Rapport TD4

Louis TEILLET

2021-12-19

```
knitr::opts_chunk$set(message = FALSE)
```

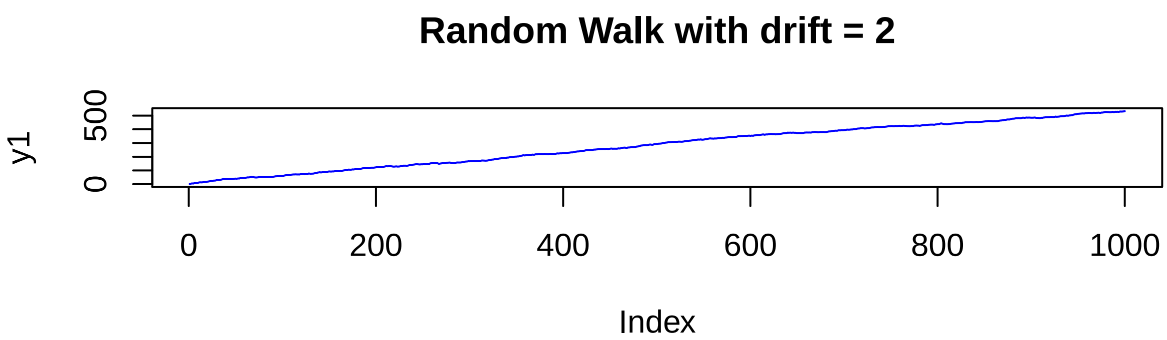
Exercise 1 : TS and DS process

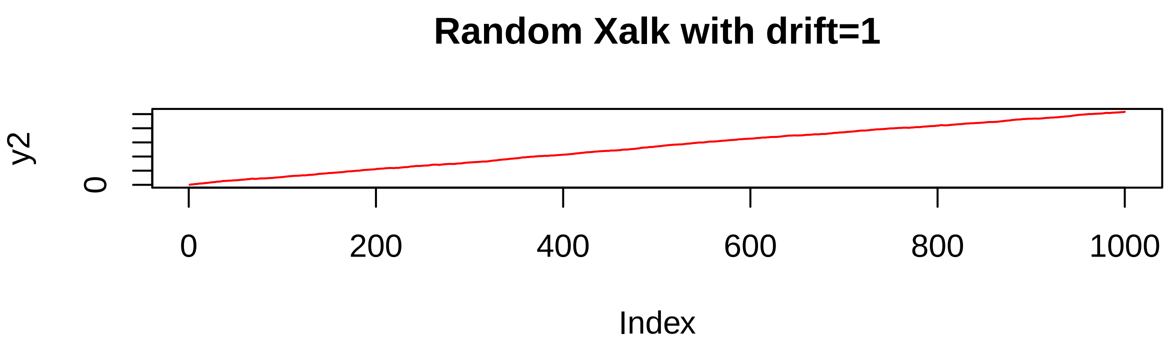
1. Computing 2 random walk with a drift and store them within a matrix

```
library(tseries)
N = 1000
u1 = rnorm(N) # Simulation of 1000 Gaussian observations
y1=c()
y2=c()
y1[1]=u1[1] # First random walk
y2[1]=u1[1] # Second random walk

for (k in 2:N)
{
 y1[k] = y1[k-1] + 0.5 +u1[k] #With a drift=0.5
 y2[k]= y2[k-1]+1+u1[k] #With a drift=1

}
```

2. Plot them all in order to verify(visually) the existence of a trend

```
par(mfrow=c(2,1))
plot(y1,type="l",col="blue", main="Random Walk with drift = 2") #Plotting the first one
plot(y2,type="l",col="red",main ="Random Xalk with drift=1") #Plotting the second one
```





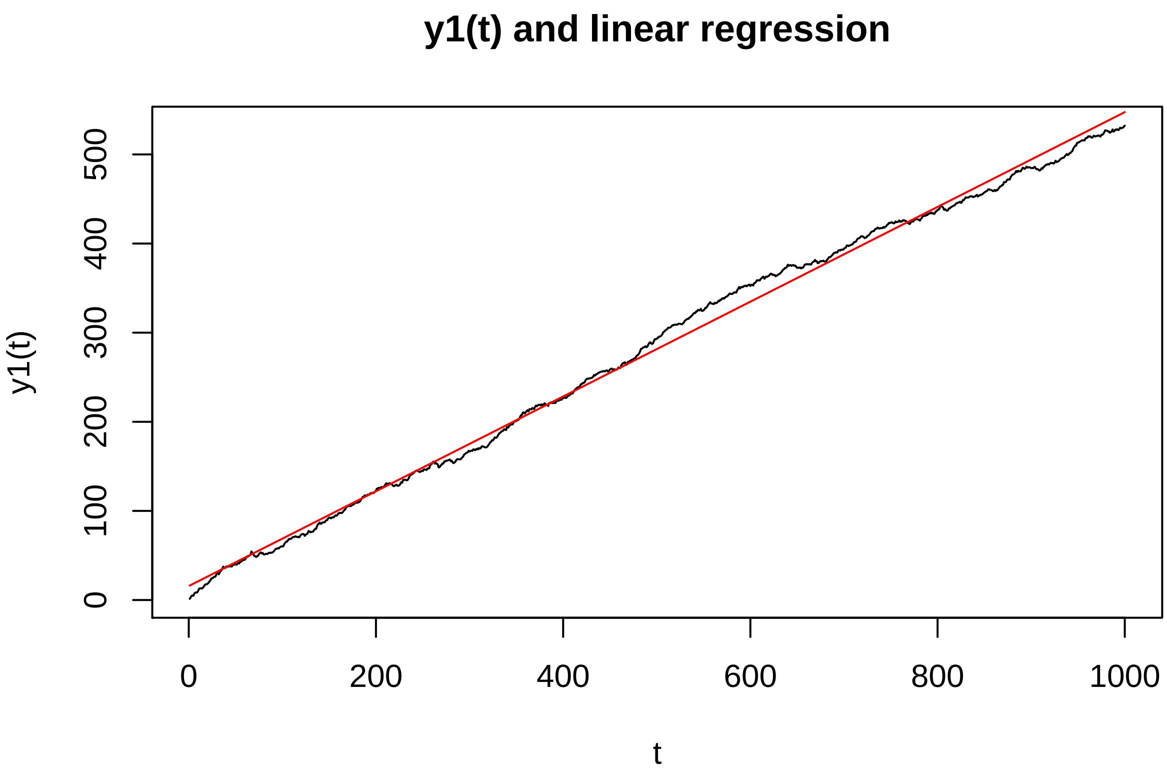
We observe a trend but it is hard to say if it's a deterministic trend or a stochastic one

3. Select one of the random walk you have generated and compute a linear regression where the explanatory variable is a trend t

```r
time = seq(1:N)
lr = lm(y1~time)
summary(lr)
```

```
##
## Call:
## lm(formula = y1 ~ time)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.483  -7.675  -1.895   6.369  22.814
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 15.707939   0.629487   24.95   <2e-16 ***
## time         0.531756   0.001089  488.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.946 on 998 degrees of freedom
## Multiple R-squared:  0.9958, Adjusted R-squared:  0.9958
## F-statistic: 2.382e+05 on 1 and 998 DF,  p-value: < 2.2e-16
```

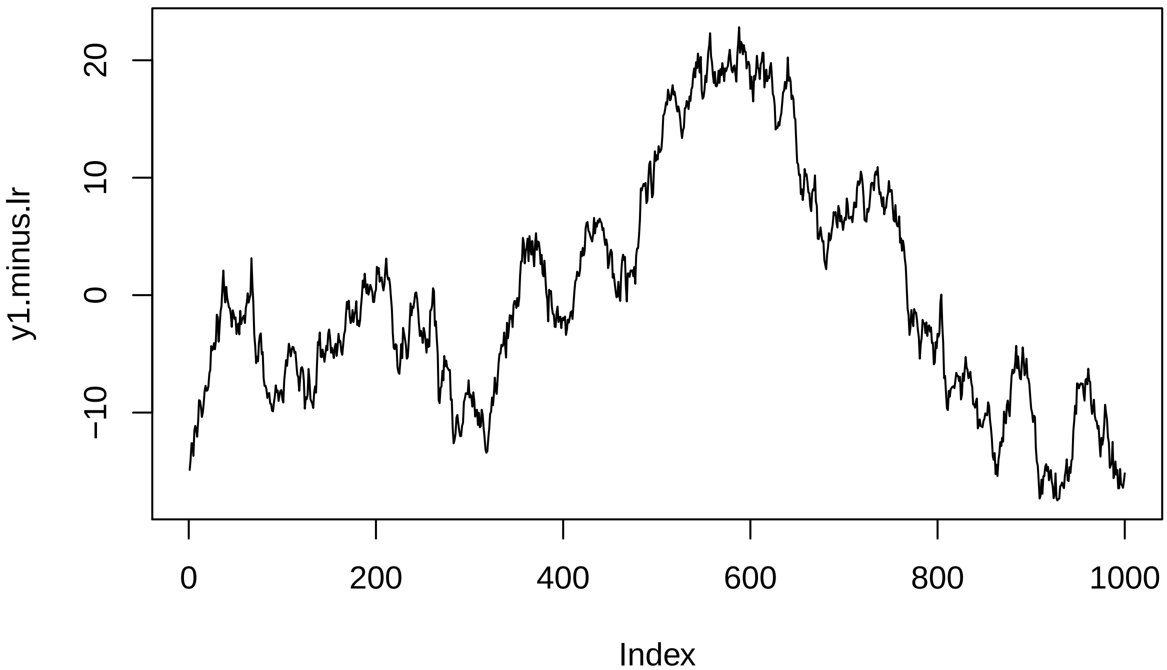
```
plot(y1,type="l",main="y1(t) and linear regression",xlab="t",ylab="y1(t)")
lines(time,lr$fitted.values,col="red")
```



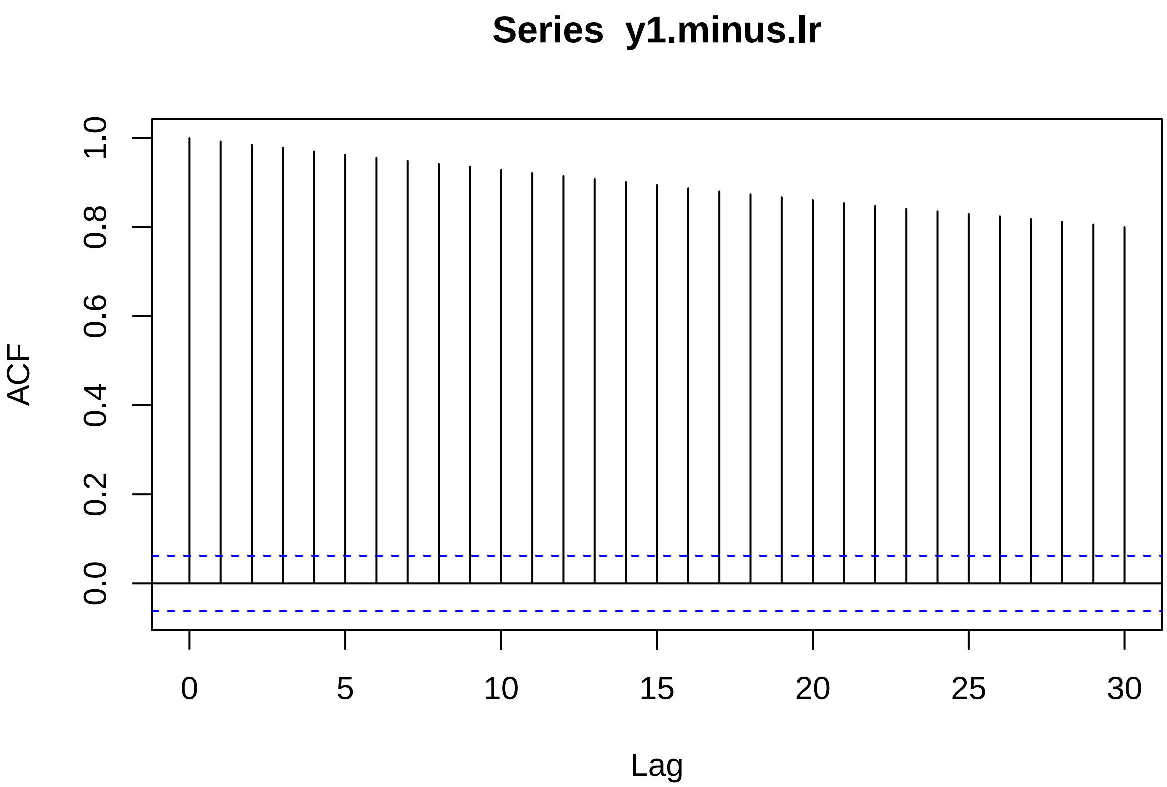
4. Is it the right way to neutralize the source of non stationary ?

