Séries temporelles sous R

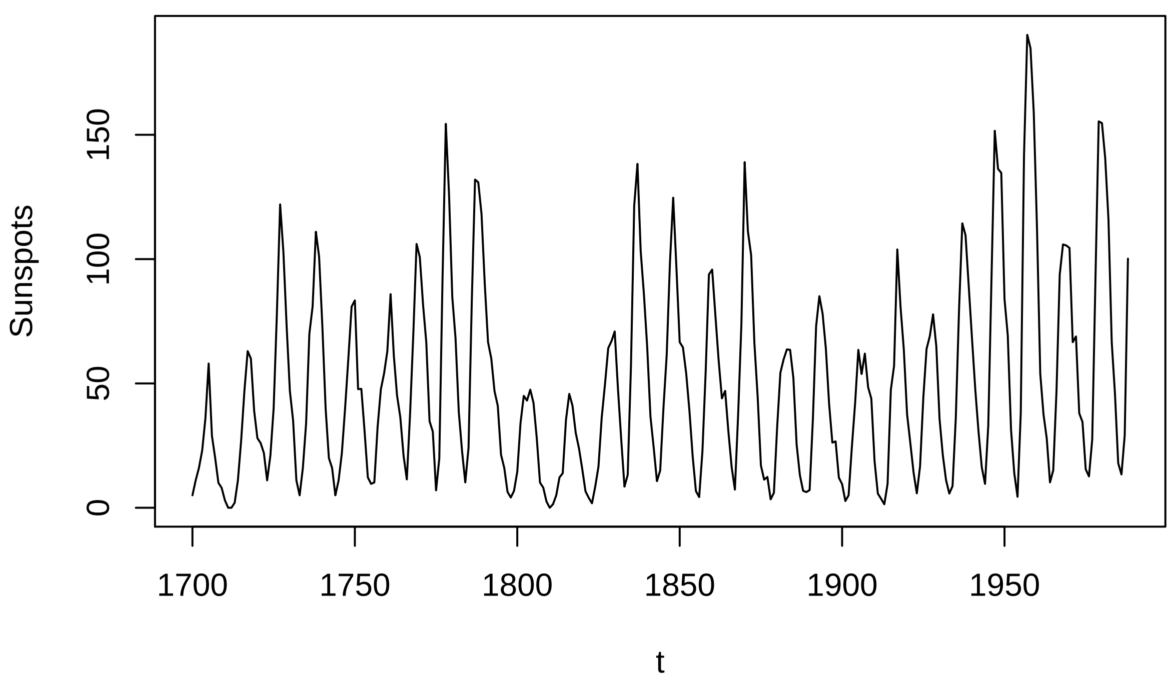
V. Lefieux

EXEMPLES DE SERIES TEMPORELLES

Les séries apparaissent dans l'ordre du cours.

Série *sunspot* : nombre annuel de tâches solaires de 1790 à 1970

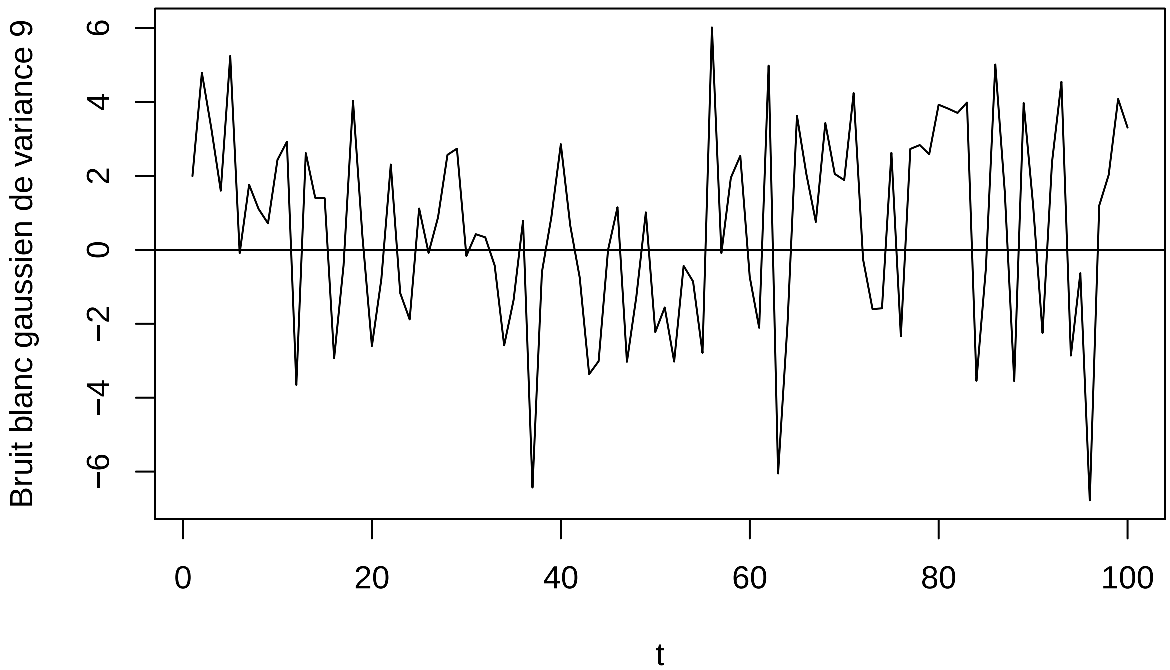
```
plot(sunspot.year,xlab="t",ylab="Sunspots")
```



Bruit blanc gaussien de loi $\mathcal{N}(0, 3^2)$

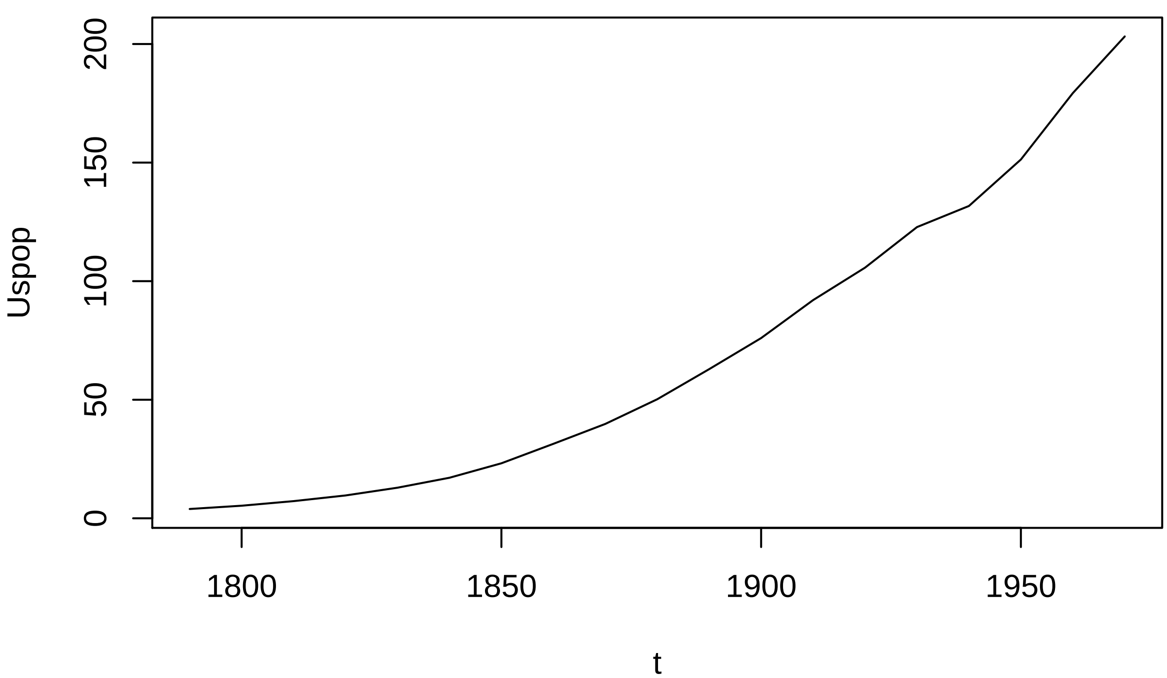
Pour les simulations effectuées dans ce document, on fixe arbitrairement la racine (seed) à 1789.

```
set.seed(1789)
plot(ts(rnorm(100,sd=3),start=1,end=100),xlab="t",ylab="Bruit blanc gaussien de variance 9")
abline(h=0)
```



Série *uspop* : population des Etats-Unis, en millions, de 1790 à 1990 (Pas de temps décennal)

