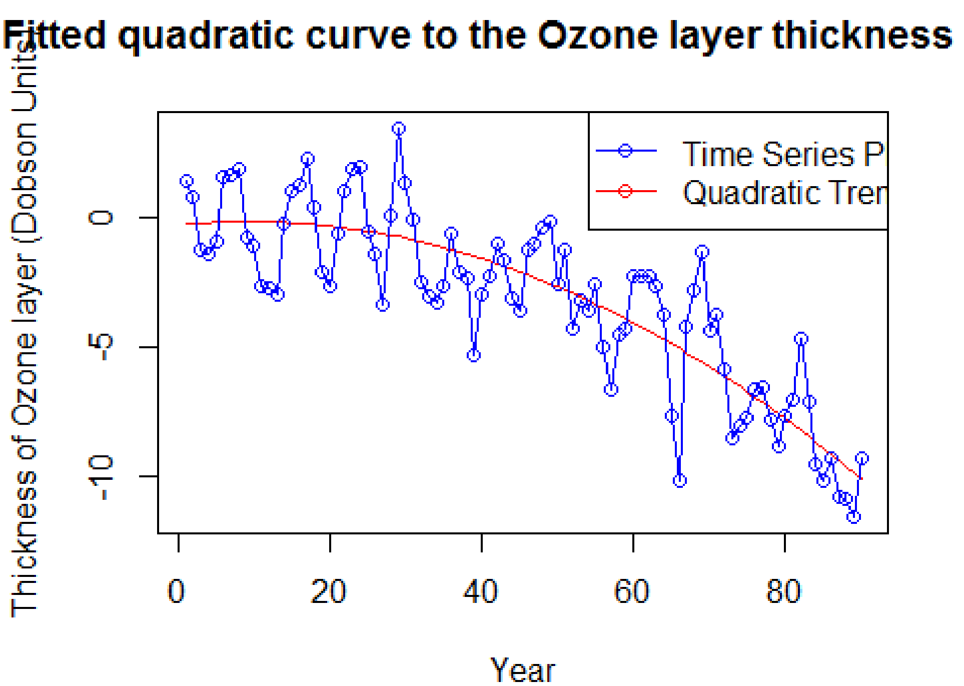
Fitting the series to Quadratic model.

```
# Quadratic model
t = time(ozone)
t2 = t^2
ozone.model2 = lm(ozone~t+t2) # label the model as ozone.model2
summary(ozone.model2)

##
## Call:
## lm(formula = ozone ~ 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
```

p-value is less than 0.05 which is statistically significant at 5% significance level. 73.31% of the variation in ozone data time series is explained by estimated Quadratic trend model which is significant compared to the Linear trend model.

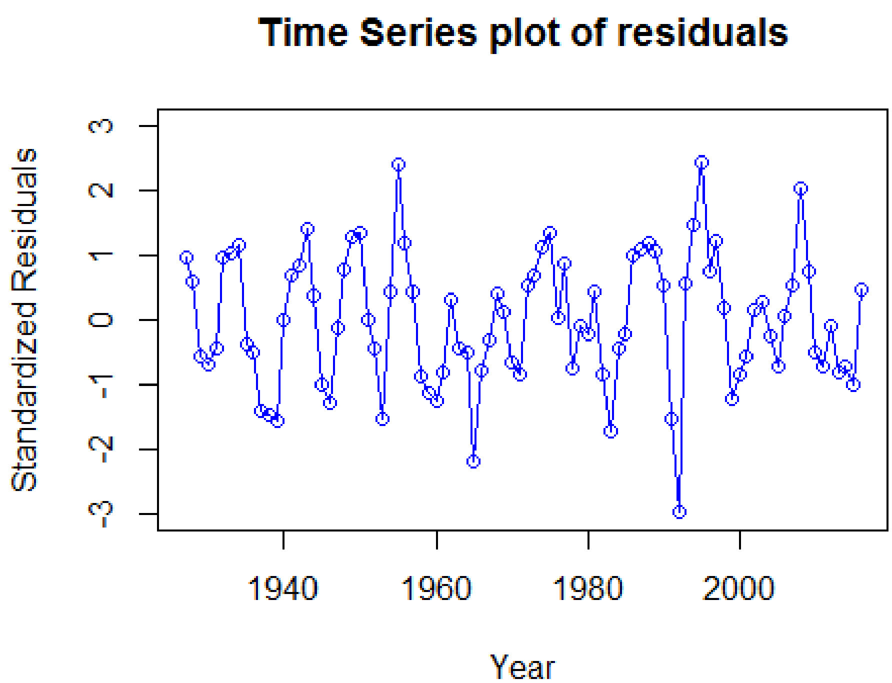
```
# Plotting fitted quadratic curve
plot(ts(fitted(ozone.model2)), ylim = c(min(c(fitted(ozone.model2),
as.vector(ozone))),
                                        max(c(fitted(ozone.model2),
as.vector(ozone)))),
                                        ylab='Thickness of Ozone layer
(Dobson Units)' , xlab='Year',
                                        main = "Fitted quadratic curve to the
Ozone layer thickness data", col = "red")
lines(as.vector(ozone), type="o", col = "blue")
legend ("topright", lty = 1, pch = 1, col = c("blue","red"), text.width = 25,
        c("Time Series Plot","Quadratic Trend Line"))
```



The quadratic curve (red line) fits much better compared to that of linear trend.

