ARIMA Model EUR And CAD

Jane

01/05/2021

# Forcasting Exchange Rate Using ARIMA Model for EUR And CAD

## Reading EUR and CAD 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

EURCADARIMA<- read.csv ("EURCAD\_Candlestick\_1\_D\_BID\_01.01.2000-31.12.2020.csv")%>%  
 select('GMT.TIME', CLOSE)%>%  
 rename(Date = ('GMT.TIME'), RateEURCAD = ("CLOSE"))  
  
   
head(EURCADARIMA)

## Date RateEURCAD  
## 1 2000-01-03 1.4817  
## 2 2000-01-04 1.4969  
## 3 2000-01-05 1.4963  
## 4 2000-01-06 1.5064  
## 5 2000-01-07 1.4992  
## 6 2000-01-10 1.4928

## Conversion of Gmt time to date format

library(dplyr)  
library(lubridate)

##   
## Attaching package: 'lubridate'

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

EURCADARIMA$Date <- lubridate::ymd(EURCADARIMA$Date)  
head(EURCADARIMA)

## Date RateEURCAD  
## 1 2000-01-03 1.4817  
## 2 2000-01-04 1.4969  
## 3 2000-01-05 1.4963  
## 4 2000-01-06 1.5064  
## 5 2000-01-07 1.4992  
## 6 2000-01-10 1.4928

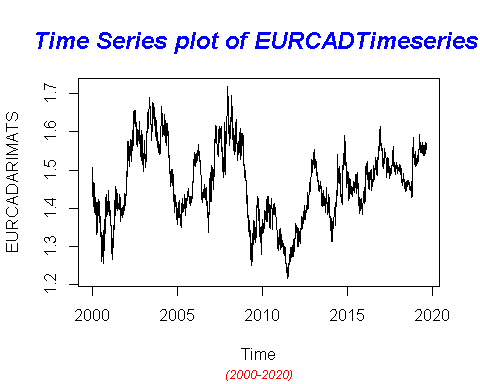
##Checking for obvious errors or missingg value

#Checking for obvious errors  
which(is.na(EURCADARIMA))

## integer(0)

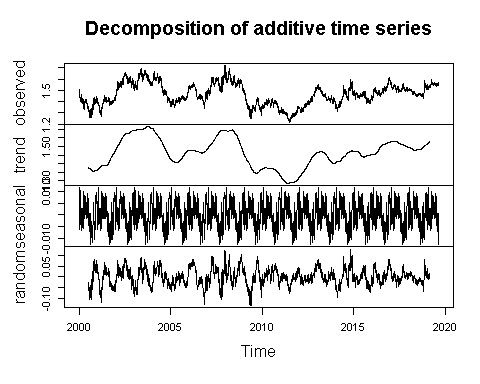
##Converting the data set into time series object

#Converting the data set into time series object  
EURCADARIMATS<- ts(as.vector(EURCADARIMA$Rate), frequency = 322, start= c(2000,01,03))  
plot.ts(EURCADARIMATS)  
title("Time Series plot of EURCADTimeseries ", sub = "(2000-2020)",  
 cex.main = 1.5, font.main= 4, col.main= "blue",  
 cex.sub = 0.75, font.sub = 3, col.sub = "red")



## Finding the component of the Time Series

ComponentEURCAD <- decompose(EURCADARIMATS)  
plot(ComponentEURCAD)