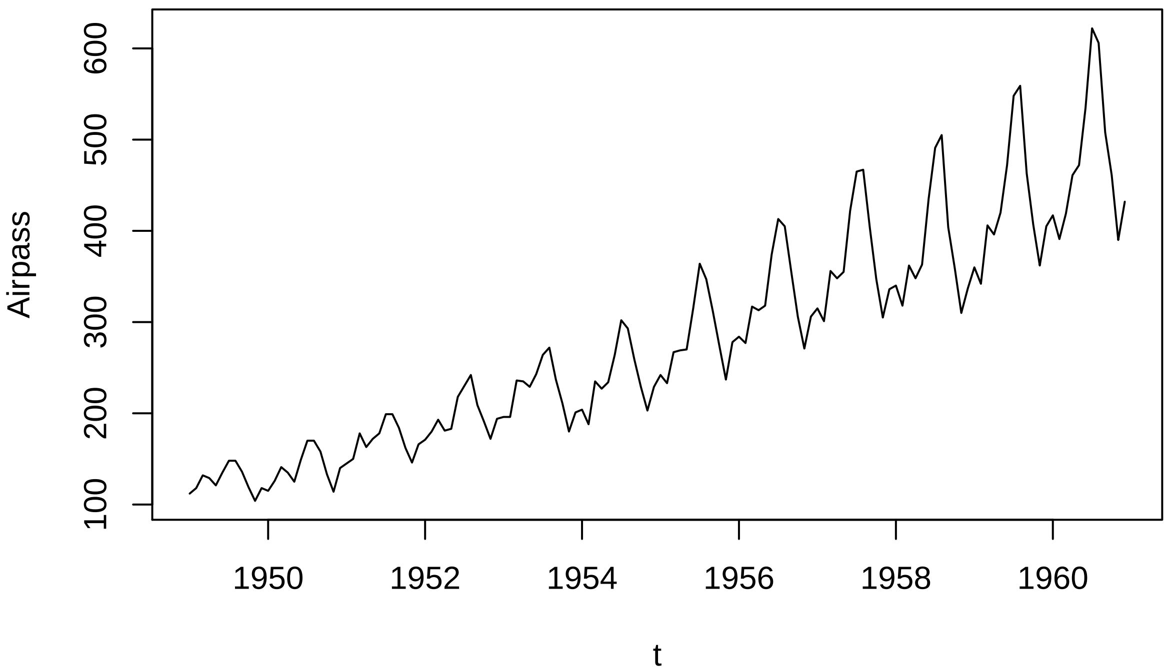
```
plot(uspop,xlab="t",ylab="Uspop")
```



Série *airpass* : nombre mensuel de passagers aériens, en milliers, de janvier 1949 à décembre 1960

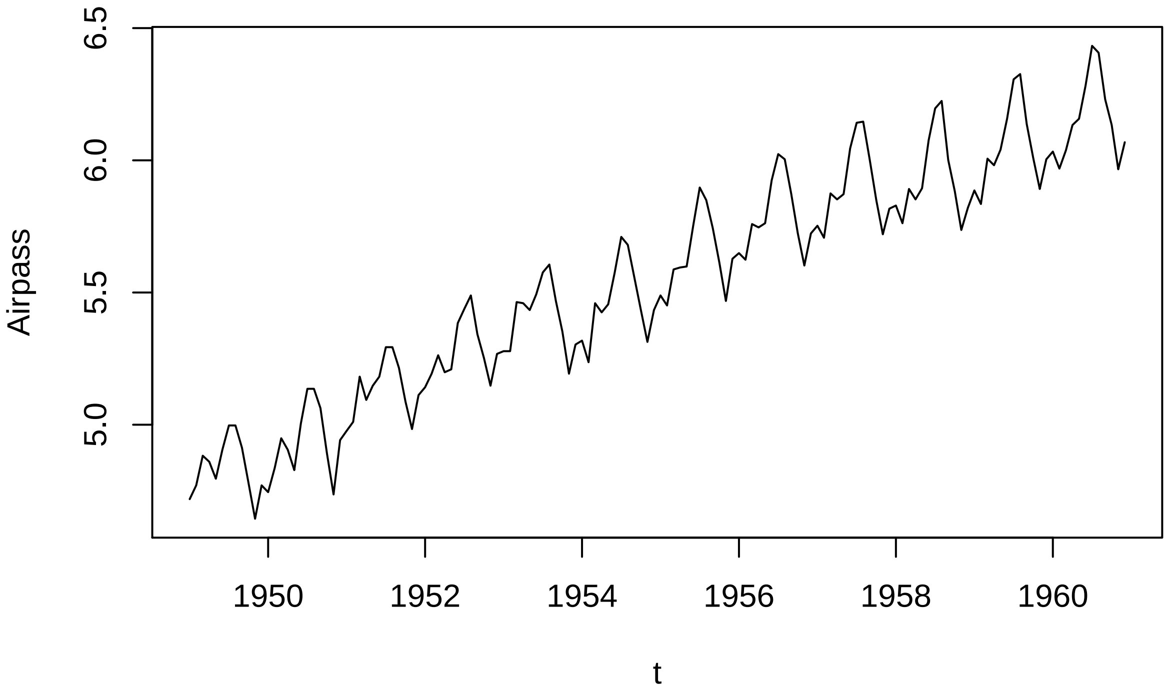
Série Brute

```
plot(AirPassengers,xlab="t",ylab="Airpass")
```



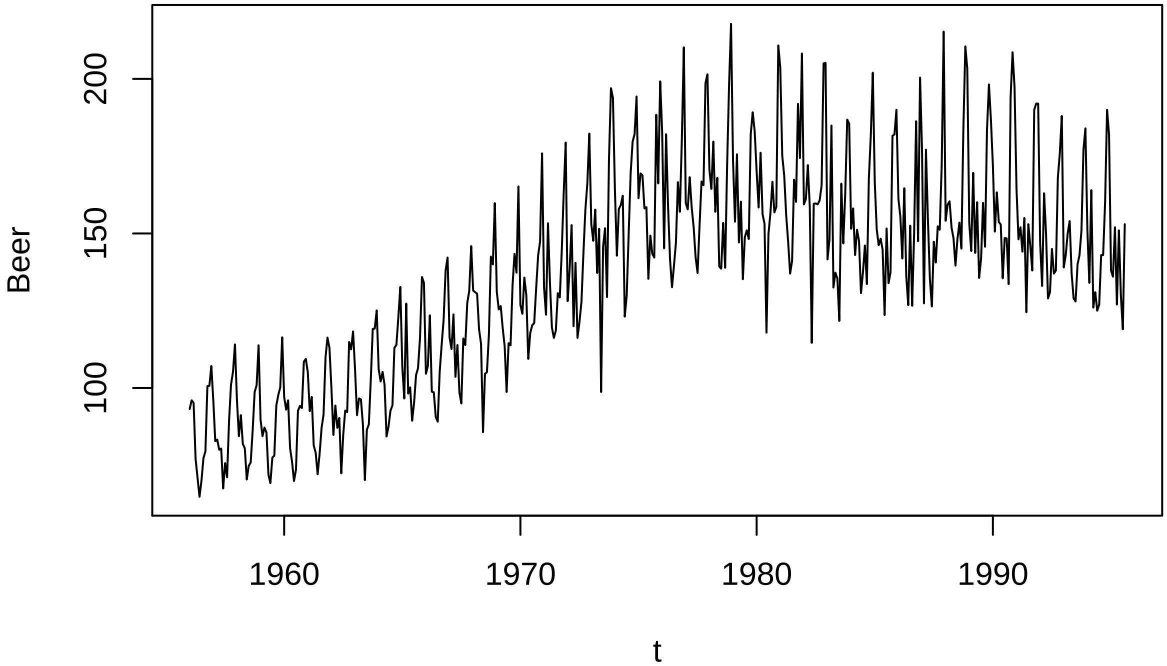
Logarithme de la série *airpass*

```
plot(log(AirPassengers),xlab="t",ylab="Airpass")
```



