Étude de la série temporelle 16

OSMAN Ahmed

2/9/2022

Importation :

```
data <- scan("serie_16.dat")

# transformation en série temporelle
x <- ts(data[-205], frequency = 12, start = c(2000, 1), end = c(2016, 12))
```

Décomposition de la série temporelle :

Éstimation de la tendance m :

On crée un vecteur `m` tel que :
- les 6 premières et dernières coordonnées valent `NA`.
- sinon :

$$m_t = \frac{1}{12}\left(\frac{1}{2}X_{t-6} + \sum_{i=t-5}^{t+5} X_i + \frac{1}{2}X_{t+6}\right)$$

```
# Fonction qui estime la tendance
tendance_estim <- function(X) {
  m <- rep(NA, times = length(X))

  for(i in 7:(length(X)-6))
    m[i] <- (0.5*X[i-6] + sum(X[(i-5):(i+5)]) + 0.5*X[i+6])/12

  return(ts(m, frequency = 12))
}

# Estimation de notre serie
estimation_m <- tendance_estim(x)
estimation_m
```

```
##         Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1        NA       NA       NA       NA       NA       NA 351.9497 350.5489
## 2  344.6337 343.5893 343.4390 343.6190 343.2521 342.0900 342.5381 343.4889
## 3  344.9259 345.4638 345.5980 345.0488 344.7813 345.7779 346.3492 346.0856
## 4  340.8438 340.3142 339.4893 339.3178 338.7704 337.8393 337.0295 336.5217
## 5  341.5866 341.1719 341.0665 340.5389 340.1165 338.9578 337.3225 336.5810
```

```
## 6  329.2394 327.8871 326.5007 325.3449 324.3960 325.1418 326.1817 326.4827
## 7  325.6454 325.8245 326.1779 326.3250 326.8207 326.9959 326.4880 325.4460
## 8  326.7259 327.3722 327.2324 327.0921 326.3035 324.8457 324.3400 324.2212
## 9  324.9543 324.8911 325.0318 325.7741 326.2167 326.8893 327.0943 327.0099
## 10 321.2357 319.6891 318.7832 317.3191 316.7521 316.7250 316.4100 316.0398
## 11 313.0356 312.2612 312.1750 311.8546 311.1394 310.8323 311.1184 310.9703
## 12 316.7504 319.1324 320.5776 321.5535 321.8269 320.9435 319.8051 319.5970
## 13 312.3563 310.5056 309.0154 308.2838 307.7266 306.9447 306.6744 306.6978
## 14 310.3191 310.2305 309.3679 308.6590 308.7849 308.9986 309.0671 308.5734
## 15 299.7338 298.5671 298.1133 297.2008 296.4983 296.8606 296.1770 295.1389
## 16 297.4538 297.7853 297.7977 298.1761 297.7108 296.6775 296.3193 296.3974
## 17 299.2854 300.3068 301.2057 301.5568 302.3698 303.0435       NA       NA
##         Sep      Oct      Nov      Dec
## 1  349.0069 347.5610 346.3438 345.5036
## 2  343.8554 344.3768 344.8533 344.8275
## 3  345.2716 344.0256 343.1812 341.9487
## 4  337.3262 338.4945 339.7599 341.2244
## 5  335.4034 333.7442 332.0174 330.2638
## 6  326.6581 326.5623 325.2184 325.0674
## 7  325.1170 324.9966 325.7197 326.3291
## 8  323.6316 324.0134 325.0197 325.1257
## 9  327.1275 326.2257 324.6070 323.2589
## 10 315.3641 315.1783 315.0135 313.9262
## 11 311.0315 311.4824 312.4145 314.4140
## 12 318.7300 317.4668 315.9187 314.0494
## 13 307.4112 308.3116 309.4229 310.0395
## 14 306.9551 304.8962 302.8419 301.4537
## 15 295.6877 296.1723 296.0985 296.3951
## 16 296.0291 296.3571 297.6409 298.5533
## 17       NA       NA       NA       NA
```

Estimation de la saisonnalité s :

- On crée un vecteur w_k qu'on le recentra pour estimer la saisonnalité, avec w_k :

$$w_k = \frac{1}{17}\sum_{j=1}^{17}\left(X_{k+12(j-1)} - m_{k+12(j-1)}\right)$$

```r
saisonalite_estim <- function(X) {
  m <- tendance_estim(X)
  w <- rep(NA, times = 12)

  for (k in 1:12) {
    ind <- k+12*((1:17)-1)
    w[k] <- mean(x[ind] - m[ind], na.rm=T)
  }

  s <- w - mean(w)
  s <- ts(rep(s, 15), frequency = 12)
  return(s)
}
```

```r
estimation_s <- saisonalite_estim(x)
estimation_s
```

```
##           Jan        Feb        Mar        Apr        May        Jun        Jul
## 1  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 2  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 3  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 4  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 5  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 6  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 7  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 8  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 9  -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 10 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 11 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 12 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 13 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 14 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
## 15 -4.7794929 -1.0876196 -0.8265307  1.8418969  2.2299608  5.5496442  4.3249870
##           Aug        Sep        Oct        Nov        Dec
## 1   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 2   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 3   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 4   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 5   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 6   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 7   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 8   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 9   3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 10  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 11  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 12  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 13  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 14  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
## 15  3.7991884 -0.2915365  0.3790321 -5.5119916 -5.6275381
```

```r
# plot(estimation_s)
```

Estimation du bruit :

On estime le bruit en retranchant la tendance et la saisonnalité de notre série temporelle.

```r
# Bruit
estimation_z <- c(x) - c(estimation_m) - c(estimation_s)
estimation_z <- ts(estimation_z)
estimation_z
```