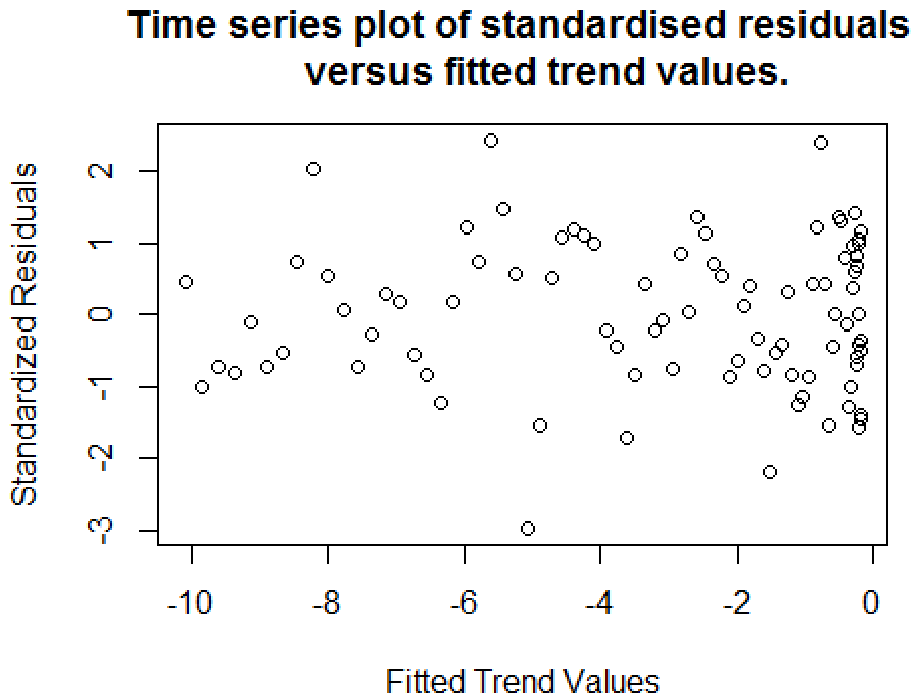
Residual analysis

```r
# Residual analysis
# Standardized Residuals
plot(y = rstudent(ozone.model2), x = as.vector(time(ozone)), type='o', ylab =
'Standardized Residuals', xlab = 'Year', main = "Time Series plot of
residuals", col="blue", ylim = c(-3,3))
```



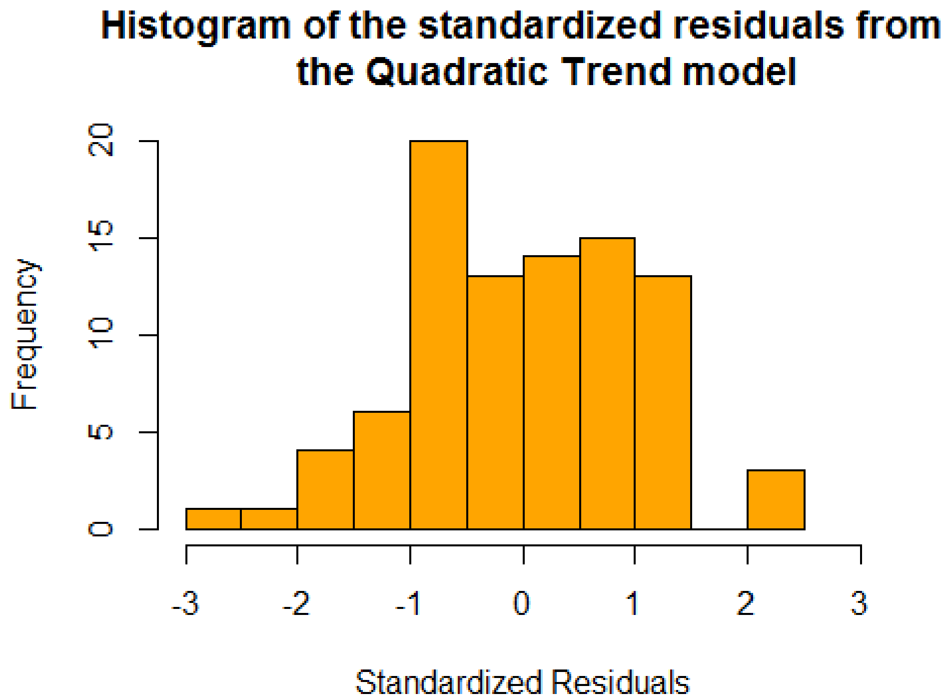
The Quadratic Residual trend has been improved in contrast with the Linear trend.

```r
plot(y = rstudent(ozone.model2), x = as.vector(fitted(ozone.model2)), type =
'n', ylab = 'Standardized Residuals', xlab = 'Fitted Trend Values', main =
"Time series plot of standardised residuals
     versus fitted trend values.", col = "blue")
points(y = rstudent(ozone.model2), x = as.vector(fitted(ozone.model2)))
```



