Final Project R File

Jane

07/04/2021

Forcasting Exchange Rate

Reading Canadian and Japanes Currency into r

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
CanJapCurrency <- readxl::read_xlsx ("CADJPY_Candlestick_1_D_BID_01.01.2000-31.12.2020.xlsx")%>%
  select('Gmt time', Close)%>%
  rename(Date = ('Gmt time'), Rate = ("Close"))

head(CanJapCurrency)
```

```
## # A tibble: 6 x 2
##   Date                 Rate
##   <dttm>              <dbl>
## 1 2000-01-03 00:00:00  70.1
## 2 2000-01-04 00:00:00  71.0
## 3 2000-01-05 00:00:00  71.9
## 4 2000-01-06 00:00:00  72.1
## 5 2000-01-07 00:00:00  72.3
## 6 2000-01-10 00:00:00  72.2
```

Conversion of Gmt time to date format

```r
library(dplyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
CanJapCurrency$Date <- lubridate::ymd(CanJapCurrency$Date)
head(CanJapCurrency)
```

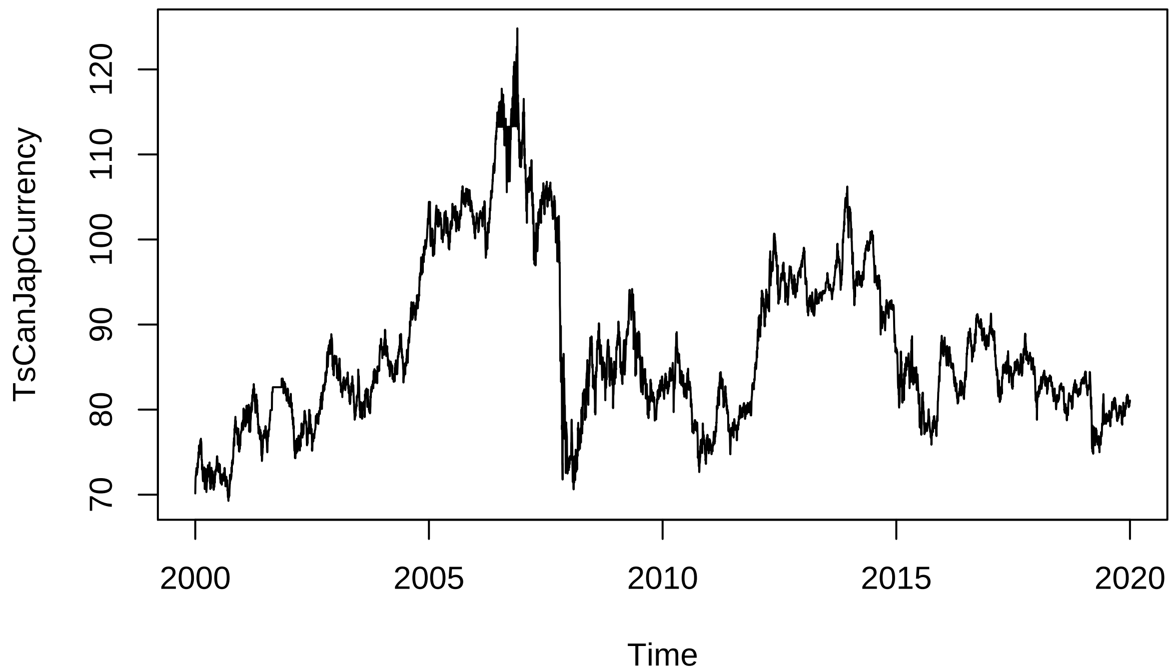
```
## # A tibble: 6 x 2
##   Date        Rate
##   <date>     <dbl>
## 1 2000-01-03  70.1
## 2 2000-01-04  71.0
## 3 2000-01-05  71.9
## 4 2000-01-06  72.1
## 5 2000-01-07  72.3
## 6 2000-01-10  72.2
```

##Converting to Time Series and Ploting a graph to view the dataset