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

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

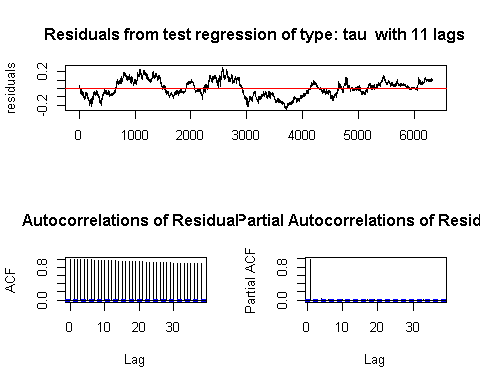
## Loading required package: timeSeries

## Warning: package 'timeSeries' was built under R version 4.0.5

## Loading required package: fBasics

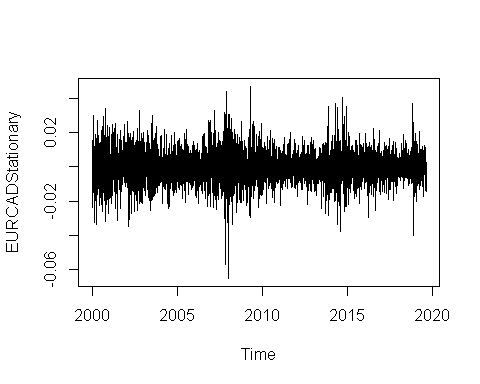
## Warning: package 'fBasics' was built under R version 4.0.5

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



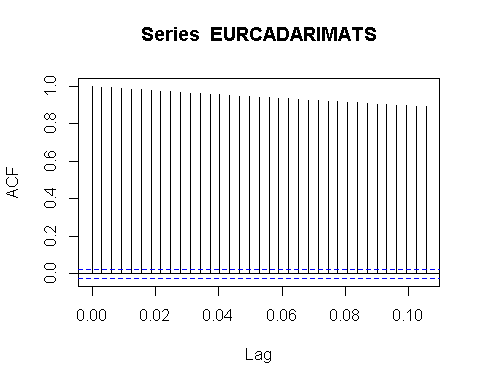
##   
## Title:  
## KPSS Unit Root Test  
##   
## Test Results:  
## NA  
##   
## Description:  
## Thu May 06 09:33:17 2021 by user: janeo

EURCADStationary= diff(EURCADARIMATS, differences=1)  
plot(EURCADStationary)

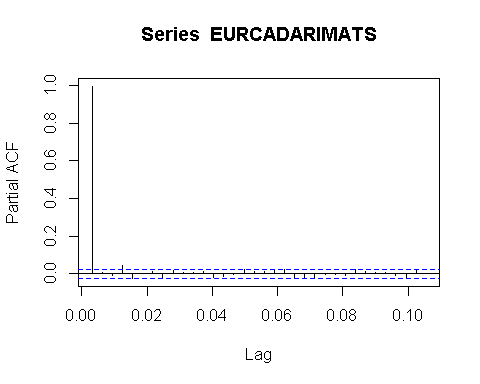


## Calculating Autocorrlation function and partil autocorlation function

acf(EURCADARIMATS,lag.max=34)

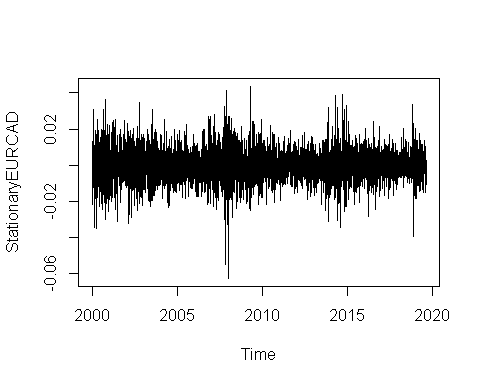


pacf(EURCADARIMATS, lag.max = 34)



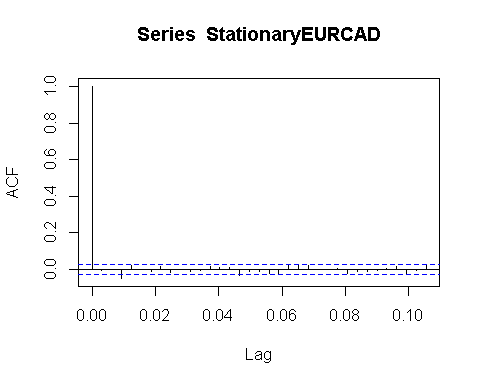
## Adjusting and ensuring there are no seasonality

TSseasonallyadjustedEURCAD <- EURCADARIMATS- ComponentEURCAD$seasonal   
StationaryEURCAD <- diff(TSseasonallyadjustedEURCAD, differences=1)  
plot(StationaryEURCAD)

