TIME SERIES ANALYSIS OF XYZ’S GDP

sammy waiyaki

3/23/2020

# Time series analysis of XYZ’s GDP from 1963-2016 using R

The data set contains the value of GDPs for country XYZ from 1963 to 2016. We aim to explore the data to find insights into how the country has been performing economically and also to forecast the GDP values for the next ten years.

### Reading GDP data set into R for analysis

GDP <- read.csv(file.choose(), header = T)  
View(GDP)

#### Transforming the data into a time series.

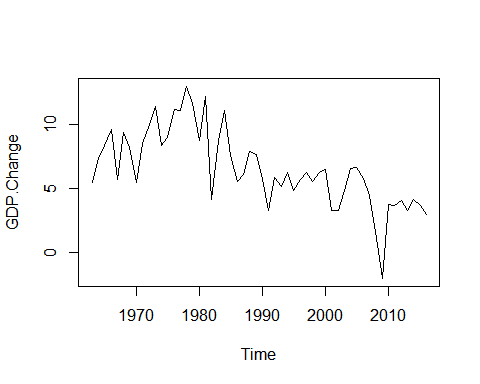
When the data is imported, it is not initialized as a time series. Therefore you must first transform it into a time series data to carry out the analysis.

GDP\_time\_series <- ts(GDP, frequency = 1, start = c(1963, 1))

#### Plotting the time series data

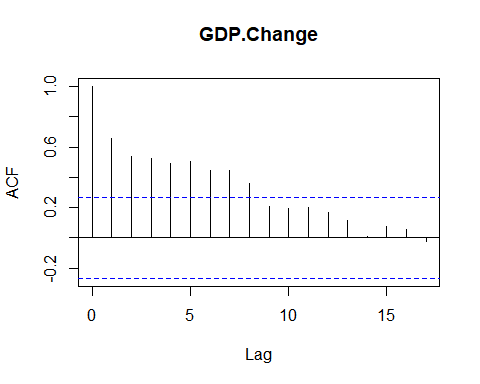
Plotting helps in identifying any trend or seasonality that data might have. Also, through studying, we can know whether the data is stationary or not.

plot.ts(GDP\_time\_series)



The plot above does not show a proper trend or seasonality property contained in the data. Besides, the data is not stationary as the mean, variance, and autocorrelation is not time-invariant. Confirmation for the non-stationarity of the data is indicated by the correlogram, where the lags decrease gradually with time and are above the bounds of the correlogram.

acf(GDP\_time\_series)



### Decomposition of the GDP time series data.

From the plot, the GDP time series data is non-seasonal. We, therefore, need to decompose it to estimate the trend and irregular components of the data. Since the data set can be described by an additive model, the simple moving average is a smoothing method to determine the trend and irregular components of the time series.

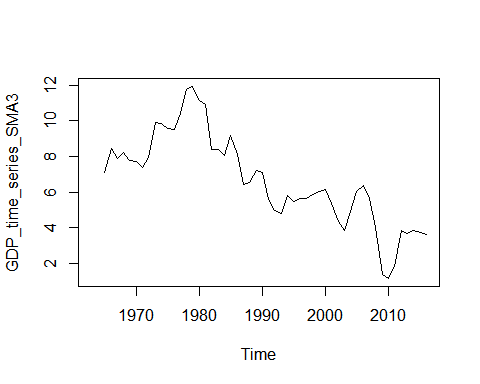
#### Library for simple moving average

library(TTR)

## Warning: package 'TTR' was built under R version 3.6.3

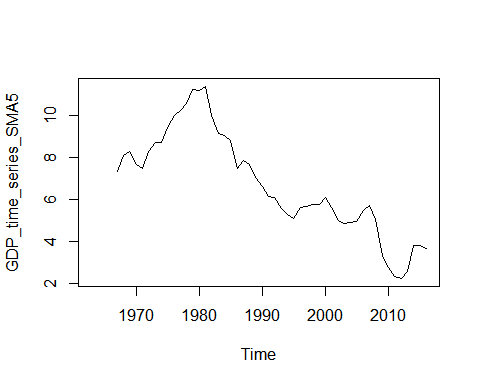
#### Simple moving average of order 3

GDP\_time\_series\_SMA3 <- SMA(GDP\_time\_series, n=3)  
plot.ts(GDP\_time\_series\_SMA3)