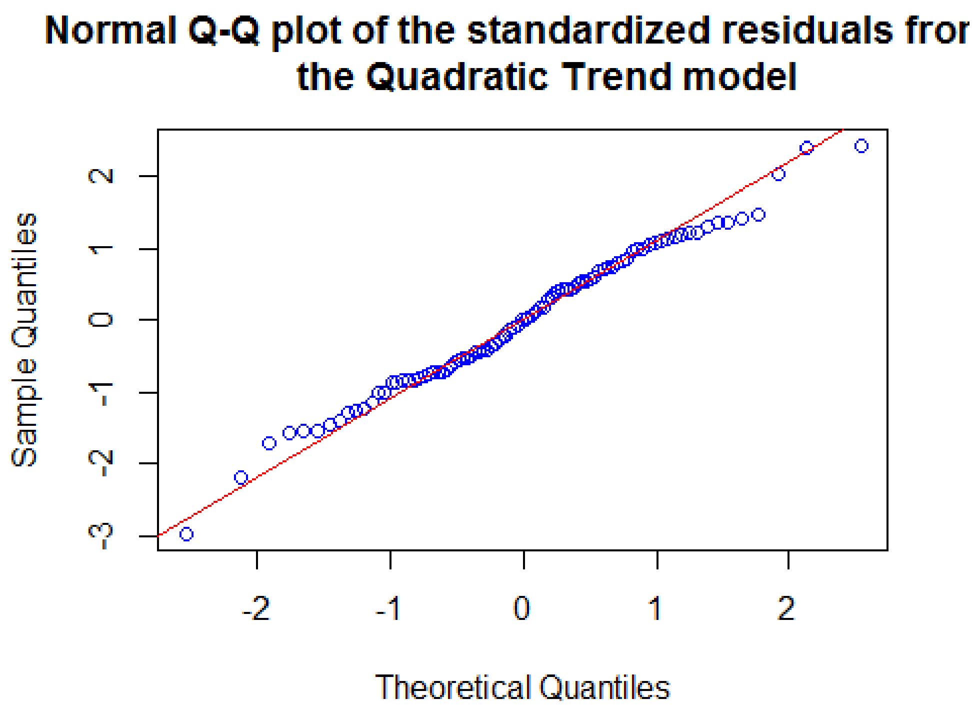
The scatter plot points does not look random.

```
# Histogram
hist(rstudent(ozone.model2), xlab = 'Standardized Residuals', main =
"Histogram of the standardized residuals from
     the Quadratic Trend model", col = "orange", xlim = c(-3,3))
```



We do not see a smooth bell shaped curve for normal distribution.

```
# Normal QQ plot
qqnorm(y = rstudent(ozone.model2), main = "Normal Q-Q plot of the
standardized residuals from
     the Quadratic Trend model", col = "blue")
qqline(y = rstudent(ozone.model2), col = "red", lwd = 1)
```

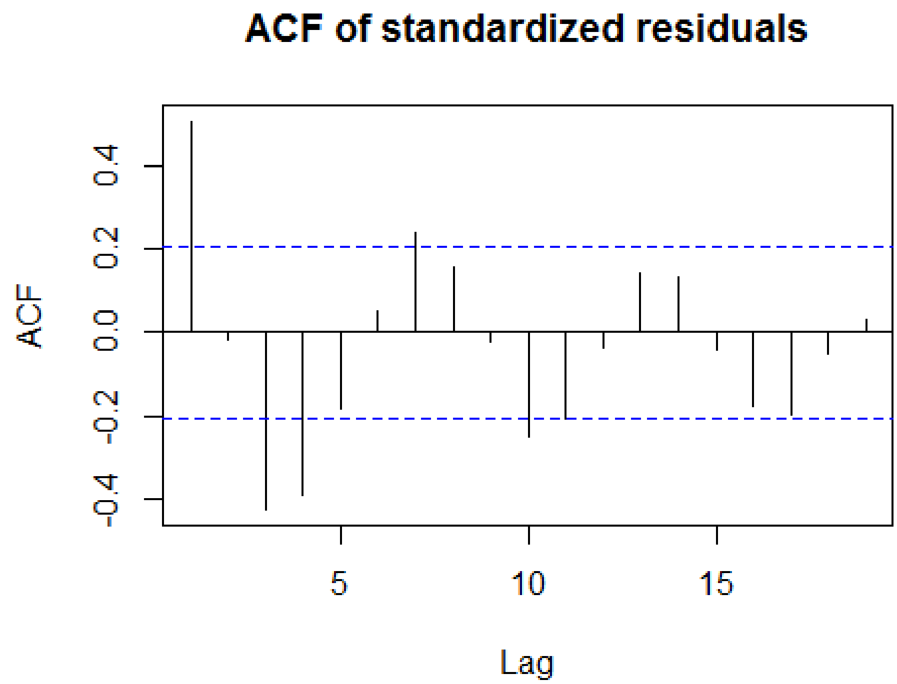


The tail ends depart from the straight line but looks improved considering the Linear model.

```
# Shapiro-Wilk Normality test
shapiro.test(rstudent(ozone.model2))

##
##  Shapiro-Wilk normality test
##
## data:  rstudent(ozone.model2)
## W = 0.98889, p-value = 0.6493
```