We try to neutralize the source of non stationary upon making the difference between y(t) and lr(t)

```
y1.minus.lr = y1-lr$fitted.values
plot(y1.minus.lr,type="l") #There is still a trend
```

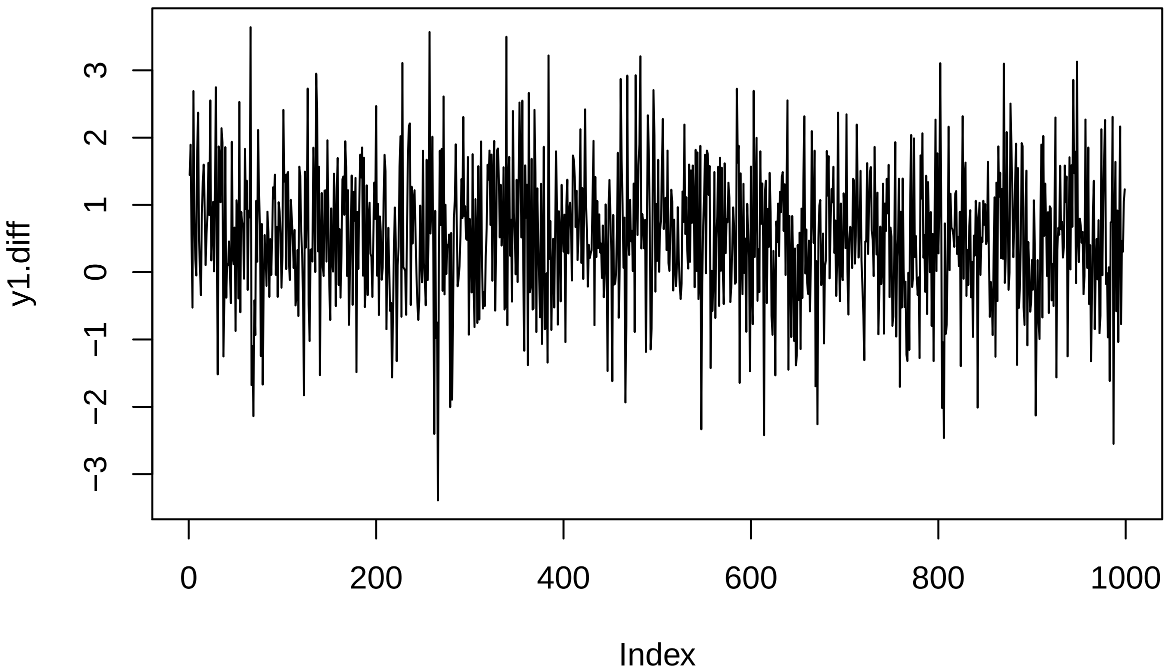



```
acf(y1.minus.lr) #The acf show a really high persistence, the process is not stationary
```

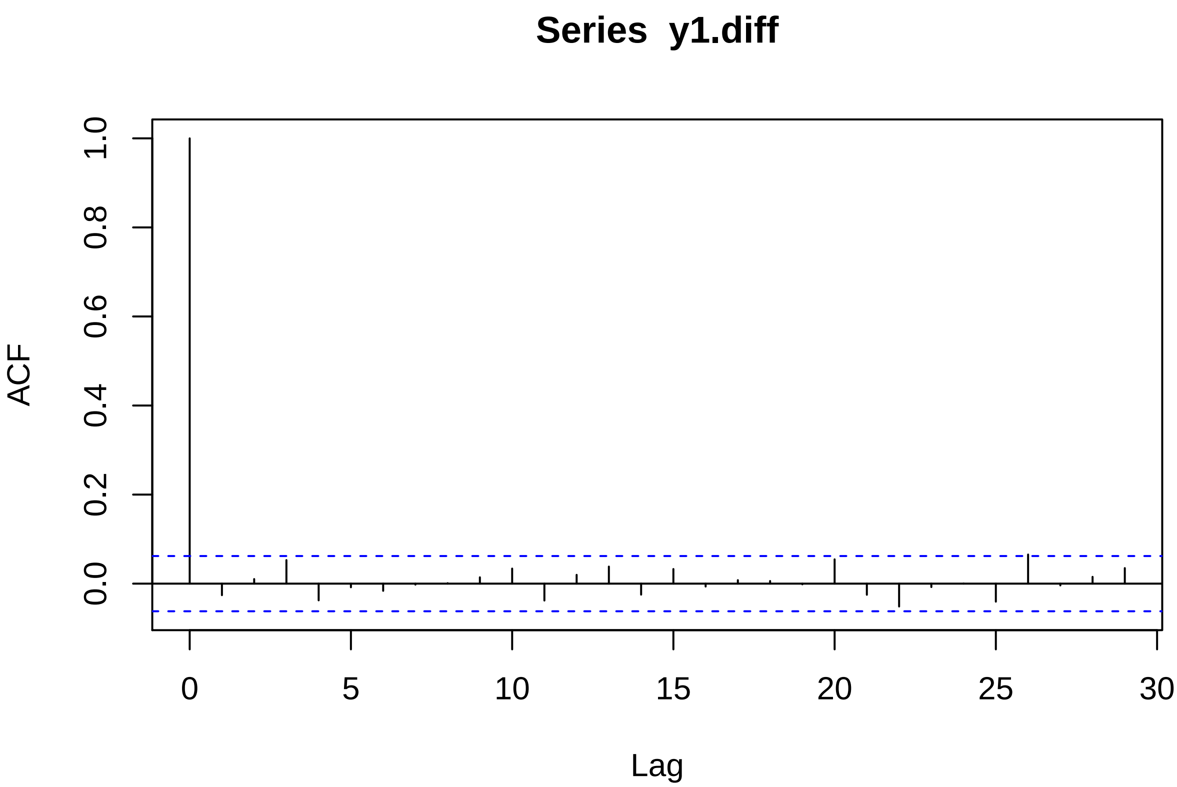


5. Differentiate the selected variable and check the autocorrelation function

```
y1.diff = diff(y1)
plot(y1.diff,type="l")
```




```
acf(y1.diff) # The acf doesn't show persistence anymore
```



Exercise 2 : ADF, PP and KPSS unit root tests

Dickey Fuller

```r
library("urca") #Loading "urca" package
```

The Dickey Fuller tests the type of non stationary of a process.

The first type of non stationary is a model without drift and temporal trend:

$y_t = \rho_1.y_{t-1} + \epsilon_t$

The second one is a model with drift but without temporal trend:

$y_t = \rho_1.y_{t-1} + c + \epsilon_t$

The last one is a model with drift and temporal trend:

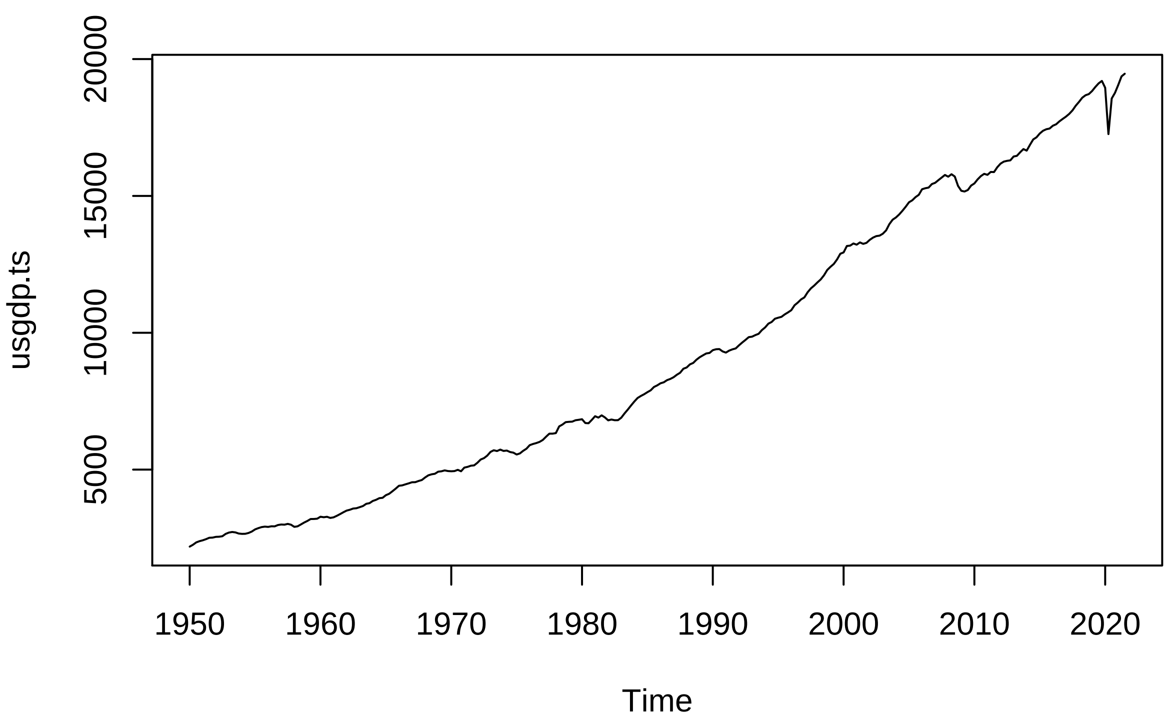
$y_t = \rho_1.y_{t-1} + c + a.t + \epsilon_t$

The null hypothesis is $H_0 : \rho = 0$

We start to test the third model, if the non-nullity of the temporal trend coefficient is not significant we go trough the second one, else we check if ρ_1 is significantly different from 0.