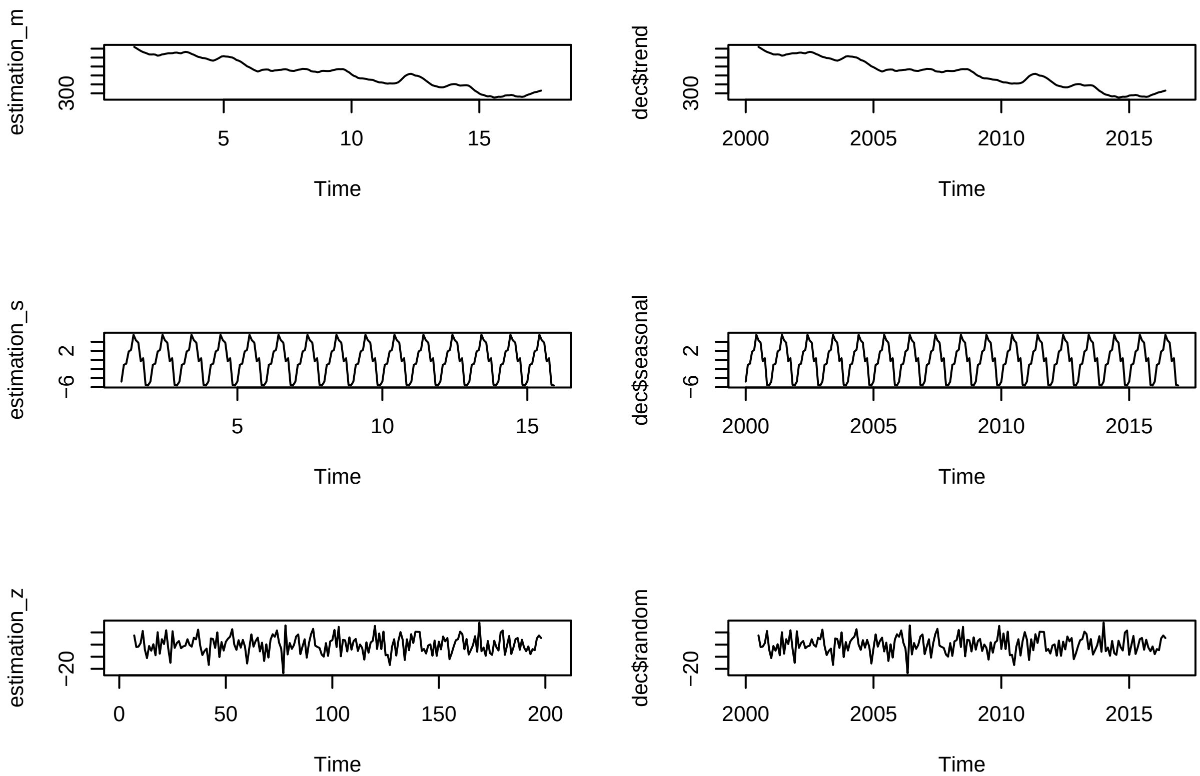
```
## Time Series:
## Start = 1
## End = 204
## Frequency = 1
```

```
##   [1]          NA          NA          NA          NA          NA
##   [6]          NA  7.56531899 -2.07122973 -1.70633159  1.22630216
##  [11] 11.18885576 -4.03733387 -11.18636417 -1.19988288 -5.07187449
##  [16]  0.26473242 -8.83373528 10.17755198 -7.52088484  4.42215218
##  [21]  0.40286562 11.77390674 -3.48948382 -15.08253640 10.99746986
##  [26] -2.73155583  1.22112160  2.89815638 -2.99054671 -1.50220602
##  [31] -0.97985328  4.38305350 -0.34844463 -1.72143221  5.61046333
##  [36]  4.28519062 12.30288815 -1.13283941 -8.65567650 -5.28730259
##  [41] -3.33027069 -16.79291169  5.05244201  4.52306374 -2.77580647
##  [46] 10.06569936 -10.36135794  2.05626047 -4.92264616  2.30464036
##  [51]  4.77999216  6.51898900 12.66599486 -0.10592998 -4.35329751
##  [56]  3.62241069 -2.54045878  3.84165948 -1.78334532 -15.62591331
##  [61] -3.17998743  8.39655677 -1.72336947  2.96215606  5.69463089
##  [66] -5.68645117  1.60009033 -13.54836050  0.20115236 -10.71144901
##  [71]  3.97735255  8.50643896  6.43517313 11.66293750  1.60626371
##  [76] -3.32479568 -23.67675305 15.78209836 -8.15729031  1.23783032
##  [81] -3.52489809 -0.34790773  6.57369527  8.35341579 -7.94570994
##  [86] -1.59091552  4.36229370 -10.79241805  0.89440648  8.50519115
##  [91] 12.94004417 -0.97599581 -1.95396892 -2.81841535 -8.19859252
##  [96] -9.95747416  1.20130261 -9.33633418  2.63942013  3.61162062
## [101] 12.04864150 -2.06355511 14.59744079 -9.69165596  3.85265169
## [106]  3.48181171 -5.67571379  5.94096495 -4.19192040  2.95271833
## [111]  4.62371045 -5.31265789  0.04466851 -2.78749596 -12.38574812
## [116]  1.82774893 -6.67486885  1.68168800  3.15857674 15.38146422
## [121] -3.65975343  9.16285548 -3.76616423 10.69060631 -8.83681819
## [126] -8.49526599 -16.86895778 -1.91283453  4.39875138 -9.05238608
## [131]  3.02112515 10.26055705  4.12602694 -12.76397707  4.35609749
## [136] -4.71102884  8.54858816  1.30670966 10.60508249 10.46745223
## [141] 10.37929839 -5.29495602 -3.66261155 -7.39757061 -0.77676851
## [146]  0.16401111 -9.19188243  3.35252051 -9.29978941  2.39231399
## [151] -3.98651035  6.67355640  2.62497903  5.37608114 -12.05563910
## [156] -7.26520380 -1.34999242  3.61143494  4.40570382 10.63748852
## [161]  8.65307535 -3.87421752  4.54253405 -8.25008315 -4.57311296
## [166] -0.56852126  6.90747075 -5.93188529 18.13113590 -5.47009415
## [171] -2.43293711 -9.22538666  2.95787966 -7.07150975 -8.51013977
## [176]  3.12553540 -2.13626248 -4.91539310  9.86216999 11.61165618
## [181] -8.47985862 -0.71307826  7.07752465 -7.76374628 -2.46205678
## [186]  4.43643033  5.43191135 -4.26046148  3.94663654 -2.44450561
## [191] -5.50078327 -1.52584857 -7.92881329 -3.74429496 -4.65804126
## [196]  5.05324838  7.49426693  5.35142995          NA          NA
## [201]          NA          NA          NA          NA
```

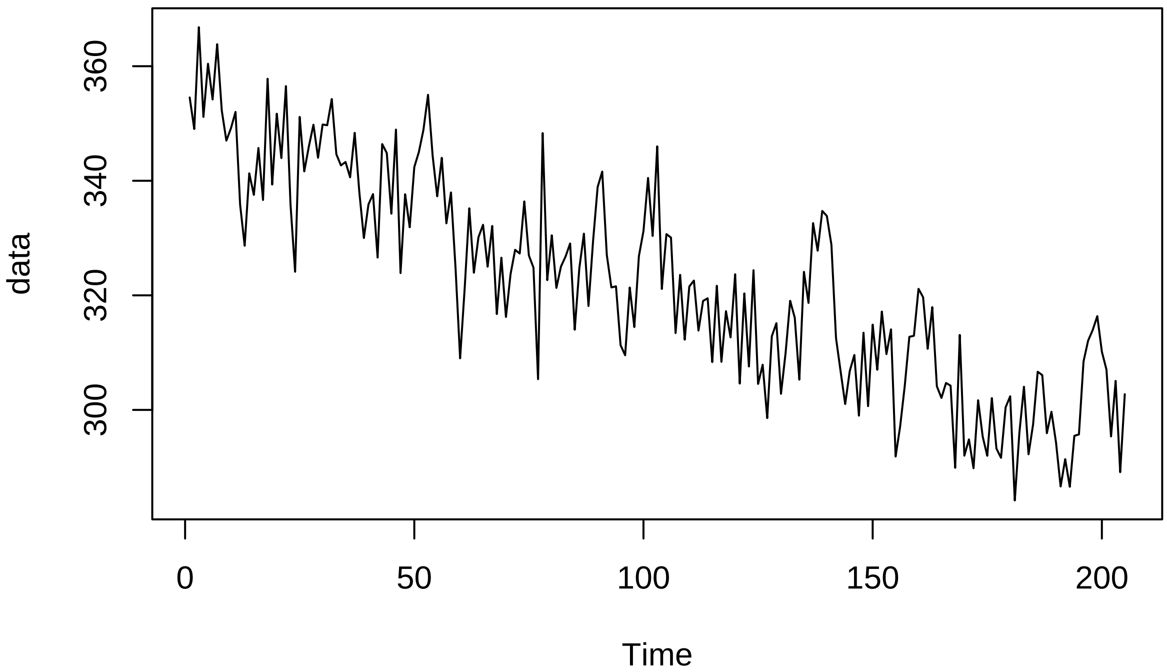
```r
# décomposition du bruit en base train et test.
n <- length(na.omit(estimation_z))

# on choisit les 6 dernière valeurs qu'on prédira
z_test <- window(estimation_z, 187, 192)

# on prend le reste comme la base d'apprentissage
z_train <- window(estimation_z, 1, 186)
```