```r
library(tseries)
```

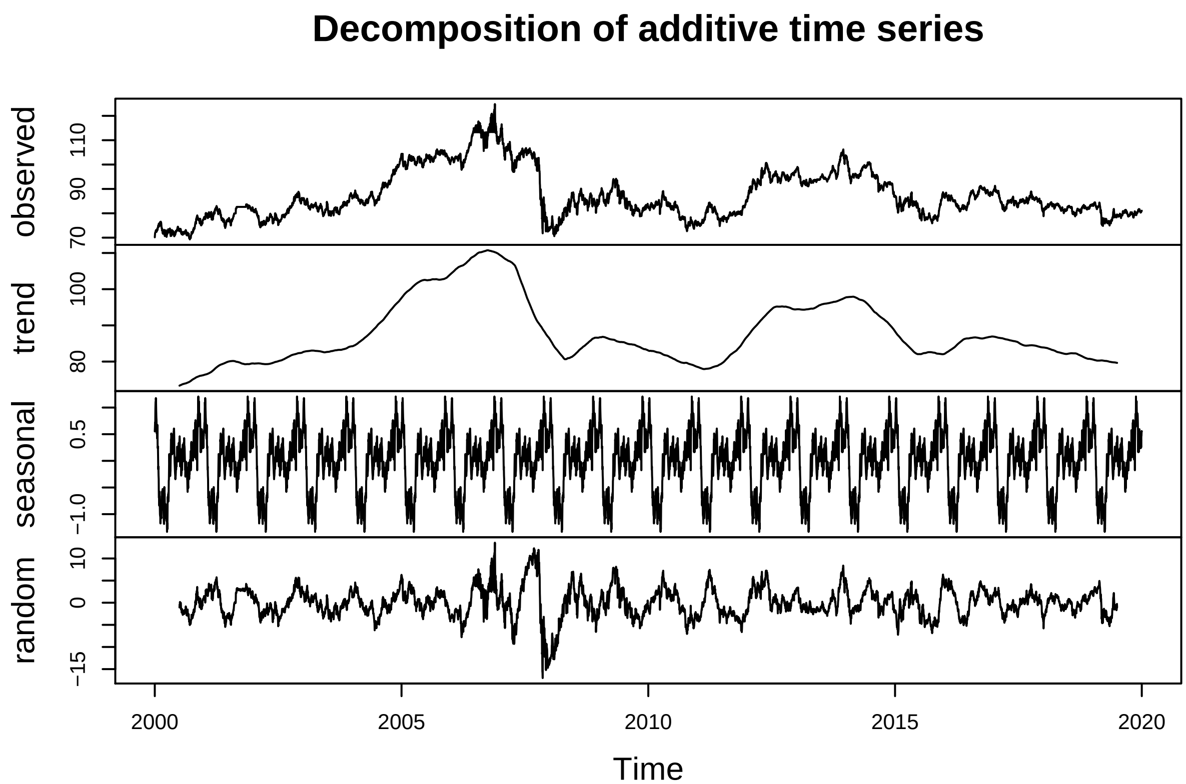
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
TsCanJapCurrency = ts( CanJapCurrency$Rate, frequency =314, start = c(2000,01,03))

plot(TsCanJapCurrency)
```



Finding the component of the Time Series

```
ComponentCanJapCurrency <- decompose(TsCanJapCurrency)
plot(ComponentCanJapCurrency)
```

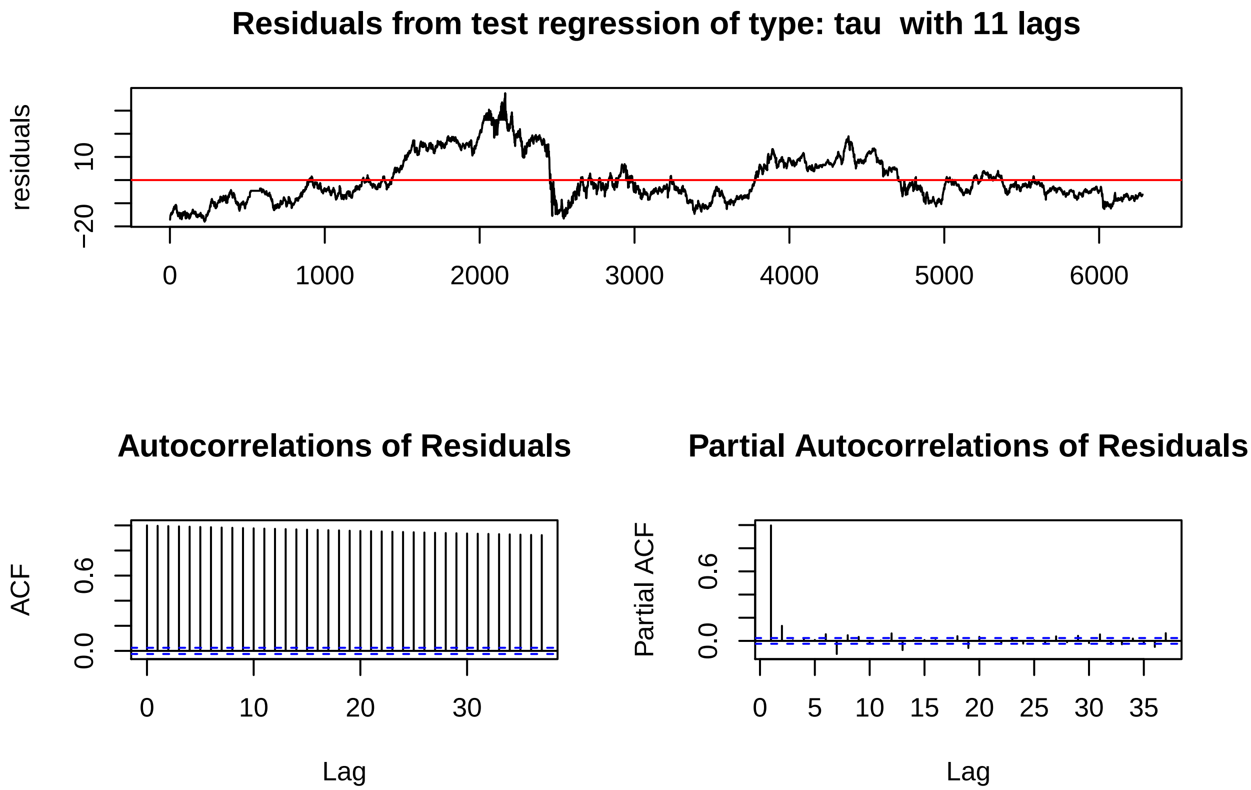


To To achieve stationarity by differencing the data – compute the differences between consecutive observations

```r
library("fUnitRoots")
```