We do the same thing for the model 2, and then for the model 1.

```r
data = read.csv("usgdp.csv",sep=";")
usgdp.ts = ts(data$usgdp,frequency=4,start=c(1950,1))
plot(usgdp.ts)
```



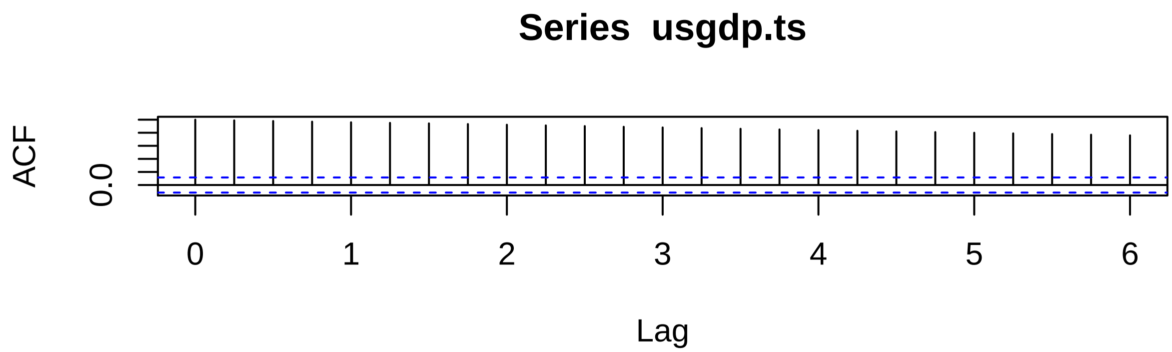

```r
summary(ur.df(usgdp.ts,type="drift",selectlags="AIC"))
```

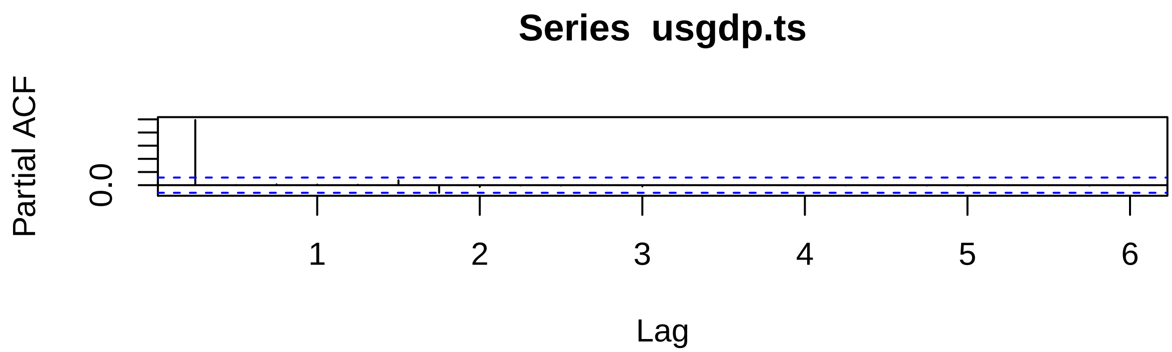
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
```

```
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1838.99   -31.15     7.24    43.45   944.35
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.123904  17.544786   1.945   0.0528 .
## z.lag.1      0.003852   0.001672   2.304   0.0220 *
## z.diff.lag  -0.152089   0.059070  -2.575   0.0105 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 145.2 on 282 degrees of freedom
## Multiple R-squared:  0.03581,    Adjusted R-squared:  0.02897
## F-statistic: 5.236 on 2 and 282 DF,  p-value: 0.00585
##
##
## Value of test-statistic is: 2.3039 29.7293
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.44 -2.87 -2.57
## phi1  6.47  4.61  3.79
```

We can't reject Ho : The presence of a unit root, hence the serie is a random walk

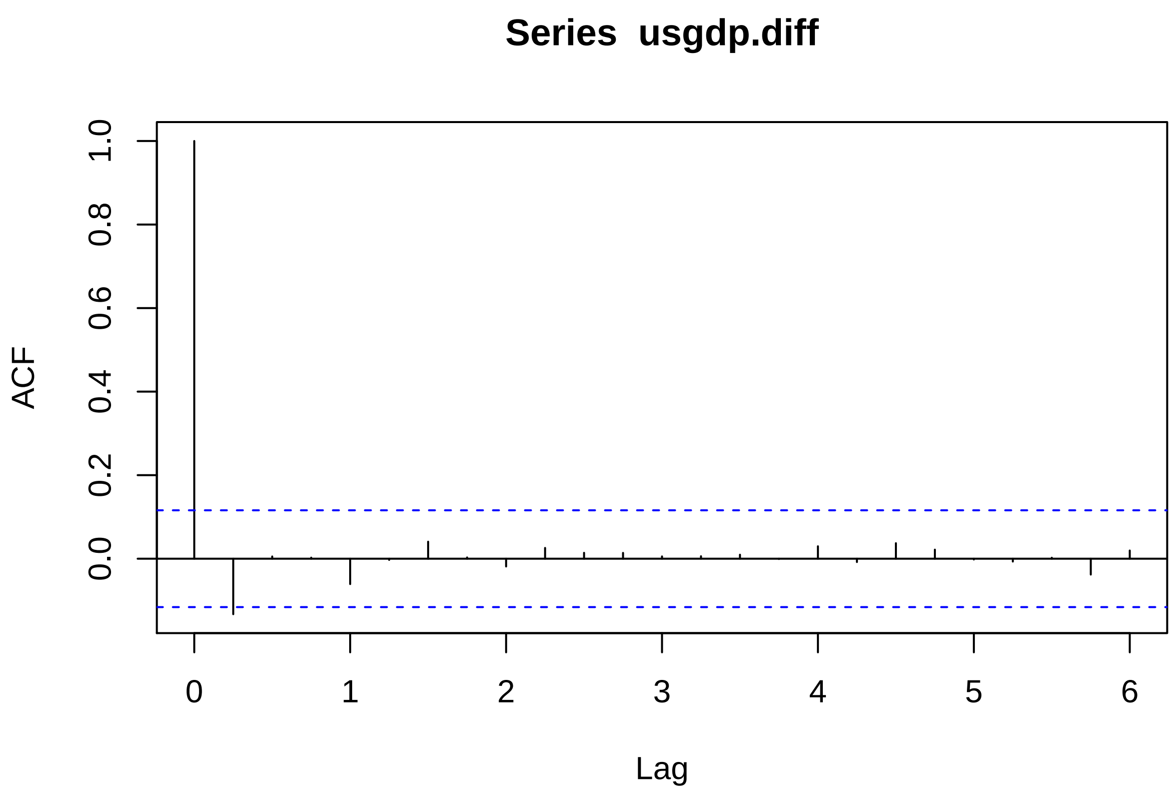
```
par(mfrow=c(2,1))
acf(usgdp.ts)
pacf(usgdp.ts)
```



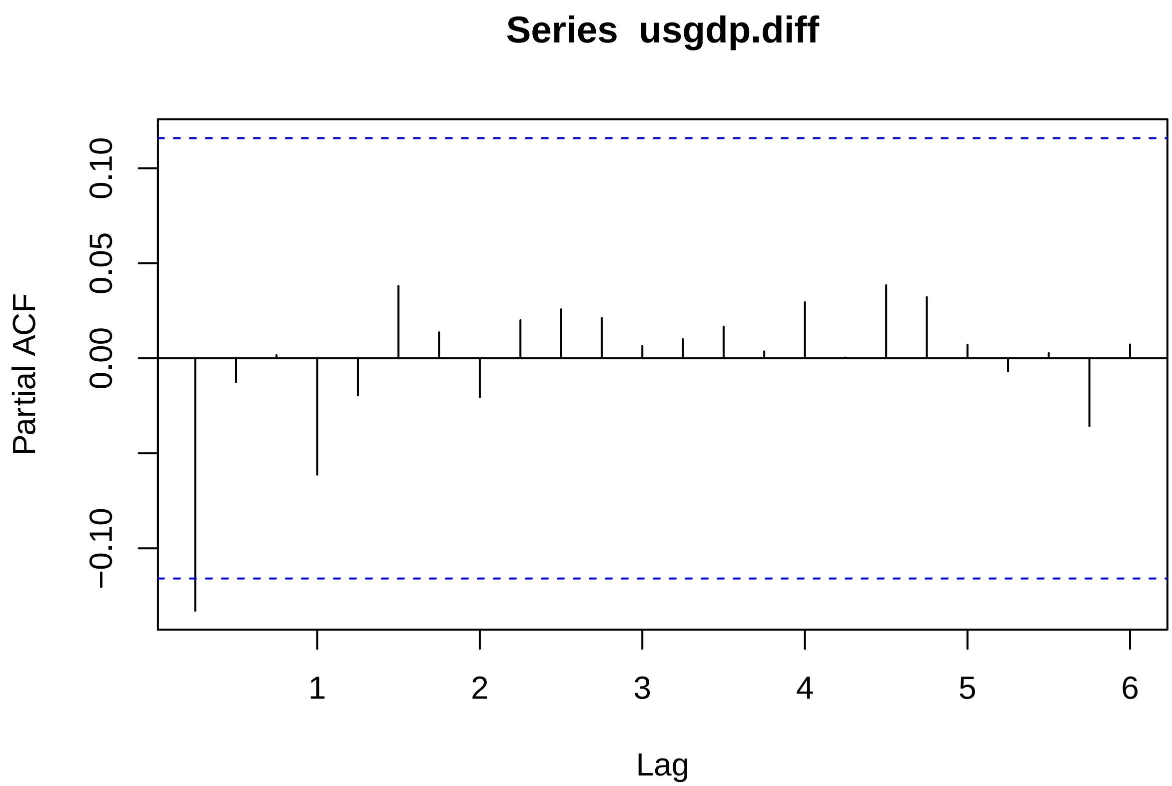


The dickey fuller result is confirmed by the autocorrelograms

```
usgdp.diff= diff(usgdp.ts) # We differentiate the serie one time
acf(usgdp.diff)
```



```r
pacf(usgdp.diff)
```



Autocorrelograms seems to indicate the stationnarity of the time serie

```r
summary(ur.df(usgdp.diff,type="drift",selectlags="AIC"))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1795.52   -40.67     4.02    49.59  1002.08
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 69.18062   10.24577   6.752 8.34e-11 ***
## z.lag.1     -1.14755    0.09014 -12.730  < 2e-16 ***
## z.diff.lag   0.01289    0.05998   0.215     0.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146.8 on 281 degrees of freedom
```

```
## Multiple R-squared:  0.5665, Adjusted R-squared:  0.5634
## F-statistic: 183.6 on 2 and 281 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.7305 81.0345
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.44 -2.87 -2.57
## phi1  6.47  4.61  3.79
```

We now can reject Ho : there is a unit root and accept H1 : the serie is stationnary

Conclusion : usGDP is I(1)

Phillips-Perron

The Phillips-Perron (PP) unit root tests differ from the ADF tests mainly in how they deal with serial correlation and heteroskedasticity in the errors. In particular, where the ADF tests use a parametric autoregression to approximate the ARMA structure of the errors in the test regression, the PP tests ignore any serial correlation in the test regression.