Comparaison avec la fonction decompose

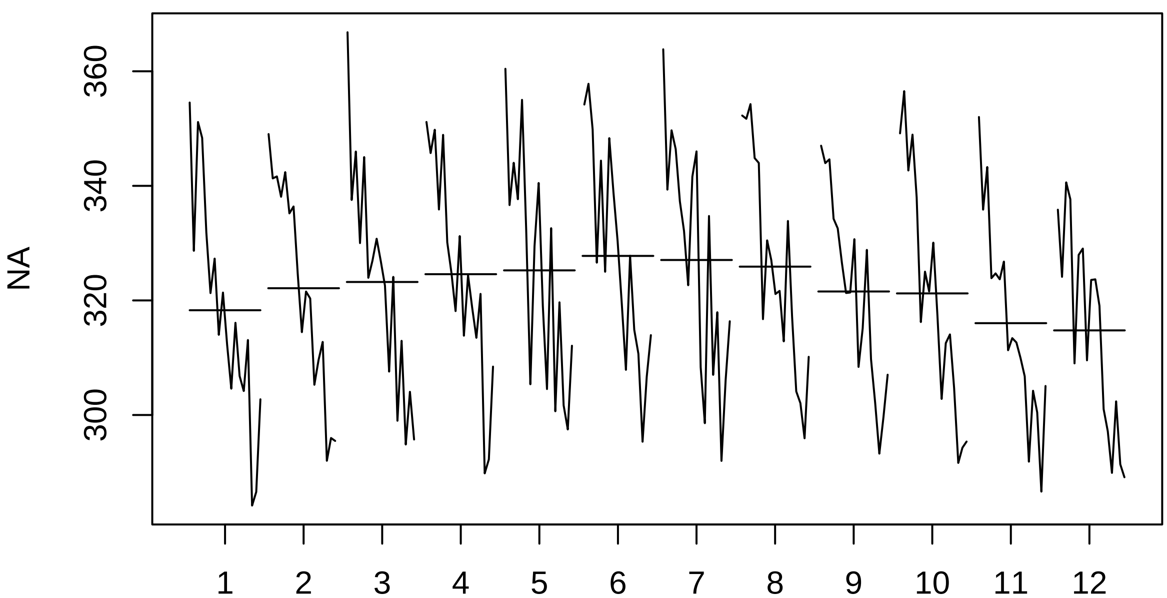
```r
dec <- decompose(x)

par(mfrow = c(3, 2))

# comparaison de la tendance m :
plot(estimation_m)
plot(dec$trend)

# comparaison de la saisonnalité s :
plot(estimation_s)
plot(dec$seasonal)

# comparaison du bruit z :
plot(estimation_z)
plot(dec$random)
```



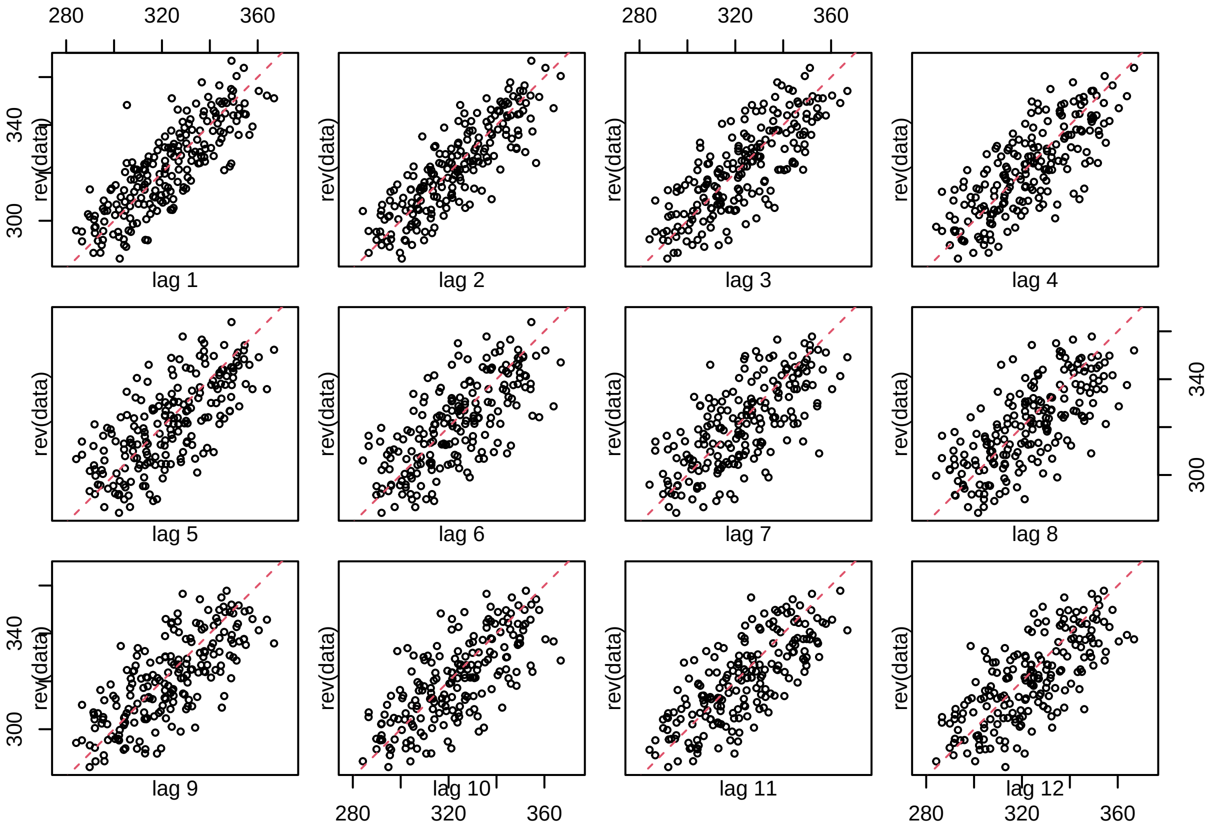

```r
ts.plot(data)
```




```r
#plot.ts(data)
monthplot(data)
```