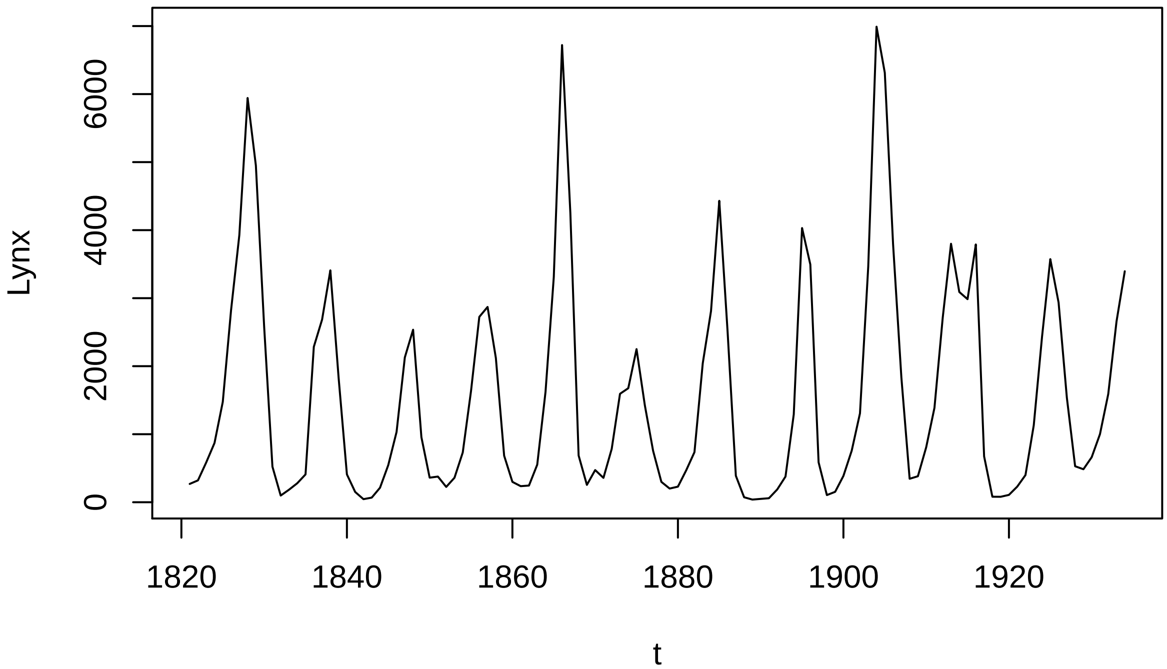
Série *beer* : production mensuelle de bière en Australie, en mégalitres, de janvier 1956 à février 1991

```
beer=read.csv("../Data/beer.csv",header=F,dec=".",sep=",")
beer=ts(beer[,2],start=1956,freq=12)
plot(beer,xlab="t",ylab="Beer")
```



Série *lynx* : nombre annuel de lynx capturés au Canada, de 1821 à 1934

```
plot(lynx,xlab="t",ylab="Lynx")
```



Sauf mention contraire, on travaille dans la suite sur la série temporelle *airpass*.

```
x=AirPassengers
y=log(x)
```

CHAPITRE 1 : DECOMPOSITION SAISONNIERE

Décomposition saisonnière à l'aide de la régression linéaire

Création des bases tendancielle et saisonnière

```
t=1:144

for (i in 1:12)
{
  su=rep(0,times=12)
  su[i]=1
  s=rep(su,times=12)
  assign(paste("s",i,sep=""),s)
}
```

Régression linéaire

```
reg=lm(y~t+s1+s2+s3+s4+s5+s6+s7+s8+s9+s10+s11+s12-1)
summary(reg)
```

```
##
## Call:
## lm(formula = y ~ t + s1 + s2 + s3 + s4 + s5 + s6 + s7 + s8 +
##     s9 + s10 + s11 + s12 - 1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## t   0.0100688  0.0001193    84.4   <2e-16 ***
## s1  4.7267804  0.0188935   250.2   <2e-16 ***
## s2  4.7047255  0.0189443   248.3   <2e-16 ***
## s3  4.8349527  0.0189957   254.5   <2e-16 ***
## s4  4.8036838  0.0190477   252.2   <2e-16 ***
## s5  4.8013112  0.0191003   251.4   <2e-16 ***
## s6  4.9234574  0.0191535   257.1   <2e-16 ***
## s7  5.0273997  0.0192073   261.7   <2e-16 ***
## s8  5.0181049  0.0192617   260.5   <2e-16 ***
## s9  4.8734703  0.0193167   252.3   <2e-16 ***
## s10 4.7353120  0.0193722   244.4   <2e-16 ***
## s11 4.5915943  0.0194283   236.3   <2e-16 ***
## s12 4.7054593  0.0194850   241.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 9.734e+04 on 13 and 131 DF,  p-value: < 2.2e-16
```

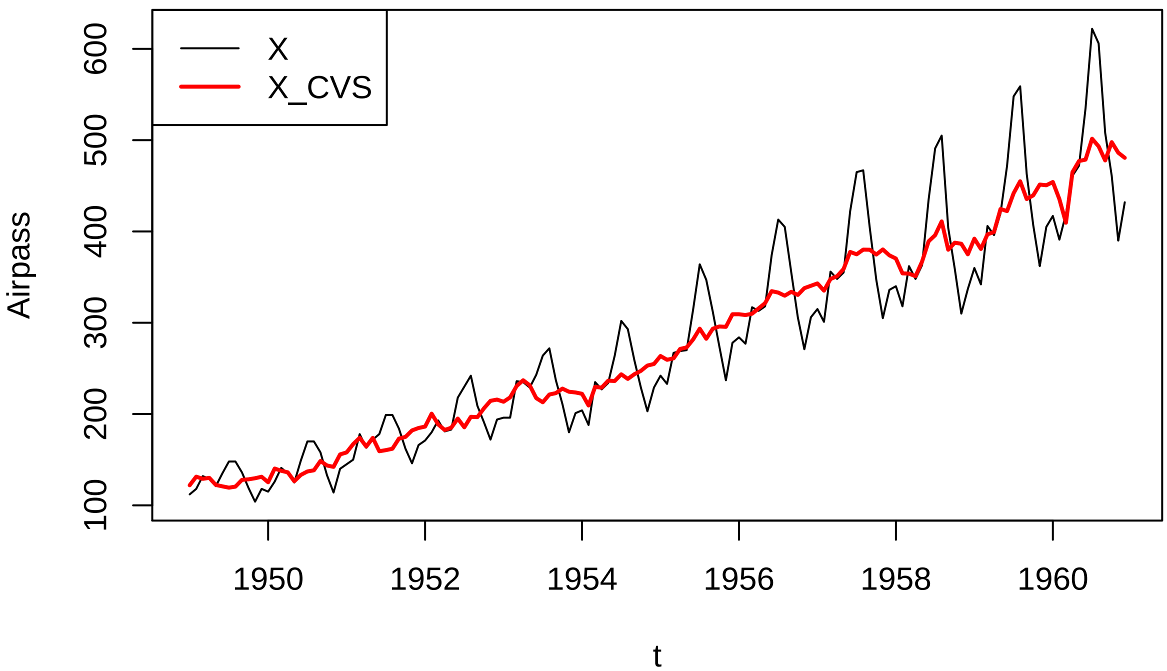
```
reg$coefficients
```

```
##         t        s1        s2        s3        s4        s5        s6
## 0.0100688 4.7267804 4.7047255 4.8349527 4.8036838 4.8013112 4.9234574
##        s7        s8        s9       s10       s11       s12
## 5.0273997 5.0181049 4.8734703 4.7353120 4.5915943 4.7054593
```

```
a=mean(reg$coefficients[2:13])
b=reg$coefficients[1]
c=reg$coefficients[2:13]-mean(reg$coefficients[2:13])
```

Calcul de la série corrigée des variations saisonnières

```
y_cvs=y-(c[1]*s1+c[2]*s2+c[3]*s3+c[4]*s4+c[5]*s5+c[6]*s6+c[7]*s7+c[8]*s8+c[9]*s9+c[10]*s10+c[11]*s11+c[
x_cvs=exp(y_cvs)
ts.plot(x,x_cvs,xlab="t",ylab="Airpass",col=c(1,2),lwd=c(1,2))
legend("topleft",legend=c("X","X_CVS"),col=c(1,2),lwd=c(1,2))
```



Décomposition saisonnière à l'aide des moyennes mobiles

On utilise les moyennes mobiles $M_{2\times 12}$ et M_3 dans la première étape de l'algorithme **X11**.

```
m2_12=function(x){
  y=(1/12)*filter(x,c(0.5,rep(1,times=11),0.5))
  return(y)
}

m3=function(x){
  y=(1/3)*filter(x,rep(1,times=3))
  return(y)
}
```

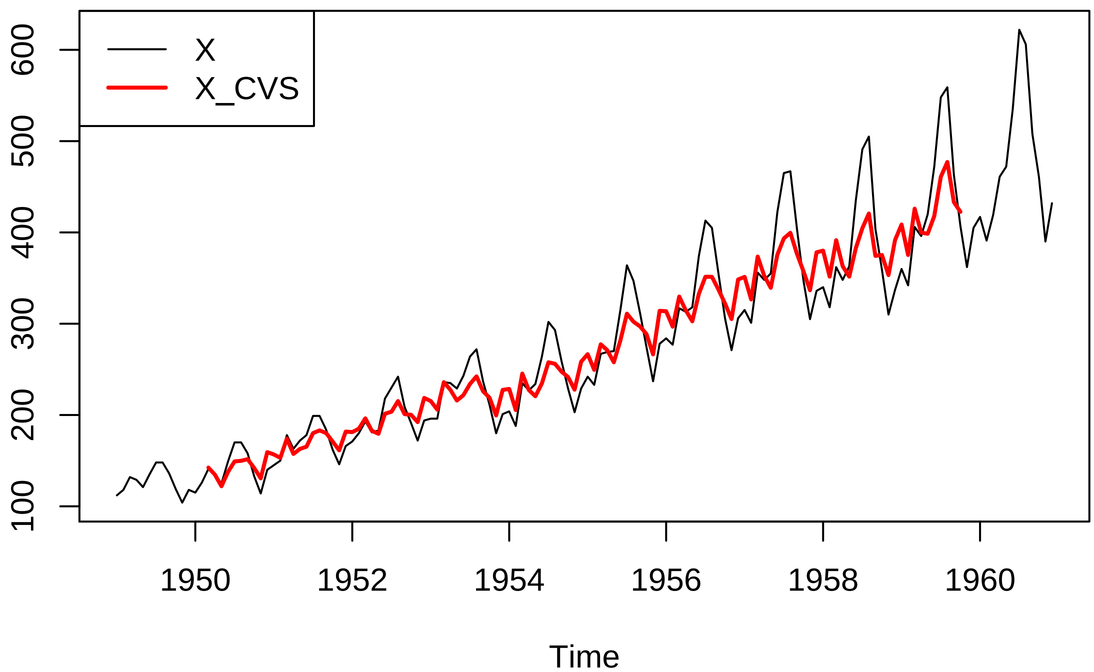
On utiliserait les moyennes mobiles M_{13}^H et M_5 dans la deuxième étape de l'algorithme **X11**.

```
m13h=function(x){
  y=(1/16796)*filter(x,c(-325,-468,0,1100,2475,3600,4032,3600,2475,1100,0,-468,-325))
  return(y)
}

m5=function(x){
  y=(1/5)*filter(x,rep(1,times=5))
  return(y)
}
```