```
library(aTSA)
pp.test(data$usgdp)
```

```
## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##  lag Z_rho p.value
##    5  1.66   0.976
## -----
##  Type 2: with drift no trend
##  lag Z_rho p.value
##    5 0.958   0.987
## -----
##  Type 3: with drift and trend
##  lag Z_rho p.value
##    5 -3.53    0.91
## ---------------
## Note: p-value = 0.01 means p.value <= 0.01
```

```
pp.test(usgdp.diff)
```

```
## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##  lag Z_rho p.value
##    5  -334    0.01
## -----
##  Type 2: with drift no trend
##  lag Z_rho p.value
##    5  -313    0.01
## -----
##  Type 3: with drift and trend
```

```
##  lag Z_rho p.value
##    5  -307    0.01
## ----------------
## Note: p-value = 0.01 means p.value <= 0.01
```

p value <=0.01 so we reject Ho : the serie has a unit root.

Conclusion: The Phillips-Perron unit root tests return the same result than Dickey Fuller. **Usgdp is I(1) for this test**

KPSS

Here we just have to be care of the fact that the null hypothesis is the stationnarity of the serie

```r
kpss.test(data$usgdp)
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag  stat p.value
##    3 0.987     0.1
## -----
##  Type 2: with drift no trend
##  lag   stat p.value
##    3 0.0728     0.1
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    3 0.0711     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

```r
kpss.test(usgdp.diff)
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag stat p.value
##    3  8.3    0.01
## -----
##  Type 2: with drift no trend
##  lag stat p.value
##    3 0.75    0.01
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    3 0.0409     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

```r
kpss.test(diff(data$usgdp,2))
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag stat p.value
##    3 5.66    0.01
## -----
##  Type 2: with drift no trend
##  lag stat p.value
##    3 0.61  0.0218
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    3 0.0325     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

```
kpss.test(diff(data$usgdp,3))
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag stat p.value
##    3 3.96    0.01
## -----
##  Type 2: with drift no trend
##  lag  stat p.value
##    3 0.544  0.0317
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    3 0.0297     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

```
kpss.test(diff(data$usgdp,4))
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag stat p.value
##    3 2.96    0.01
## -----
##  Type 2: with drift no trend
##  lag  stat p.value
##    3 0.512   0.039
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    3 0.0304     0.1
## -----------
```

```
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

```
kpss.test(diff(data$usgdp,5)) #p value >0.05 we can't reject the null hypothesis
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag  stat p.value
##    3 0.731     0.1
## -----
##  Type 2: with drift no trend
##  lag stat p.value
##    3 0.42  0.0685
## -----
##  Type 1: with drift and trend
##  lag  stat p.value
##    3 0.022     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

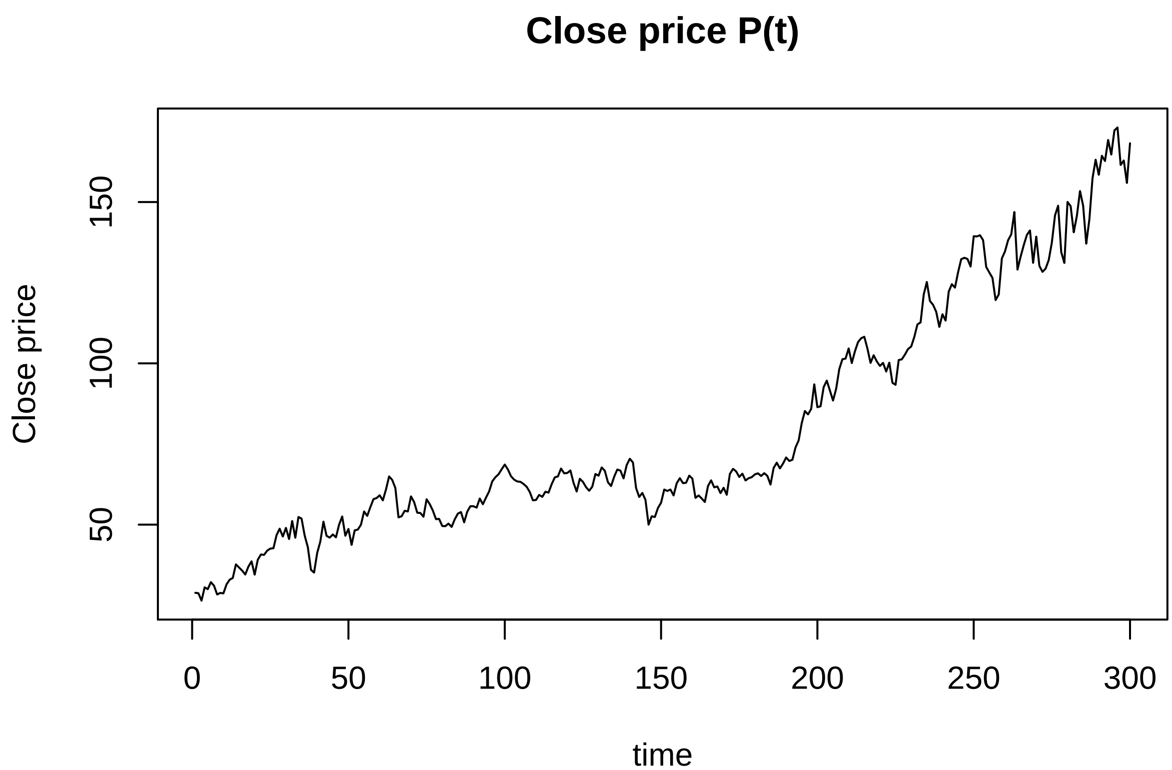
Kpss returns an order of integration of 5. **Conclusion : With KPSS test, usgdp is I(5)

Exercise 3 : Estimating ARIMA(p,d,q)

```
library(tidyquant)
jnj = tq_get("JNJ",get="stock.prices",from="1997-01-01") %>% tq_transmute(mutate_fun = to.period,period=
```

1. Determine the degree of integration of the Johnson & Johnson stock price.

```
plot(jnj$close,type="l",main="Close price P(t)",ylab="Close price",xlab="time")
```




```
df.test=ur.df(jnj$close,type="trend",selectlags="AIC")
summary(df.test)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.3585  -2.2245   0.0024   2.3343  17.8293
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.738301   0.590602   1.250   0.2123
## z.lag.1     -0.030390   0.016732  -1.816   0.0704 .
## tt           0.014503   0.007053   2.056   0.0406 *
## z.diff.lag  -0.083907   0.059168  -1.418   0.1572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.077 on 294 degrees of freedom
## Multiple R-squared:  0.02349,    Adjusted R-squared:  0.01352
## F-statistic: 2.357 on 3 and 294 DF,  p-value: 0.07191
```

```
##
##
## Value of test-statistic is: -1.8163 2.9585 2.1344
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.98 -3.42 -3.13
## phi2  6.15  4.71  4.05
## phi3  8.34  6.30  5.36
```

All of our test value are within the "fail to reject" zone, so jnj is a time serie which has a unit root (it's a random walk), but neither trend nor constant drift

We try to differentiate the time serie, one time

```
jnj.diff = diff(jnj$close)
df.test.diff=ur.df(jnj.diff,type="trend",selectlags="AIC")
summary(df.test.diff)
```

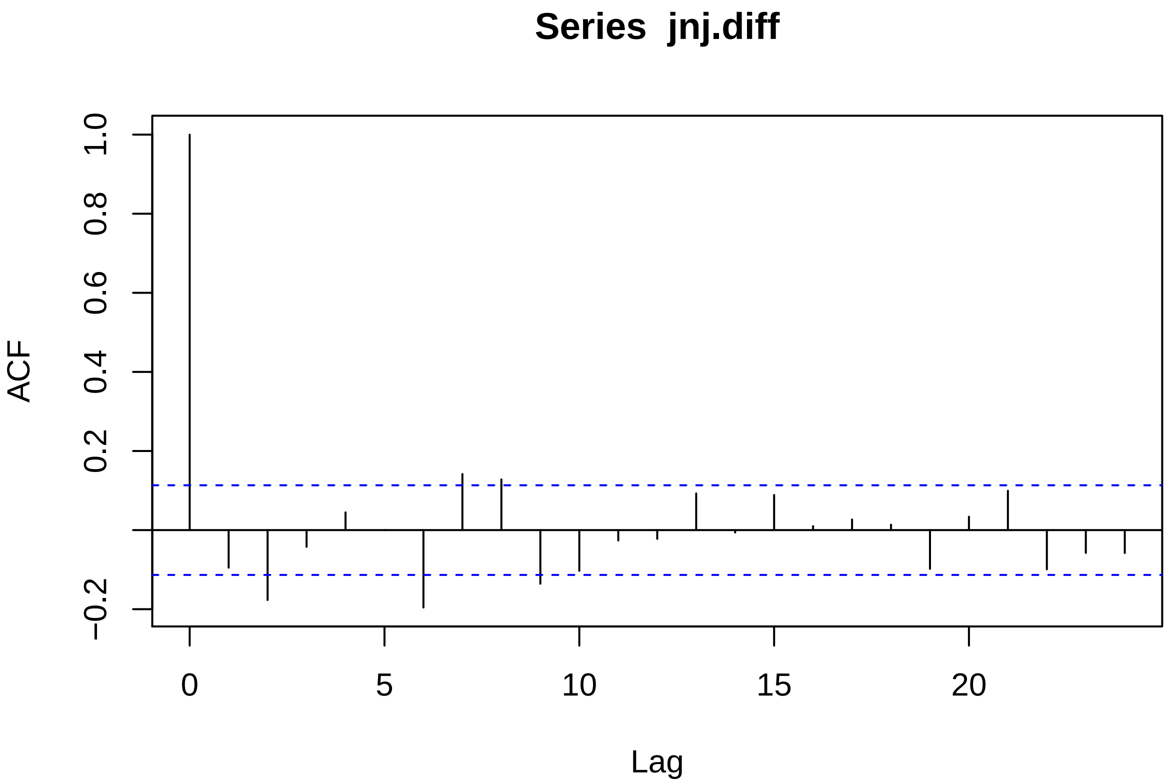
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.643  -1.995  -0.036   2.255  14.690
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.163175   0.471086   0.346 0.729305
## z.lag.1     -1.312812   0.085561 -15.343  < 2e-16 ***
## tt           0.003018   0.002731   1.105 0.270069
## z.diff.lag   0.196560   0.058333   3.370 0.000854 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.027 on 293 degrees of freedom
## Multiple R-squared:  0.5607, Adjusted R-squared:  0.5562
## F-statistic: 124.6 on 3 and 293 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -15.3435 78.4894 117.7205
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.98 -3.42 -3.13
## phi2  6.15  4.71  4.05
## phi3  8.34  6.30  5.36
```

The three test-statistic are within the “reject” zone, so the serie is staionnary but with less than 1% of error, The p-value of the linear trend is 0.27 so we fail to reject the null hypothesis of the coeff, the model has not linear trend same thing for the constant, we can’t reject the null hypothesis.