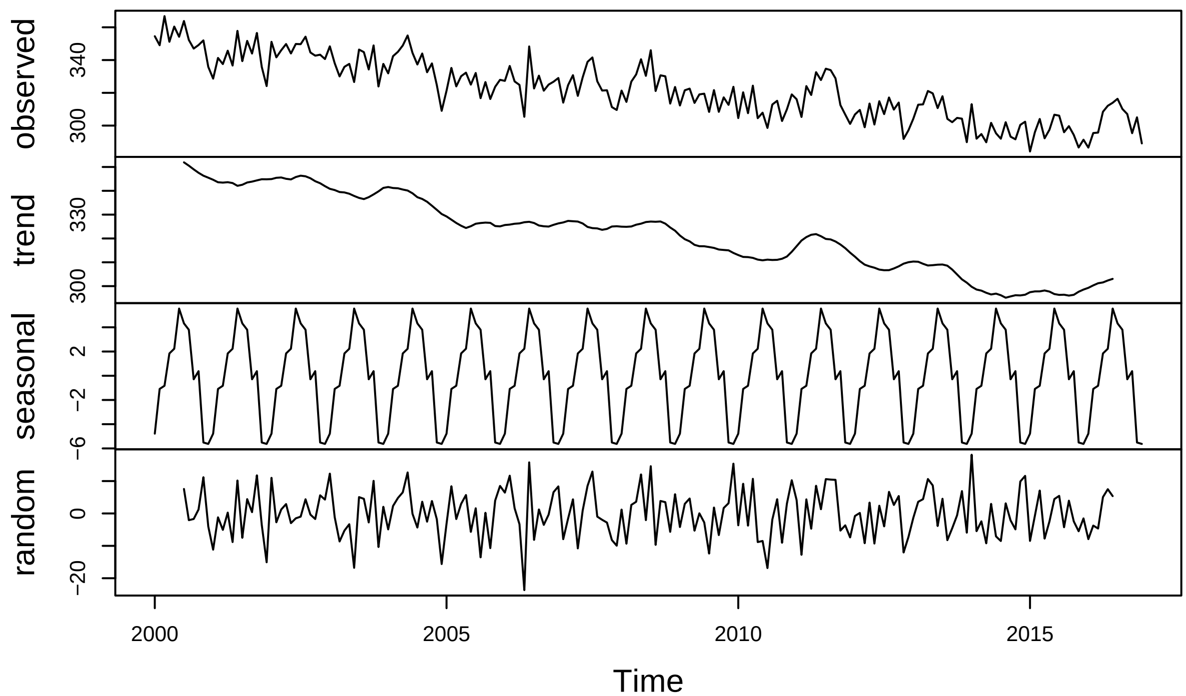



```r
lag.plot(rev(data), 12, layout = c(3, 4), diag.col = 2)
```



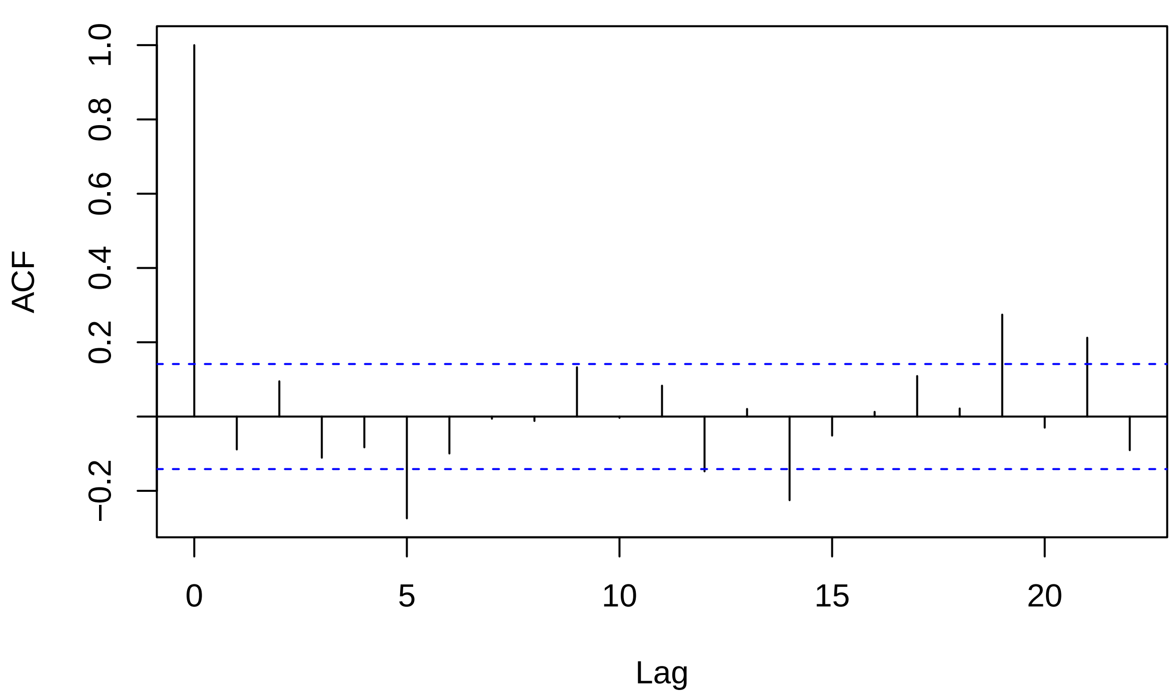

```
dec <- decompose(x)
plot(dec)
```

Decomposition of additive time series

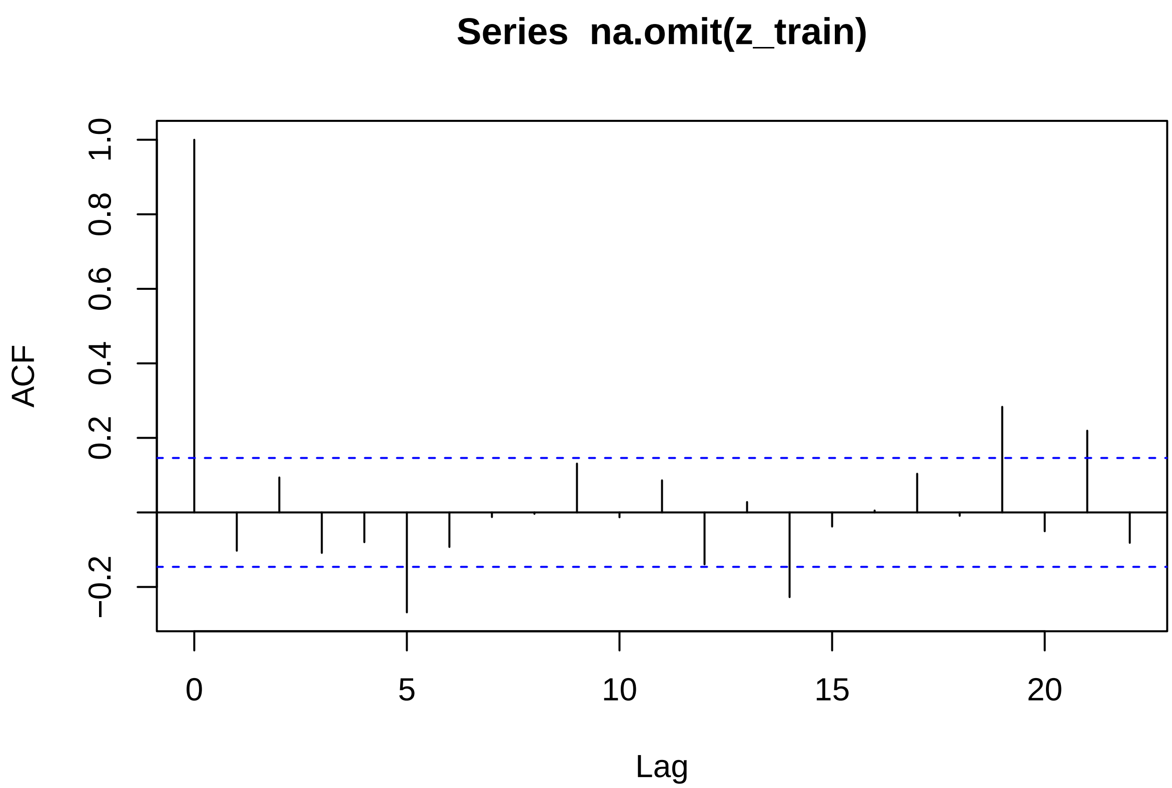



```
# stationnarité des résidus Z
acf(na.omit(estimation_z))
```

Series na.omit(estimation_z)



```r
# paril qu'avec z_train
acf(na.omit(z_train))
```