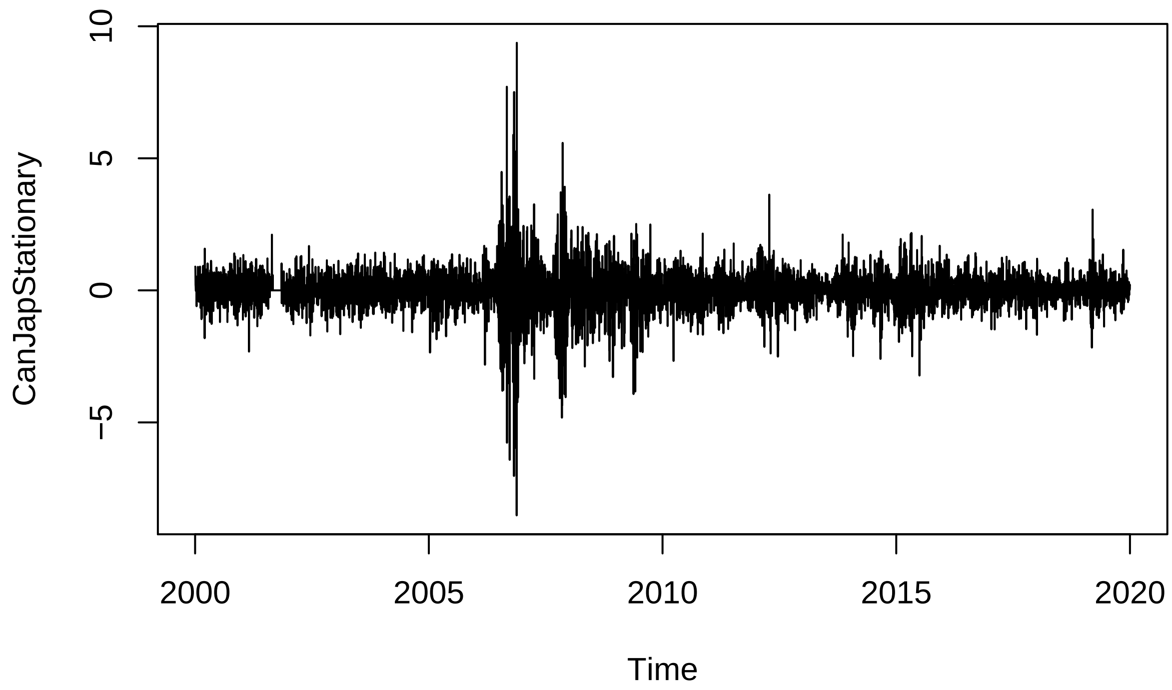
```
## Warning: package 'fUnitRoots' was built under R version 4.0.5
## Loading required package: timeDate
## Warning: package 'timeDate' was built under R version 4.0.4
## Loading required package: timeSeries
## Warning: package 'timeSeries' was built under R version 4.0.5
## Loading required package: fBasics
## Warning: package 'fBasics' was built under R version 4.0.5
```

```r
urkpssTest(TsCanJapCurrency, type = c("tau"), lags = c("short"),use.lag = NULL, doplot = TRUE)
```



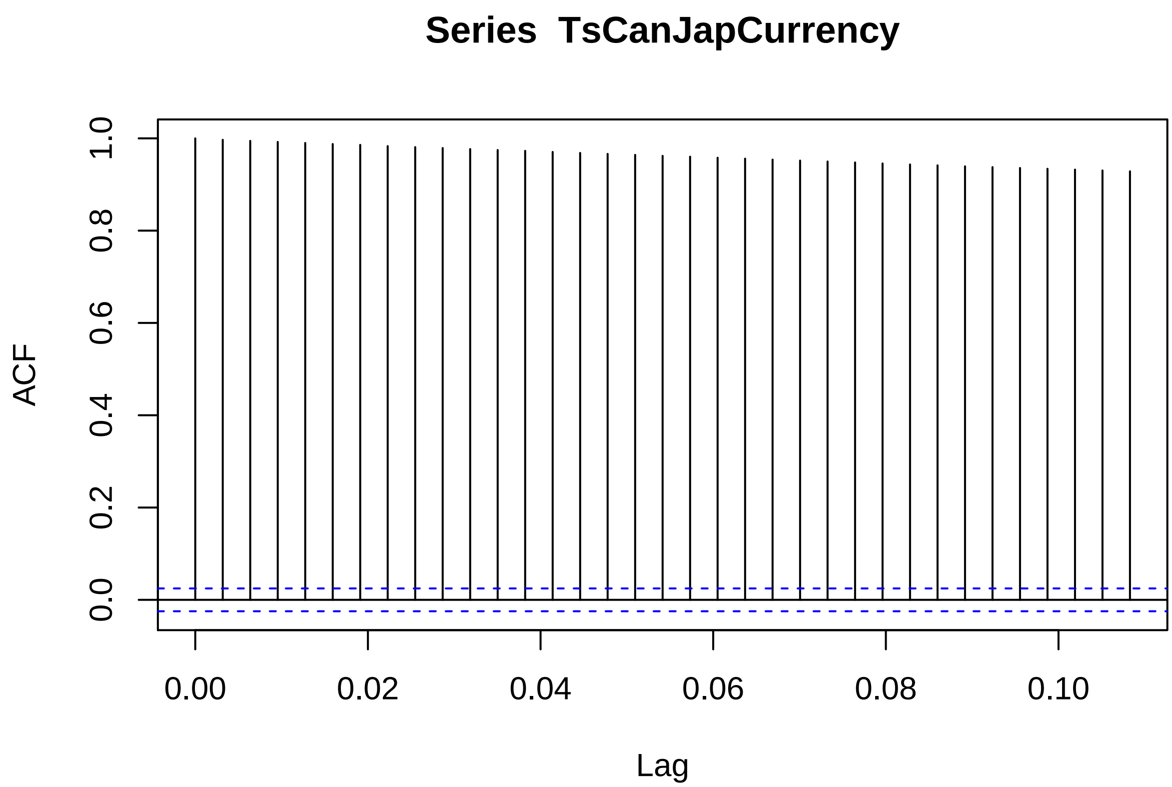

```
##
## Title:
##  KPSS Unit Root Test
##
## Test Results:
##   NA
##
## Description:
##  Mon May 03 23:23:45 2021 by user: janeo
```

```r
CanJapStationary= diff(TsCanJapCurrency, differences=1)
plot(CanJapStationary)
```