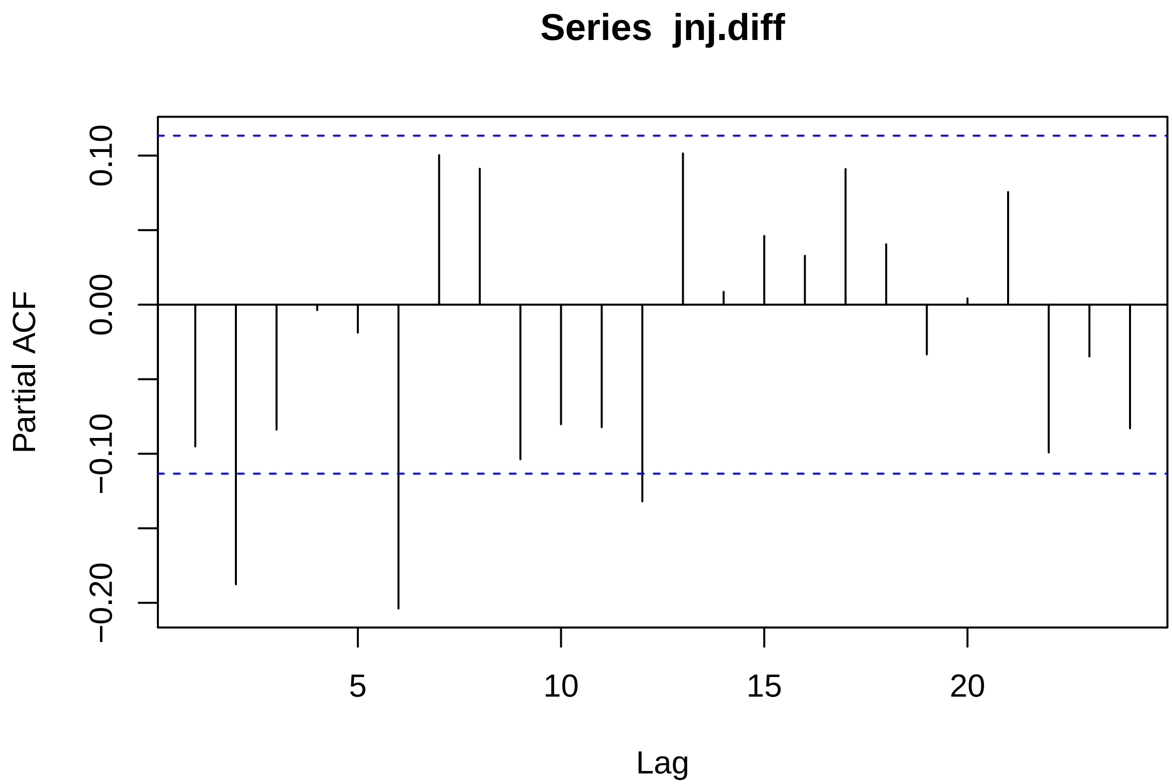
The time serie is I(1)

2. Determine the order of the required ARIMA model, i.e the values of p,d,q.

```
par(mfrow=c(1,1))
acf(jnj.diff)
```




```
pacf(jnj.diff)
```



We observe 2 pics out of the significance limit, for the lag 2 and 6

p = 2 or p =6, we will choose 2 to respect the parsimony of the model

We observe 3 pics out of the significance limit, for the lag 2,6 and 7

q = 2 or q = 9 we will choose 2 to respect the parsimony of the model

```
#determiner (p,q) avec les criteres d'informations
library(qpcR)
model = arima(jnj.diff,order=c(1,0,0))
min_AIC = c(AIC(model),1,0)
min_BIC=c(BIC(model),1,0)
for(p in 1:8)
{
  for(q in 1:8)
  {
    model = arima(jnj.diff,order=c(p,0,q))
    ci1 = AIC(model)
    ci2 = BIC(model)
    if (ci1<min_AIC[1])
    {
      min_AIC=c(ci1,p,q)
    }
    if (ci2<min_BIC[1])
    {
      min_BIC=c(ci2,p,q)
    }


  }
```

```
}
```

```
## Warning in arima(jnj.diff, order = c(p, 0, q)): possible convergence problem:
## optim gave code = 1

## Warning in log(s2): production de NaN

## Warning in log(s2): production de NaN

## Warning in log(s2): production de NaN

## Warning in arima(jnj.diff, order = c(p, 0, q)): possible convergence problem:
## optim gave code = 1

## Warning in arima(jnj.diff, order = c(p, 0, q)): possible convergence problem:
## optim gave code = 1
```

```
cat("Akaike criteria : p =",min_AIC[2], "and q =", min_AIC[3])
```

```
## Akaike criteria : p = 7 and q = 4
```

```
cat("Baysian criteria : p =",min_BIC[2], "and q =", min_BIC[3])
```

```
## Baysian criteria : p = 1 and q = 1
```

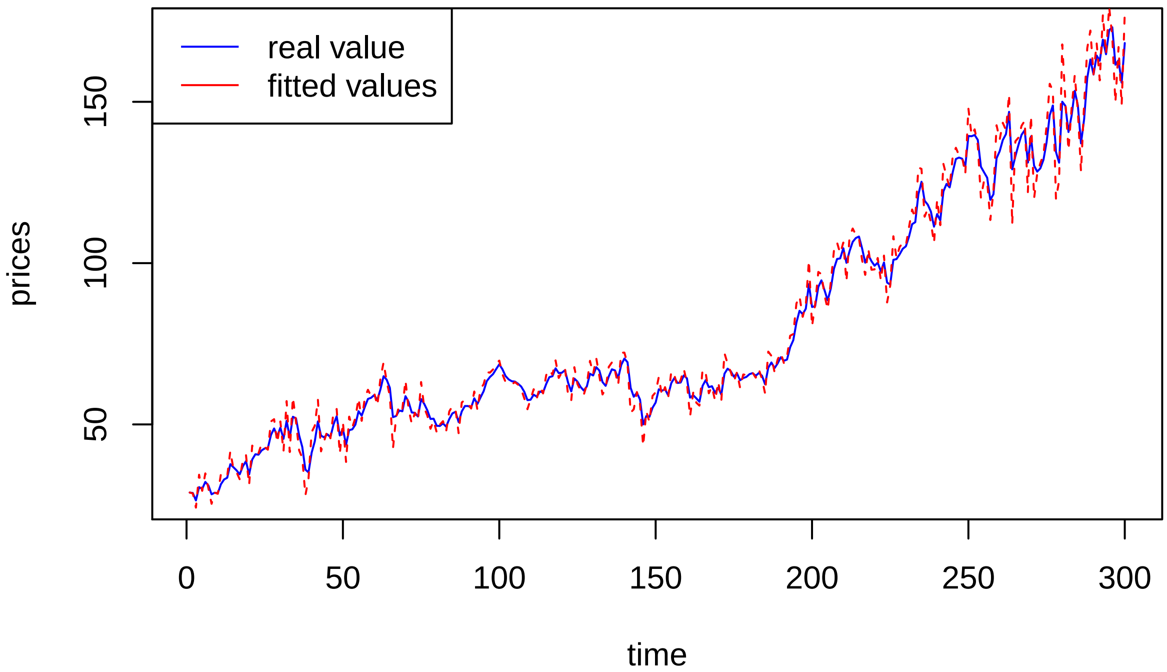
We choose to keep p=1 and q=1 to respect the parsimony of the model The final model is an ARIMA(1,1,1)

3. Estimating our ARIMA(1,1,1)

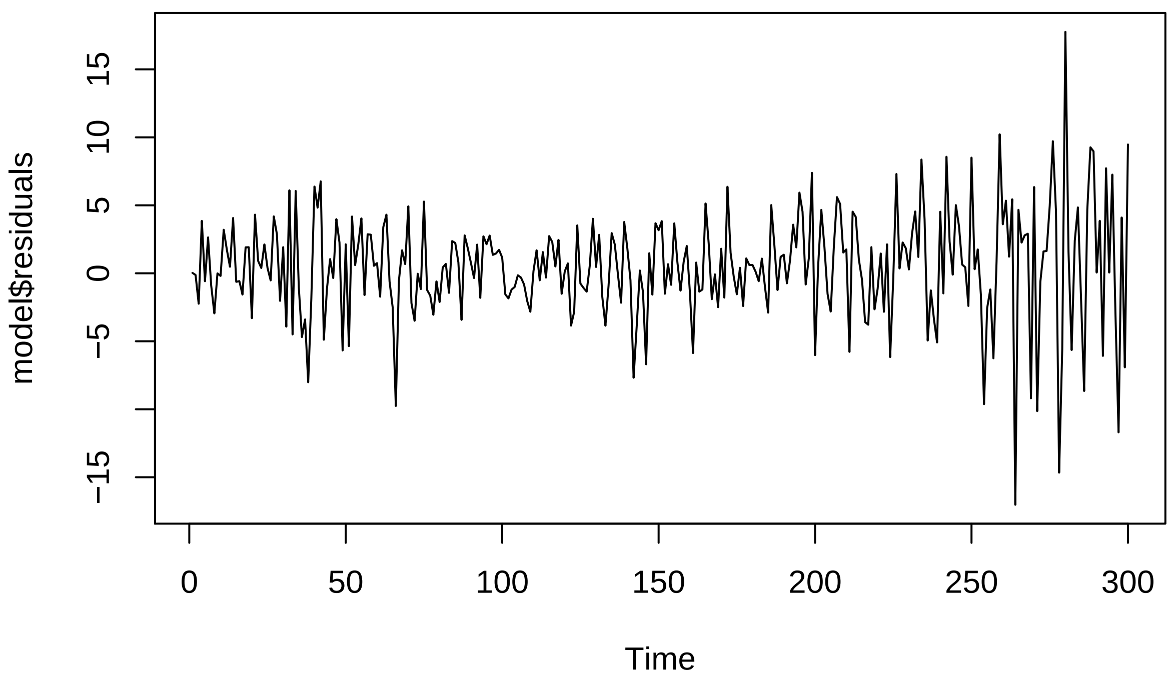
```
model = arima(jnj$close,order=c(2,1,2))
model
```

```
##
## Call:
## arima(x = jnj$close, order = c(2, 1, 2))
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       0.1158  -0.9490  -0.1828  0.8959
## s.e.  0.0752   0.0528   0.0846  0.0905
##
## sigma^2 estimated as 15.91:  log likelihood = -838.27,  aic = 1686.54
```

```
plot(jnj$close,type="l",ylab="prices",xlab="time",col="blue")
points(jnj$close+model$residuals,type="l",col="red",lty=2)
legend("topleft",legend=c("real value","fitted values"),col=c("blue","red"),lty=1)
```




```
plot(model$residuals)
```



4. Check if the residuals are gaussian

```
jarque.bera.test(model$residuals) #p-value < $2^{-16}$ we reject Ho :residuals have a normal kurtosis
```

