Time Series Analysis using R

A.Tharwinvikram Nadar

20/11/2020

Motivation:

To study the Britania stock price data perform various time series model and it's analysis

```
library(ggplot2)
library(forecast)
library(tseries)
library(ISLR)
raww_data = read.csv("/home/rocky/Desktop/Vikram2019-20/dataset/Stock-Data/BRITANNIA.csv"
                     ,header = TRUE)
```

In data we see Open close and VWAP prices of the stock.

To build a forecasting model for the closing price of the Britannia stock data

```
head(raww_data,10)
```

```
##          Date    Symbol Series Prev.Close  Open   High    Low   Last  Close
## 1  2000-01-03 BRITANNIA     EQ     703.25 705.0 759.50 705.00 758.00 756.90
## 2  2000-01-04 BRITANNIA     EQ     756.90 710.0 770.00 710.00 740.00 754.55
## 3  2000-01-05 BRITANNIA     EQ     754.55 755.0 759.00 705.00 740.00 735.30
## 4  2000-01-06 BRITANNIA     EQ     735.30 740.0 794.15 740.00 770.00 785.65
## 5  2000-01-07 BRITANNIA     EQ     785.65 808.0 848.50 798.00 848.50 848.50
## 6  2000-01-10 BRITANNIA     EQ     848.50 900.0 916.40 865.00 916.40 912.20
## 7  2000-01-11 BRITANNIA     EQ     912.20 920.0 920.00 839.25 865.00 853.75
## 8  2000-01-12 BRITANNIA     EQ     853.75 900.0 900.00 860.55 890.95 882.70
## 9  2000-01-13 BRITANNIA     EQ     882.70 890.0 920.00 875.00 885.00 881.40
## 10 2000-01-14 BRITANNIA     EQ     881.40 872.5 880.00 864.00 870.00 869.65
##      VWAP Volume     Turnover Trades Deliverable.Volume X.Deliverble
## 1  741.01   7512 5.566488e+11     NA                 NA           NA
## 2  742.52   8135 6.040391e+11     NA                 NA           NA
## 3  739.92   6095 4.509784e+11     NA                 NA           NA
## 4  788.83  19697 1.553756e+12     NA                 NA           NA
## 5  827.53  33107 2.739708e+12     NA                 NA           NA
## 6  905.42  29575 2.677784e+12     NA                 NA           NA
## 7  858.02  20635 1.770516e+12     NA                 NA           NA
## 8  885.18   9312 8.242756e+11     NA                 NA           NA
## 9  898.96  19526 1.755313e+12     NA                 NA           NA
## 10 873.91  15675 1.369847e+12     NA                 NA           NA
```

```
tail(raww_data,10)
```

```
##            Date    Symbol Series Prev.Close   Open    High     Low    Last
## 5153 2020-09-17 BRITANNIA     EQ    3844.50 3848.0 3890.95 3768.75 3803.00
## 5154 2020-09-18 BRITANNIA     EQ    3815.65 3837.7 3839.75 3774.85 3804.00
## 5155 2020-09-21 BRITANNIA     EQ    3797.50 3790.0 3795.00 3613.50 3643.50
## 5156 2020-09-22 BRITANNIA     EQ    3629.30 3660.0 3678.00 3540.05 3590.00
## 5157 2020-09-23 BRITANNIA     EQ    3588.00 3618.9 3652.90 3562.80 3630.00
## 5158 2020-09-24 BRITANNIA     EQ    3624.90 3590.0 3655.00 3560.20 3611.00
## 5159 2020-09-25 BRITANNIA     EQ    3612.75 3640.0 3716.95 3615.00 3703.75
## 5160 2020-09-28 BRITANNIA     EQ    3686.40 3710.9 3778.00 3689.00 3731.05
## 5161 2020-09-29 BRITANNIA     EQ    3737.35 3769.0 3796.00 3701.65 3715.05
## 5162 2020-09-30 BRITANNIA     EQ    3736.85 3734.0 3825.00 3714.05 3795.50
##        Close    VWAP Volume     Turnover Trades Deliverable.Volume X.Deliverble
## 5153 3815.65 3845.28 741010 2.849393e+14  49657             152843       0.2063
## 5154 3797.50 3801.11 947698 3.602309e+14  41566             631679       0.6665
## 5155 3629.30 3687.23 604282 2.228128e+14  43918             251550       0.4163
## 5156 3588.00 3590.40 670648 2.407896e+14  45992             179945       0.2683
## 5157 3624.90 3611.53 405856 1.465763e+14  30346              61928       0.1526
## 5158 3612.75 3612.21 517316 1.868652e+14  38631             152823       0.2954
## 5159 3686.40 3668.71 507368 1.861385e+14  33389             133845       0.2638
## 5160 3737.35 3737.45 390640 1.459997e+14  23905              96348       0.2466
## 5161 3736.85 3756.82 449330 1.688051e+14  24309             126432       0.2814
## 5162 3798.15 3789.81 535771 2.030472e+14  32948             170100       0.3175
```

Some Terminology

VWAP(volume weighted average price)-In finance, volume-weighted average price is the ratio of the value traded to total volume traded over a particular time horizon. It is a measure of the average price at which a stock is traded over the trading horizon.

Volume - In the context of a single stock trading on a stock exchange, the volume is commonly reported as the number of shares that changed hands during a given day. The transactions are measured on stocks, bonds, options contracts, futures contracts and commodities.

Here tsclean function is used to identify and replace outliers and missing values

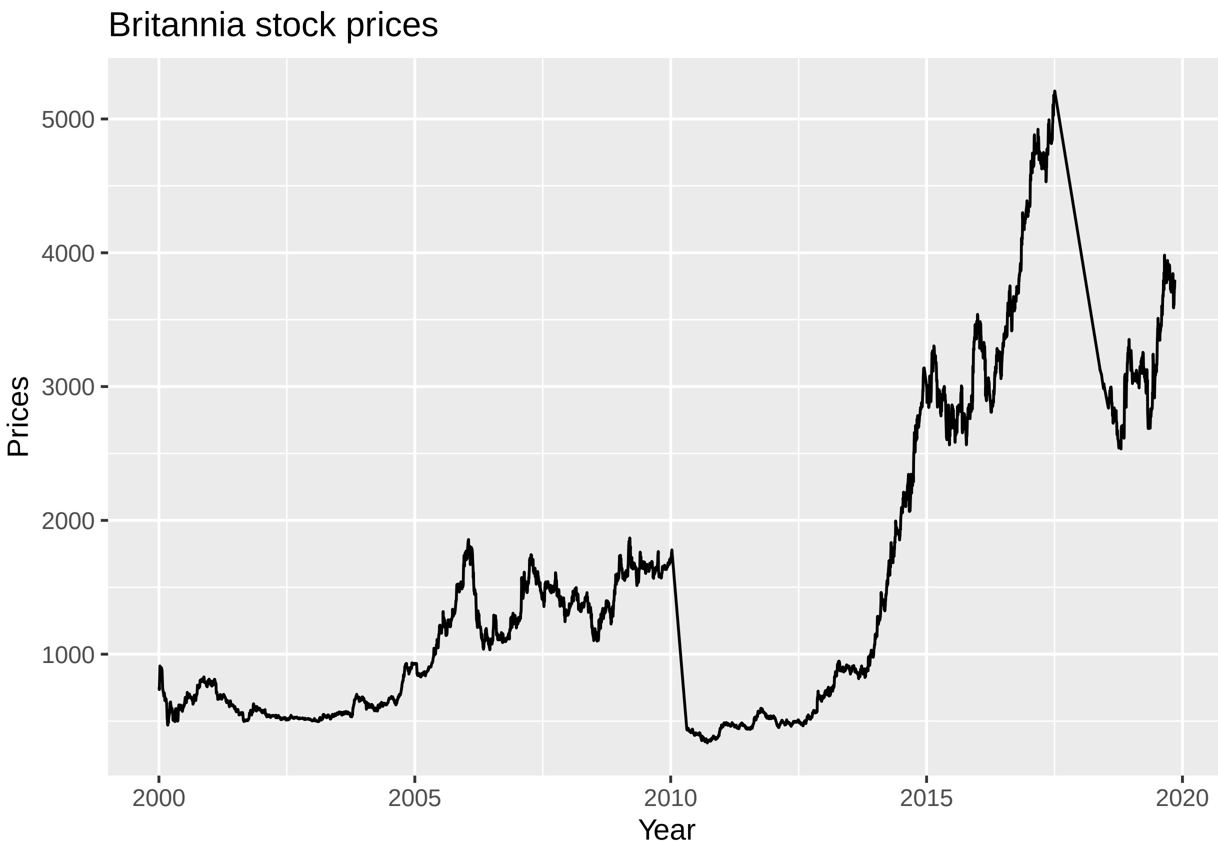
Here frequency is set to 260 because in a year stock market functions for 260 days

```r
clean_dataa = tsclean(raww_data$Close)
my_time = ts(clean_dataa,start = 2000,frequency = 260)
```

Plotting the time series

```r
autoplot(my_time)+ggtitle("Britannia stock prices")+xlab("Year")+ylab("Prices")
```

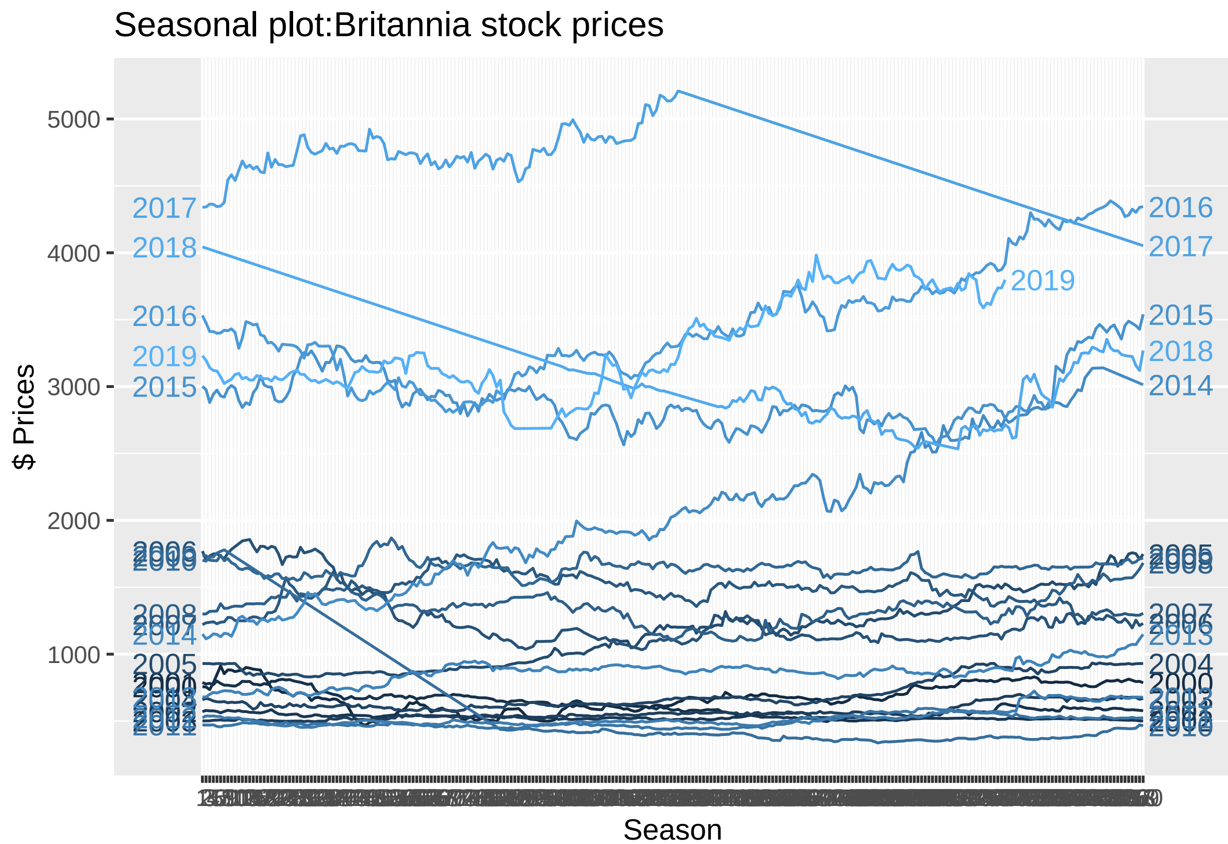
```
## Warning in is.na(main): is.na() applied to non-(list or vector) of type 'NULL'
```