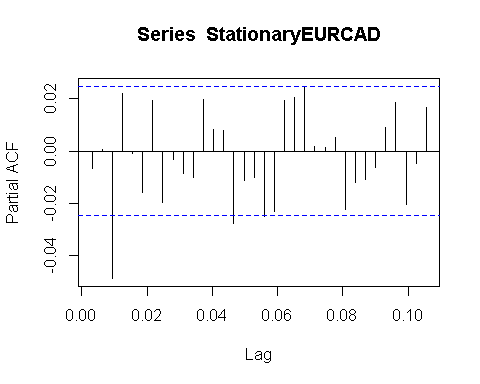


## Calculating again for ACF and PACF after finding stationality

acf(StationaryEURCAD, lag.max=34)



pacf(StationaryEURCAD, lag.max=34)



# Fitting The ARIMA Model

## ARIMA fitting (1,1,0)

fitArima1EURCAD <- arima(EURCADARIMATS, order = c(1,0,0), include.mean = TRUE)  
fitArima1EURCAD

##   
## Call:  
## arima(x = EURCADARIMATS, order = c(1, 0, 0), include.mean = TRUE)  
##   
## Coefficients:  
## ar1 intercept  
## 0.9964 1.4630  
## s.e. 0.0011 0.0275  
##   
## sigma^2 estimated as 6.832e-05: log likelihood = 21365.44, aic = -42724.87

##Arima Fitting (1,0,1)

fitArima2EURCAD <- arima(EURCADARIMATS, order = c(1,0,1), include.mean = TRUE)  
fitArima2EURCAD

##   
## Call:  
## arima(x = EURCADARIMATS, order = c(1, 0, 1), include.mean = TRUE)  
##   
## Coefficients:  
## ar1 ma1 intercept  
## 0.9964 -0.0086 1.4631  
## s.e. 0.0011 0.0126 0.0277  
##   
## sigma^2 estimated as 6.831e-05: log likelihood = 21365.66, aic = -42723.32

## Arima Fitting (2,1,1)

fitArima3EURCAD <- arima(EURCADARIMATS, order = c(2,1,1), include.mean = TRUE)  
fitArima3EURCAD

##   
## Call:  
## arima(x = EURCADARIMATS, order = c(2, 1, 1), include.mean = TRUE)  
##   
## Coefficients:

## Warning in sqrt(diag(x$var.coef)): NaNs produced

## ar1 ar2 ma1  
## -0.0055 0.0041 -0.0048  
## s.e. NaN 0.0081 NaN  
##   
## sigma^2 estimated as 6.844e-05: log likelihood = 21358.79, aic = -42709.58

##Fitting Arima (3,1,0)

fitArima4EURCAD <- arima(EURCADARIMATS, order = c(3,1,0), include.mean = TRUE)  
fitArima4EURCAD

##   
## Call:  
## arima(x = EURCADARIMATS, order = c(3, 1, 0), include.mean = TRUE)  
##   
## Coefficients:  
## ar1 ar2 ar3  
## -0.0101 0.0038 -0.0493  
## s.e. 0.0126 0.0126 0.0126  
##   
## sigma^2 estimated as 6.828e-05: log likelihood = 21366.48, aic = -42724.96

##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

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

## 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

ARIMAModelSelectionEURCAD = AIC(fitArima1EURCAD,fitArima2EURCAD,fitArima3EURCAD,fitArima4EURCAD)

## Warning in AIC.default(fitArima1EURCAD, fitArima2EURCAD, fitArima3EURCAD, :  
## models are not all fitted to the same number of observations

sortScore(ARIMAModelSelectionEURCAD, score ="aic")

## df AIC  
## fitArima4EURCAD 4 -42724.96  
## fitArima1EURCAD 3 -42724.87  
## fitArima2EURCAD 4 -42723.32  
## fitArima3EURCAD 4 -42709.58