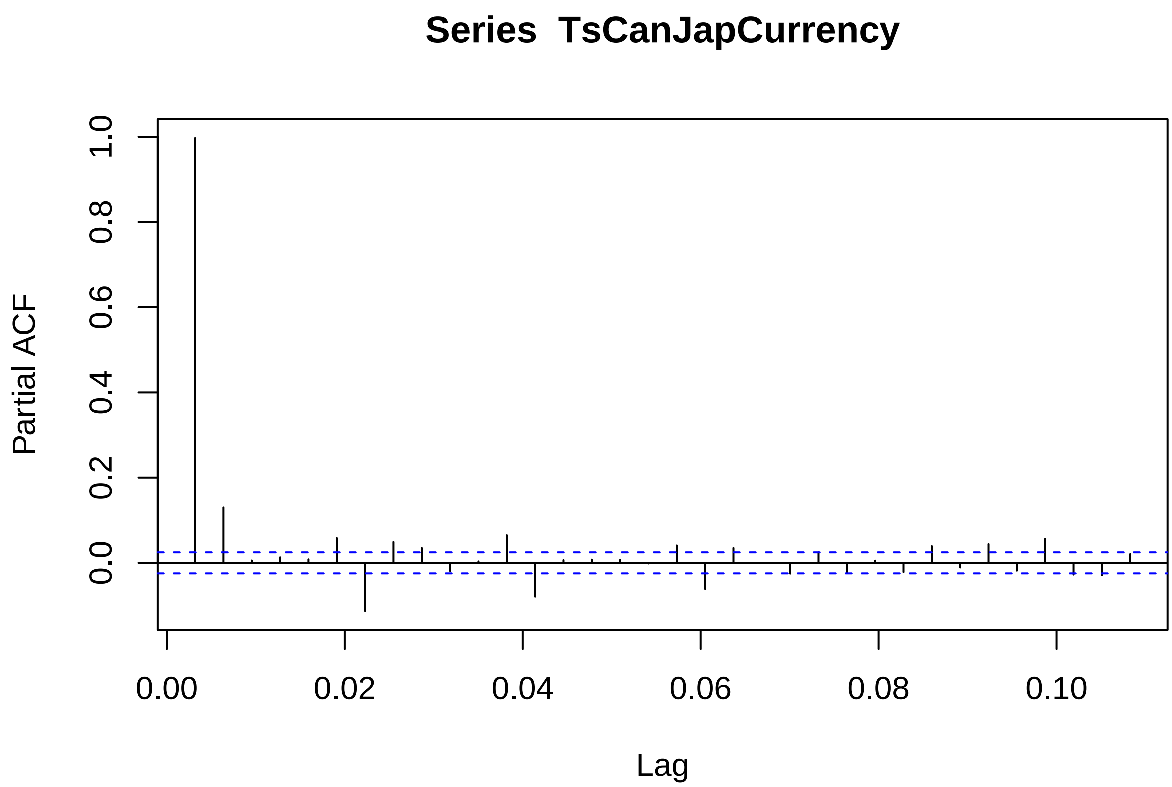


Calculating Autocorrlation function and partil autocorlation function

```
acf(TsCanJapCurrency,lag.max=34)
```

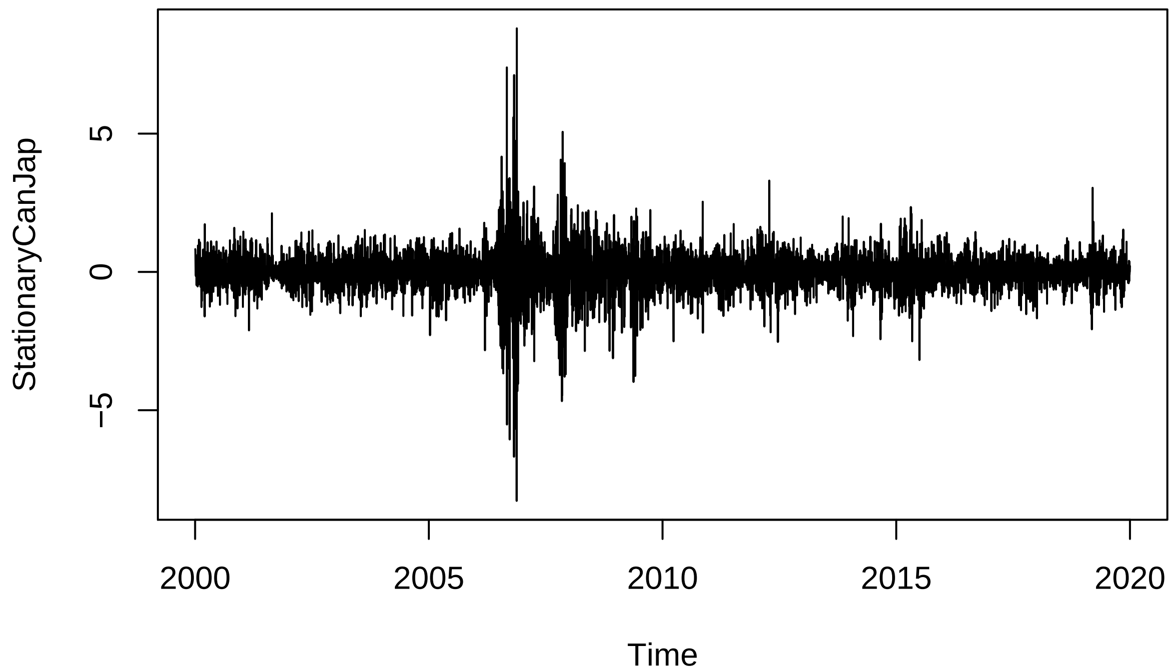



```
pacf(TsCanJapCurrency, lag.max = 34)
```