Le premier jeu d'estimation donne :

```
t1=m2_12(y)
sig1=y-t1
s1=m3(m3(sig1))
shat1=s1-m2_12(s1)
ycvs1=y-shat1
xcvs1=exp(ycvs1)
ts.plot(x,xcvs1,col=c(1,2),lwd=c(1,2))
legend("topleft",legend=c("X","X_CVS"),col=c(1,2),lwd=c(1,2))
```



Il faudrait effectuer les 4 étapes suivantes et compléter les données éliminées par des moyennes mobiles asymétriques.

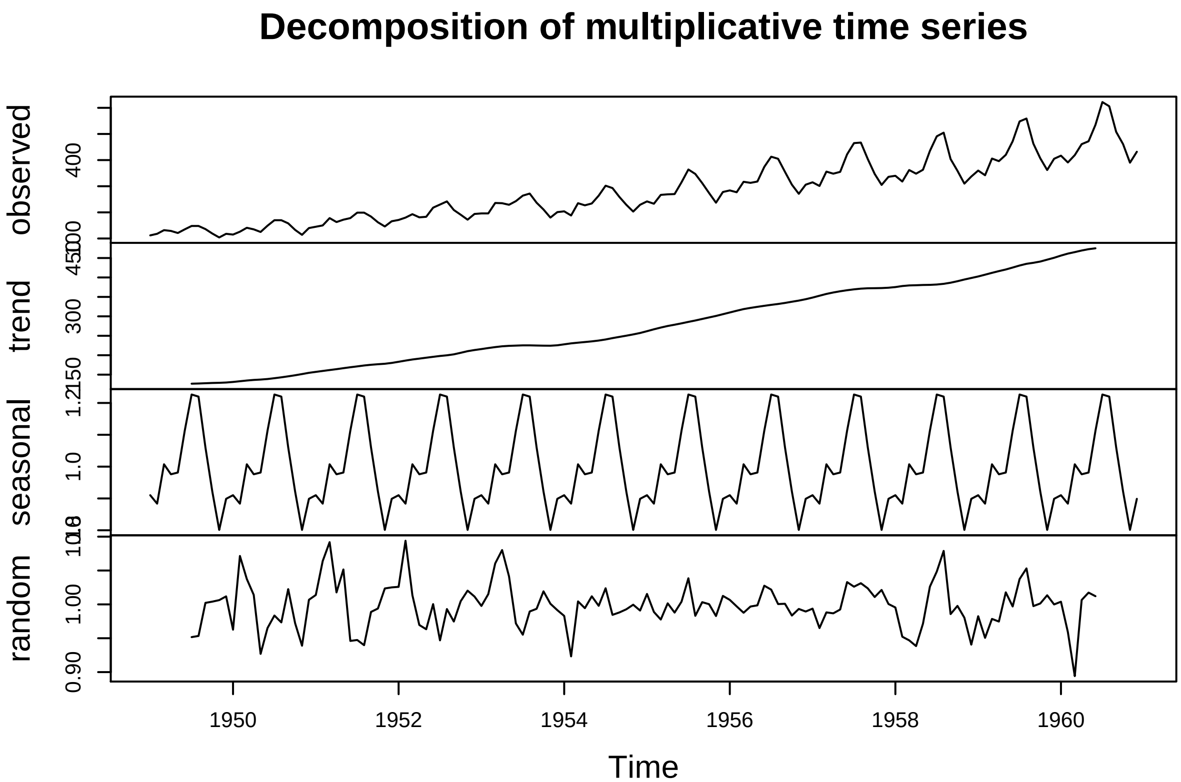
Notons qu'il est également possible d'utiliser la librairie (complète) `X12`.

Décomposition saisonnière à l'aide de la fonction `decompose`

```
decomp.x=decompose(x,type="multiplicative")
decomp.x$figure
```

```
##  [1] 0.9102304 0.8836253 1.0073663 0.9759060 0.9813780 1.1127758 1.2265555
##  [8] 1.2199110 1.0604919 0.9217572 0.8011781 0.8988244
```

```
plot(decomp.x)
```