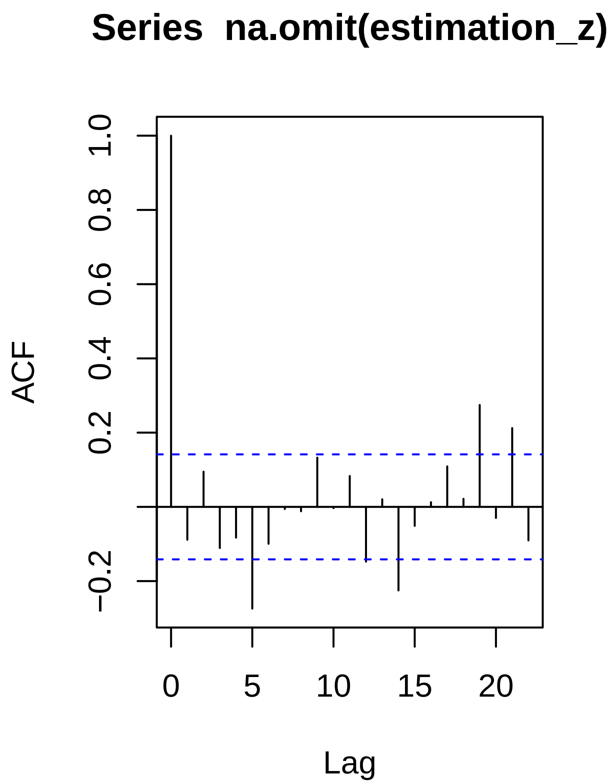


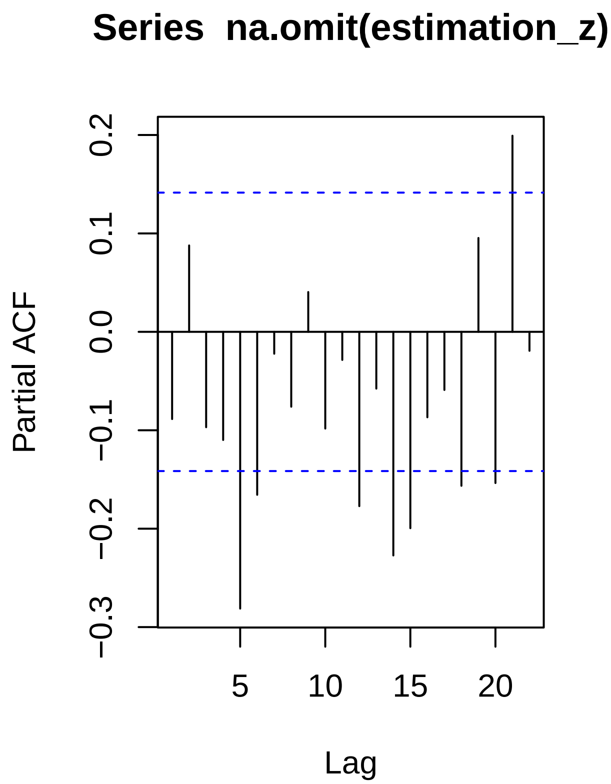

```r
# acf(na.omit(dec$random))
```

On remarque que d'après cette figure que la stationnarité Z des résidus est bien vérifiée.

Proposition et estimation des modèles ARMA pour le résidu

```r
par(mfrow = c(1, 2))

# Pour l'estimation de MA
acf(na.omit(estimation_z)) # MA(5)

# Pour l'estimation de AR
pacf(na.omit(estimation_z)) # AR(6)
```





D'après les graphes précédents, on peut proposer un modèle ARMA(6, 5).

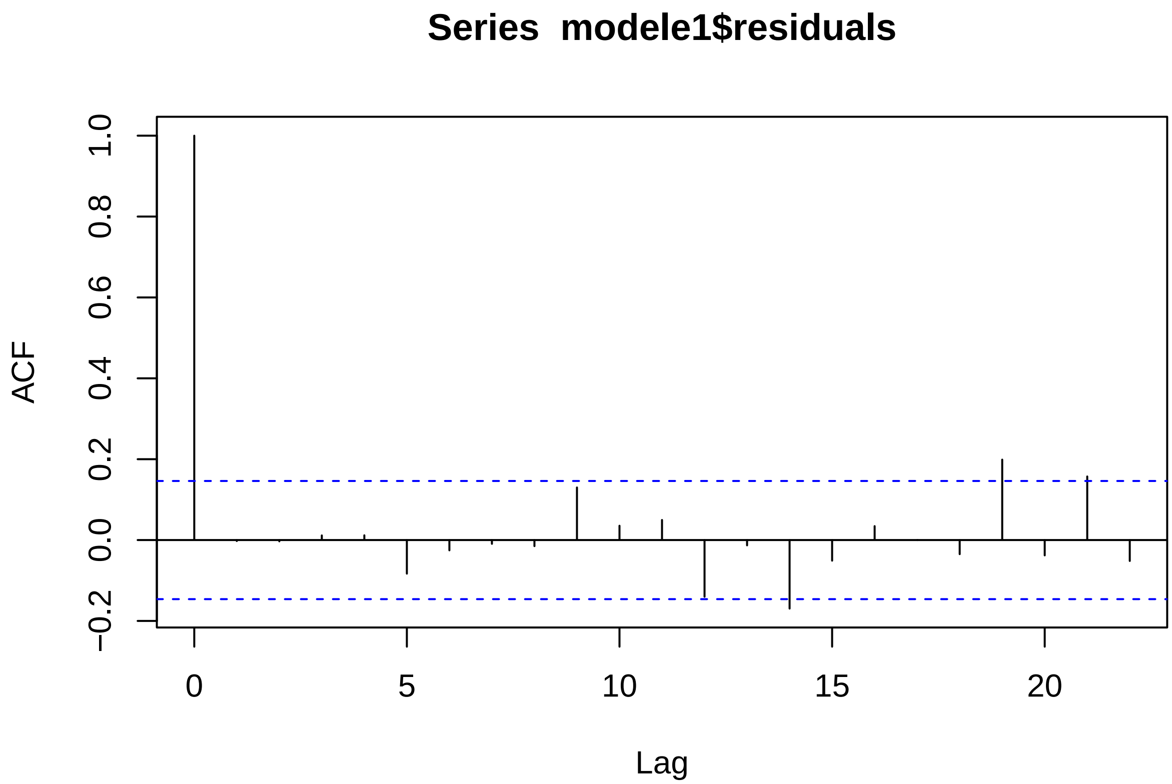
On procedera par la méthode backward et on enlevera les derniers coefficients non significatifs.

```
modele1 <- arima(na.omit(z_train), order = c(6, 0, 5))
modele1
```

```
##
## Call:
## arima(x = na.omit(z_train), order = c(6, 0, 5))
##
## Coefficients:
##          ar1      ar2     ar3     ar4      ar5     ar6      ma1     ma2
##       1.5458  -1.0509  0.1674  0.6125  -0.5098  0.0592  -1.9665  1.6278
## s.e.  0.2609   0.5276  0.5504  0.3678   0.1502  0.0885   0.2128  0.6312
##           ma3      ma4     ma5  intercept
##       -0.7827  -0.4461  0.5681     0.0000
## s.e.   0.7739   0.6345  0.2285     0.0038
##
## sigma^2 estimated as 35.62:  log likelihood = -584.45,  aic = 1194.9
```

Testons maintenant la blancheur des résidus :

```
acf(modele1$residuals)
```



On voit d'après le graphique que les résidus sont corrélés (le 19ème individu), ainsi on peut constater que les résidus ne sont pas blancs, on rejette donc le modèle.