The p-value is 0.6493 and hence we cannot reject the null hypothesis which states that the the stochastic component of this model is normally distributed.

```
# Sample Auto-correlation Function
acf(rstudent(ozone.model2), main = "ACF of standardized residuals")
```

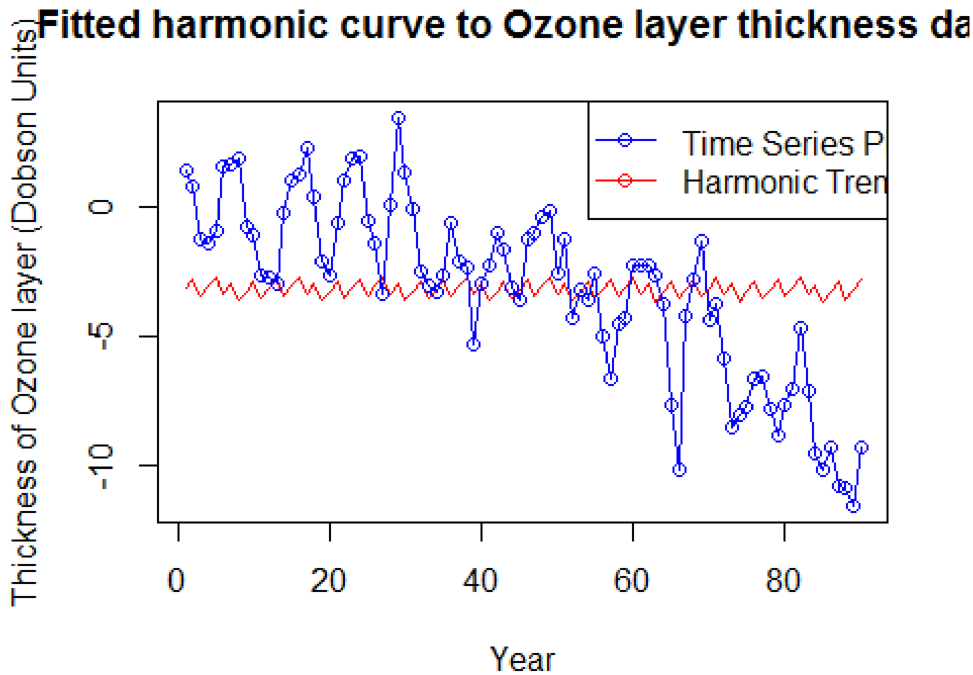


The plot displays several lags with higher values over the confidence interval and so the we can say that the stochastic component of the series is not white noise.

Harmonic model

```
# Harmonic model
har.=harmonic(ozone, 0.45) # calculate cos(2*pi*t) and sin(2*pi*t)
ozone.model3 = lm(ozone~har.)
summary(ozone.model3)

##
## Call:
## lm(formula = ozone ~ har.)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3520 -1.8905  0.4837  2.3643  6.4248
##
## Coefficients: (1 not defined because of singularities)
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -2.970e+00  4.790e-01  -6.199 1.79e-08 ***
## har.cos(2*pi*t)         NA         NA      NA       NA
## har.sin(2*pi*t)  5.462e+11  7.105e+11   0.769    0.444
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.522 on 88 degrees of freedom
```

```
## Multiple R-squared:  0.006672,	Adjusted R-squared:  -0.004616
## F-statistic: 0.5911 on 1 and 88 DF,  p-value: 0.4441

# Plotting fitted cosine curve
plot(ts(fitted(ozone.model3)), ylim =
c(min(c(fitted(ozone.model3),as.vector(ozone))),

max(c(fitted(ozone.model3),as.vector(ozone)))),
     ylab="Thickness of Ozone layer (Dobson Units)", xlab="Year",
     main = "Fitted harmonic curve to Ozone layer thickness data", type =
"l", lty = 1, col = "red")
lines(as.vector(ozone), type="o", col = "blue")
legend ("topright", lty = 1, pch = 1, col = c("blue","red"), text.width = 25,
        c("Time Series Plot","Harmonic Trend Line"))
```



The p-value of the model is more than 0.05 and the adjusted R2 value is less than the other two models which implies that the model is insignificant. Also, seasonality is not present in the series and so the cosine trend couldn't fit to this data.

Summary

Comparing all the models, we can conclude that Quadratic model is the best fit to the series since it has higher multiple R2 square value and Shapiro-Wilk Normality test value.