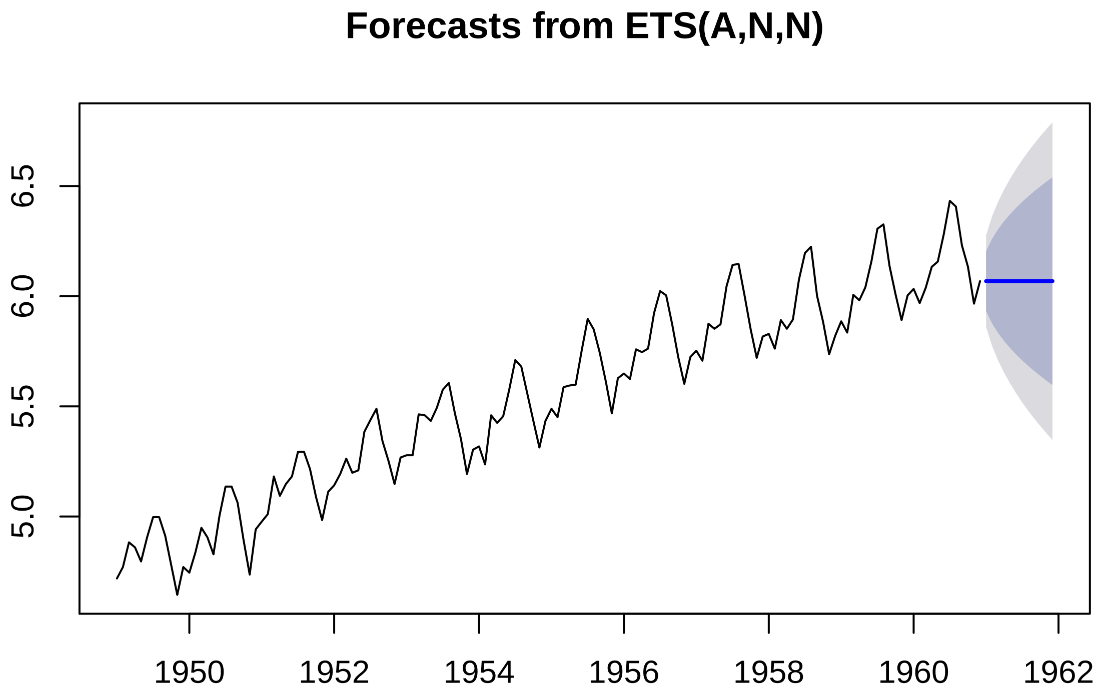
CHAPITRE 1 : LISSAGE EXPONENTIEL

On utilise la librairie `forecast`.

```
library(forecast)
```

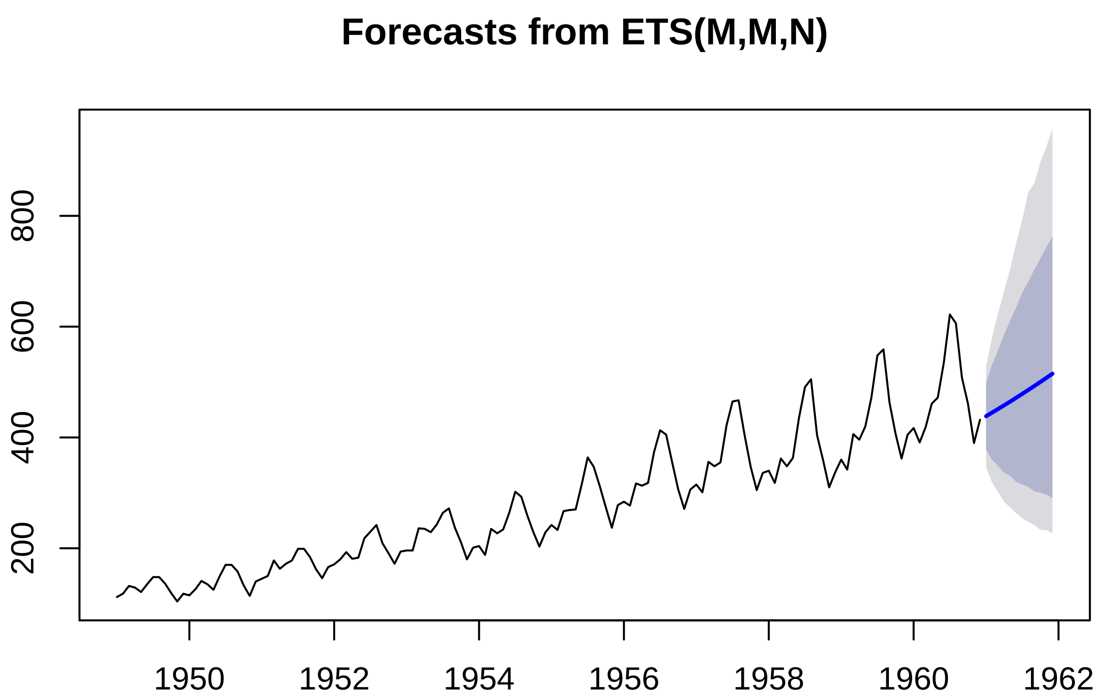
Lissage exponentiel simple

```
les=ets(y,model="ANN")
les.pred=predict(les,12)
plot(les.pred)
```



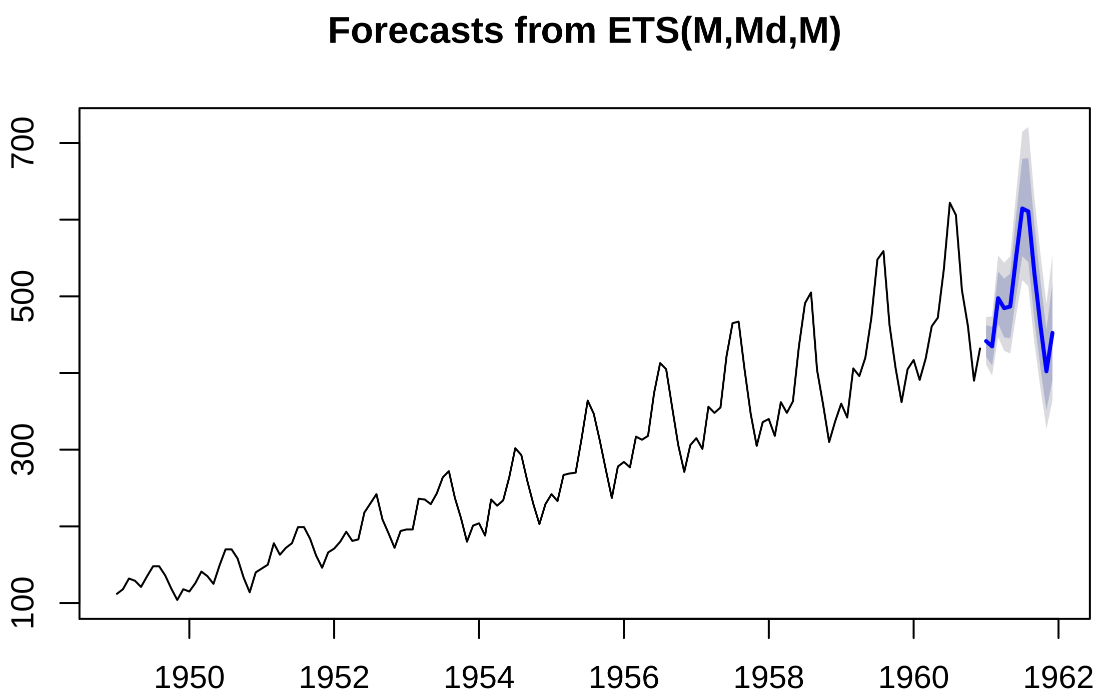
Lissage exponentiel double

```
led=ets(x,model="MMN")
led.pred=predict(led,12)
plot(led.pred)
```



Méthode de Holt-Winters

```r
hw=ets(x,model="MMM")
hw.pred=predict(hw,12)
plot(hw.pred)
```



CHAPITRE 2

Blancheur

On utilise la librairie `caschrono`.

```r
library(caschrono)
```

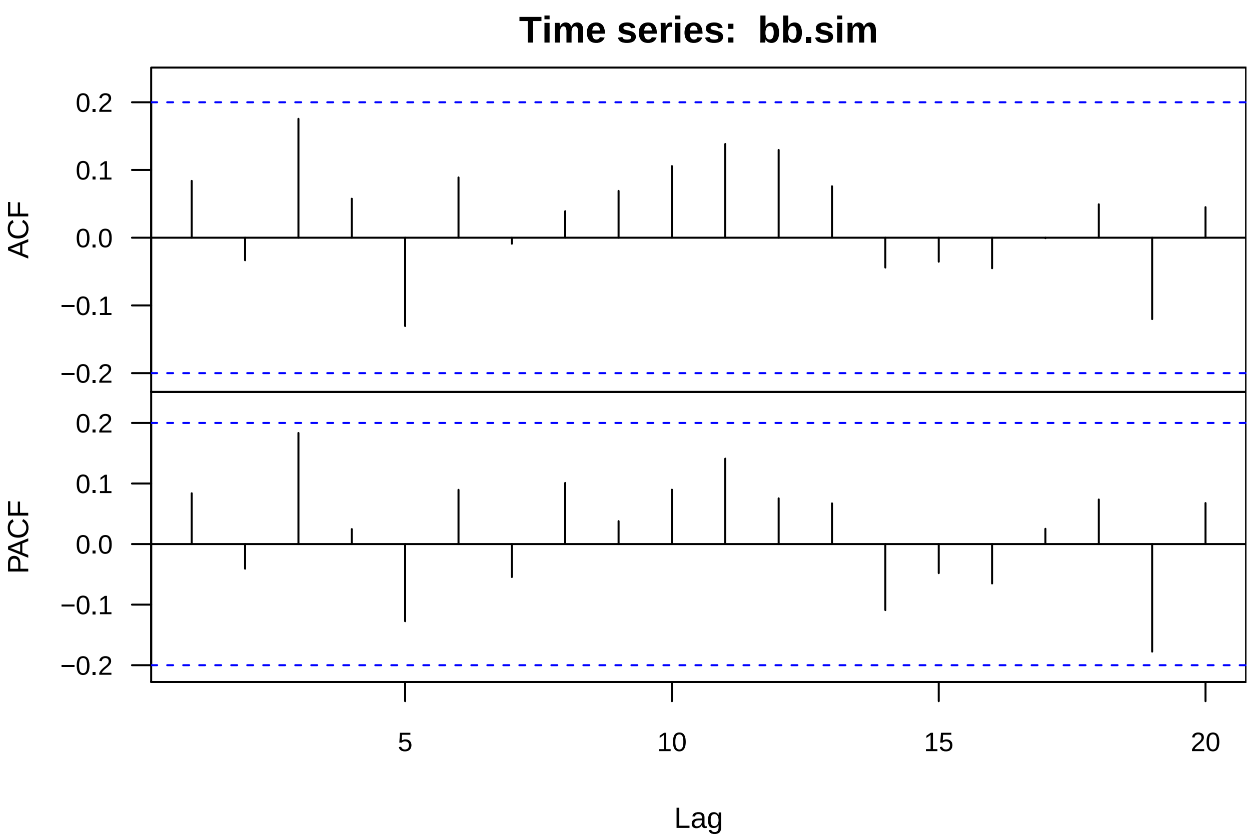
On obtient sur un bruit blanc gaussien de loi $\mathcal{N}\left(0,3^2\right)$:

```r
set.seed(1789)
bb.sim=ts(rnorm(100,sd=3),start=1,end=100)
Box.test.2(bb.sim,nlag=c(5,10,20),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      5 0.31297
## [2,]     10 0.58200
## [3,]     20 0.77246
```

On peut également visualiser ses autocorrélogrammes empiriques simple et partiel.

```r
acf2y(bb.sim,lag.max=20)
```




```
##        LAG          ACF1        PACF
##  [1,]    1  0.0838892617  0.08388926
##  [2,]    2 -0.0333372907 -0.04066085
##  [3,]    3  0.1755783814  0.18349229
##  [4,]    4  0.0575819857  0.02470018
##  [5,]    5 -0.1304585184 -0.12741415
##  [6,]    6  0.0890363233  0.08961168
##  [7,]    7 -0.0088651265 -0.05443103
##  [8,]    8  0.0391275977  0.10096243
##  [9,]    9  0.0691433921  0.03789974
## [10,]   10  0.1057201344  0.08982110
## [11,]   11  0.1384150148  0.14111357
## [12,]   12  0.1296123238  0.07558802
## [13,]   13  0.0759349274  0.06719900
## [14,]   14 -0.0441275912 -0.10905173
## [15,]   15 -0.0354845680 -0.04801316
## [16,]   16 -0.0451179166 -0.06502711
## [17,]   17 -0.0007088443  0.02522153
## [18,]   18  0.0493552774  0.07364464
## [19,]   19 -0.1201841157 -0.17759044
## [20,]   20  0.0451003221  0.06774346
```