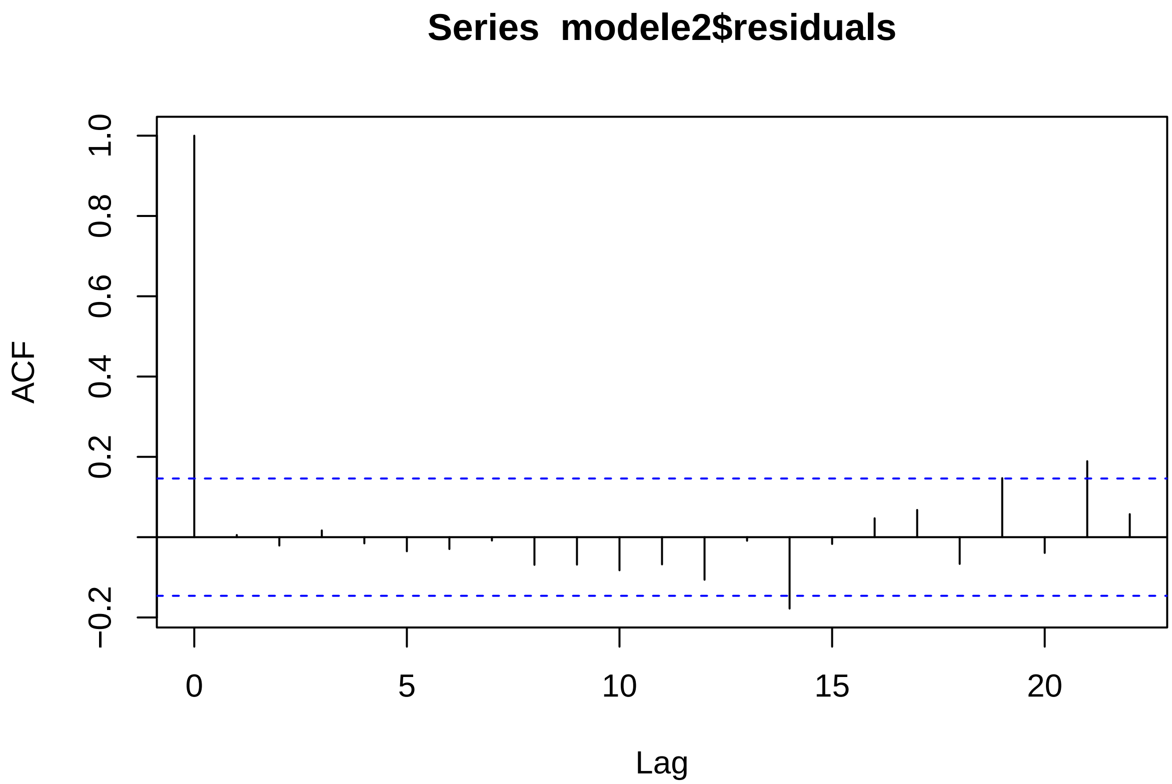
Proposons un autre modèle : ARMA(5, 5)

```
modele2 <- arima(na.omit(z_train), order = c(5, 0, 5))
modele2
```

```
##
## Call:
## arima(x = na.omit(z_train), order = c(5, 0, 5))
##
## Coefficients:
##           ar1     ar2      ar3      ar4     ar5      ma1      ma2     ma3
##       -0.0992  0.6842  -0.1642  -0.5895  0.1782  -0.1537  -0.8015  0.0945
## s.e.   0.2150  0.1418   0.2365   0.0951  0.1650   0.1967   0.1363  0.2537
##          ma4      ma5  intercept
##       0.4536  -0.5928    -0.0044
## s.e.  0.1464   0.1756     0.0229
##
## sigma^2 estimated as 37.72:  log likelihood = -586.48,  aic = 1196.95
```

Testons maintenant la blancheur des résidus :

```
acf(modele2$residuals)
```



On voit d'après le graphique que les résidus ne sont pas corrélés, ainsi on peut constater que les résidus sont blancs.

Testons la significativité des coefficients

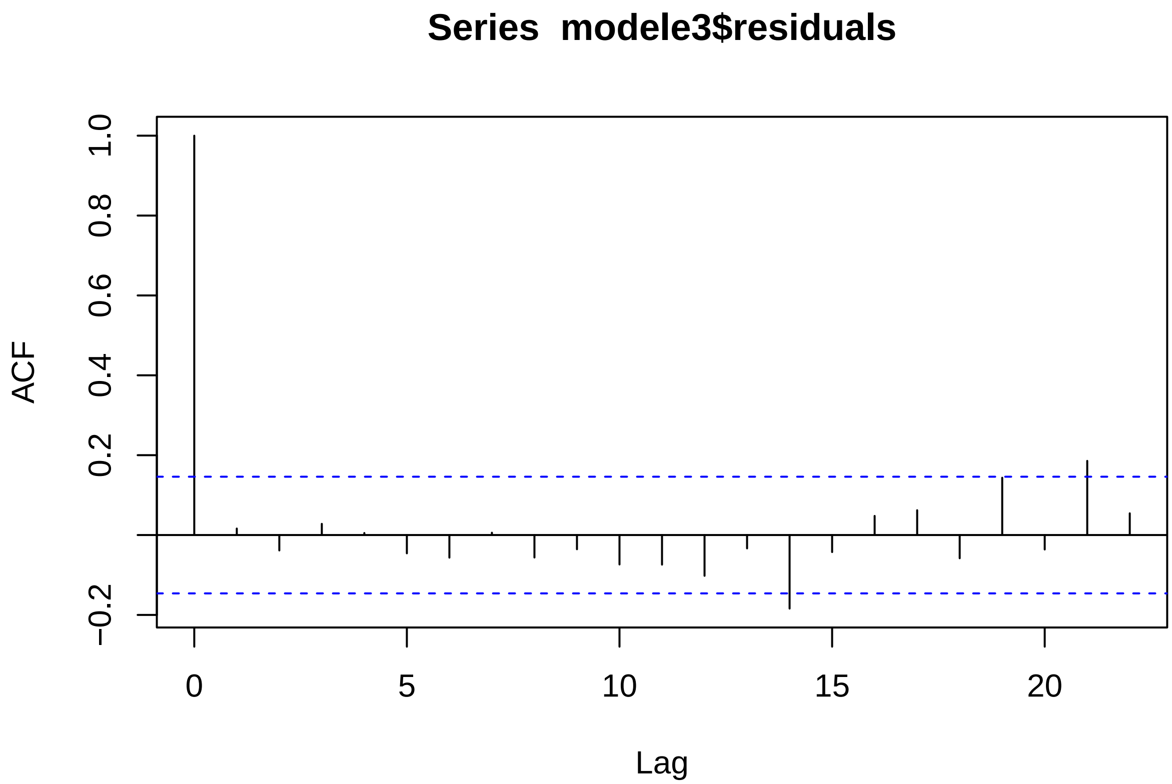
```
confint(modele2)
```

```
##                2.5 %      97.5 %
## ar1       -0.52064930  0.32230279
## ar2        0.40623488  0.96212845
## ar3       -0.62781801  0.29940528
## ar4       -0.77590043 -0.40316063
## ar5       -0.14527930  0.50169015
## ma1       -0.53932365  0.23190886
## ma2       -1.06867703 -0.53430197
## ma3       -0.40277847  0.59170492
## ma4        0.16665739  0.74050192
## ma5       -0.93697382 -0.24865060
## intercept -0.04921901  0.04044964
```

On remarque bien que 0 appartient à l'intervalle de confiance des estimateurs : ar1, ar3, ar5, ma1, ma3. ainsi ces coefficients ne sont pas significatifs.