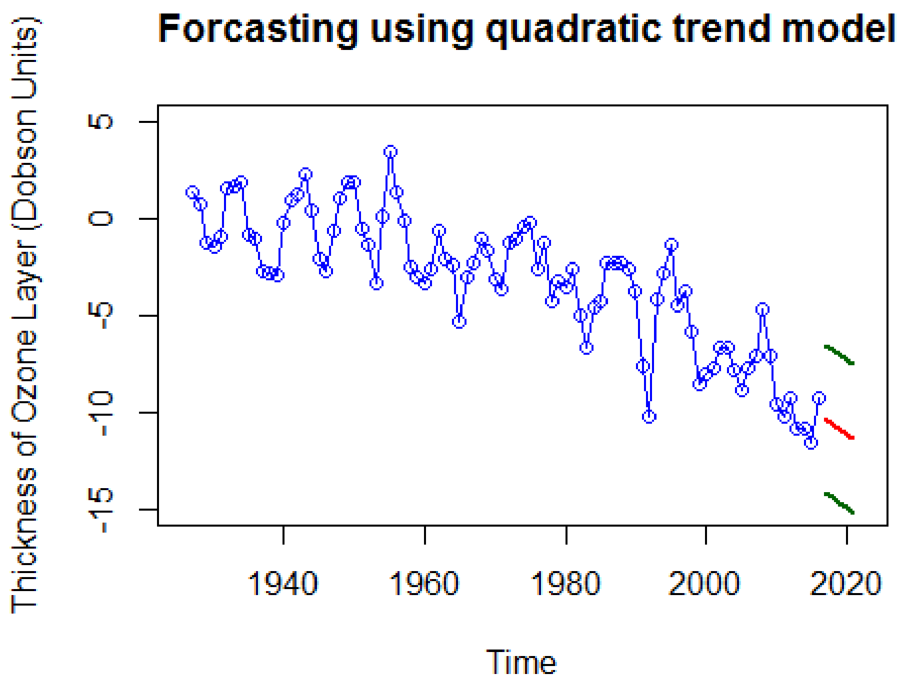
Forecasting

Predicting the yearly changes for 5 years with the Quadratic model.

```
# Forcasting
t = c(2017, 2018, 2019, 2020, 2021)
t2 = t^2
pred = data.frame(t,t2)
forecast = predict(ozone.model2, pred, interval = "prediction")
print(forecast)
```

```
##         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
```

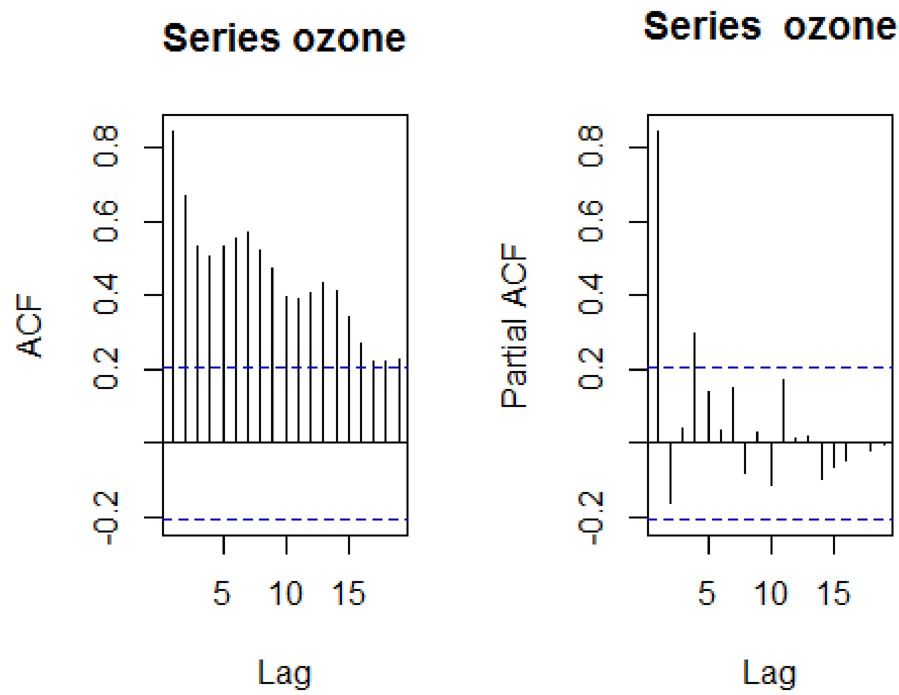
```
# Plotting the forcast data
plot(ozone, xlim = c(1926, 2022), ylim = c(-15, 5), type="o", ylab="Thickness
of Ozone Layer (Dobson Units)" , main = " Forcasting using quadratic trend
model", col="blue")
lines(ts(as.vector(forecast [,1]), start = c(2017,1), frequency = 1),
col="red", type="l", lwd=2)
lines(ts(as.vector(forecast [,2]), start = c(2017,1), frequency = 1),
col="darkgreen", type="l", lwd=2)
lines(ts(as.vector(forecast [,3]), start = c(2017,1), frequency = 1),
col="darkgreen", type="l", lwd=2)
```



Through the plot, we can say that the ozone layer thickness will gradually decrease after 5 years.

Task 2

```
class(ozone)

## [1] "ts"

par(mfrow=c(1,2))
acf(ozone)
pacf(ozone)
```




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

A downward trend can be observed in the ACF plot whereas we can see that the PACF plot contains the first lag and confirms the trend.