From visual inspection we don't see trend and seasonality.We observe there is a sudden price drop around the year 2010.We will try ploting some seasonality plots to check seasonality

```
ggseasonplot(my_time, year.labels=TRUE, year.labels.left=TRUE,continuous = TRUE) +
  ylab("$ Prices") +
  ggtitle("Seasonal plot:Britannia stock prices ")
```

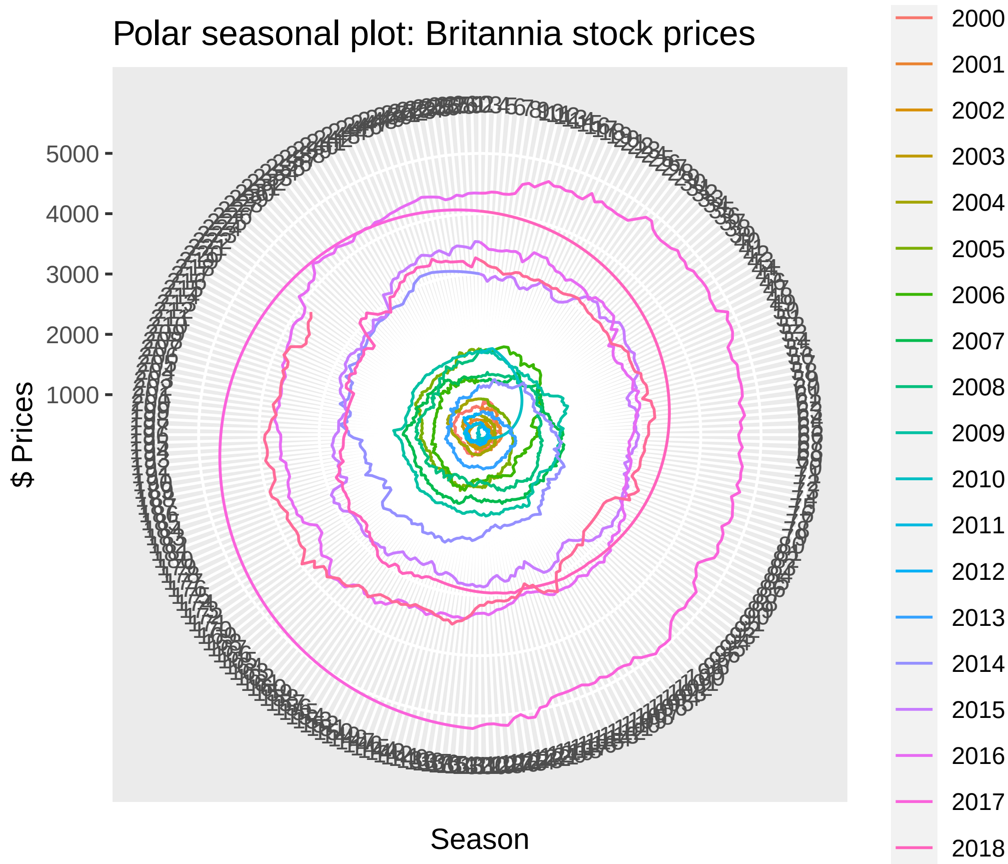
```
## Warning in is.na(ylab): is.na() applied to non-(list or vector) of type 'NULL'
```



Plotting Seasonal Polar plot

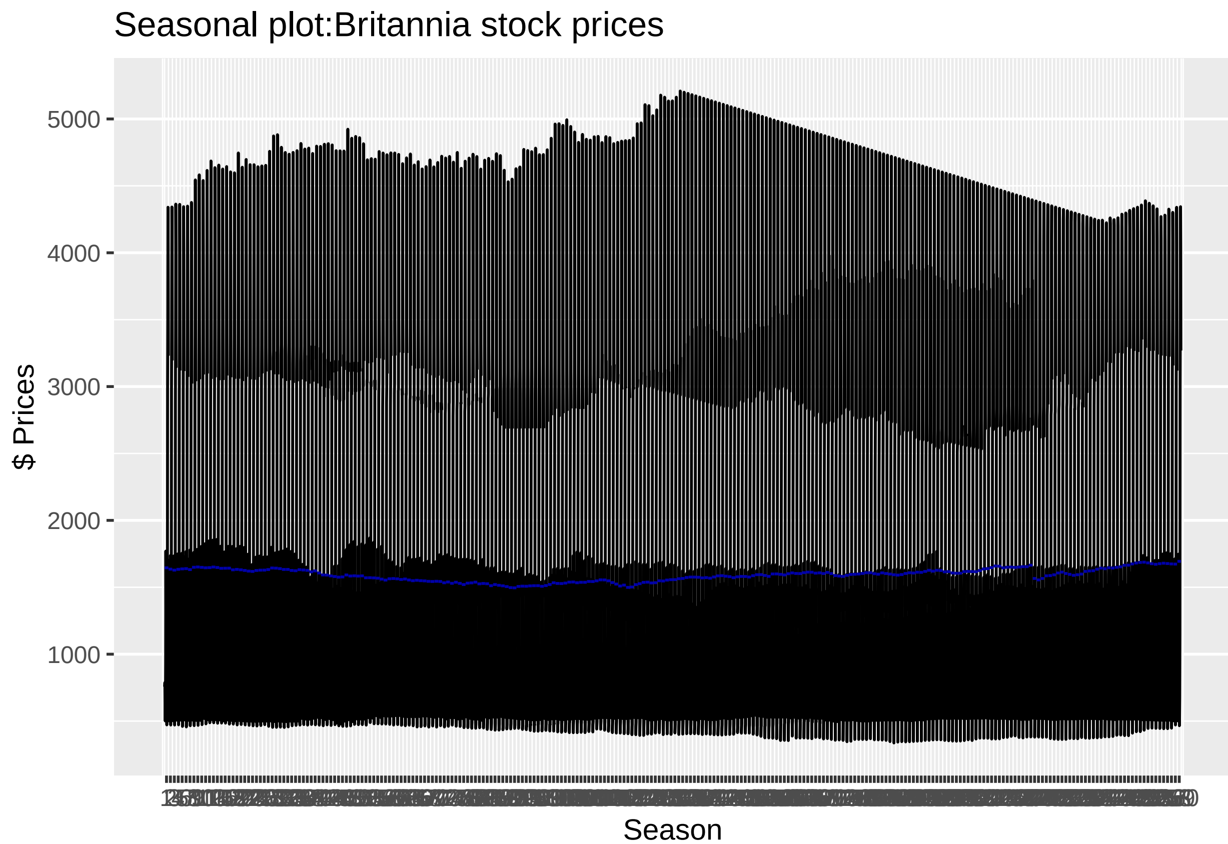
```r
ggseasonplot(my_time, polar=TRUE) +ylab("$ Prices") +
  ggtitle("Polar seasonal plot: Britannia stock prices")
```

```
## Warning in is.na(ylab): is.na() applied to non-(list or vector) of type 'NULL'
```



Plotting seasonal subseries plot

```
ggsubseriesplot(my_time)+ylab("$ Prices") +
  ggtitle("Seasonal plot:Britannia stock prices ")
```



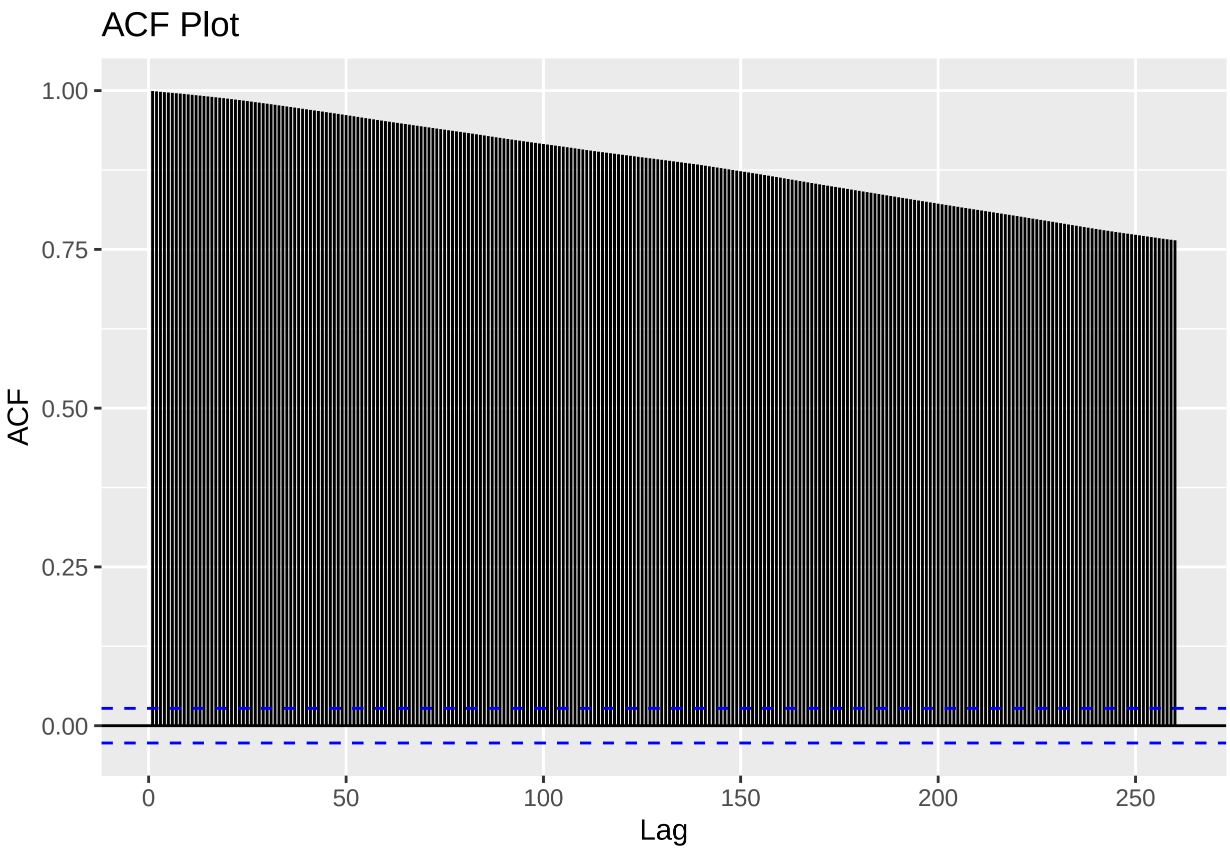
Hence from the seasonal plots presence of seasonality in the series is not ensured.

Autocorrelation function:

Let x_{t} be a series s and t be a point in the series then the auto correlation function is defined as $\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}}$. The ACF measures the linear predictability of the series at time t, say x_{t} , using only the value of x_{s}.

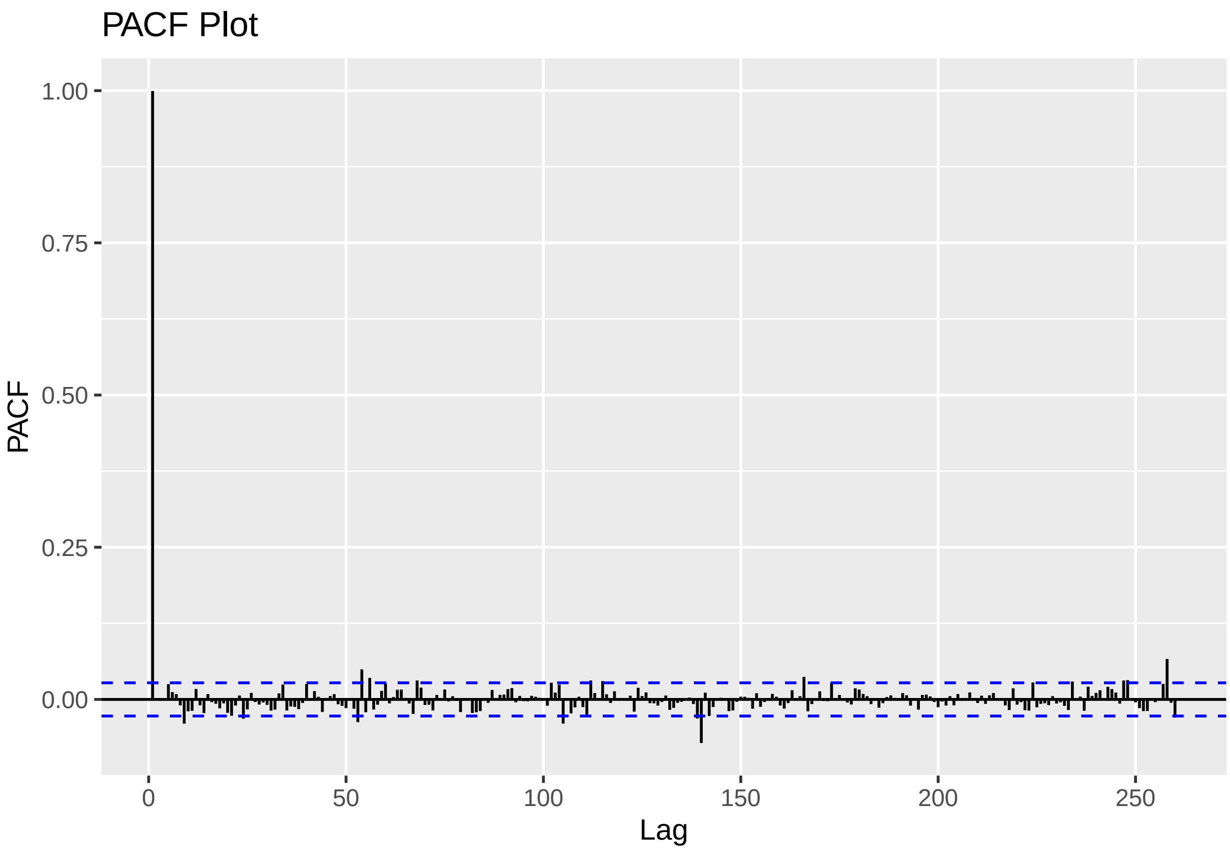
From ACF plot we can say that the data is highly auto corelated and there is decresing trend present in the data

```
ggAcf(my_time,lag.max = 260)+ggtitle("ACF Plot")
```