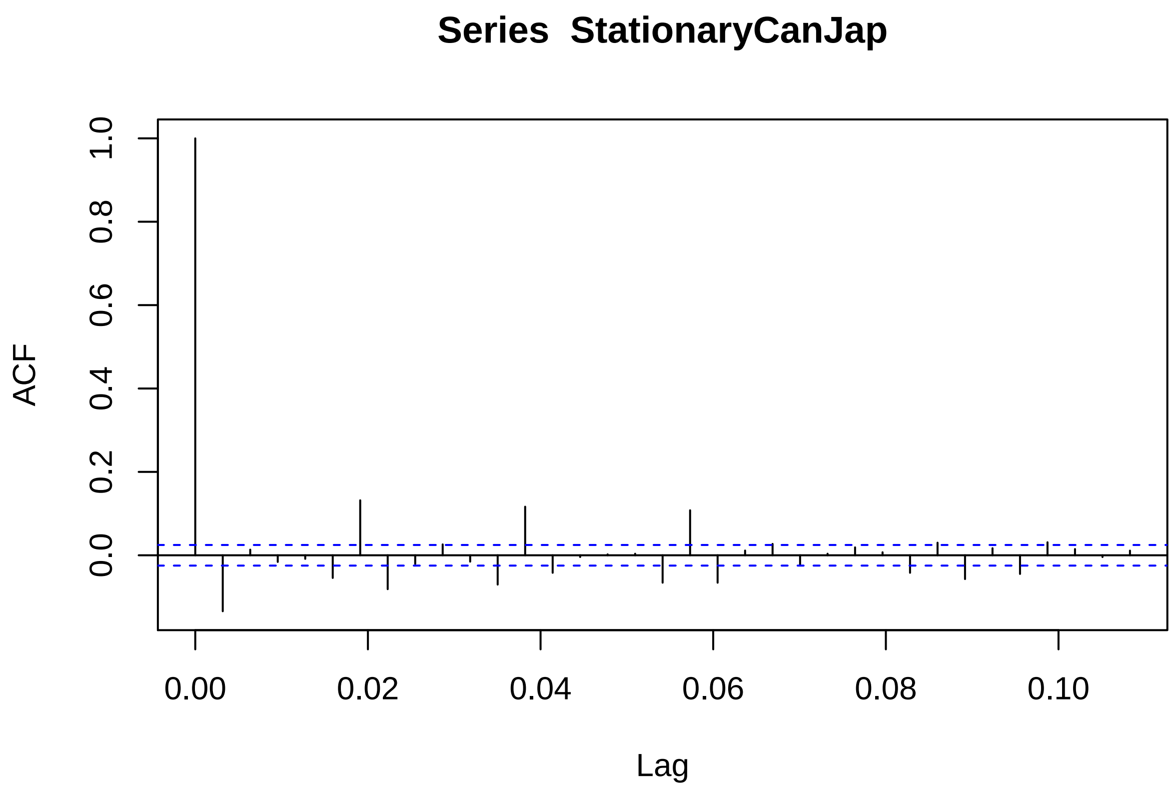
Adjusting and ensuring there are no seasonality

```
TSseasonallyadjustedCanJap <- TsCanJapCurrency- ComponentCanJapCurrency$seasonal
StationaryCanJap <- diff(TSseasonallyadjustedCanJap, differences=1)
plot(StationaryCanJap)
```