### Base on the above the fitArima1CanJap is selected

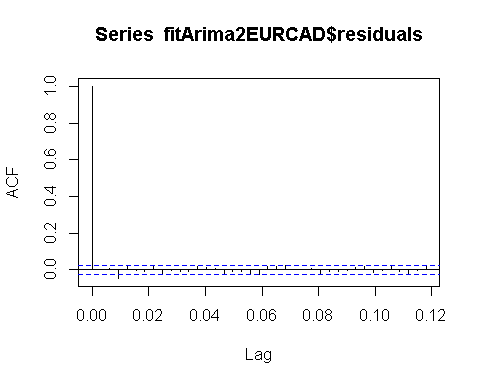
## 

confint(fitArima2EURCAD)

## 2.5 % 97.5 %  
## ar1 0.99433005 0.99853870  
## ma1 -0.03323634 0.01600139  
## intercept 1.40871624 1.51739070

## Runing code to obtain Box Test Rest

acf(fitArima2EURCAD$residuals)



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

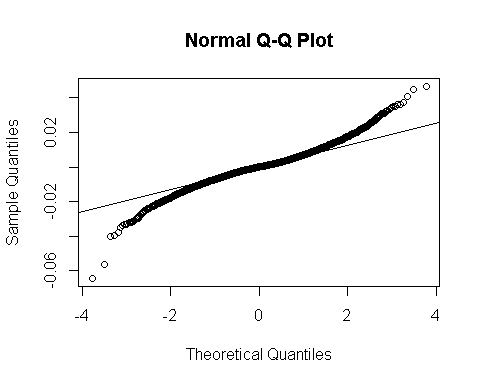
## Loading required package: bestglm

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

library(bestglm)  
 Box.test(resid(fitArima2EURCAD),type="Ljung",lag=20,fitdf=1)

##   
## Box-Ljung test  
##   
## data: resid(fitArima2EURCAD)  
## X-squared = 44.116, df = 19, p-value = 0.00091

qqnorm(fitArima2EURCAD$residuals)  
qqline(fitArima2EURCAD$residuals)



## Using Auto.arima to find the best model fit

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

auto.arima(EURCADARIMATS, trace=TRUE)

##   
## Fitting models using approximations to speed things up...  
##   
## ARIMA(2,1,2)(1,0,1)[322] with drift : Inf  
## ARIMA(0,1,0) with drift : -42703.22  
## ARIMA(1,1,0)(1,0,0)[322] with drift : -43003.27  
## ARIMA(0,1,1)(0,0,1)[322] with drift : Inf  
## ARIMA(0,1,0) : -42705.21  
## ARIMA(1,1,0) with drift : -42704.27  
## ARIMA(1,1,0)(2,0,0)[322] with drift : Inf  
## ARIMA(1,1,0)(1,0,1)[322] with drift : Inf  
## ARIMA(1,1,0)(0,0,1)[322] with drift : Inf  
## ARIMA(1,1,0)(2,0,1)[322] with drift : Inf  
## ARIMA(0,1,0)(1,0,0)[322] with drift : Inf  
## ARIMA(2,1,0)(1,0,0)[322] with drift : Inf  
## ARIMA(1,1,1)(1,0,0)[322] with drift : Inf  
## ARIMA(0,1,1)(1,0,0)[322] with drift : Inf  
## ARIMA(2,1,1)(1,0,0)[322] with drift : -43004.71  
## ARIMA(2,1,1) with drift : Inf  
## ARIMA(2,1,1)(2,0,0)[322] with drift : Inf  
## ARIMA(2,1,1)(1,0,1)[322] with drift : Inf  
## ARIMA(2,1,1)(0,0,1)[322] with drift : Inf  
## ARIMA(2,1,1)(2,0,1)[322] with drift : Inf  
## ARIMA(3,1,1)(1,0,0)[322] with drift : Inf  
## ARIMA(2,1,2)(1,0,0)[322] with drift : Inf  
## ARIMA(1,1,2)(1,0,0)[322] with drift : Inf  
## ARIMA(3,1,0)(1,0,0)[322] with drift : Inf  
## ARIMA(3,1,2)(1,0,0)[322] with drift : Inf  
## ARIMA(2,1,1)(1,0,0)[322] : Inf  
##   
## Now re-fitting the best model(s) without approximations...  
##   
## ARIMA(2,1,1)(1,0,0)[322] with drift : Inf  
## ARIMA(1,1,0)(1,0,0)[322] with drift : Inf  
## ARIMA(0,1,0) : -42714.8  
##   
## Best model: ARIMA(0,1,0)