PACF plot talks about the coorelation of consecutive points of the series.From the plot we see that lag 1,53 and other lag outside the dotted line are statiscally siginificant and the rest are statiscally siginifcant to 0

```
ggPacf(my_time,lag.max = 260)+ggtitle("PACF Plot")
```



Now we will split the data. #### Splitting the data into train and test data

```
train_data = head(my_time,round(length(my_time)*0.8))
h = length(my_time)-length(train_data)
test_data = tail(my_time,h)
```

Now we will try to build basic models on our series

1) Average method

2) Naive method

3) Seasonal Naive

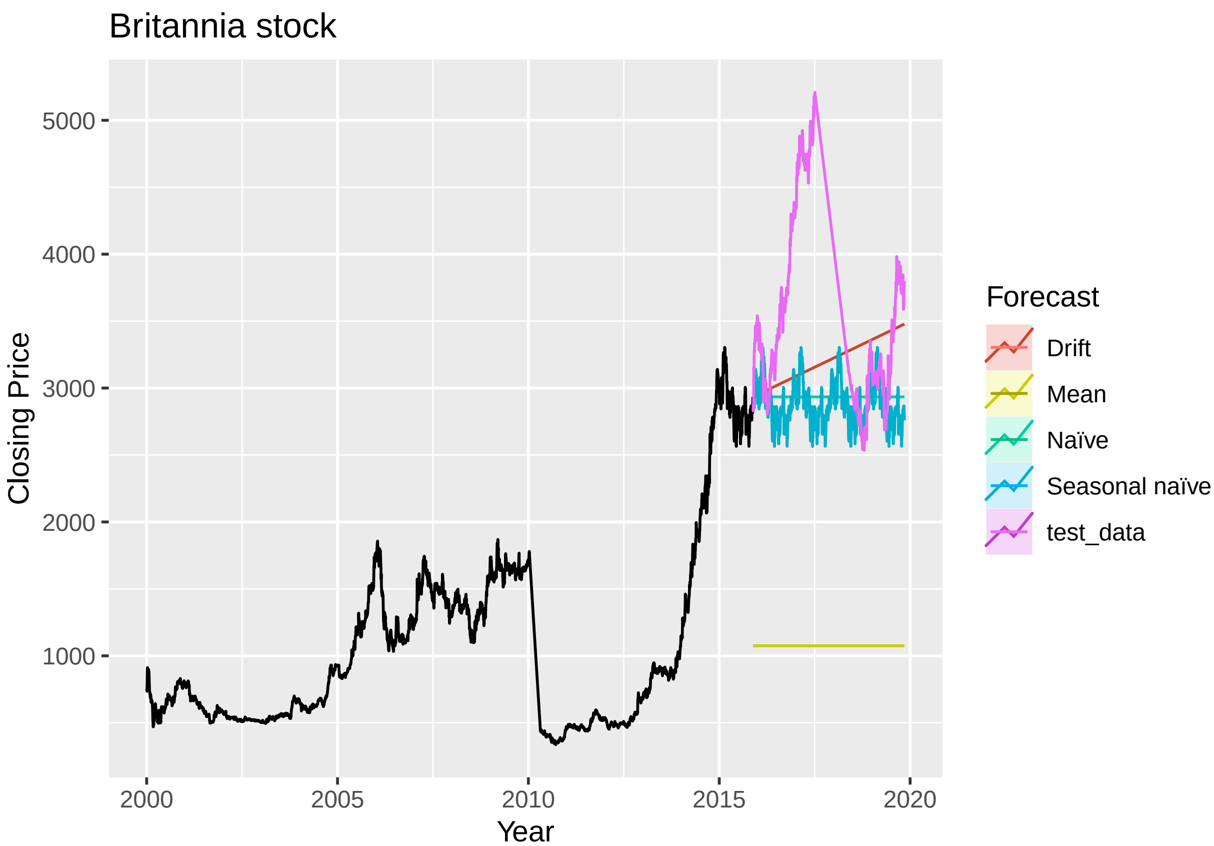
4) Random walk drift

Building all the models in the series

```
model_1 = meanf(train_data, h=h)
model_2 = rwf(train_data, h=h)
model_3 = rwf(train_data, drift=TRUE, h=h)
model_4 = snaive(train_data, h=h)
```

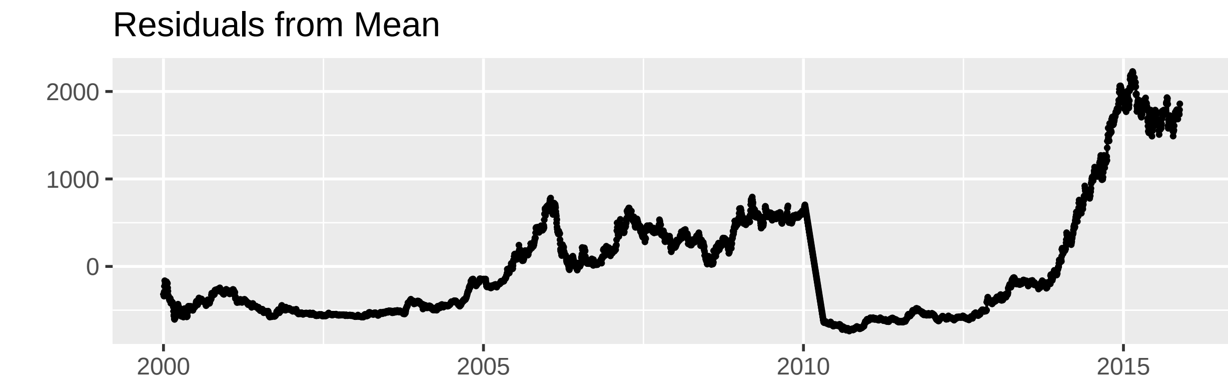
```r
autoplot(train_data) +
  autolayer(model_1,series="Mean", PI=FALSE) +
  autolayer(model_2,series="Naïve", PI=FALSE) +
  autolayer(model_3,series="Drift", PI=FALSE) +
  autolayer(model_4,series="Seasonal naïve", PI=FALSE)+
  ggtitle("Britannia stock ") +
  xlab("Year") + ylab("Closing Price") +
  guides(colour=guide_legend(title="Forecast"))+autolayer(test_data)
```

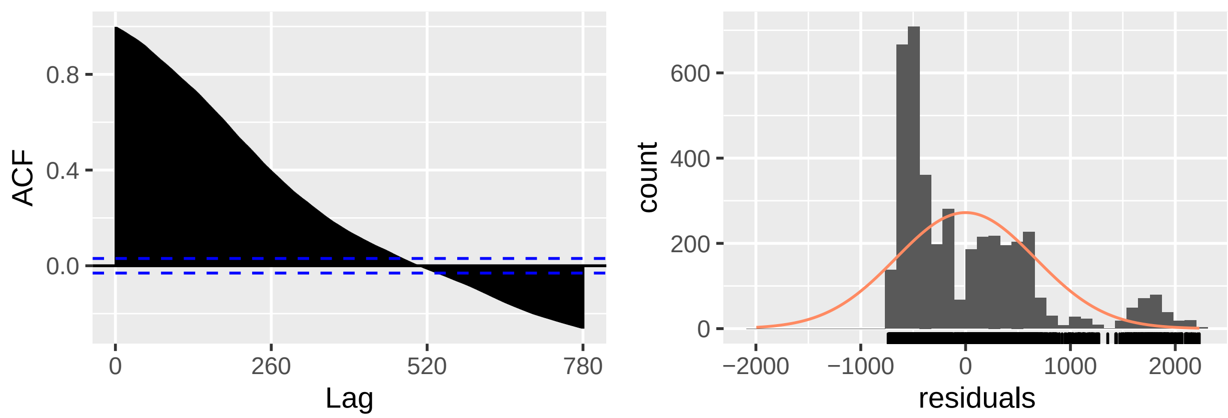
```
## Warning in is.na(main): is.na() applied to non-(list or vector) of type 'NULL'
```



From visual inspection we see that drift seem to perform well as it captures some the test data #### Now we will perform residue analysis for each model to see which performs better ## Residue analysis of model 1

```r
checkresiduals(model_1)
```





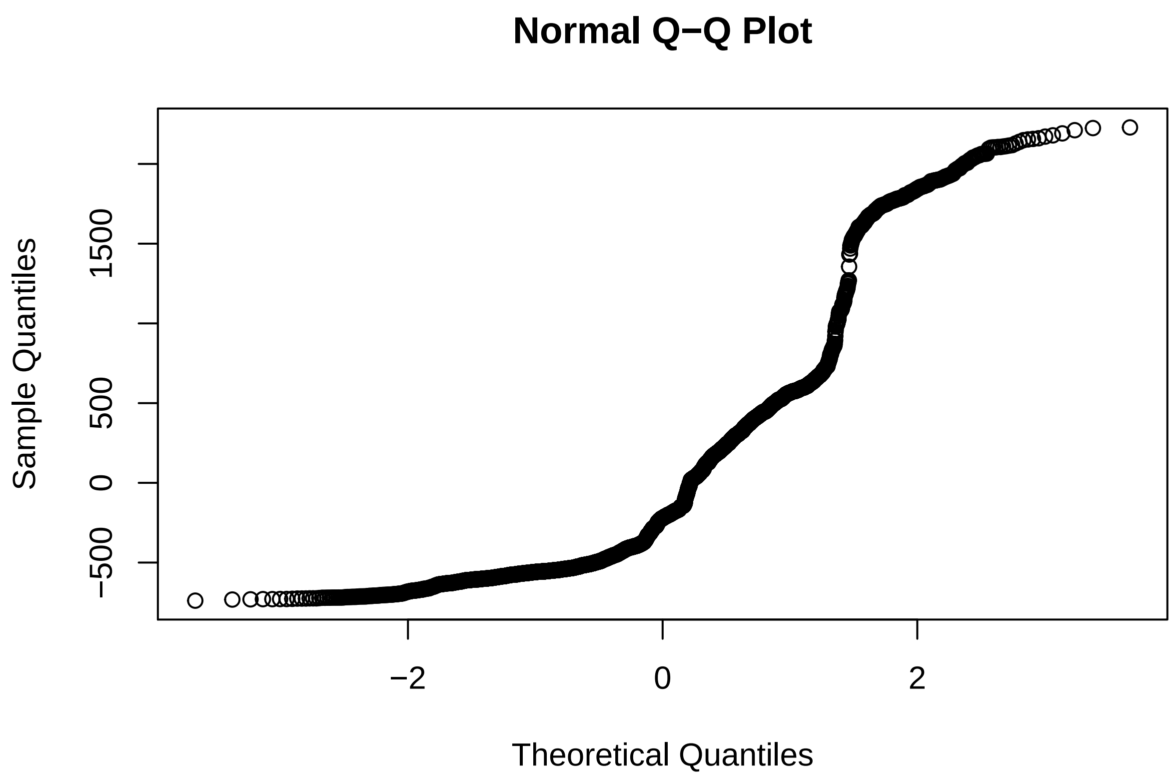

```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 654480, df = 519, p-value < 2.2e-16
##
## Model df: 1.   Total lags used: 520
```

pvalue of Ljung-Box test $\leq$ 0.05 hence we reject the null hypothesis therefore there is no co-relation in the residuals therefore it is not stationary

```r
shapiro.test(model_1$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_1$residuals
## W = 0.83878, p-value < 2.2e-16
```