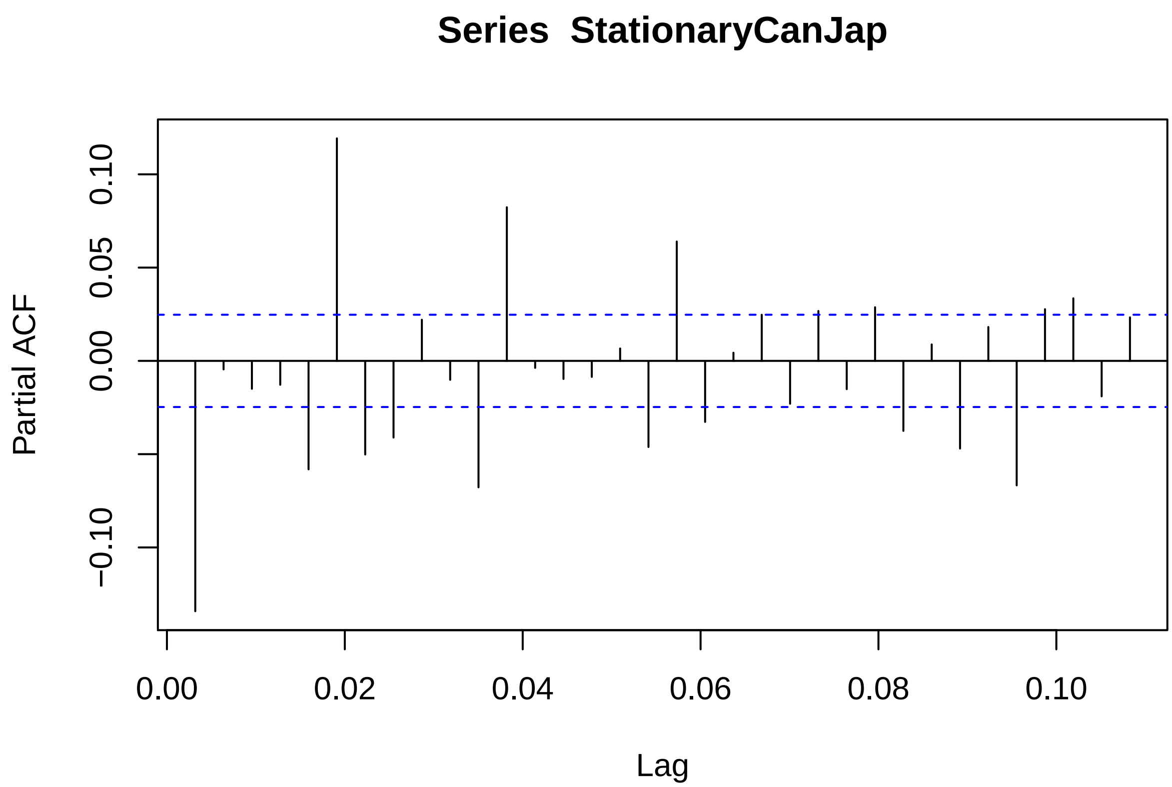


Calculating again for ACF and PACF after finding stationality

```
acf(StationaryCanJap, lag.max=34)
```




```
pacf(StationaryCanJap, lag.max=34)
```



Fitting The ARIMA Model

ARIMA fitting (1,1,0)

```
fitArima1CanJap <- arima(TsCanJapCurrency, order =  c(1,1,0), include.mean = TRUE)
fitArima1CanJap
```

```
##
## Call:
## arima(x = TsCanJapCurrency, order = c(1, 1, 0), include.mean = TRUE)
##
## Coefficients:
##           ar1
##       -0.1399
## s.e.   0.0125
##
## sigma^2 estimated as 0.5455:  log likelihood = -7007.94,  aic = 14019.89
```

##Arima Fitting (1,1,1)

```
fitArima2CanJap <- arima(TsCanJapCurrency, order =  c(1,1,1), include.mean = TRUE)
fitArima2CanJap
```

```
##
## Call:
## arima(x = TsCanJapCurrency, order = c(1, 1, 1), include.mean = TRUE)
##
## Coefficients:
##           ar1      ma1
##       -0.1024  -0.0383
## s.e.   0.1007   0.1014
##
## sigma^2 estimated as 0.5455:  log likelihood = -7007.88,  aic = 14021.77
```

Arima Fitting (2,1,1)

```
fitArima3CanJap <- arima(TsCanJapCurrency, order = c(2,1,1), include.mean = TRUE)
fitArima3CanJap
```

```
##
## Call:
## arima(x = TsCanJapCurrency, order = c(2, 1, 1), include.mean = TRUE)
##
## Coefficients:
##          ar1     ar2      ma1
##       0.6581  0.1010  -0.7998
## s.e.  0.2656  0.0439   0.2650
##
## sigma^2 estimated as 0.5453:  log likelihood = -7006.92,  aic = 14021.85
```

##Fitting Arima (0,1,3)

```
FitArima4CanJap <- arima(TsCanJapCurrency, order = c(0,1,3), include.mean = TRUE)
FitArima4CanJap
```

```
##
## Call:
## arima(x = TsCanJapCurrency, order = c(0, 1, 3), include.mean = TRUE)
##
## Coefficients:
##           ma1     ma2      ma3
##       -0.1413  0.0133  -0.0156
## s.e.   0.0126  0.0130   0.0115
##
## sigma^2 estimated as 0.5454:  log likelihood = -7007.09,  aic = 14022.18
```

##Best possible model is selected by AIC scores of the models

```
library(dLagM)
```

```
## Warning: package 'dLagM' was built under R version 4.0.5
```

```
## Loading required package: nardl
```

```
## Warning: package 'nardl' was built under R version 4.0.5

## Loading required package: dynlm

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following object is masked from 'package:timeSeries':
##
##     time<-

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
ARIMAModelSelection = AIC(fitArima1CanJap,fitArima2CanJap,fitArima3CanJap,FitArima4CanJap)
sortScore(ARIMAModelSelection, score ="aic")
```

```
##                 df      AIC
## fitArima1CanJap  2 14019.89
## fitArima2CanJap  3 14021.77
## fitArima3CanJap  4 14021.85
## FitArima4CanJap  4 14022.18
```