On obtient pour un processus $AR(1)$, $X_t = 0.6X_{t-1} + \varepsilon_t$ où $\text{Var}\,(X_t) = 3^2$:

```r
set.seed(1789)
ar.sim=arima.sim(n=100,list(ar=0.6),sd=3)
Box.test.2(ar.sim,nlag=c(1,5,10,20),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      1       0
## [2,]      5       0
## [3,]     10       0
## [4,]     20       0
```

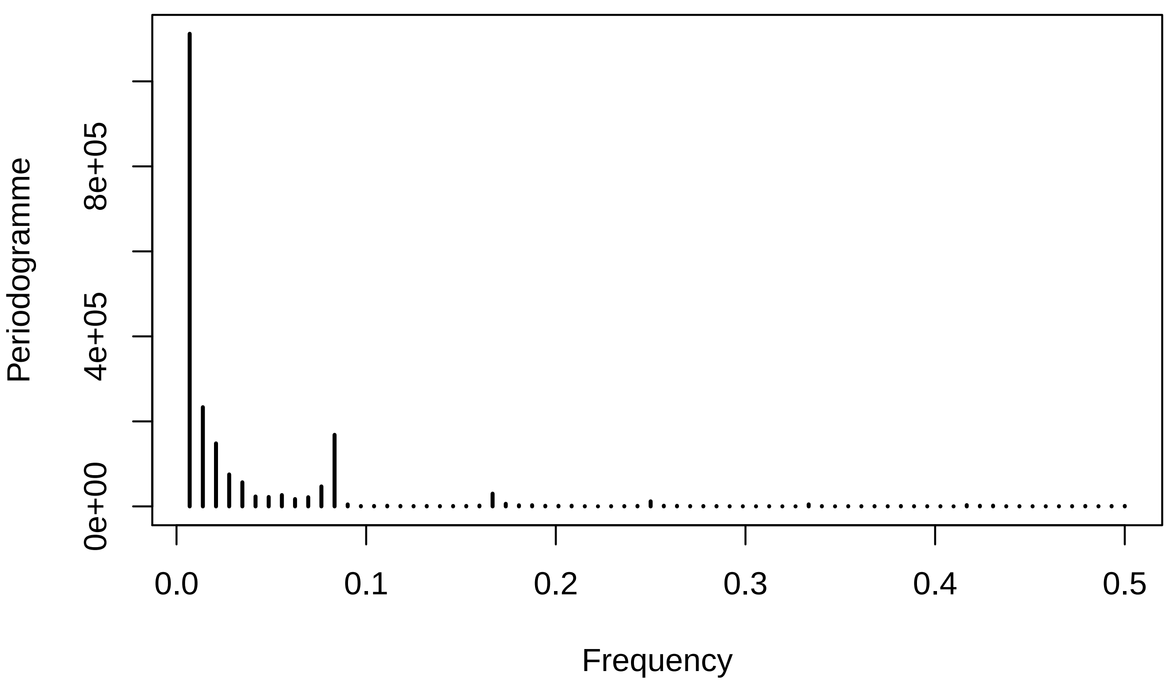
Périodogramme

On utilise la librairie `TSA`.

```r
library(TSA)
```

On obtient pour la série *lynx* :

```r
lynx.periodogram=periodogram(x,ylab="Periodogramme")
```



On peut ensuite déterminer pour quelle fréquence le périodogramme est maximal, etc.

```r
lynx.periodogram$freq[which.max(as.vector(lynx.periodogram$spec))]*114
```

```
## [1] 0.7916667
```

CHAPITRE 4 : PROCESSUS AR, MA & ARMA

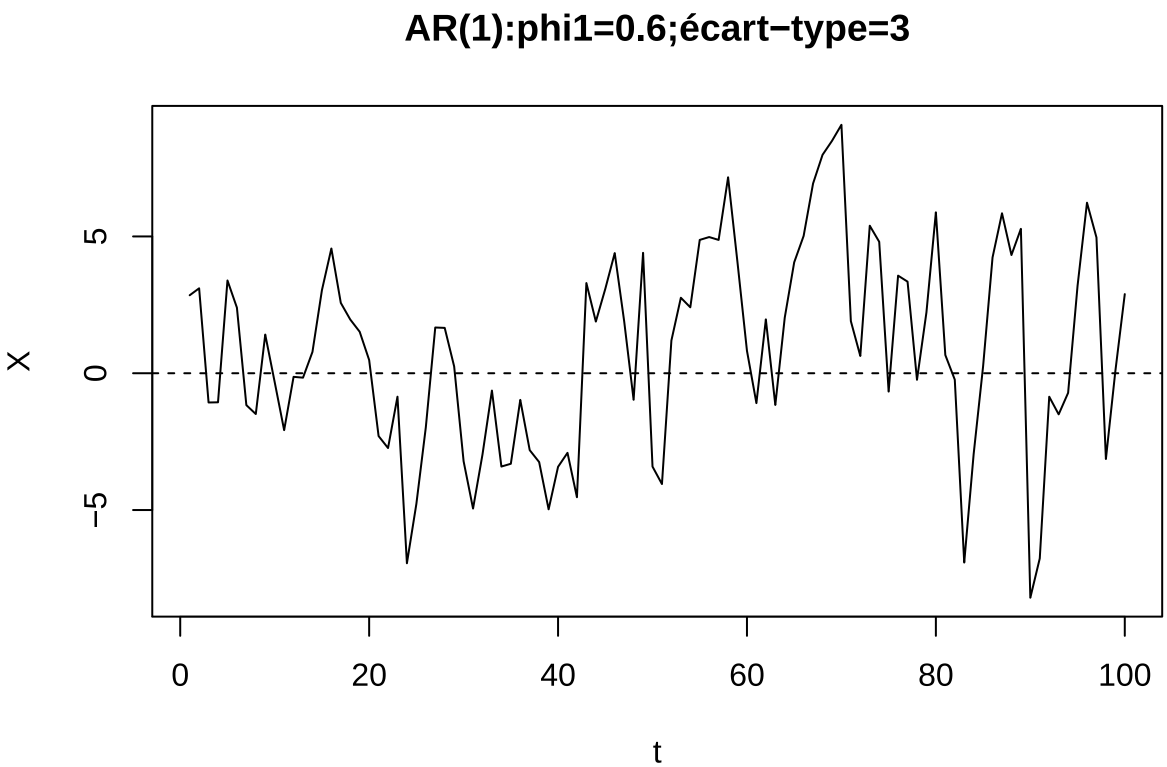
On utilise la librairie `caschrono`.

```
library(caschrono)
```

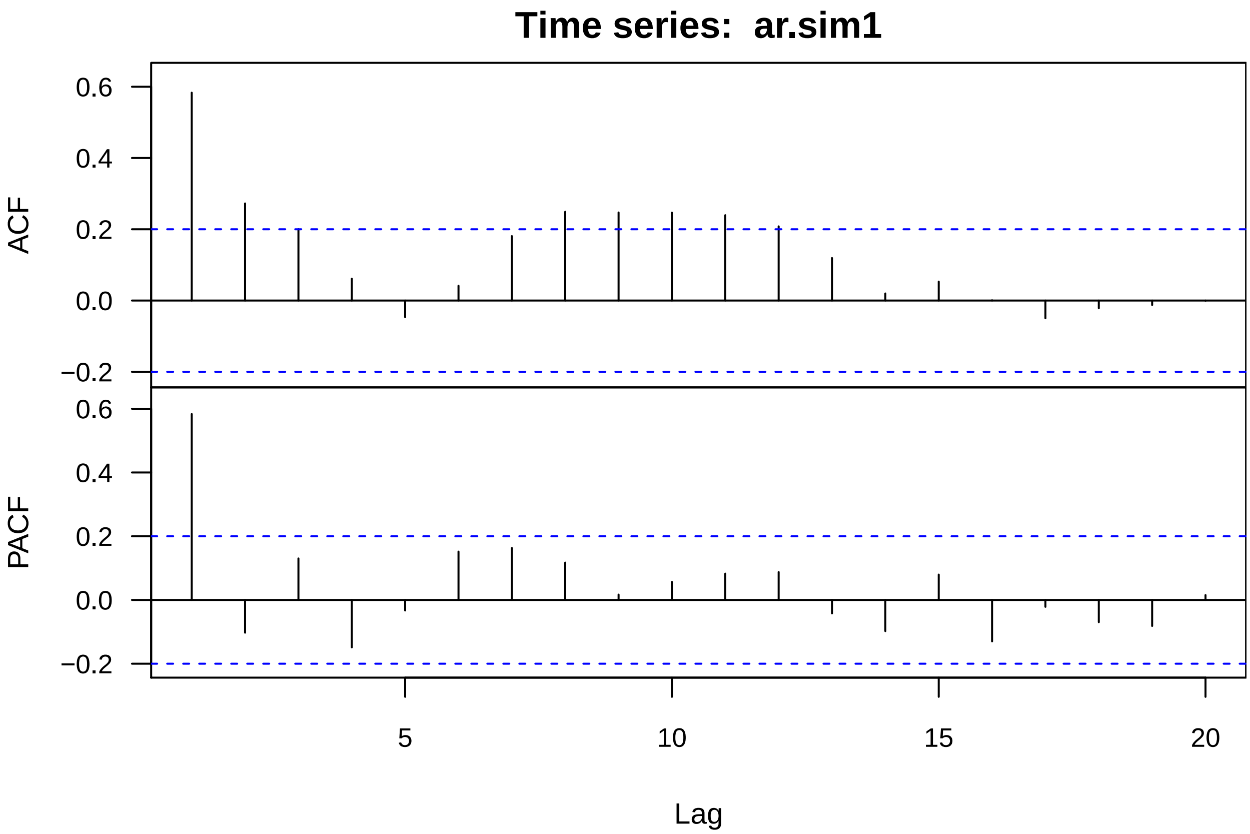
Autocorrélogrammes simple et partiel d'un processus AR