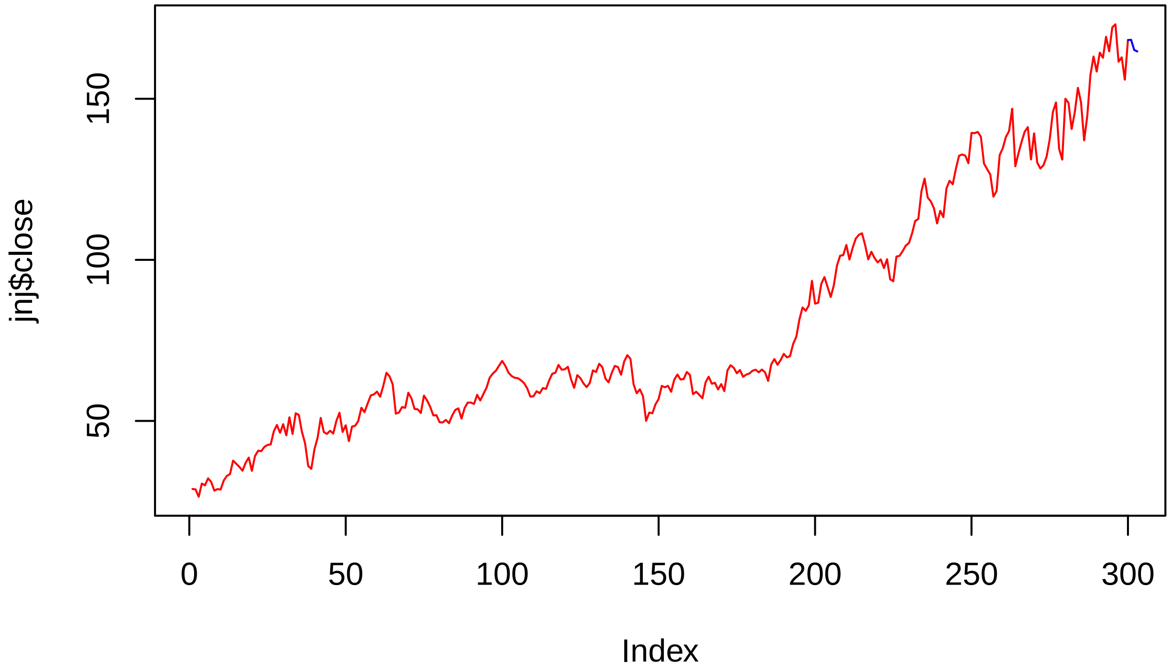
```
##
##  Jarque Bera Test
##
## data:  model$residuals
```

```
## X-squared = 103.79, df = 2, p-value < 2.2e-16
```

Maybe because of the structural break of the time serie

5. Forecast

```
jnj.forecasting = predict(model,n.ahead=3)
jnj.forecasting = c(jnj[300,]$close,jnj.forecasting$pred)
plot(jnj$close,type="l",col="red")
points(c(300,301,302,303),jnj.forecasting,col="blue",type="l")
```

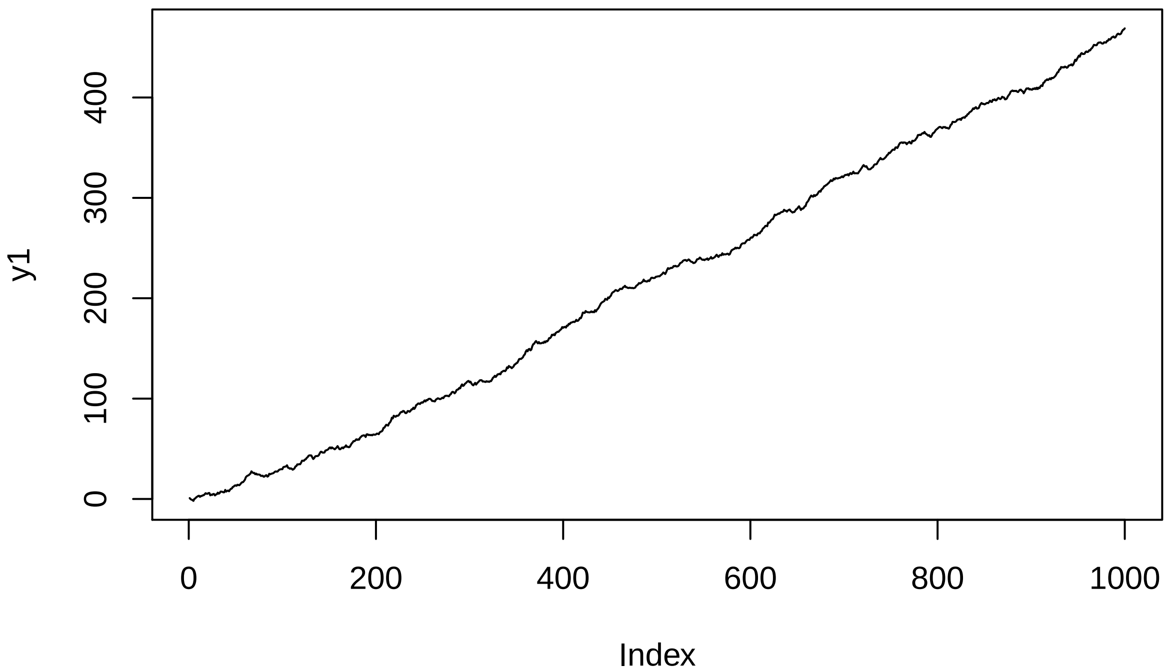


Exercise 4 : Unit root test for another one

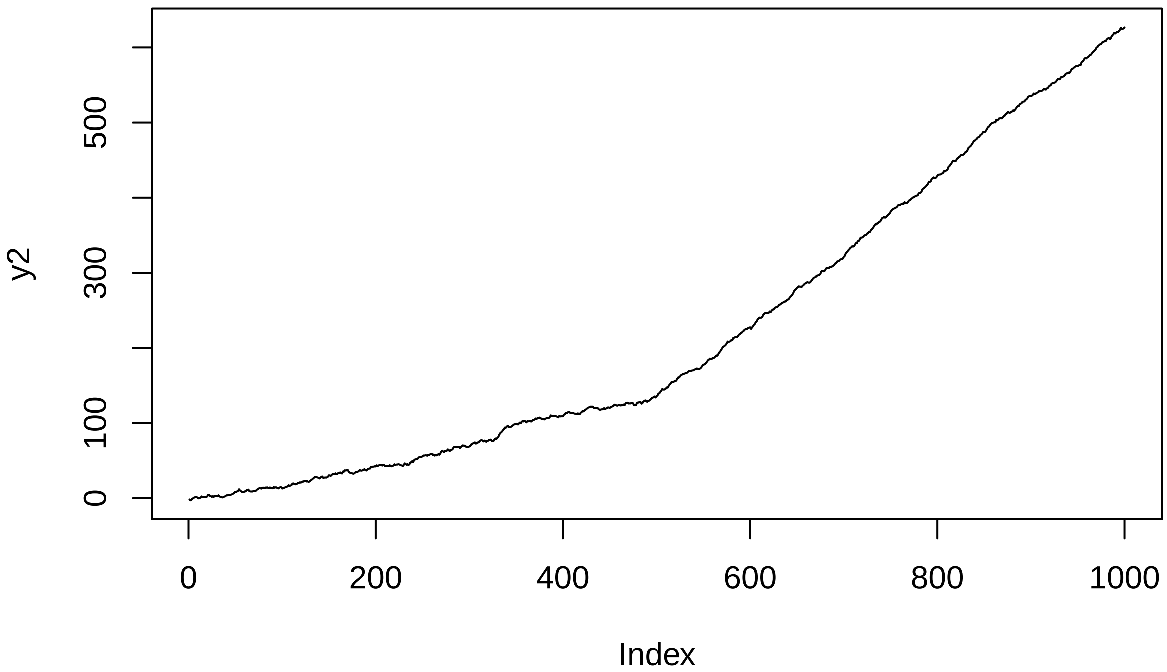
2. Generating 3 differents random walk

```
N = 1000
u1 = rnorm(N) # Simulation of 1000 Gaussian observations
u2 = rnorm(N) # Simulation of 1000 Gaussian observations
u3 = rnorm(N) # Simulation of 1000 Gaussian observationsy1=c()
y1=c()
y2=c()
y3=c()
y1[1]=u1[1] # First random walk without break
y2[1]=u2[1] # with level break
y3[1]=u3[1] # with trend and level break

for (k in 2:(N/2))
{
  y1[k] = y1[k-1] + 0.5 +u1[k]
  y2[k]= y2[k-1]+0.3+u2[k]
```

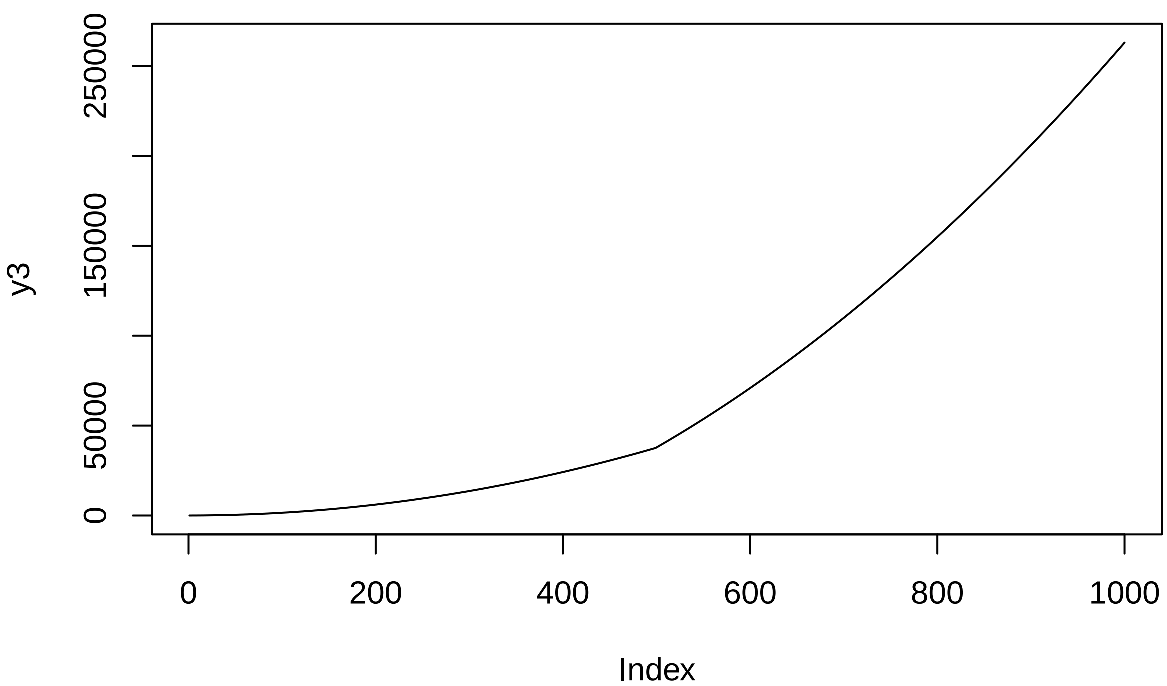
```
  y3[k]=y3[k-1]+0.2+0.3*k + u3[k]

}
for (k in (N/2):N)
{
  y1[k] = y1[k-1] + 0.5 +u1[k]
  y2[k]= y2[k-1]+1+u2[k]
  y3[k]=y3[k-1]-0.1+0.6*k + u3[k]

}
par(mfrow=c(1,1))
plot(y1,type="l")
```




```
plot(y2,type="l")
```




```
plot(y3,type="l")
```



3. Compute the Zivot and Andrews test

```
summary(ur.za(y1))
```

```
##
## ################################
## # Zivot-Andrews Unit Root Test #
## ################################
```

```
##
##
## Call:
## lm(formula = testmat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.69106 -0.68370  0.02163  0.71913  3.10009
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.185633   0.132727   1.399  0.16224
## y.l1         0.974791   0.005783 168.562  < 2e-16 ***
## trend        0.012551   0.002838   4.422 1.09e-05 ***
## du          -0.498596   0.178264  -2.797  0.00526 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.003 on 995 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 6.46e+06 on 3 and 995 DF,  p-value: < 2.2e-16
##
##
## Teststatistic: -4.3591
## Critical values: 0.01= -5.34 0.05= -4.8 0.1= -4.58
##
## Potential break point at position: 67
```

```r
summary(ur.za(y2,model="intercept"))
```

```
##
## ################################
## # Zivot-Andrews Unit Root Test #
## ################################
##
##
## Call:
## lm(formula = testmat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2303 -0.6535  0.0066  0.6663  2.8054
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.0330141  0.0839568    0.393    0.694
## y.l1         0.9979744  0.0008021 1244.220  < 2e-16 ***
## trend        0.0027199  0.0006109    4.452 9.45e-06 ***
## du          -0.3844252  0.1446615   -2.657    0.008 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9468 on 995 degrees of freedom
##   (1 observation deleted due to missingness)
```

```
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.366e+07 on 3 and 995 DF,  p-value: < 2.2e-16
##
##
## Teststatistic: -2.5254
## Critical values: 0.01= -5.34 0.05= -4.8 0.1= -4.58
##
## Potential break point at position: 167
```

```
summary(ur.za(y3,model="both"))
```