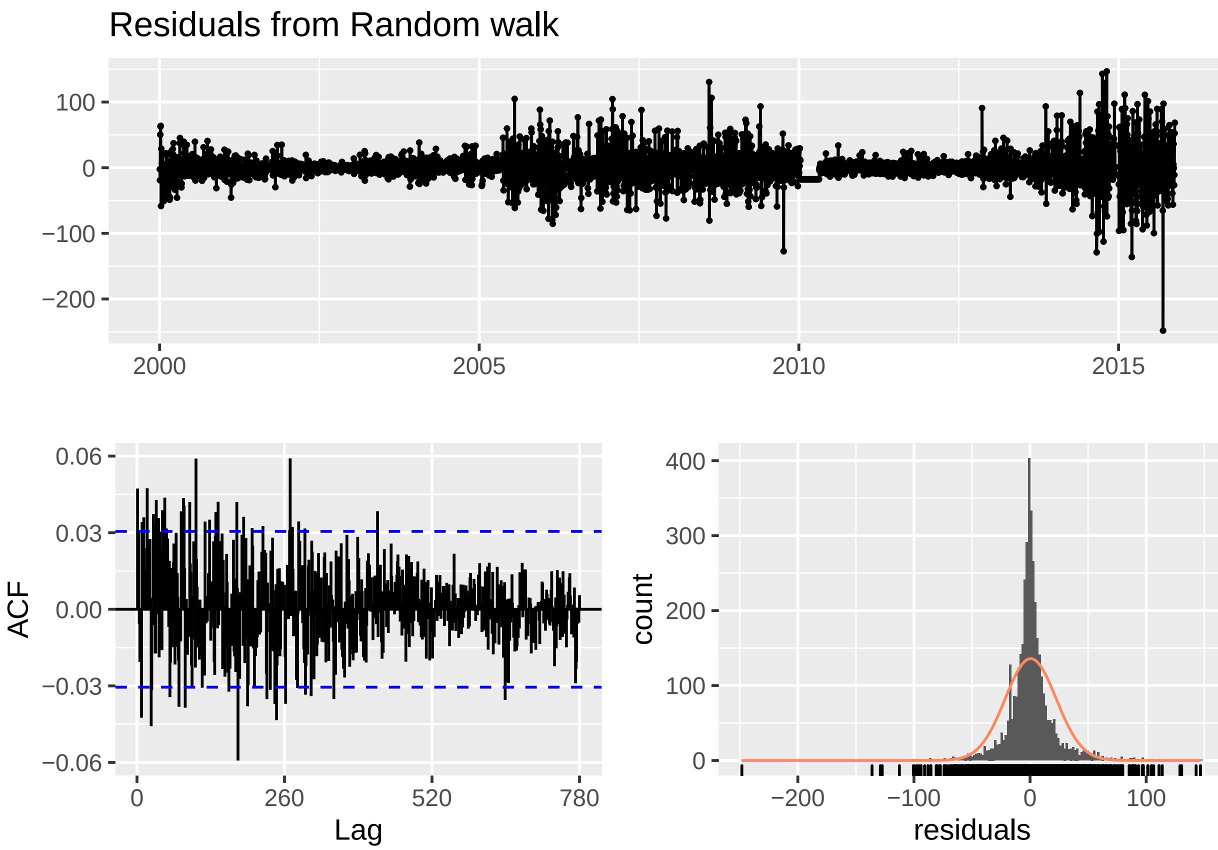
```r
qqnorm(model_1$residuals)
```



pvalue of Shapiro-Wilk test ≤ 0.05 hence we reject the null hypothesis therefore residuals not normally distributed

Residue analysis of model 2

```
checkresiduals(model_2)
```

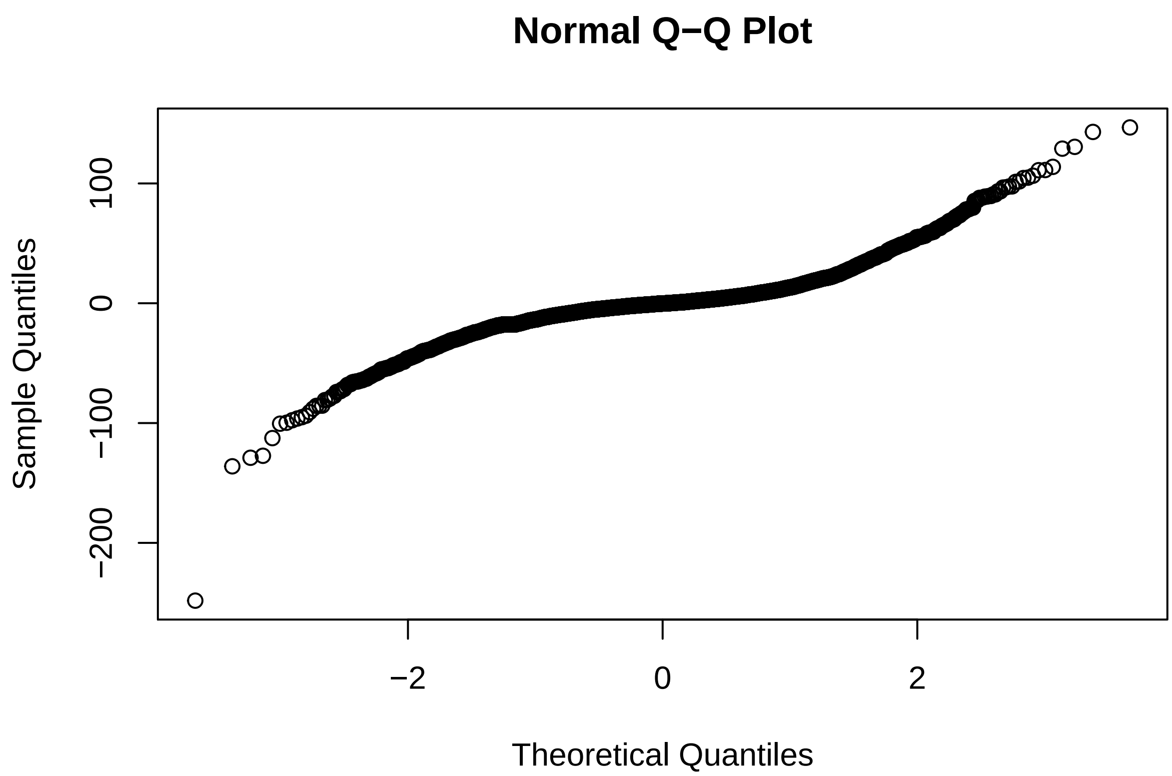



```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk
## Q* = 764.1, df = 520, p-value = 1.459e-11
##
## Model df: 0.   Total lags used: 520
```

```
shapiro.test(model_2$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_2$residuals
## W = 0.86267, p-value < 2.2e-16
```

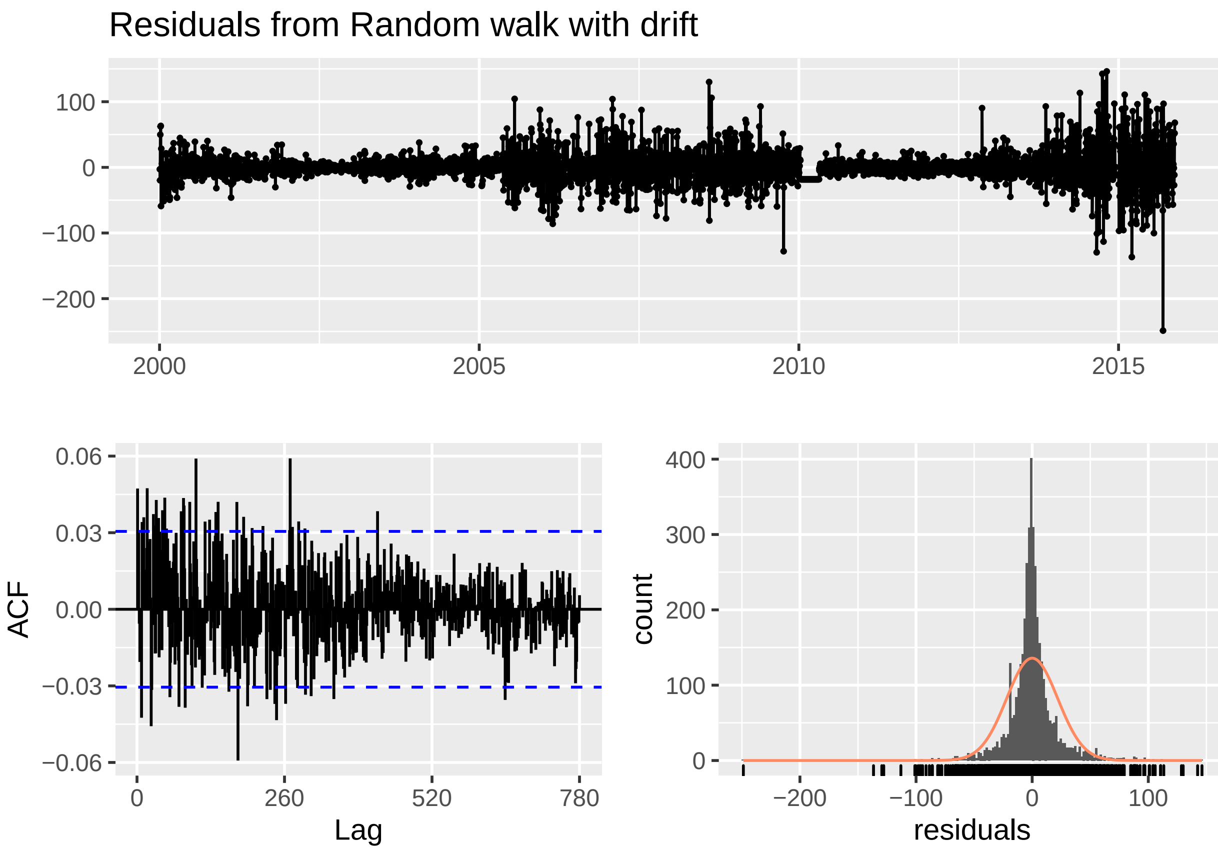
```
qqnorm(model_2$residuals)
```



pvalue of Ljung-Box test ≤ 0.05 hence we reject the null hypothesis therefore there is no co-relation in the residuals therefore it is not stationary.pvalue of Shapiro-Wilk test ≤ 0.05 hence we reject the null hypothesis therefore residuals not normally distributed

Residue analysis of model 3

```
checkresiduals(model_3)
```

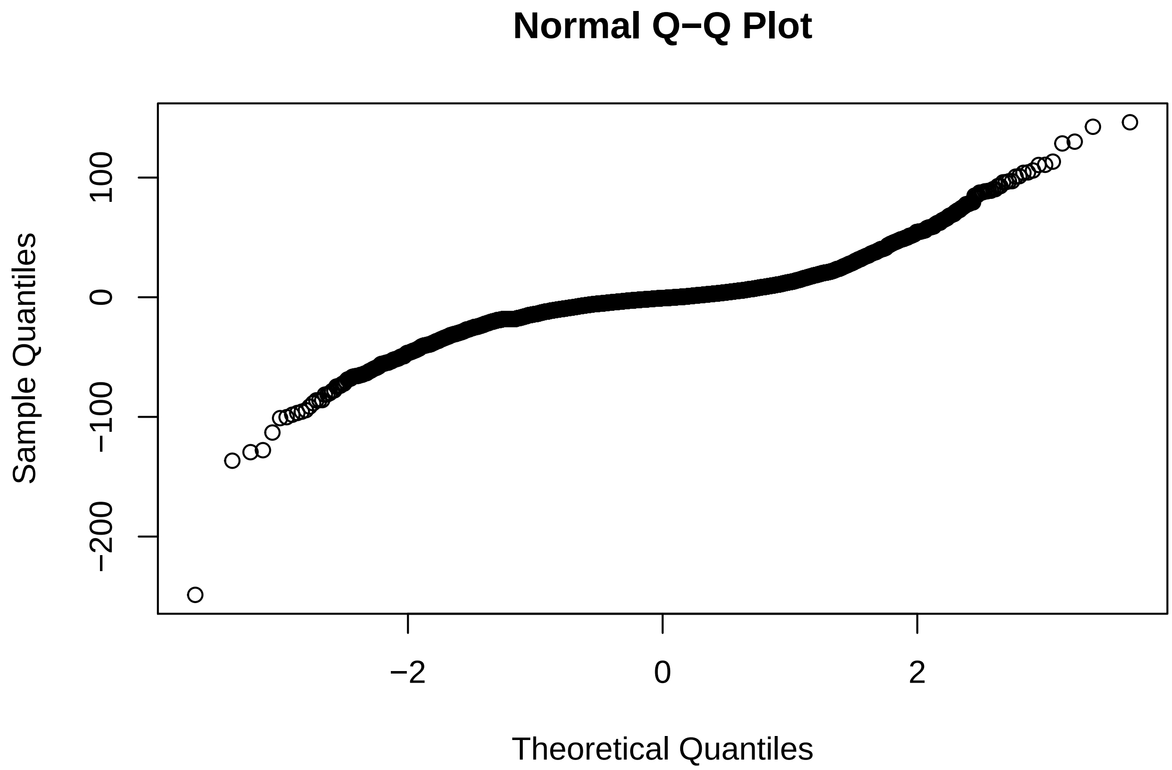



```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk with drift
## Q* = 764.1, df = 519, p-value = 1.197e-11
##
## Model df: 1.   Total lags used: 520
```

```
shapiro.test(model_3$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_3$residuals
## W = 0.86267, p-value < 2.2e-16
```