On obtient pour un processus $AR(1)$, $X_t = 0.6X_{t-1} + \varepsilon_t$ où $\text{Var}\,(X_t) = 3^2$:

```
set.seed(1789)
ar.sim1=arima.sim(n=100,list(ar=0.6),sd=3)
plot(ar.sim1,xlab="t",ylab="X",main="AR(1):phi1=0.6;écart-type=3")
abline(h=0,lty=2)
```




```
acf2y(ar.sim1,lag.max=20)
```

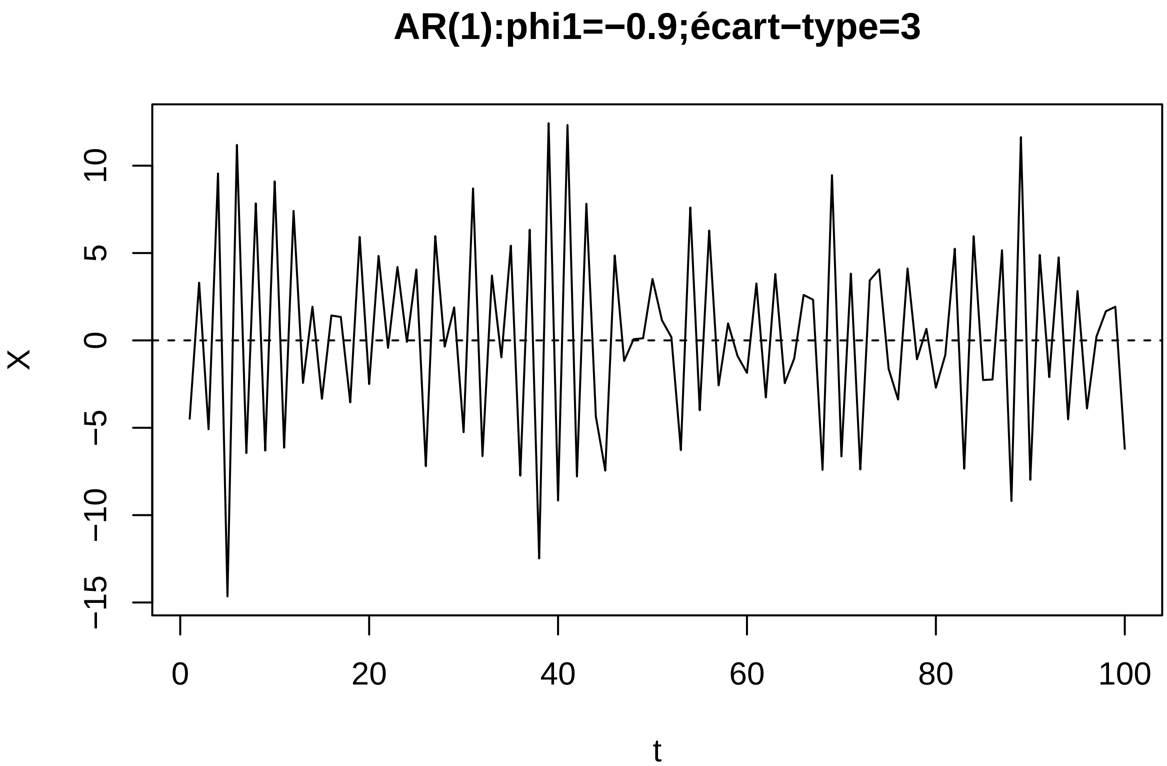



```
##       LAG          ACF1        PACF
##  [1,]   1  0.5833406576  0.58334066
##  [2,]   2  0.2725242295 -0.10271440
##  [3,]   3  0.2004274366  0.13028117
##  [4,]   4  0.0612962952 -0.14879787
##  [5,]   5 -0.0468918379 -0.03301523
##  [6,]   6  0.0415286628  0.15176161
##  [7,]   7  0.1808874892  0.16300315
##  [8,]   8  0.2491534956  0.11728415
##  [9,]   9  0.2471368054  0.01686339
## [10,]  10  0.2466945271  0.05663331
## [11,]  11  0.2395867316  0.08269296
## [12,]  12  0.2081637627  0.08766179
## [13,]  13  0.1193204973 -0.04212328
## [14,]  14  0.0198400955 -0.09789964
## [15,]  15  0.0531791690  0.07968125
## [16,]  16  0.0007248788 -0.12997743
## [17,]  17 -0.0497584201 -0.02161079
## [18,]  18 -0.0216936170 -0.06979308
## [19,]  19 -0.0125636344 -0.08161061
## [20,]  20 -0.0006498825  0.01532153
```

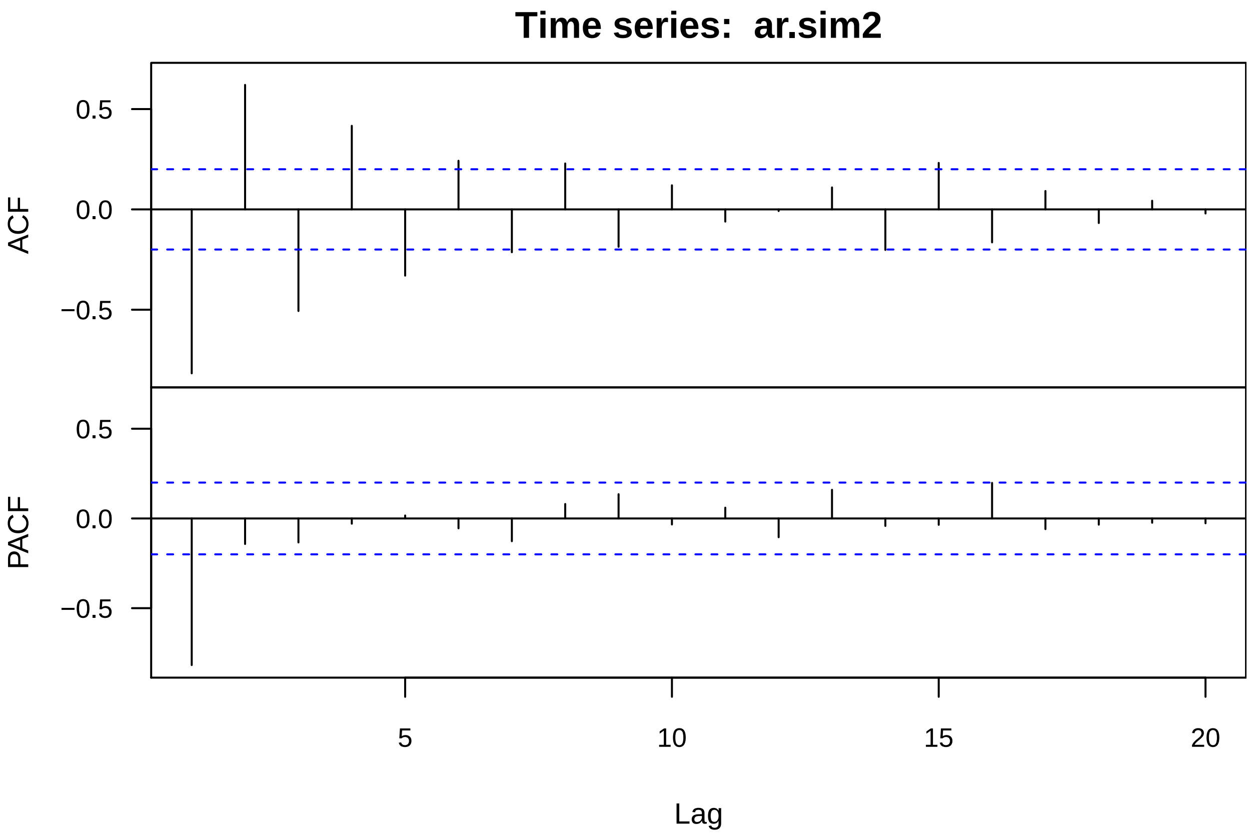
On obtient pour un processus $AR(1)$, $X_t = -0.9X_{t-1} + \varepsilon_t$ où $\text{Var}(X_t) = 3^2$:

```
set.seed(1789)
ar.sim2=arima.sim(n=100,list(ar=-0.9),sd=3)
```

```r
plot(ar.sim2,xlab="t",ylab="X",main="AR(1):phi1=-0.9;écart-type=3")
abline(h=0,lty=2)
```




```r
acf2y(ar.sim2,lag.max=20)
```