On peut enlever le coefficient ar5 et donc proposer un nouveau modèle ARMA(4, 5)

```
modele3 <- arima(na.omit(z_train), order = c(4, 0, 5))

acf(modele3$residuals)
```



On voit d'après le graphique que les résidus ne sont pas corrélés, ainsi on peut constater que les résidus sont blancs.

Testons la significativité des coefficients

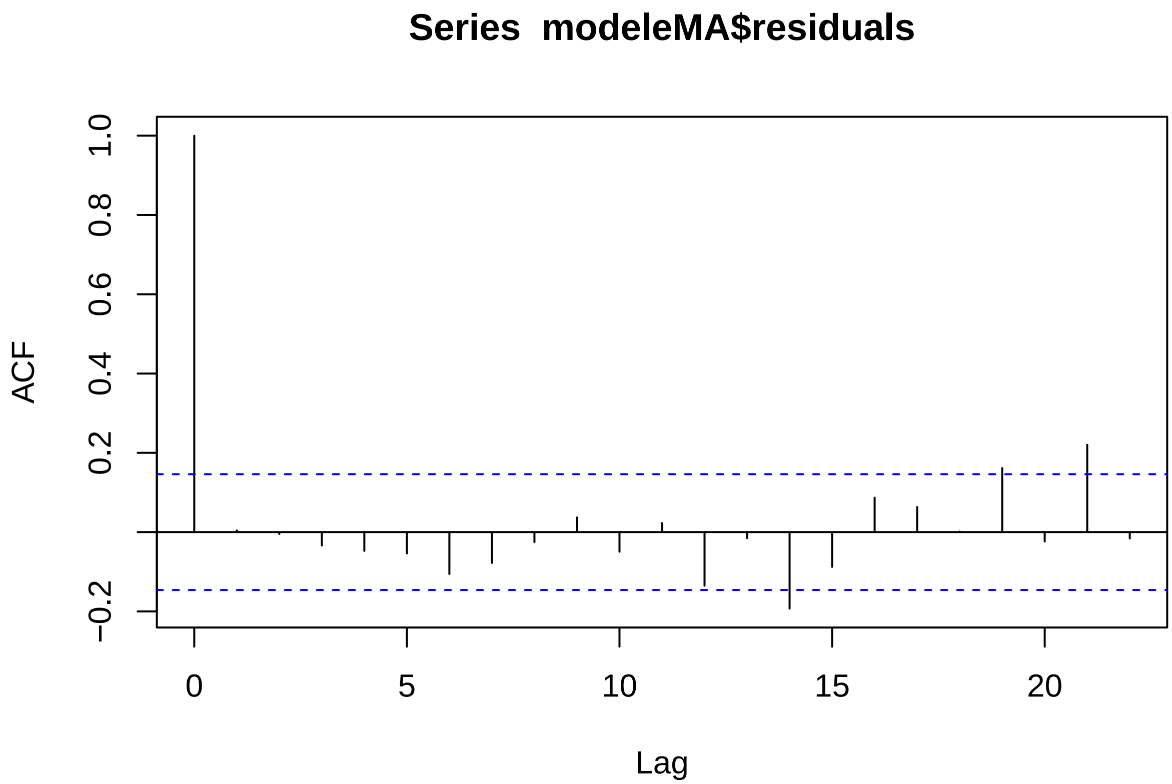
```
confint(modele3)
```

```
##                  2.5 %      97.5 %
## ar1       -0.54493396 -0.04225470
## ar2        0.40832106  0.99234761
## ar3       -0.09286460  0.28131892
## ar4       -0.75174491 -0.38106104
## ma1       -0.24258049  0.29387228
## ma2       -1.13476927 -0.57053786
## ma3       -0.46829767  0.10346776
## ma4        0.13086625  0.69472095
## ma5       -0.64267418 -0.16406058
## intercept -0.04563544  0.03751983
```

On remarque bien que 0 n'appartient à l'intervalle de confiance des estimateurs : ar4 et ma5 ainsi ces coefficients sont pas significatifs.

On peut aussi proposer un autre modèle comme MA(5) :

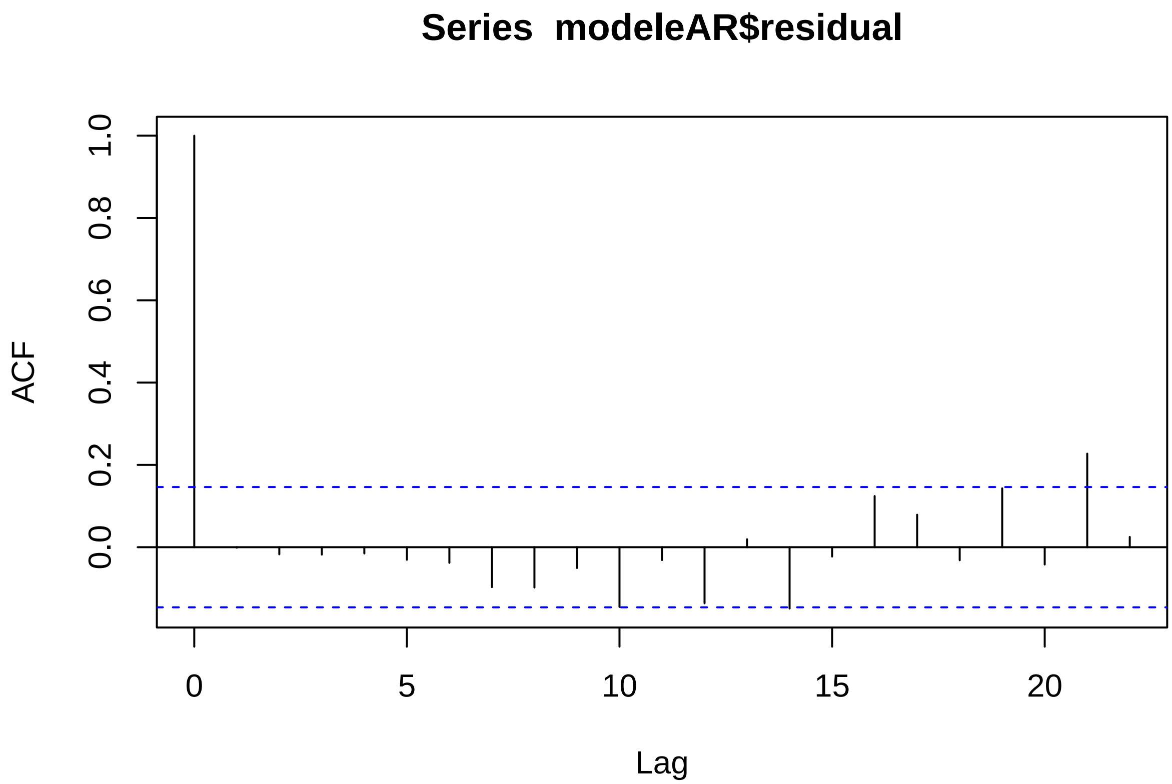
```
modeleMA <- arima(na.omit(z_train), order = c(0, 0, 5))

acf(modeleMA$residuals)
```



On voit d'après le graphique que les résidus sont corrélés (l'avant dernier individu), ainsi on peut constater que les résidus ne sont pas blancs, on rejette donc le modèle.

On pouvait aussi proposer un modèle AR(6) :

```
modeleAR <- arima(na.omit(z_train), order = c(6, 0, 0))

acf(modeleAR$residual)
```



Les résidus ne sont pas blancs, on rejette donc le modèle.