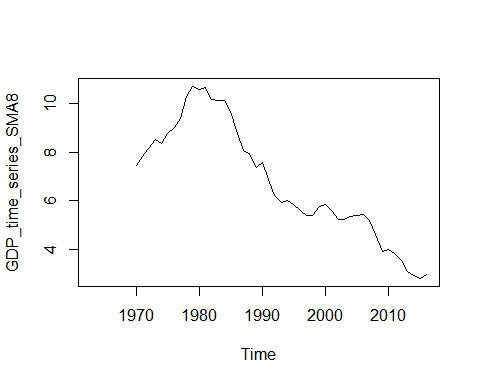
#### Simple moving average of order 5

GDP\_time\_series\_SMA5 <- SMA(GDP\_time\_series, n=5)  
plot.ts(GDP\_time\_series\_SMA5)



#### Simple moving average of order 8

GDP\_time\_series\_SMA8 <- SMA(GDP\_time\_series, n=8)  
plot.ts(GDP\_time\_series\_SMA8)



## Simple exponential smoothing and forecasting

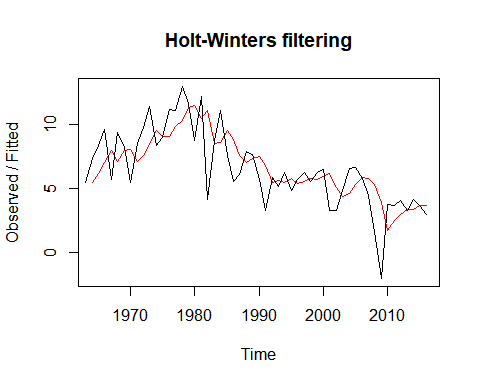
The simple exponential smoothing method provides a way of estimating the level at the current time point. Smoothing is controlled by the parameter alpha; for the estimate of the level at the current time point. The value of alpha; lies between 0 and 1. Values of alpha that are close to 0 mean that little weight is placed on the most recent observations when making forecasts of future values.

### Fitting Simple exponential smmothing to data

GDP\_time\_series\_SESM <- HoltWinters(GDP\_time\_series, beta = F, gamma = F)  
GDP\_time\_series\_SESM

## Holt-Winters exponential smoothing without trend and a seasonal component.  
##   
## Call:  
## HoltWinters(x = GDP\_time\_series, beta = F, gamma = F)  
##   
## Smoothing parameters:  
## alpha: 0.3769724  
## beta : FALSE  
## gamma: FALSE  
##   
## Coefficients:  
## [,1]  
## a 3.425426

View(GDP\_time\_series\_SESM$fitted)  
plot(GDP\_time\_series\_SESM)



GDP\_time\_series\_SESM$SSE

## [1] 238.5429

### Forecasting GDP for the next 10 years

library(forecast)

## Warning: package 'forecast' was built under R version 3.6.3

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

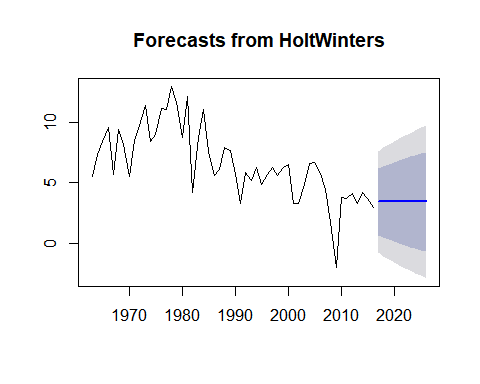
GDP\_time\_series\_SESM\_forecast <- forecast(GDP\_time\_series\_SESM, h=10)  
GDP\_time\_series\_SESM\_forecast

## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2017 3.425426 0.68387159 6.166980 -0.7674195 7.618271  
## 2018 3.425426 0.49554161 6.355310 -1.0554454 7.906297  
## 2019 3.425426 0.31860696 6.532245 -1.3260436 8.176895  
## 2020 3.425426 0.15121976 6.699632 -1.5820402 8.432892  
## 2021 3.425426 -0.00801666 6.858868 -1.8255713 8.676423  
## 2022 3.425426 -0.16018838 7.011040 -2.0582978 8.909149  
## 2023 3.425426 -0.30615978 7.157011 -2.2815418 9.132393  
## 2024 3.425426 -0.44663216 7.297484 -2.4963758 9.347227  
## 2025 3.425426 -0.58218381 7.433035 -2.7036842 9.554536  
## 2026 3.425426 -0.71329824 7.564150 -2.9042064 9.755058

GDP\_time\_series\_SESM\_forecast$residuals

## Time Series:  
## Start = 1963   
## End = 2016   
## Frequency = 1   
## x  
## [1,] NA  
## [2,] 1.90000000  
## [3,] 2.18375251  
## [4,] 2.56053816  
## [5,] -2.30471396  
## [6,] 2.26409951  
## [7,] 0.21059656  
## [8,] -2.56879252  
## [9,] 1.39957127  
## [10,] 2.17197158  
## [11,] 2.95319832  
## [12,] -1.16007584  
## [13,] -0.12275931  
## [14,] 2.12351756  
## [15,] 1.22301012  
## [16,] 2.66196911  
## [17,] 0.35848032  
## [18,] -2.67665686  
## [19,] 1.73236881  
## [20,] -6.92068636  
## [21,] 0.28822114  
## [22,] 2.47956974  
## [23,] -1.95515953  
## [24,] -3.21811842  
## [25,] -1.50497671  
## [26,] 0.86235792  
## [27,] 0.33727282  
## [28,] -1.78986972  
## [29,] -3.51513830  
## [30,] 0.40997170  
## [31,] -0.44457630  
## [32,] 0.82301668  
## [33,] -0.88723787  
## [34,] 0.24722629  
## [35,] 0.75402881  
## [36,] -0.23021921  
## [37,] 0.55656707  
## [38,] 0.54675666  
## [39,] -2.85935549  
## [40,] -1.78145749  
## [41,] 0.49010275  
## [42,] 2.00534756  
## [43,] 1.34938695  
## [44,] -0.05929464  
## [45,] -1.33694220  
## [46,] -3.63295194  
## [47,] -5.96342945  
## [48,] 2.08461865  
## [49,] 1.19877503  
## [50,] 1.14686997  
## [51,] -0.08546831  
## [52,] 0.84675088  
## [53,] 0.02754920  
## [54,] -0.68283609

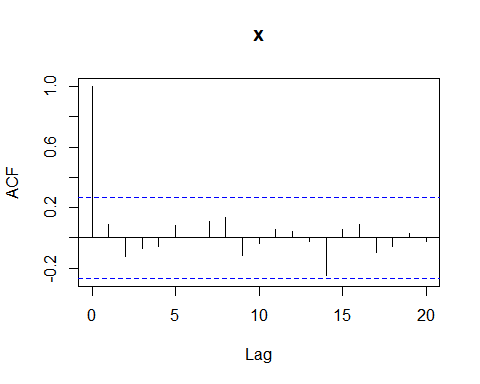
plot(GDP\_time\_series\_SESM\_forecast)



### Correlation of forecast Errors and successive prediction.

A simple exponential smoothing is considere accurate and cannot be improved by another model if there is no correlation between the errors of forecast and successive prediction. We explore the correlogram residual to identify any correlation. further, we perfom Ljung Box test to see the significance of the non-zero correlation.

acf(GDP\_time\_series\_SESM\_forecast$residuals, lag.max = 20, na.action = na.pass)

 ### Ljung Box test

Box.test(GDP\_time\_series\_SESM\_forecast$residuals, lag=20, type = 'Ljung-Box')

##   
## Box-Ljung test  
##   
## data: GDP\_time\_series\_SESM\_forecast$residuals  
## X-squared = 12.16, df = 20, p-value = 0.9104

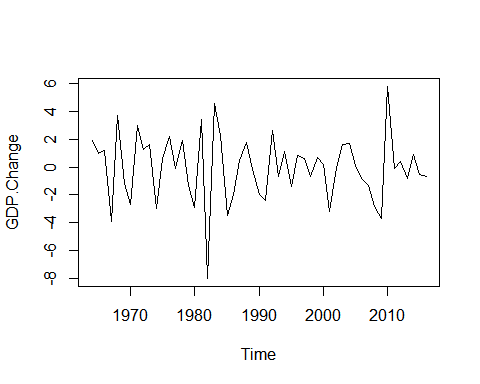
## ARIMA MODEL

ARIMA is an acronym of Autoregressive Integrated Moving Average. ARIMA models include an explicit statistical model for the irregular component of a time series, that allows for non-zero autocorrelations in the irregular component. Arima models usually have three-parameter;p is the number of lag observations, d is the order of differentiation that makes the data stationary, and q is the order on the moving average.