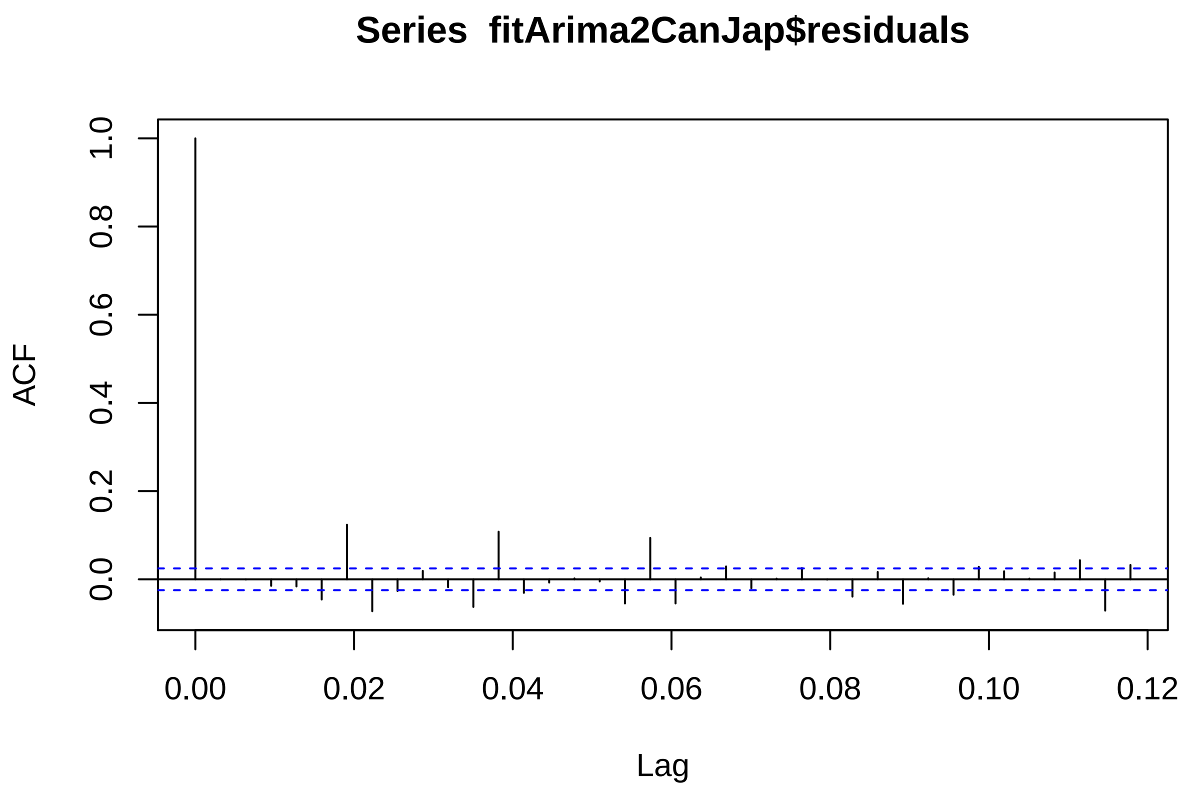
Base on the above the fitArima1CanJap is selected

```
confint(fitArima2CanJap)
```

```
##          2.5 %    97.5 %
## ar1 -0.2998151 0.0950947
## ma1 -0.2369821 0.1604028
```

Runing code to obtain Box Test Rest

```
acf(fitArima2CanJap$residuals)
```



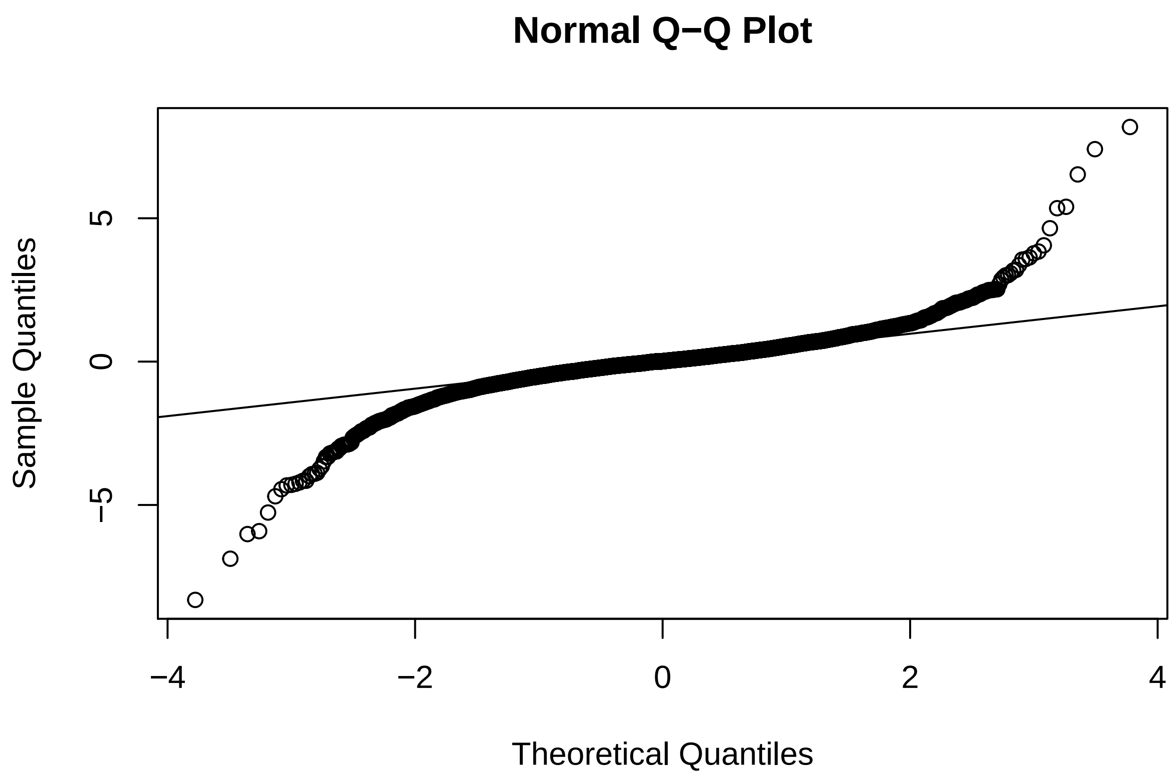

```r
library(FitAR)
```

```
## Warning: package 'FitAR' was built under R version 4.0.5

## Loading required package: lattice

## Loading required package: leaps

## Loading required package: ltsa

## Loading required package: bestglm

## Warning: package 'bestglm' was built under R version 4.0.5
```

```r
library(bestglm)
 Box.test(resid(fitArima2CanJap),type="Ljung",lag=20,fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  resid(fitArima2CanJap)
## X-squared = 352.48, df = 19, p-value < 2.2e-16
```

```r
qqnorm(fitArima2CanJap$residuals)
qqline(fitArima2CanJap$residuals)
```