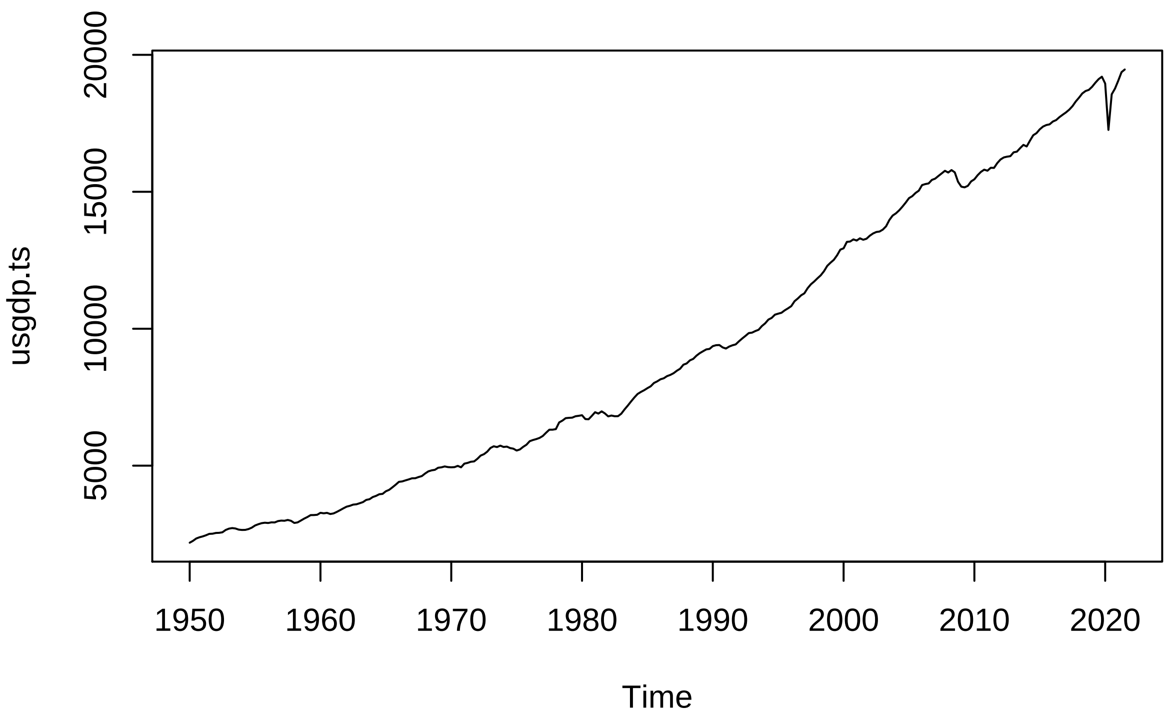
```
##
## ################################
## # Zivot-Andrews Unit Root Test #
## ################################
##
##
## Call:
## lm(formula = testmat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -88.915  -7.008  -1.734   7.948  58.830
##
## Coefficients:
##               Estimate Std. Error   t value Pr(>|t|)
## (Intercept) -1.141e+01  2.083e+00    -5.480 5.39e-08 ***
## y.l1         9.972e-01  8.325e-05 11978.251  < 2e-16 ***
## trend        4.712e-01  1.018e-02    46.280  < 2e-16 ***
## du          -2.279e+01  3.270e+00    -6.971 5.73e-12 ***
## dt           1.469e+00  3.080e-02    47.689  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.43 on 994 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.668e+09 on 4 and 994 DF,  p-value: < 2.2e-16
##
##
## Teststatistic: -33.8673
## Critical values: 0.01= -5.57 0.05= -5.08 0.1= -4.82
##
## Potential break point at position: 401
```

4. Is it relevant to use such test for the US GDP? Compute the Zivot and Andrews unit root test using the US GDP

Exercise 4.2 : Modeling

Order of integration

```
plot(usgdp.ts)
```



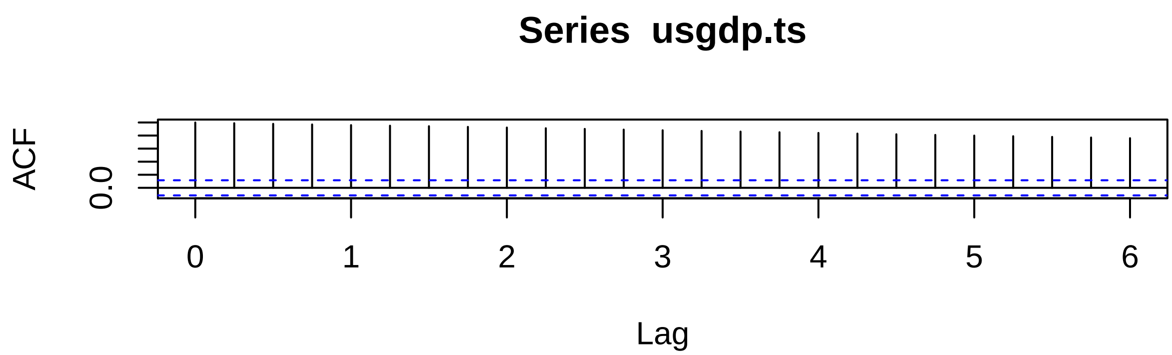

```
summary(ur.df(usgdp.ts,type="drift",selectlags="AIC"))
```

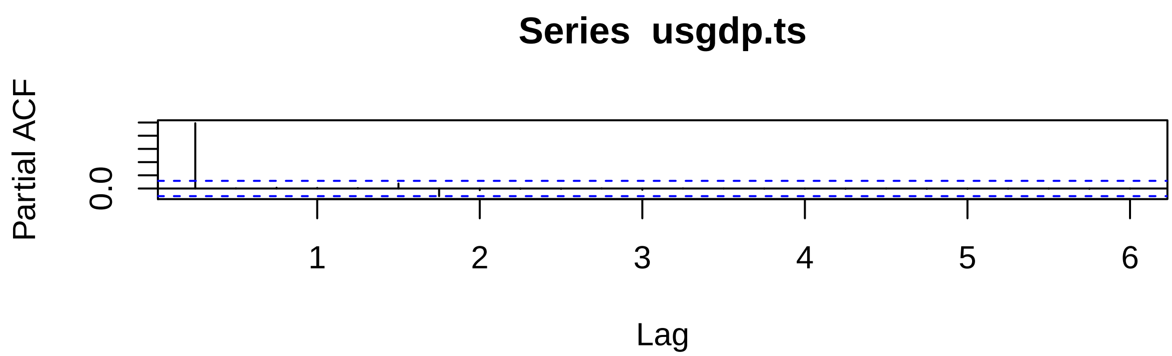
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1838.99   -31.15     7.24    43.45   944.35
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.123904  17.544786   1.945   0.0528 .
## z.lag.1      0.003852   0.001672   2.304   0.0220 *
```

```
## z.diff.lag  -0.152089   0.059070  -2.575   0.0105 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 145.2 on 282 degrees of freedom
## Multiple R-squared:  0.03581,    Adjusted R-squared:  0.02897
## F-statistic: 5.236 on 2 and 282 DF,  p-value: 0.00585
##
##
## Value of test-statistic is: 2.3039 29.7293
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.44 -2.87 -2.57
## phi1  6.47  4.61  3.79
```

We can't reject Ho : The presence of a unit root, hence the serie is a random walk

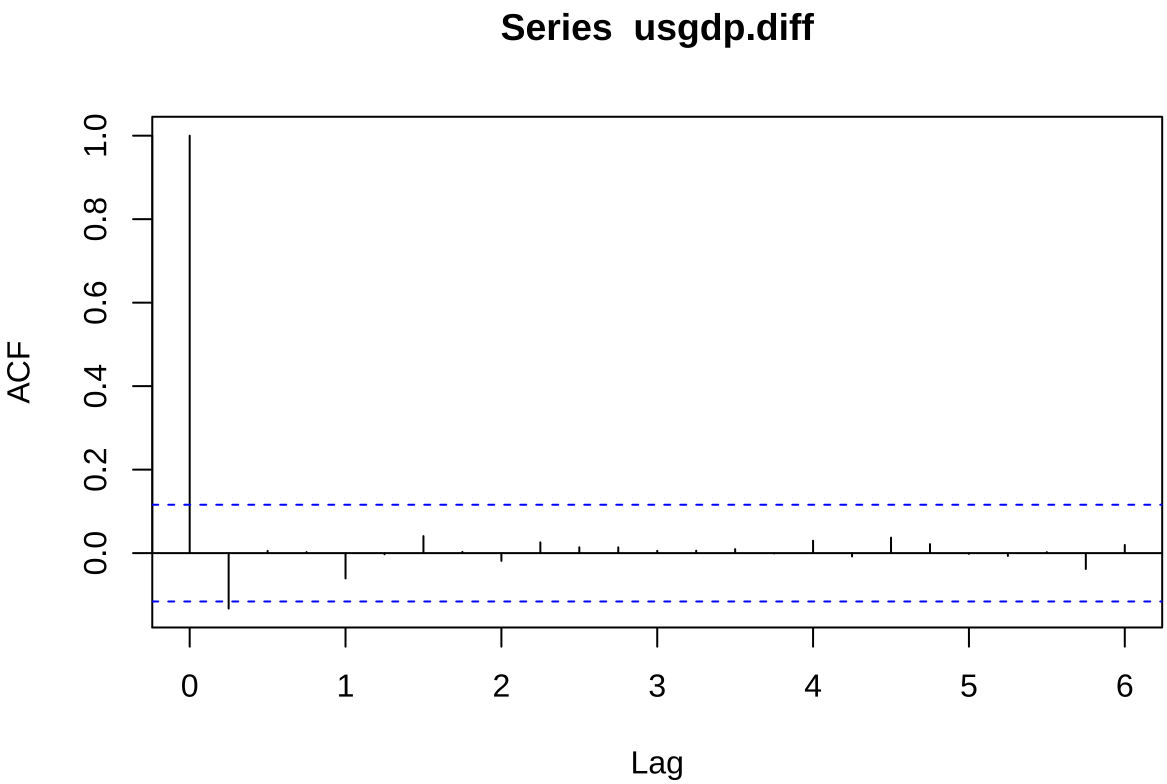
```
par(mfrow=c(2,1))
acf(usgdp.ts) # q=0
pacf(usgdp.ts) # p  =1
```



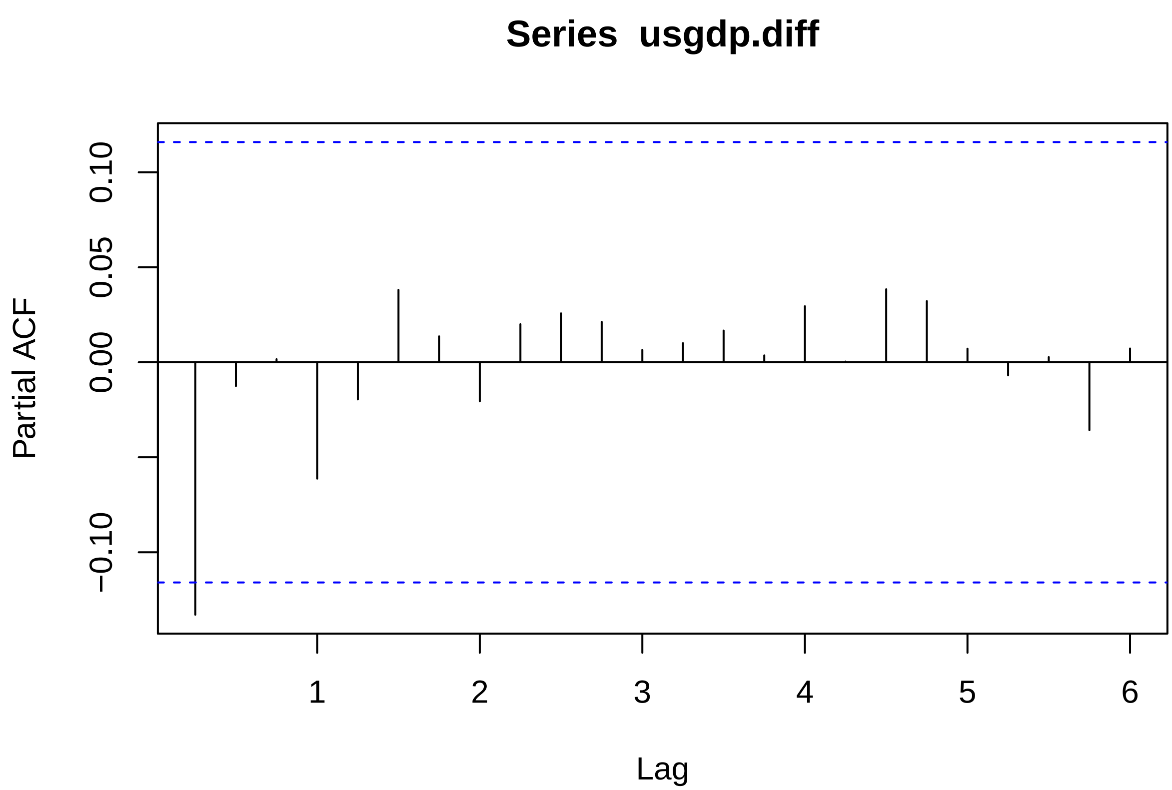


The dickey fuller result is confirmed by the autocorrelograms