Conclusion

On rejette tous les modèles sauf le modèle : ARMA(4, 5) “modele3”.

Prédiction

```
predict(modele3, n.ahead = 6)
```

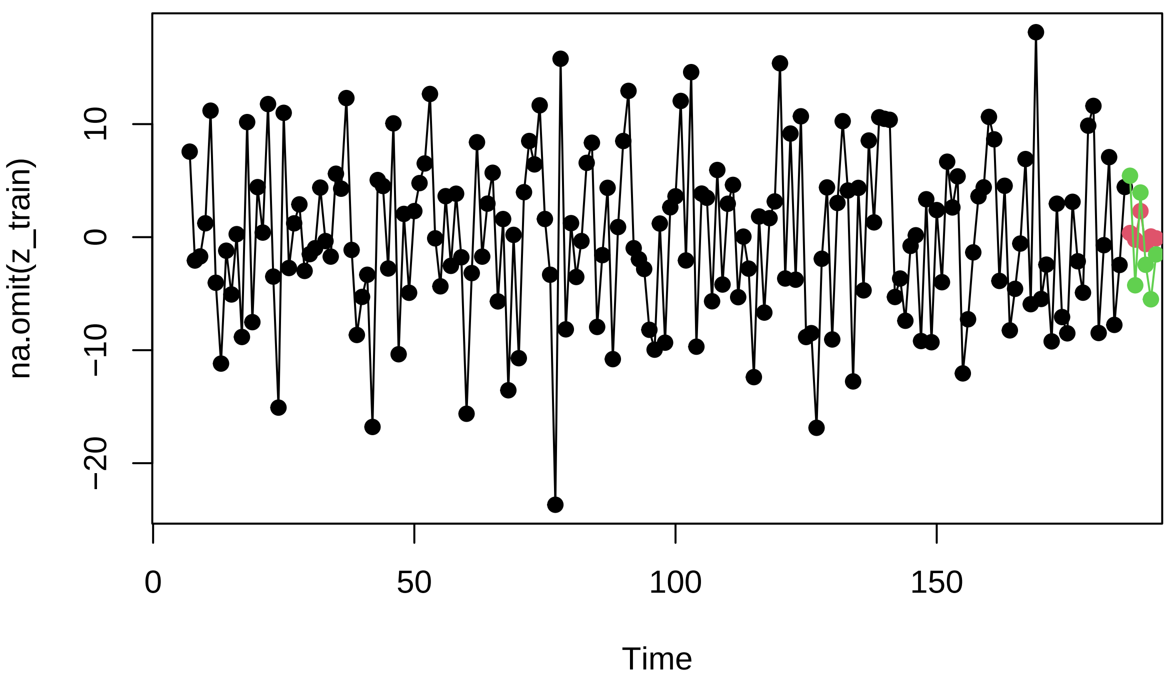
```
## $pred
## Time Series:
## Start = 187
## End = 192
## Frequency = 1
## [1]  0.36247702 -0.22133954  2.30218477 -0.61377499  0.05732246 -0.10870488
##
## $se
## Time Series:
## Start = 187
## End = 192
## Frequency = 1
## [1] 6.202426 6.426016 6.437631 6.623660 6.694264 7.096355
```

```r
z_test
```

```
## Time Series:
## Start = 187
## End = 192
## Frequency = 1
## [1]  5.431911 -4.260461  3.946637 -2.444506 -5.500783 -1.525849
```

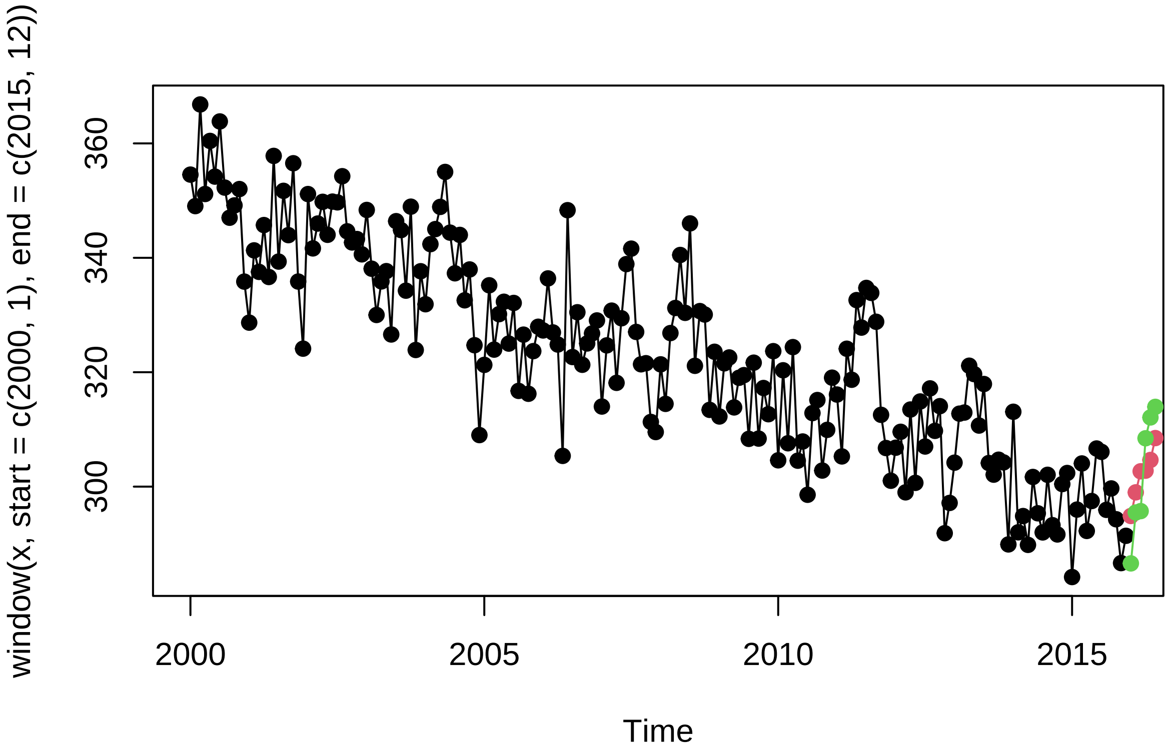
Prédiction des bruits

```r
plot(na.omit(z_train), type = "o", pch = 19)
points(predict(modele3, n.ahead = 6)$pred, col = 2, type = "o", pch = 19)
points(z_test, col = 3, type = "o", pch = 19)
```



Prédiction de la série

```r
prediction <- ts(predict(modele3, n.ahead = 6)$pred, start = c(2016, 1), end = c(2016, 6), frequency = 

estimation_m <- ts(estimation_m, start = c(2000, 1), end = c(2016, 12), frequency = 12)
estimation_s <- ts(estimation_s, start = c(2000, 1), end = c(2016, 12), frequency = 12)

plot(window(x, start = c(2000, 1), end = c(2015, 12)), type = "o", pch = 19)
points(prediction +
         window(estimation_s, start = c(2016, 1), end = c(2016, 6)) +
         window(estimation_m, start = c(2016, 1), end = c(2016, 6)),
                col = 2, type = "o", pch = 19)
points(window(x, start = c(2016, 1), end = c(2016, 6)), col = 3, type = "o", pch = 19)
```




```
ecart <- var(c(prediction) - c(z_test))
sqrt(ecart)
```

```
## [1] 3.865359
```

Si le graphe n'est pas trop clair, regarder l'image **prediction_serie_16.png** qui se trouve dans le même fichier.