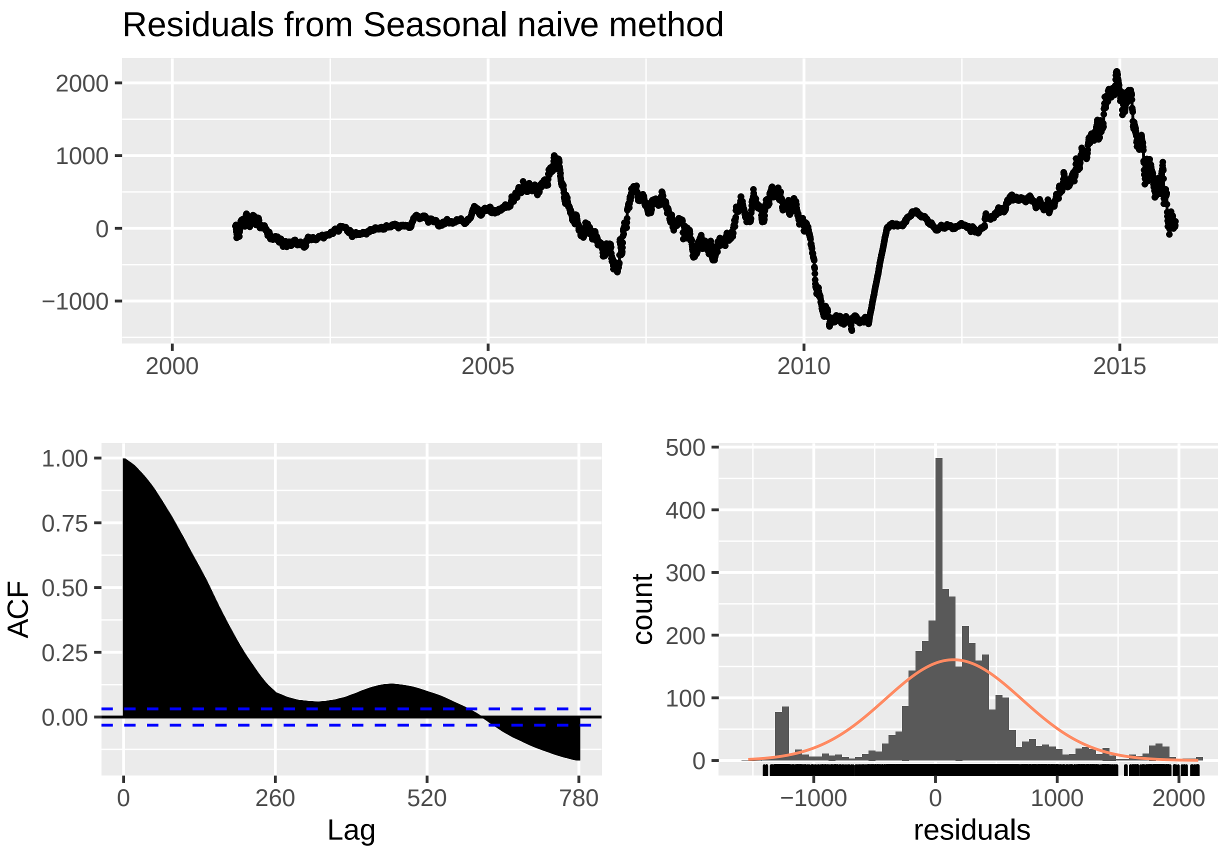
```
qqnorm(model_3$residuals)
```



pvalue of Ljung-Box test ≤ 0.05 hence we reject the null hypothesis therefore there is no co-relation in the residuals therefore it is not stationary.pvalue of Shapiro-Wilk test ≤ 0.05 hence we reject the null hypothesis therefore residuals not normally distributed

Residue analysis of model 4

```
checkresiduals(model_4)
```

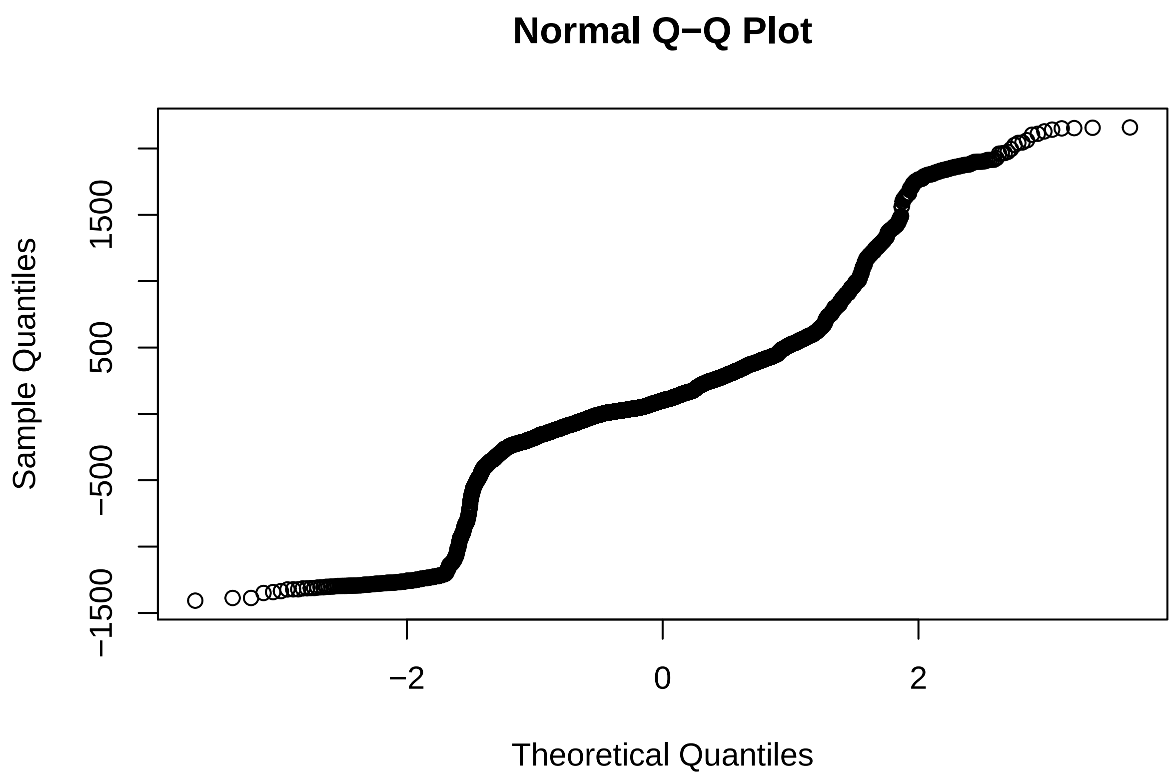



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 418620, df = 520, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 520
```

```
shapiro.test(model_4$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_4$residuals
## W = 0.90219, p-value < 2.2e-16
```

```
qqnorm(model_4$residuals)
```



pvalue of Ljung-Box test ≤ 0.05 hence we reject the null hypothesis therefore there is no co-relation in the residuals therefore it is not stationary.pvalue of Shapiro-Wilk test ≤ 0.05 hence we reject the null hypothesis therefore residuals not normally distributed

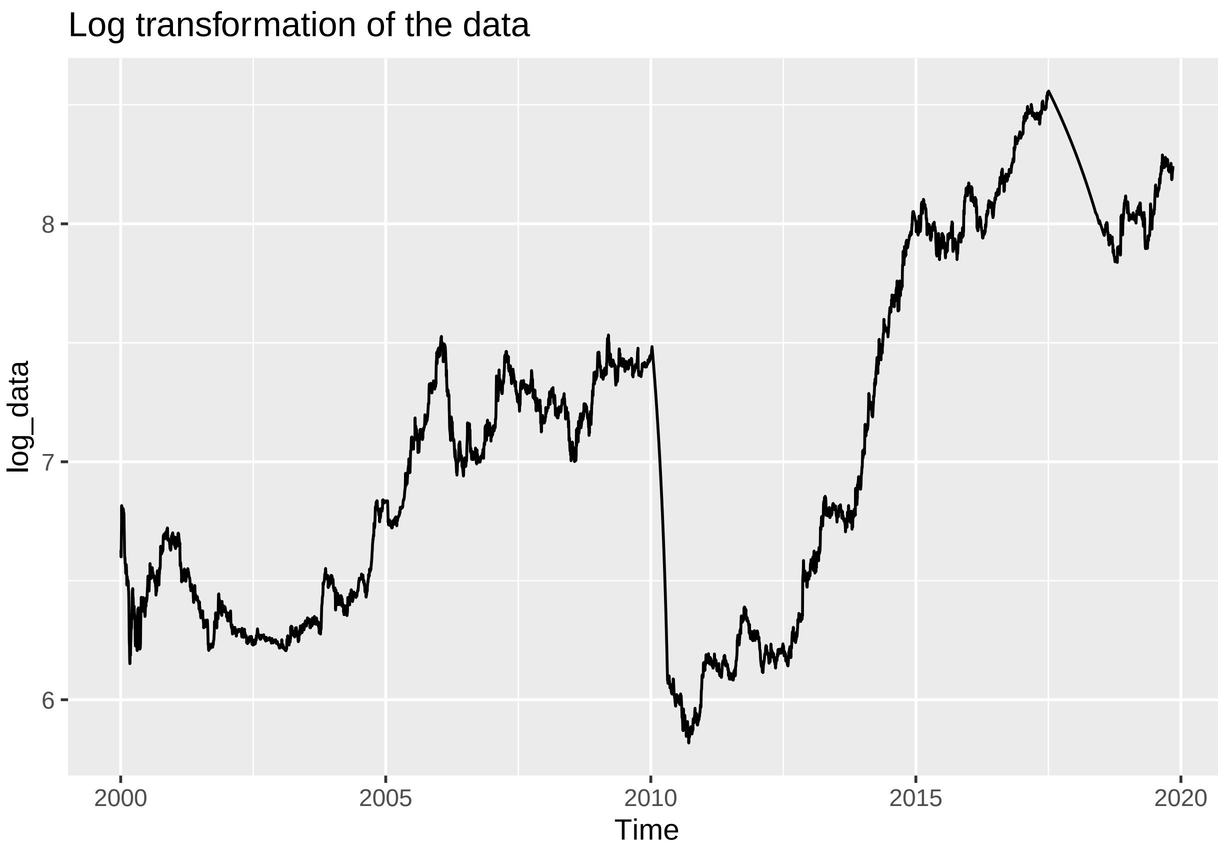
Stationarity of the series

We will use tranformation