```
usgdp.diff= diff(usgdp.ts) # We differentiate the serie one time
acf(usgdp.diff)
```




```
pacf(usgdp.diff)
```



Autocorrelograms seems to indicate the stationnarity of the time serie

```
summary(ur.df(usgdp.diff,type="drift",selectlags="AIC"))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1795.52   -40.67     4.02    49.59  1002.08
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 69.18062   10.24577   6.752 8.34e-11 ***
## z.lag.1     -1.14755    0.09014 -12.730  < 2e-16 ***
## z.diff.lag   0.01289    0.05998   0.215     0.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146.8 on 281 degrees of freedom
## Multiple R-squared:  0.5665, Adjusted R-squared:  0.5634
## F-statistic: 183.6 on 2 and 281 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.7305 81.0345
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.44 -2.87 -2.57
## phi1  6.47  4.61  3.79
```

We now can reject Ho : there is a unit root and accept H1 : the serie is stationnary

Conclusion : usGDP is I(1)

ARMA(p,q) analyse

```
model = arima(usgdp.ts,order=c(1,1,0),method="ML")
min_AIC = c(AIC(model),1,0)
min_BIC=c(BIC(model),1,0)
for(p in 1:4)
{
  for(q in 1:4)
  {
    model = arima(usgdp.ts,order=c(p,1,q),method="ML")
    ci1 = AIC(model)
    ci2 = BIC(model)
```

```
    if (ci1<min_AIC[1])
    {
      min_AIC=c(ci1,p,q)
    }
    if (ci2<min_BIC[1])
    {
      min_BIC=c(ci2,p,q)
    }


  }
}
```

```
## Warning in arima(usgdp.ts, order = c(p, 1, q), method = "ML"): possible
## convergence problem: optim gave code = 1
```

```
cat("Akaike criteria : p =",min_AIC[2], "and q =", min_AIC[3])
```

```
## Akaike criteria : p = 1 and q = 2
```

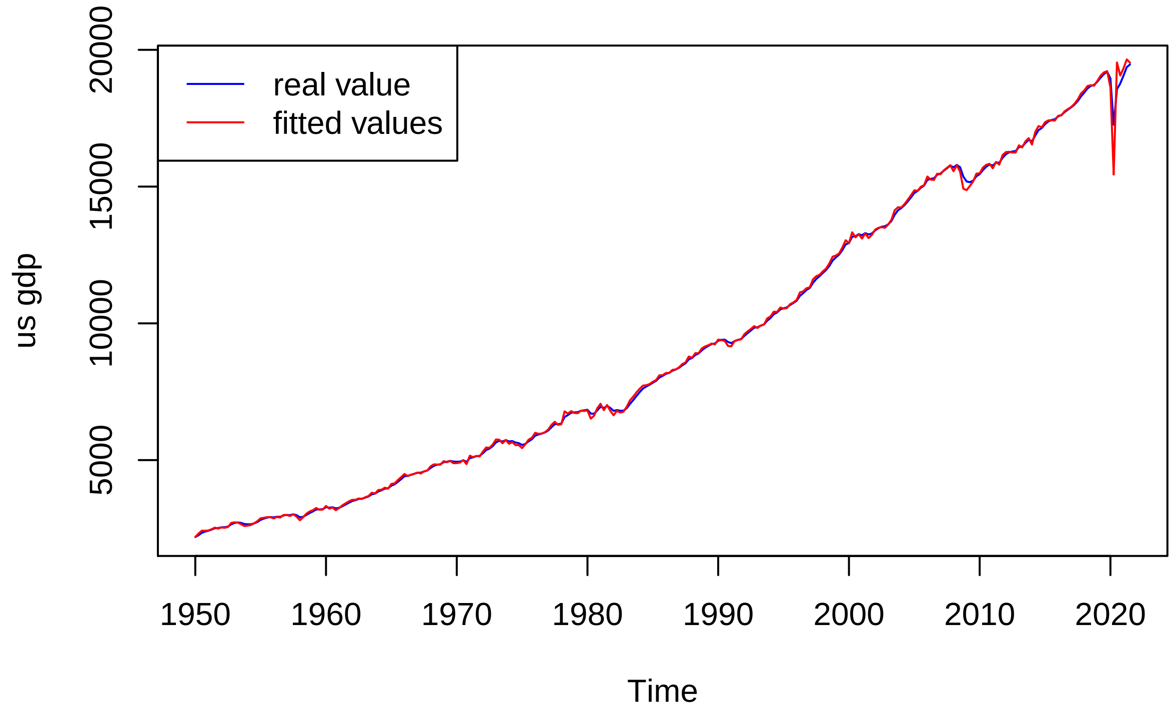
```
cat("Baysian criteria : p =",min_BIC[2], "and q =", min_BIC[3])
```

```
## Baysian criteria : p = 1 and q = 2
```

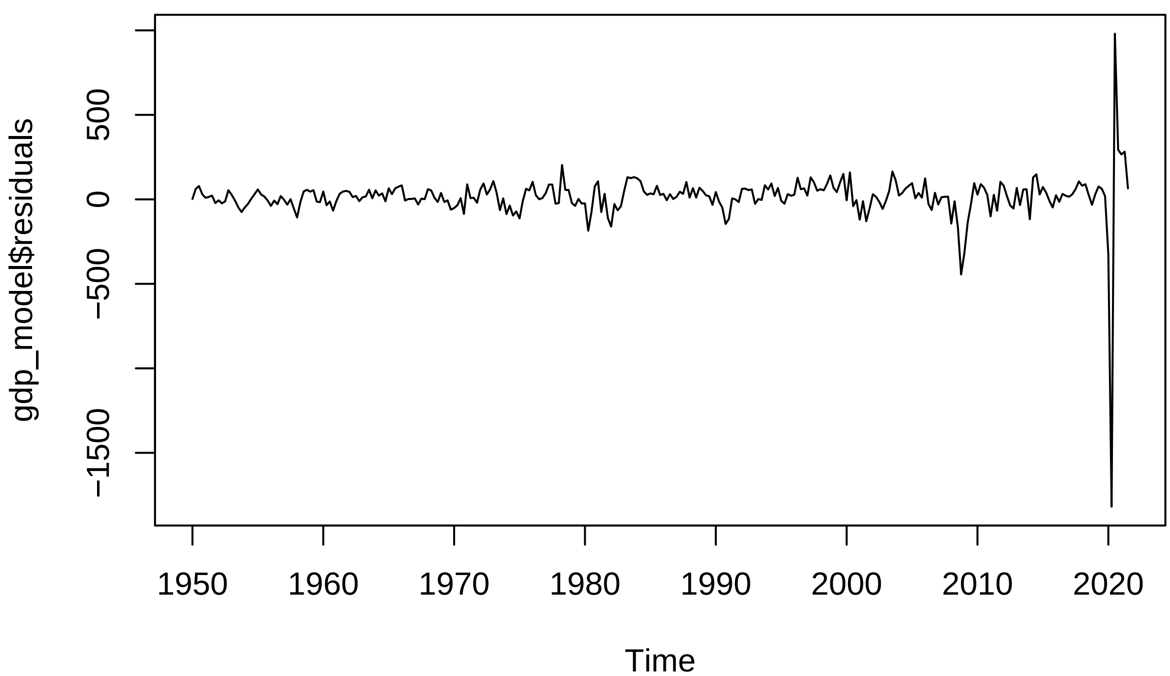
Conclusion: we choose an ARIMA(1,1,2) to modelize GDP

Quality Check

```
par(mfrow=c(1,1))
gdp_model = arima(usgdp.ts,order=c(1,1,2),method="ML")
plot(usgdp.ts,ylab="us gdp",col="blue")
points(usgdp.ts+gdp_model$residuals,col="red",type="l")
legend("topleft",legend=c("real value","fitted values"),col=c("blue","red"),lty=1)
```




```
plot(gdp_model$residuals)
```



our model seems to be not really robust when an economic crash appear (problem of structural break, but I have not really understand this concept and the test of the presence of these breaks.)

```
jarque.bera.test(model$residuals)
```

```
##
##  Jarque Bera Test
##
## data:  model$residuals
```

```
## X-squared = 104215, df = 2, p-value < 2.2e-16
```

```r
jarque.bera.test(model$residuals[1:100])
```

```
##
##  Jarque Bera Test
##
## data:  model$residuals[1:100]
## X-squared = 0.52651, df = 2, p-value = 0.7685
```

Conclusion

The final test of jarque bera is the proof that our model is good in a period of non-break because residuals have a normal kurtosis but the same model is not robust when crashes appear. Actually, is period of crash our model make high error.

Maybe we should test another model with more differentiation, or another type of model which treat the influence of break on our model.