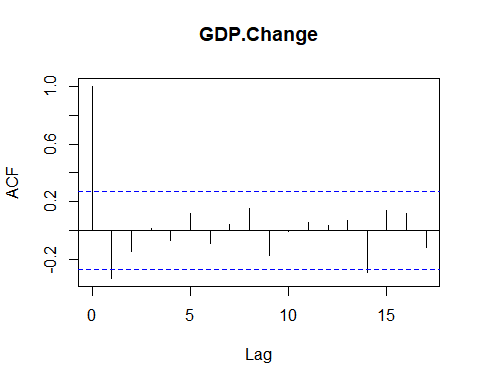
Differentiate time series to make it stationary

When a time series is non-stationary, it can be made stationary by differentiating in different orders until it attains stationarity. The order of differentiation that makes the data stationary becomes the d parameter.

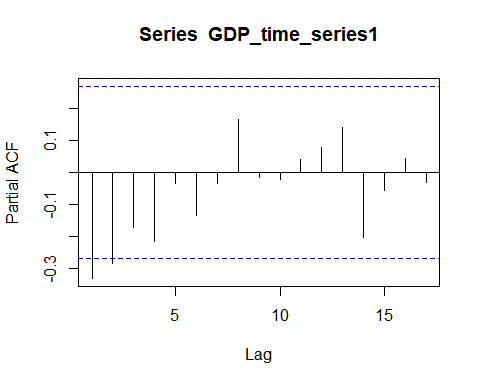
GDP\_time\_series1 <- diff(GDP\_time\_series, differences = 1)  
plot.ts(GDP\_time\_series1)

 ### Finding appropirate values for p and q

acf(GDP\_time\_series1)



pacf(GDP\_time\_series1)

 Alternatively auto.arima() function in the forecast libraby can used to find the best values of pdq

auto.arima(GDP\_time\_series)

## Series: GDP\_time\_series   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.6452  
## s.e. 0.1137  
##   
## sigma^2 estimated as 4.505: log likelihood=-114.86  
## AIC=233.72 AICc=233.96 BIC=237.66

### Fiting the ARIMA model to GDP data

GDP\_time\_series\_Arima <- arima(GDP\_time\_series, order = c(0,1,1))  
GDP\_time\_series\_Arima

##   
## Call:  
## arima(x = GDP\_time\_series, order = c(0, 1, 1))  
##   
## Coefficients:  
## ma1  
## -0.6452  
## s.e. 0.1137  
##   
## sigma^2 estimated as 4.42: log likelihood = -114.86, aic = 233.72

## Forecasting using the ARIMA model and plotting

GDP\_arima\_forecast <- forecast(GDP\_time\_series\_Arima, h=10)  
GDP\_arima\_forecast

## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2017 3.43097 0.736554241 6.125386 -0.6897835 7.551724  
## 2018 3.43097 0.572015354 6.289925 -0.9414240 7.803365  
## 2019 3.43097 0.416443991 6.445497 -1.1793499 8.041291  
## 2020 3.43097 0.268516449 6.593424 -1.4055855 8.267526  
## 2021 3.43097 0.127205804 6.734735 -1.6217015 8.483642  
## 2022 3.43097 -0.008303654 6.870244 -1.8289453 8.690886  
## 2023 3.43097 -0.138672655 7.000613 -2.0283275 8.890268  
## 2024 3.43097 -0.264445281 7.126386 -2.2206801 9.082621  
## 2025 3.43097 -0.386075914 7.248017 -2.4066982 9.268639  
## 2026 3.43097 -0.503948669 7.365889 -2.5869690 9.448910

plot(GDP\_arima\_forecast)