```
log_data = log(my_time)
lambda = BoxCox.lambda(my_time)
Box_data = BoxCox(my_time,lambda = lambda)
```

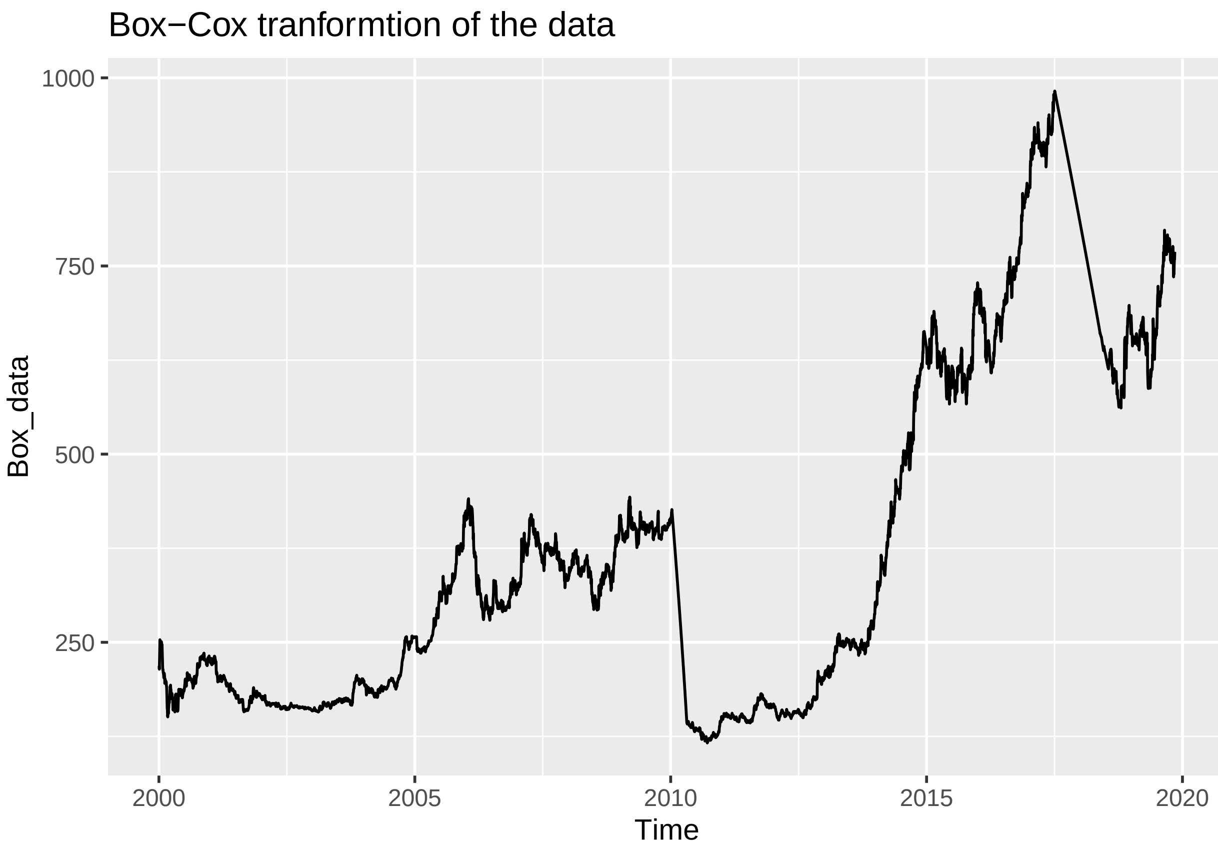
```
autoplot(log_data)+ggtitle("Log transformation of the data")
```

```
## Warning in is.na(main): is.na() applied to non-(list or vector) of type 'NULL'
```



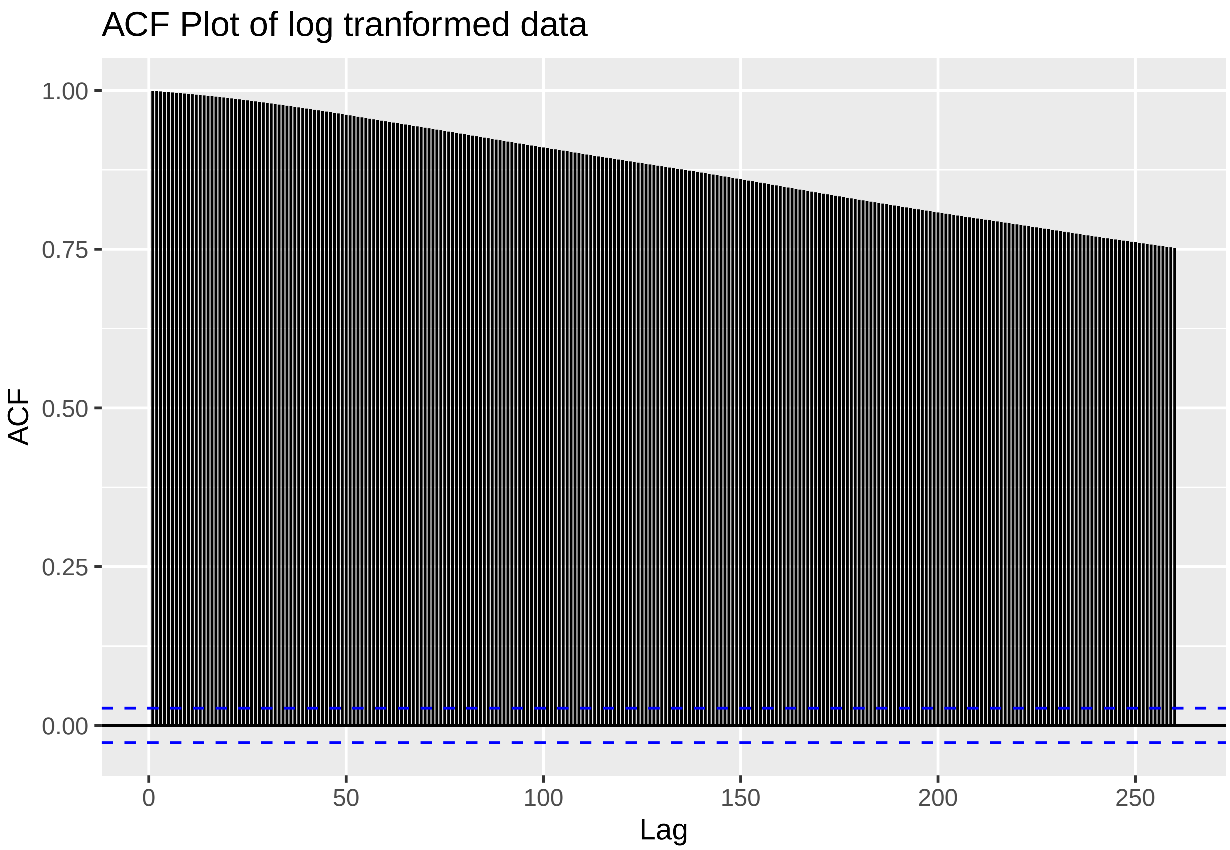

```r
autoplot(Box_data)+ggtitle("Box-Cox tranformtion of the data")
```

```
## Warning in is.na(main): is.na() applied to non-(list or vector) of type 'NULL'
```

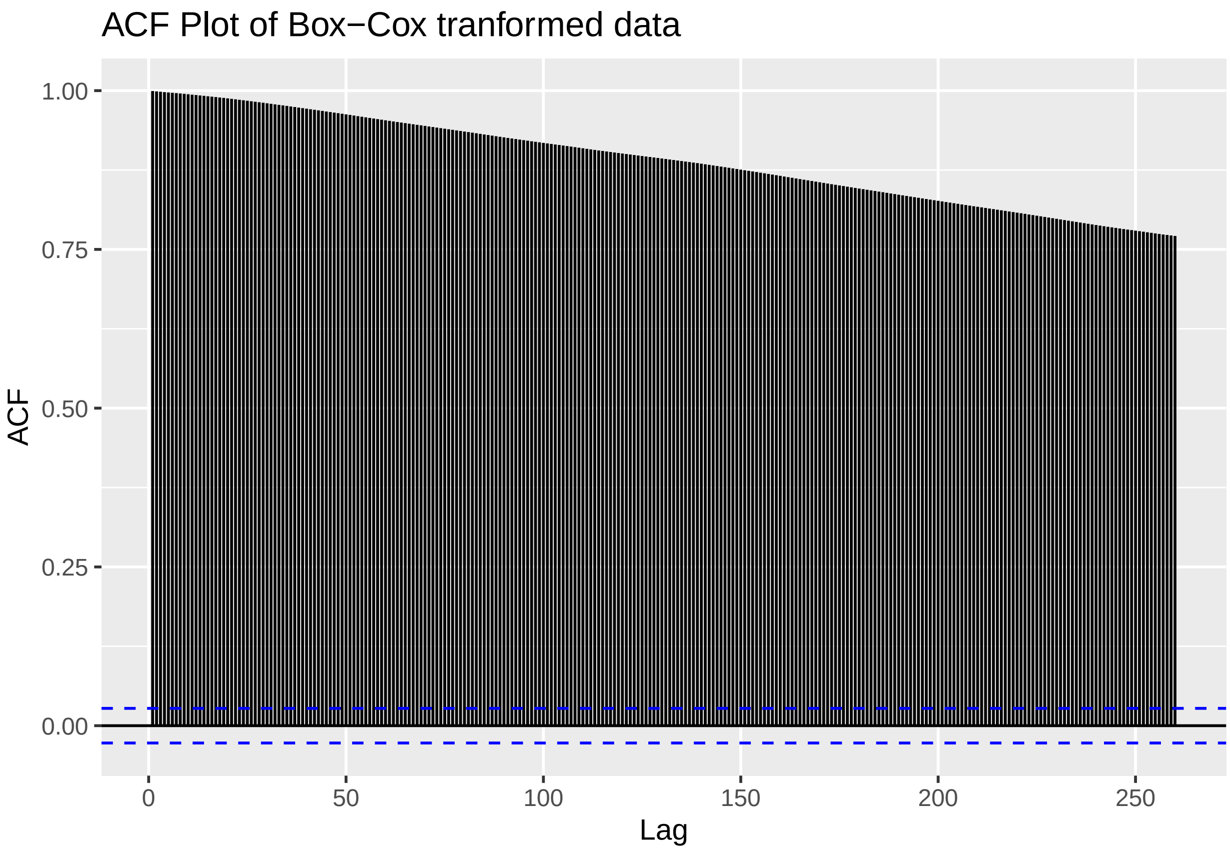


Now we will check the acf plot of tranformed data