Data Transformation

We will use Box cox transformation to check if need to transform to reduce the changing variance.

```r
ozonetr = BoxCox.ar(ozone+abs(min(ozone))+1)

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
```

```
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
```

```
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
```

```
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```

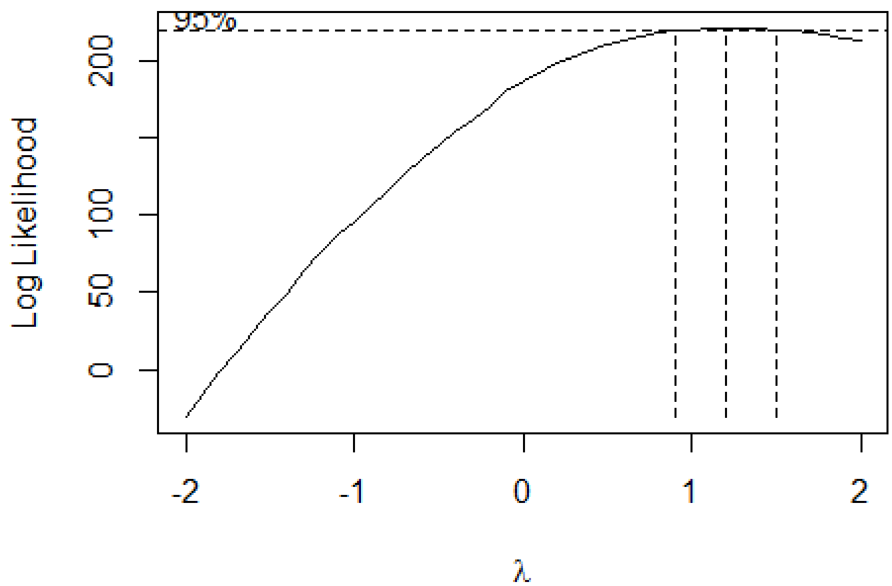
```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
```

```
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
```

```
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```

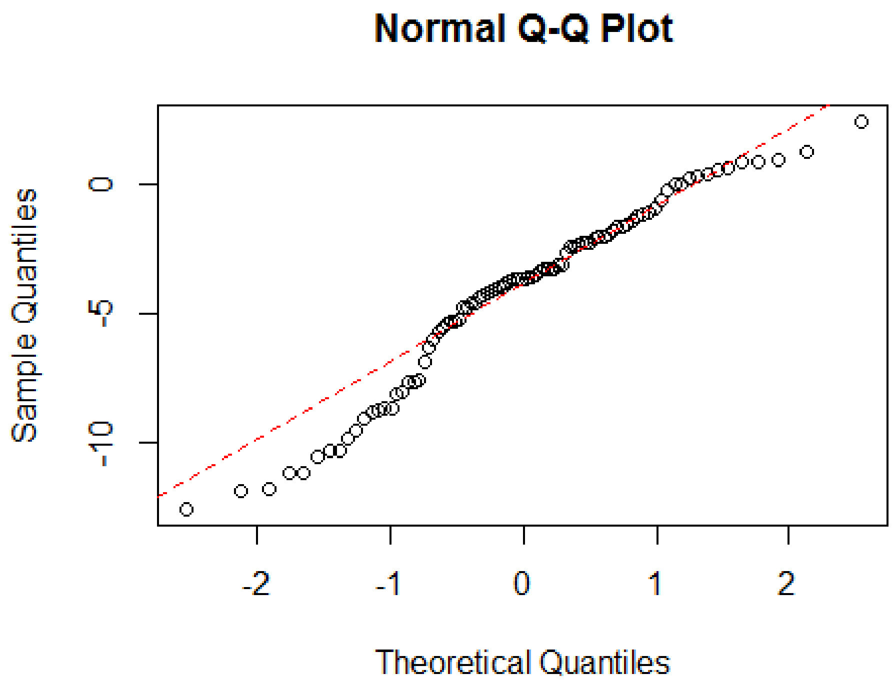
```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean):
possible
## convergence problem: optim gave code = 1
```




```
ozonetr$ci

## [1] 0.9 1.5
```

We can see that the confidence interval lies between 0.9 and 1.5 which suggests that there is no need to use log or power transformation.4