## Series: EURCADARIMATS   
## ARIMA(0,1,0)   
##   
## sigma^2 estimated as 6.845e-05: log likelihood=21358.4  
## AIC=-42714.8 AICc=-42714.8 BIC=-42708.05

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

forecastarimaEURCAD<- predict(fitArima2EURCAD,n.ahead = 100)  
forecastarimaEURCAD

## $pred  
## Time Series:  
## Start = c(2019, 211)   
## End = c(2019, 310)   
## Frequency = 322   
## [1] 1.553637 1.553314 1.552992 1.552671 1.552352 1.552033 1.551716 1.551400  
## [9] 1.551085 1.550771 1.550458 1.550147 1.549836 1.549527 1.549218 1.548911  
## [17] 1.548605 1.548300 1.547996 1.547693 1.547391 1.547091 1.546791 1.546492  
## [25] 1.546195 1.545899 1.545603 1.545309 1.545015 1.544723 1.544432 1.544142  
## [33] 1.543853 1.543565 1.543278 1.542992 1.542706 1.542422 1.542139 1.541857  
## [41] 1.541576 1.541297 1.541018 1.540740 1.540463 1.540187 1.539911 1.539637  
## [49] 1.539364 1.539092 1.538821 1.538551 1.538282 1.538014 1.537746 1.537480  
## [57] 1.537215 1.536950 1.536687 1.536424 1.536162 1.535902 1.535642 1.535383  
## [65] 1.535125 1.534868 1.534612 1.534357 1.534103 1.533850 1.533597 1.533346  
## [73] 1.533095 1.532845 1.532596 1.532348 1.532101 1.531855 1.531610 1.531365  
## [81] 1.531122 1.530879 1.530637 1.530396 1.530156 1.529917 1.529678 1.529441  
## [89] 1.529204 1.528968 1.528733 1.528499 1.528266 1.528033 1.527802 1.527571  
## [97] 1.527341 1.527111 1.526883 1.526655  
##   
## $se  
## Time Series:  
## Start = c(2019, 211)   
## End = c(2019, 310)   
## Frequency = 322   
## [1] 0.008265238 0.011617827 0.014183058 0.016336255 0.018224146 0.019922383  
## [7] 0.021476081 0.022914730 0.024258876 0.025523516 0.026720003 0.027857176  
## [13] 0.028942083 0.029980450 0.030977009 0.031935722 0.032859949 0.033752568  
## [19] 0.034616069 0.035452620 0.036264130 0.037052283 0.037818578 0.038564357  
## [25] 0.039290825 0.039999070 0.040690078 0.041364749 0.042023902 0.042668290  
## [31] 0.043298604 0.043915482 0.044519513 0.045111242 0.045691179 0.046259794  
## [37] 0.046817531 0.047364799 0.047901987 0.048429456 0.048947546 0.049456580  
## [43] 0.049956858 0.050448668 0.050932278 0.051407945 0.051875911 0.052336406  
## [49] 0.052789649 0.053235847 0.053675201 0.054107897 0.054534116 0.054954031  
## [55] 0.055367806 0.055775599 0.056177560 0.056573834 0.056964561 0.057349872  
## [61] 0.057729896 0.058104756 0.058474570 0.058839452 0.059199512 0.059554855  
## [67] 0.059905583 0.060251794 0.060593584 0.060931045 0.061264264 0.061593327  
## [73] 0.061918318 0.062239316 0.062556400 0.062869645 0.063179123 0.063484905  
## [79] 0.063787061 0.064085656 0.064380756 0.064672422 0.064960717 0.065245700  
## [85] 0.065527427 0.065805956 0.066081340 0.066353634 0.066622888 0.066889153  
## [91] 0.067152479 0.067412912 0.067670499 0.067925287 0.068177319 0.068426638  
## [97] 0.068673287 0.068917307 0.069158738 0.069397619

par(mfrow = c(1,1))