```
ggAcf(log_data,lag.max = 260)+ggtitle("ACF Plot of log tranformed data")
```




```
ggAcf(Box_data,lag.max = 260)+ggtitle("ACF Plot of Box-Cox tranformed data")
```



Tranformed data doesn't seem to work well

Decomposing the data

The following two structures are considered for basic decomposition models:

Additive: = Trend + Seasonal + Random

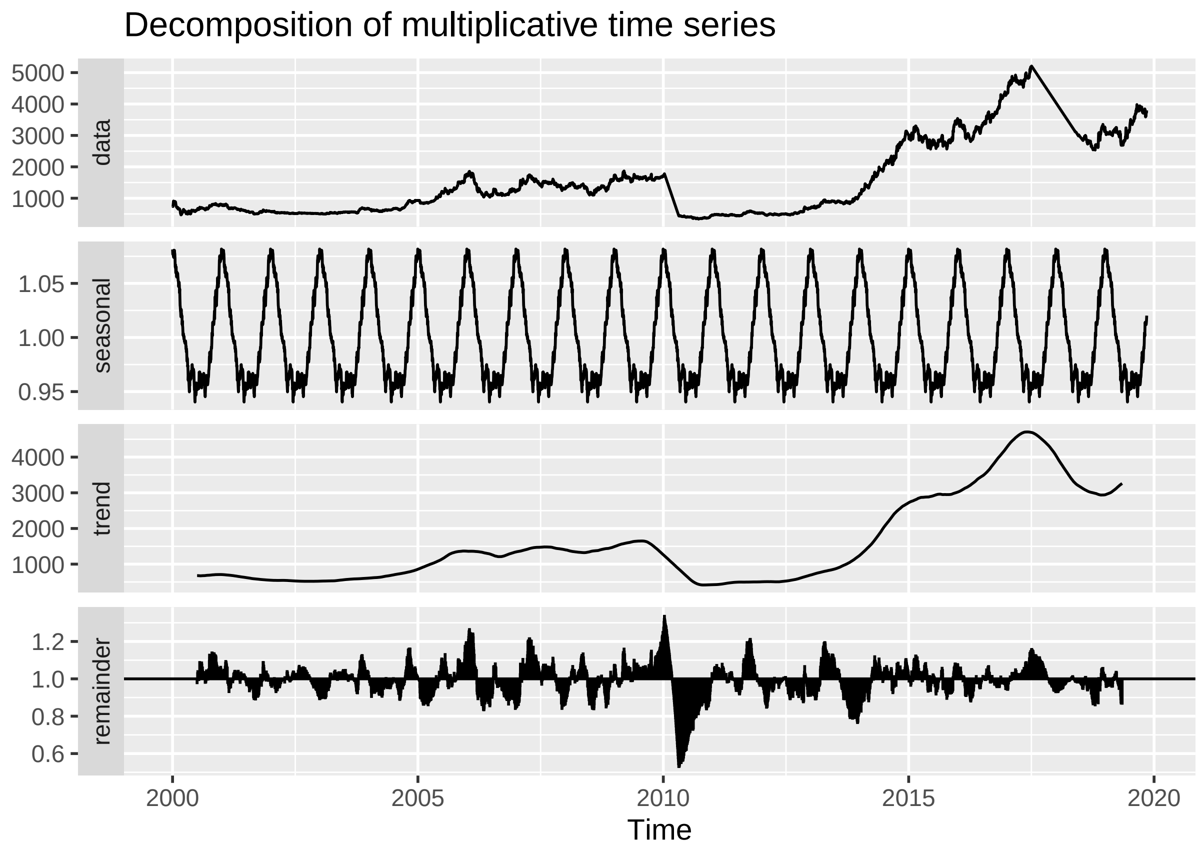
Multiplicative: = Trend * Seasonal * Random

How to Choose Between Additive and Multiplicative Decompositions

The additive model is useful when the seasonal variation is relatively constant over time.

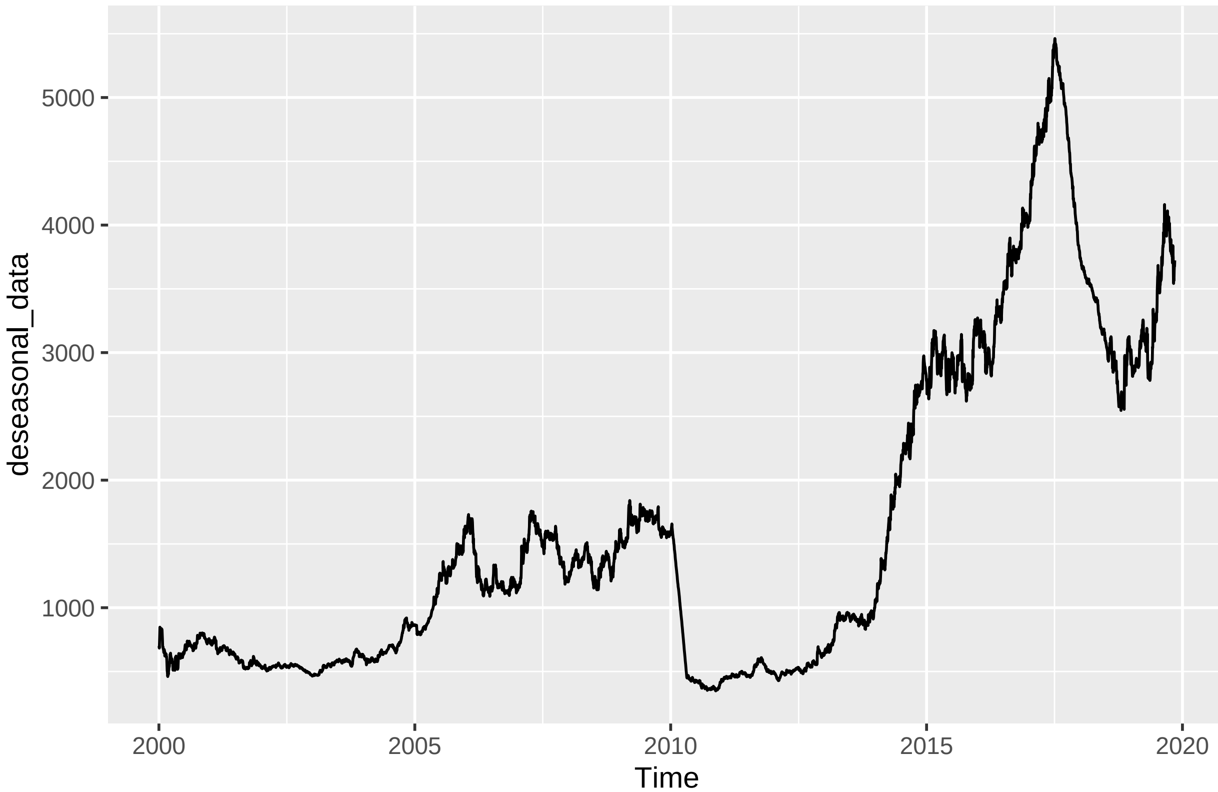
The multiplicative model is useful when the seasonal variation increases over time.

```
decomp_data = decompose(my_time,type = "multiplicative")
autoplot(decomp_data)
```




```r
deseasonal_data = my_time/decomp_data$seasonal
autoplot(deseasonal_data)
```

```
## Warning in is.na(main): is.na() applied to non-(list or vector) of type 'NULL'
```



Differencing ideas

lets try for differincing methods for the deseasonal data

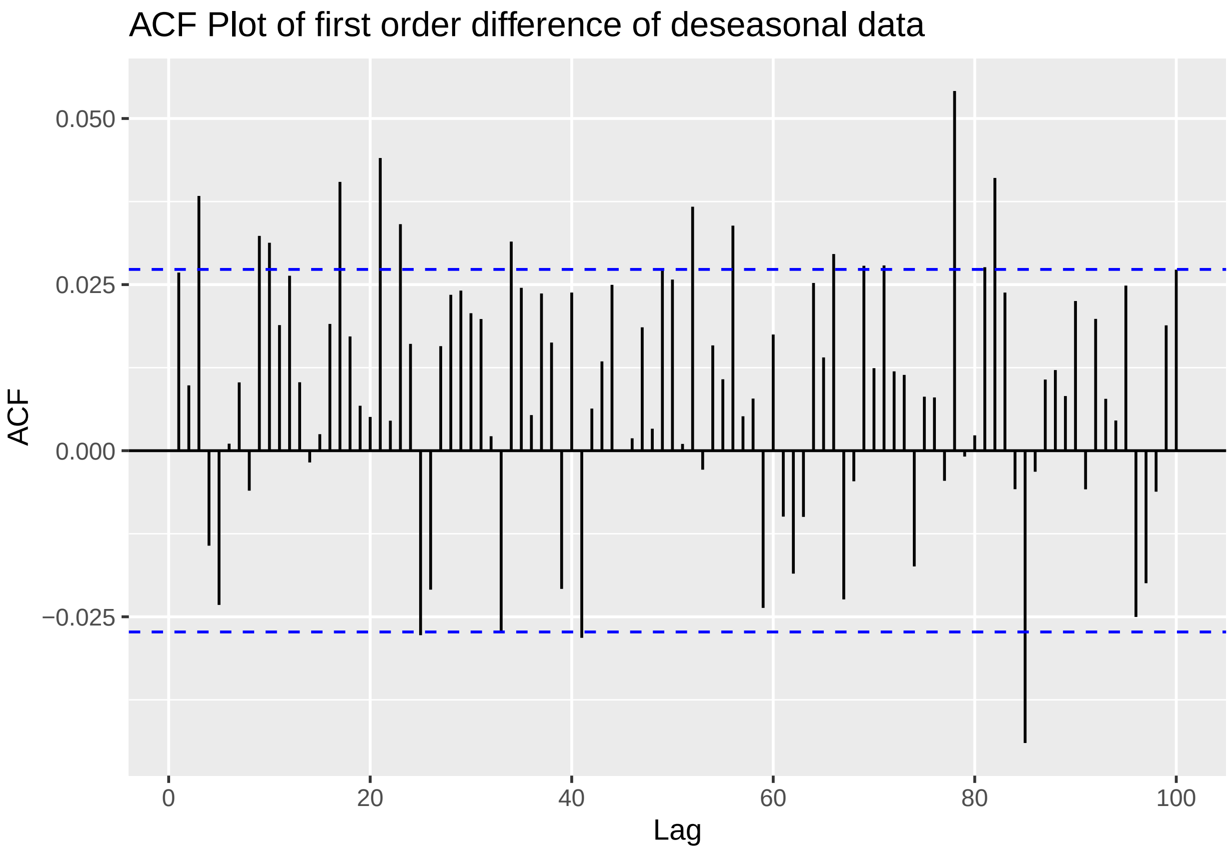
```
ndiffs(deseasonal_data,test = "kpss")
```

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

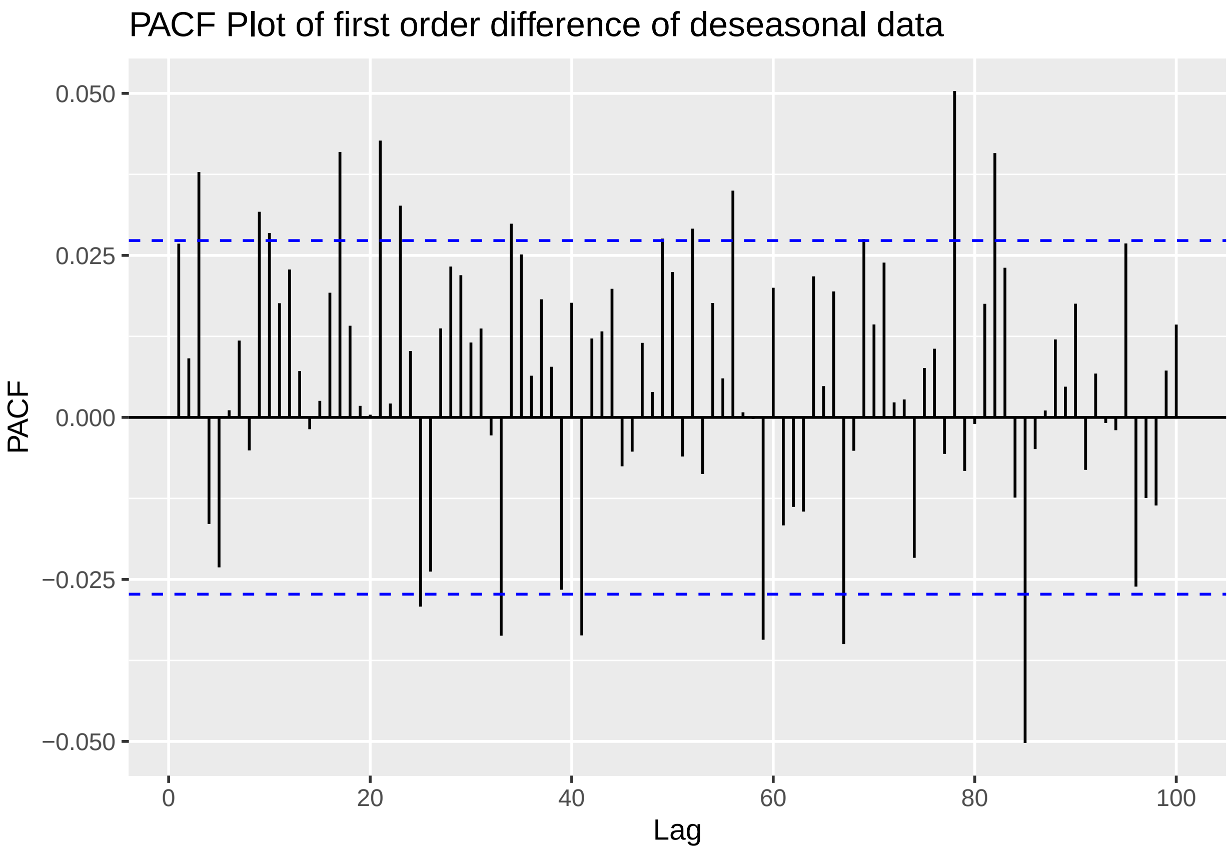
```
# using kpss test to find number of differencing required for the data
first_order = diff(deseasonal_data,1)
```

Now we will look onto the acf plots of diffrenced data

```
ggAcf(first_order,lag.max = 100)+ggtitle("ACF Plot of first order difference of deseasonal data")
```




```
ggPacf(first_order,lag.max = 100)+ggtitle("PACF Plot of first order difference of deseasonal data")
```



We observe significant difference but still we are not sure whether the series is stationry or not.
We will use ADF test to determine whether our diffrenced series is statinary or not

```
adf.test(first_order,alternative = "stationary")
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  first_order
## Dickey-Fuller = -14.669, Lag order = 17, p-value = 0.01
## alternative hypothesis: stationary
```

p value is less than 0.05 hence hence series is stationary

ARIMA model

From PACF plot of first order we see suggestive AR(3) and AR(9) so our intial models will ARIMA(3,1,0) ARIMA(9,1,0) as differenced lag is 1

```
fit_1 = Arima(deseasonal_data,order = c(3,1,0))
fit_1
```

```
## Series: deseasonal_data
## ARIMA(3,1,0)
##
```

```
## Coefficients:
##          ar1     ar2     ar3
##       0.0267  0.0085  0.0383
## s.e.  0.0139  0.0139  0.0139
##
## sigma^2 estimated as 723.2:  log likelihood=-24310.87
## AIC=48629.75   AICc=48629.75   BIC=48655.94
```