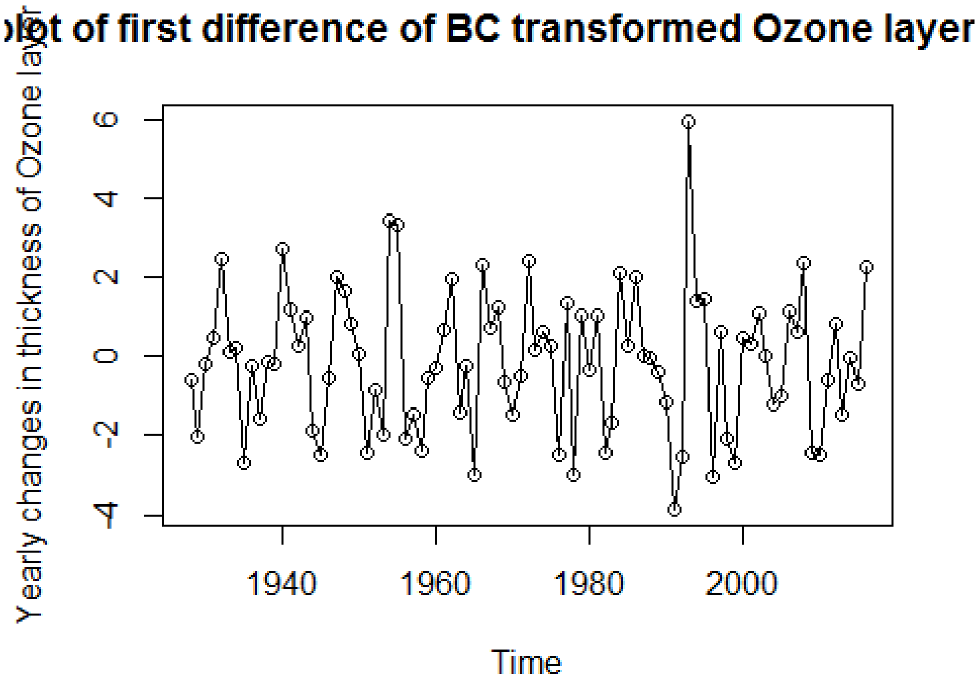
```
lambda = 1
BC.ozone = (ozone^lambda-1)/lambda
qqnorm(BC.ozone)
qqline(BC.ozone, col = 2, lwd = 1, lty = 2)
```



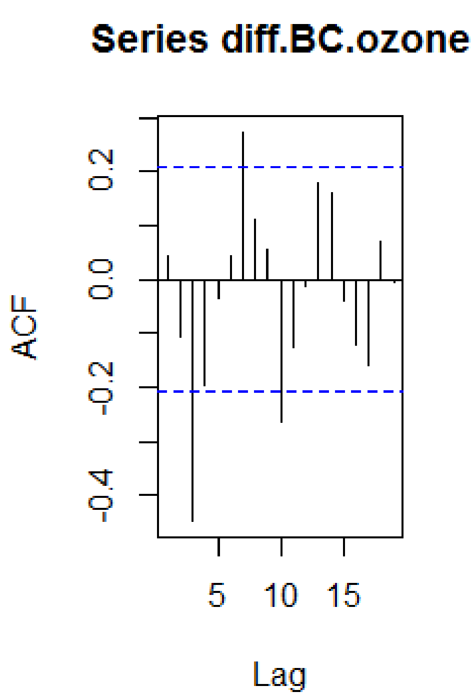

```
shapiro.test(BC.ozone)

##
##  Shapiro-Wilk normality test
##
## data:  BC.ozone
## W = 0.95605, p-value = 0.004031
```

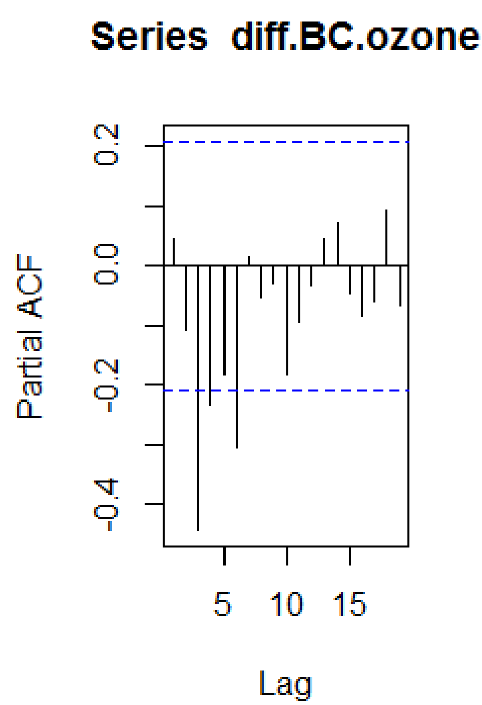
The shapiro test implies that the the series does not have normal distribution as p-value is below 0.05.

```
diff.BC.ozone = diff(BC.ozone, differences = 1)
plot(diff.BC.ozone,type='o',
     ylab='Yearly changes in thickness of Ozone layer',
     main = 'Time series plot of first difference of BC transformed Ozone
layer thickness series')
```