Using Auto.arima to find the best model fit

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.0.5
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:FitAR':
##
##     BoxCox
```

```
## The following object is masked from 'package:dLagM':
##
##     forecast
```

```r
auto.arima(TsCanJapCurrency, trace=TRUE)
```

```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,1,2)(1,0,1)[314] with drift         : Inf
##  ARIMA(0,1,0)             with drift         : 14144.65
##  ARIMA(1,1,0)(1,0,0)[314] with drift         : 14196.37
##  ARIMA(0,1,1)(0,0,1)[314] with drift         : Inf
##  ARIMA(0,1,0)                                : 14142.68
##  ARIMA(0,1,0)(1,0,0)[314] with drift         : Inf
##  ARIMA(0,1,0)(0,0,1)[314] with drift         : Inf
##  ARIMA(0,1,0)(1,0,1)[314] with drift         : Inf
##  ARIMA(1,1,0)             with drift         : 14021.97
##  ARIMA(1,1,0)(0,0,1)[314] with drift         : Inf
##  ARIMA(1,1,0)(1,0,1)[314] with drift         : Inf
##  ARIMA(2,1,0)             with drift         : 14023.06
##  ARIMA(1,1,1)             with drift         : 14023.71
##  ARIMA(0,1,1)             with drift         : 14023.37
##  ARIMA(2,1,1)             with drift         : 14023.27
##  ARIMA(1,1,0)                                : 14020
##  ARIMA(1,1,0)(1,0,0)[314]                    : Inf
##  ARIMA(1,1,0)(0,0,1)[314]                    : Inf
##  ARIMA(1,1,0)(1,0,1)[314]                    : Inf
##  ARIMA(2,1,0)                                : 14021.09
##  ARIMA(1,1,1)                                : 14021.74
##  ARIMA(0,1,1)                                : 14021.41
##  ARIMA(2,1,1)                                : 14021.31
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(1,1,0)                                : 14019.89
##
##  Best model: ARIMA(1,1,0)
```

```
## Series: TsCanJapCurrency
## ARIMA(1,1,0)
##
## Coefficients:
##           ar1
##       -0.1399
## s.e.   0.0125
##
## sigma^2 estimated as 0.5456:  log likelihood=-7007.94
## AIC=14019.89   AICc=14019.89   BIC=14033.38
```

forecasting using Best model: ARIMA(1,1,0)

```r
forecastarimaCanJa<- predict(fitArima1CanJap,n.ahead = 100)
forecastarimaCanJa
```

```
## $pred
## Time Series:
## Start = c(2020, 2)
## End = c(2020, 101)
## Frequency = 314
##   [1] 81.04503 81.04908 81.04852 81.04860 81.04859 81.04859 81.04859 81.04859
##   [9] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [17] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [25] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [33] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [41] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [49] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [57] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [65] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [73] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [81] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [89] 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859 81.04859
##  [97] 81.04859 81.04859 81.04859 81.04859
##
## $se
## Time Series:
## Start = c(2020, 2)
## End = c(2020, 101)
## Frequency = 314
##   [1] 0.7385804 0.9741674 1.1709354 1.3381148 1.4867346 1.6217758 1.7464080
##   [8] 1.8627196 1.9721835 2.0758832 2.1746435 2.2691095 2.3597968 2.4471258
##  [15] 2.5314438 2.6130425 2.6921691 2.7690356 2.8438251 2.9166976 2.9877932
##  [22] 3.0572359 3.1251360 3.1915918 3.2566918 3.3205158 3.3831359 3.4446178
##  [29] 3.5050214 3.5644016 3.6228086 3.6802888 3.7368849 3.7926366 3.8475805
##  [36] 3.9017507 3.9551792 4.0078954 4.0599272 4.1113006 4.1620399 4.2121680
##  [43] 4.2617065 4.3106758 4.3590950 4.4069823 4.4543547 4.5012287 4.5476195
##  [50] 4.5935418 4.6390095 4.6840360 4.7286337 4.7728147 4.8165904 4.8599719
##  [57] 4.9029695 4.9455933 4.9878529 5.0297574 5.0713157 5.1125362 5.1534270
##  [64] 5.1939959 5.2342504 5.2741976 5.3138445 5.3531978 5.3922639 5.4310491
##  [71] 5.4695591 5.5078000 5.5457771 5.5834960 5.6209617 5.6581794 5.6951539
##  [78] 5.7318898 5.7683918 5.8046643 5.8407115 5.8765376 5.9121466 5.9475424
##  [85] 5.9827288 6.0177095 6.0524880 6.0870678 6.1214523 6.1556447 6.1896482
##  [92] 6.2234659 6.2571009 6.2905560 6.3238341 6.3569381 6.3898705 6.4226341
##  [99] 6.4552314 6.4876648
```

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