```r
fit_2 = Arima(deseasonal_data,order = c(9,1,0))
fit_2
```

```
## Series: deseasonal_data
## ARIMA(9,1,0)
##
## Coefficients:
##          ar1     ar2     ar3      ar4      ar5    ar6     ar7      ar8     ar9
##       0.0271  0.0094  0.0389  -0.0153  -0.0224  0.000  0.0121  -0.0055  0.0326
## s.e.  0.0139  0.0139  0.0139   0.0139   0.0140  0.014  0.0140   0.0140  0.0140
##
## sigma^2 estimated as 722.6:  log likelihood=-24305.71
## AIC=48631.42   AICc=48631.46   BIC=48696.91
```

```r
fit_3 = Arima(deseasonal_data,order = c(3,1,1))
fit_3
```

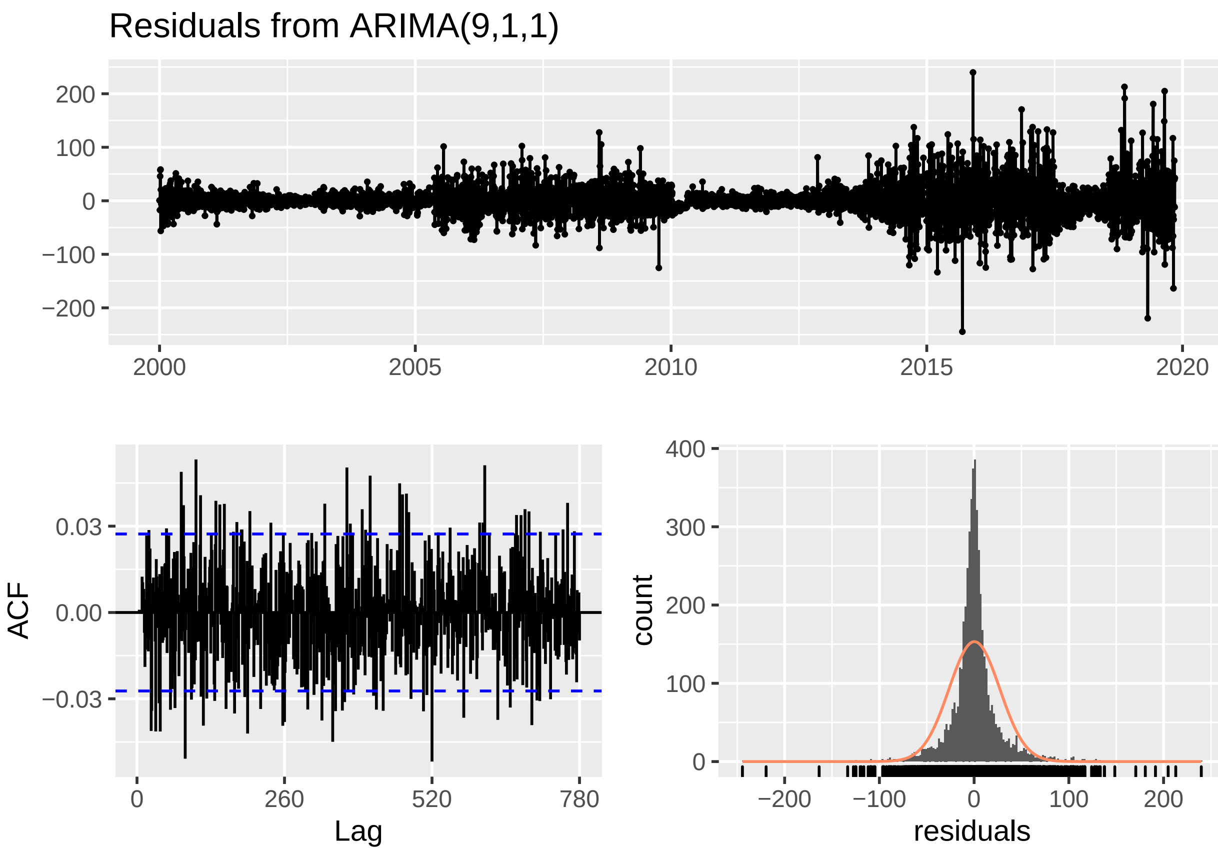
```
## Series: deseasonal_data
## ARIMA(3,1,1)
##
## Coefficients:
##           ar1     ar2     ar3     ma1
##       -0.1585  0.0135  0.0409  0.1855
## s.e.   0.2387  0.0155  0.0139  0.2387
##
## sigma^2 estimated as 723.3:  log likelihood=-24310.57
## AIC=48631.14   AICc=48631.16   BIC=48663.89
```

```r
fit_4 = Arima(deseasonal_data,order = c(9,1,1))
fit_4
```

```
## Series: deseasonal_data
## ARIMA(9,1,1)
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ar5     ar6     ar7      ar8
##       0.9911  -0.0164  0.0294  -0.0532  -0.0072  0.0225  0.0110  -0.0172
## s.e.  0.0188   0.0196  0.0196   0.0196   0.0196  0.0196  0.0196   0.0197
##          ar9      ma1
##       0.0240  -0.9696
## s.e.  0.0145   0.0127
##
## sigma^2 estimated as 719.4:  log likelihood=-24293.71
## AIC=48609.42   AICc=48609.47   BIC=48681.46
```

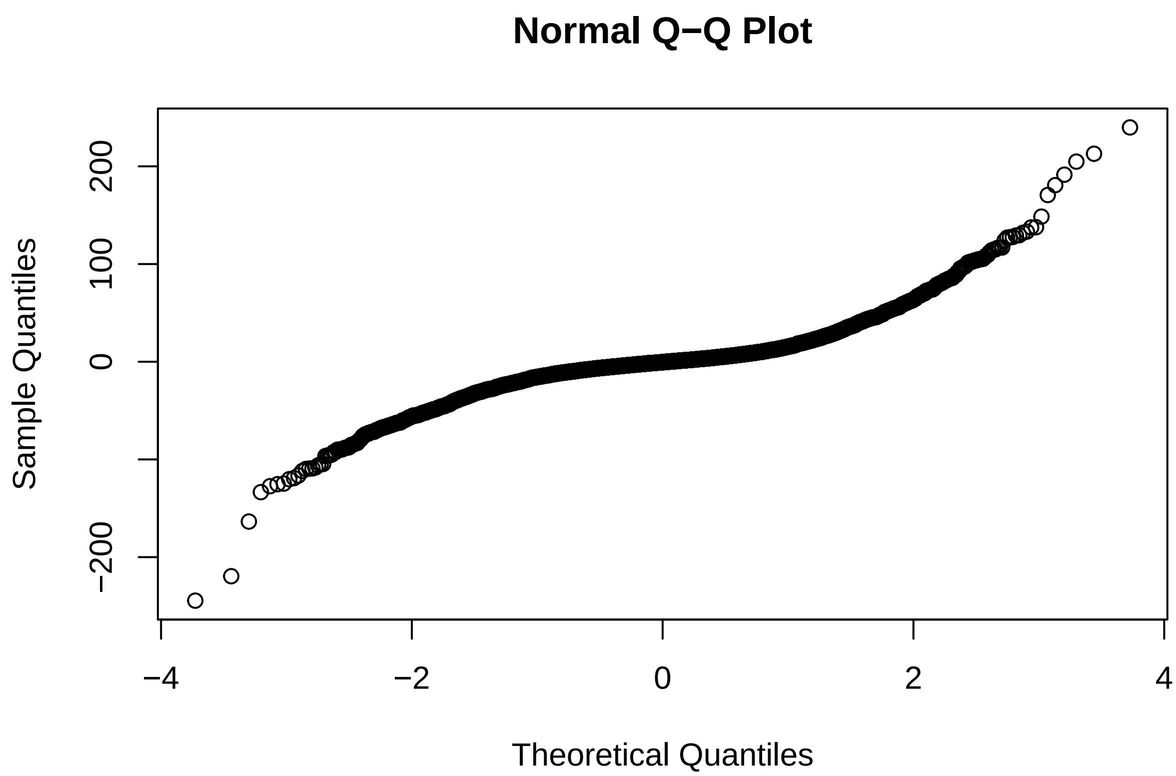
From the 4 models we AIC of model 4 seem to be less compared to other

```r
checkresiduals(fit_4)
```



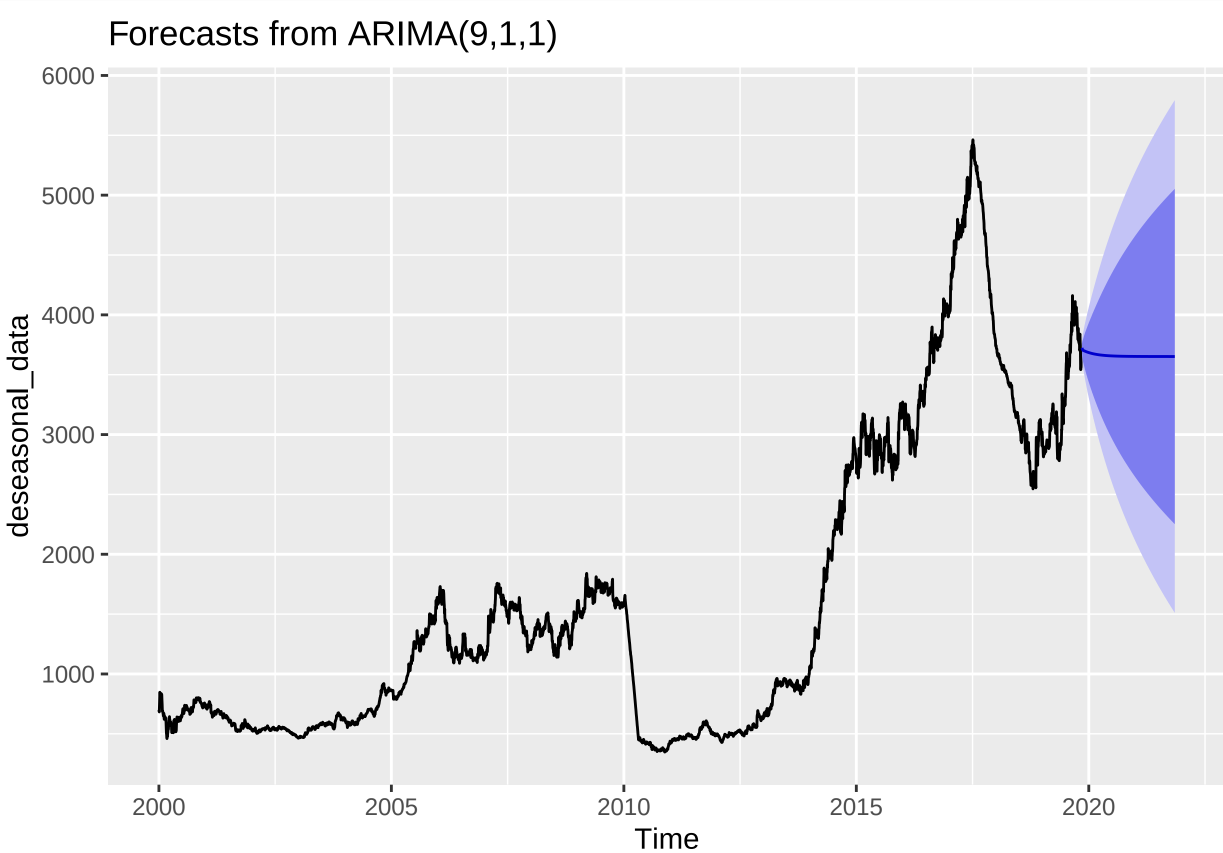

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(9,1,1)
## Q* = 902.93, df = 510, p-value < 2.2e-16
##
## Model df: 10.   Total lags used: 520
```

```r
qqnorm(fit_4$residuals)
```




```r
# normality of the ARIMA(9,1,1) is not acheived
```

```r
autoplot(forecast(fit_4)) # forecast using ARIMA(9,1,1)
```



```r
fi = auto.arima(deseasonal_data,seasonal = FALSE)
fi
```

```
## Series: deseasonal_data
## ARIMA(0,1,1) with drift
##
## Coefficients:
##          ma1   drift
##       0.0264  0.5857
## s.e.  0.0138  0.3843
##
## sigma^2 estimated as 723.9:  log likelihood=-24313.77
## AIC=48633.54   AICc=48633.54   BIC=48653.19
```