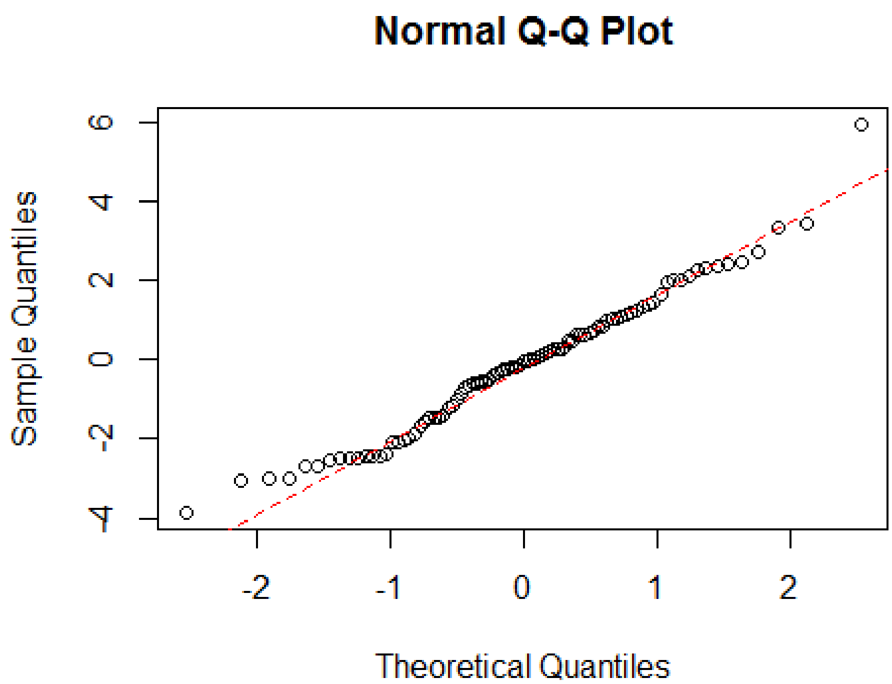



```
par(mfrow=c(1,2))
acf(diff.BC.ozone)
pacf(diff.BC.ozone)
```





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

qqnorm(diff.BC.ozone)
qqline(diff.BC.ozone, col = 2, lwd = 1, lty = 2)
```



The differencing box cox QQ looks improved considering the previous QQ plot.

```r
shapiro.test(diff.BC.ozone)

##
##  Shapiro-Wilk normality test
##
## data:  diff.BC.ozone
## W = 0.97907, p-value = 0.1606
```

The p-value is more than 0.05 so we conclude to no reject the null hypothesis.

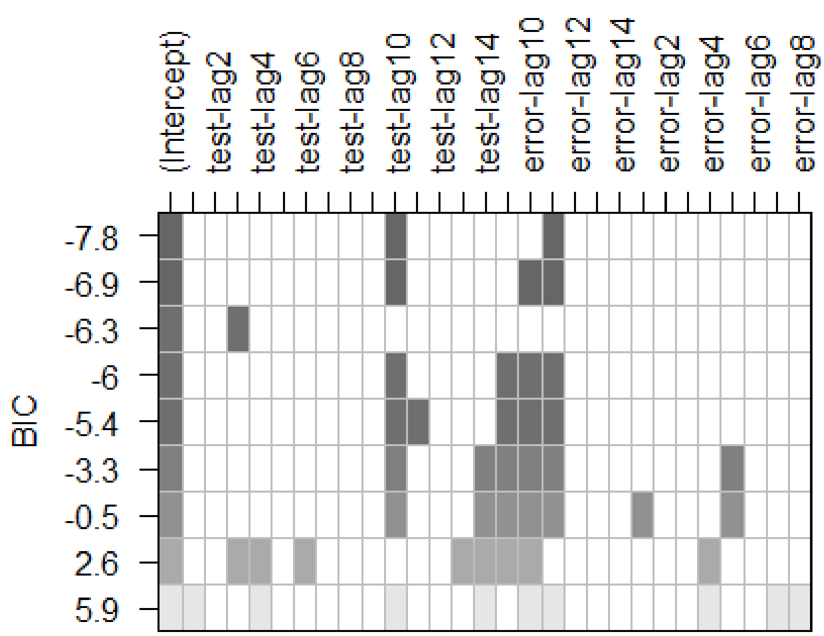
```r
adf.test(diff.BC.ozone)

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

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

We can say that time series after first lag is stationary as p value is not significant.

```
eacf(diff.BC.ozone)

## 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
```

the set of models are ARIMA(1,1,3), ARIMA(2,1,3), and ARIMA(3,1,3)

```
ozoneBIC = armasubsets(y=diff.BC.ozone, nar = 14,nma = 14,y.name = 'test',
                       ar.method = 'ols')

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
force.in =
## force.in, : 8 linear dependencies found

## Reordering variables and trying again:

plot(ozoneBIC)
```



Through the shaded columns the coefficients are AR(11) and MA(10) which is ARIMA(11,1,3).

Conclusion

The ozone layer thickness showed the downward trend with no seasonality, change in variation & intervention followed by which linear, quadratic and harmonic modeling was performed. The quadratic model was suggested as the best fit to the ozone layer series data. However, there are some flaws associated with it. In the ACF plot, there were some significant lags which confirmed the smoothness of the time series plot. This wasn't expected in the white noise process. This could be due to the series being non-stationary. The non-stationary series was then converted into stationary to have a best fit model using differencing. The set of ARIMA models were ARIMA(1,1,3), ARIMA(2,1,3), ARIMA(3,1,3) andARIMA(11,1,3).

References

https://ourworldindata.org/ozone-layer

MATH1318 Time Series Analysis notes by Dr. Haydar Demirhan