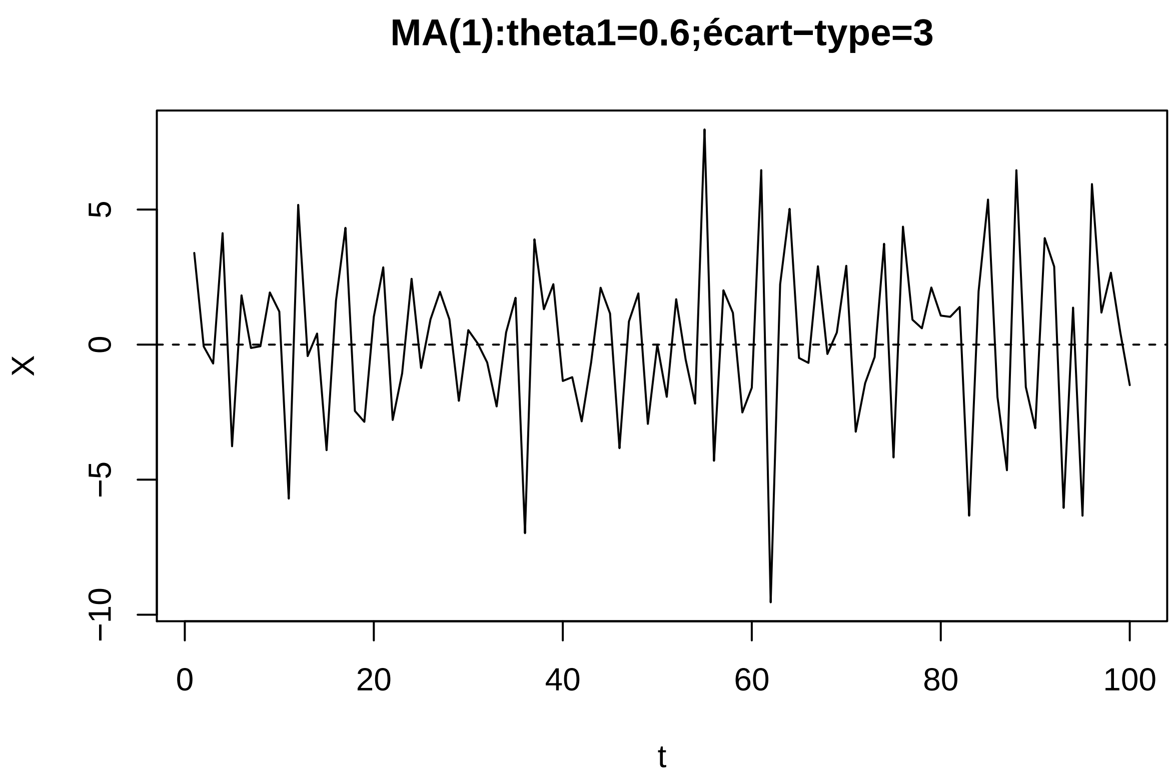



```
##       LAG         ACF1        PACF
##  [1,]   1 -0.817205024 -0.81720502
##  [2,]   2  0.620523013 -0.14239754
##  [3,]   3 -0.506458739 -0.13373628
##  [4,]   4  0.416536919 -0.02934504
##  [5,]   5 -0.330180309  0.01654439
##  [6,]   6  0.242301861 -0.05504116
##  [7,]   7 -0.213658638 -0.12691071
##  [8,]   8  0.228957221  0.08054664
##  [9,]   9 -0.186520555  0.13522429
## [10,]  10  0.120132476 -0.03361949
## [11,]  11 -0.060838458  0.06018788
## [12,]  12 -0.008241237 -0.10460821
## [13,]  13  0.109500966  0.15971297
## [14,]  14 -0.202098515 -0.04166016
## [15,]  15  0.231776812 -0.03563727
## [16,]  16 -0.164477217  0.19813360
## [17,]  17  0.091655820 -0.05943177
## [18,]  18 -0.068015167 -0.03475955
## [19,]  19  0.043092488 -0.02396456
## [20,]  20 -0.020384778 -0.02693162
```

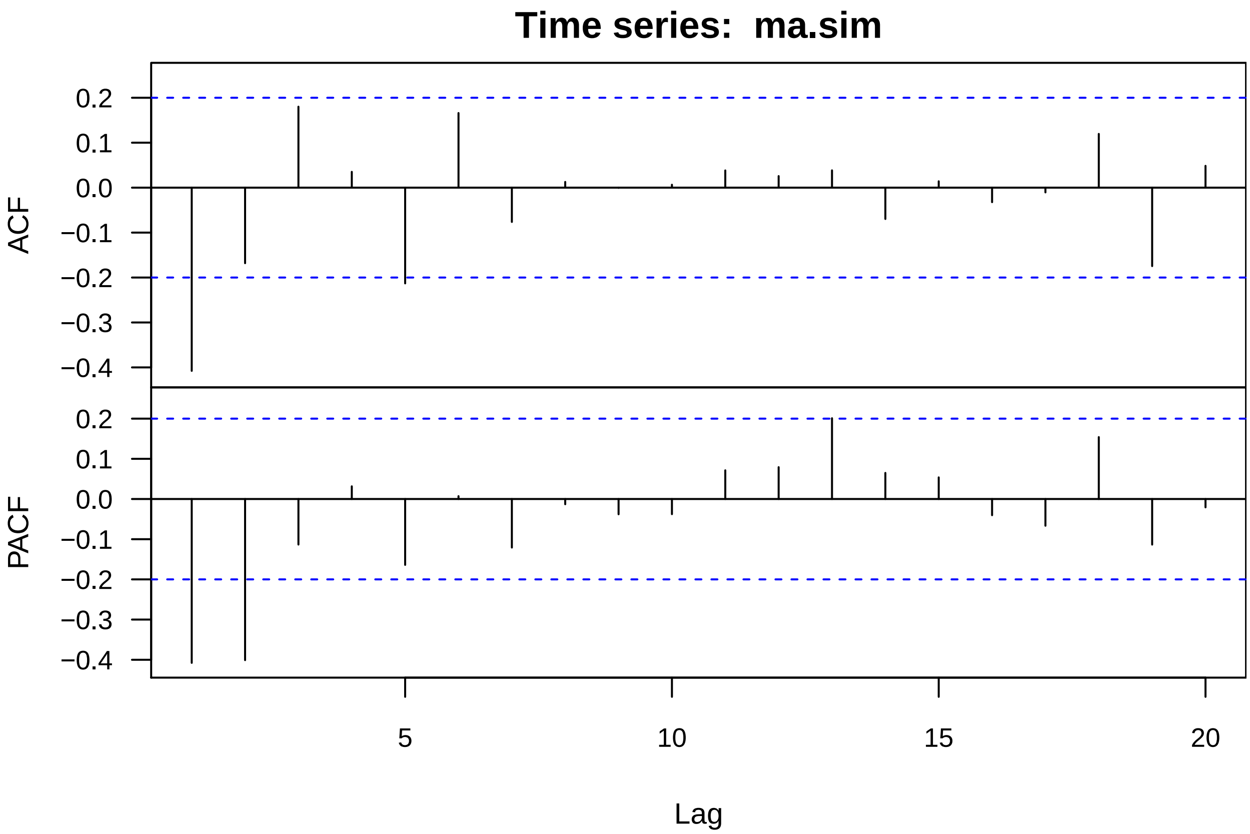
Autocorrélogrammes simple et partiel d'un processus MA

On obtient pour un processus $MA(1)$, $X_t = \varepsilon_t - 0.7\varepsilon_{t-1}$ où $\mathrm{Var}\,(X_t) = 3^2$:

```r
set.seed(1789)
ma.sim=arima.sim(n=100,list(ma=-0.7),sd=3)
plot(ma.sim,xlab="t",ylab="X",main="MA(1):theta1=0.6;écart-type=3")
abline(h=0,lty=2)
```




```r
acf2y(ma.sim,lag.max=20)
```

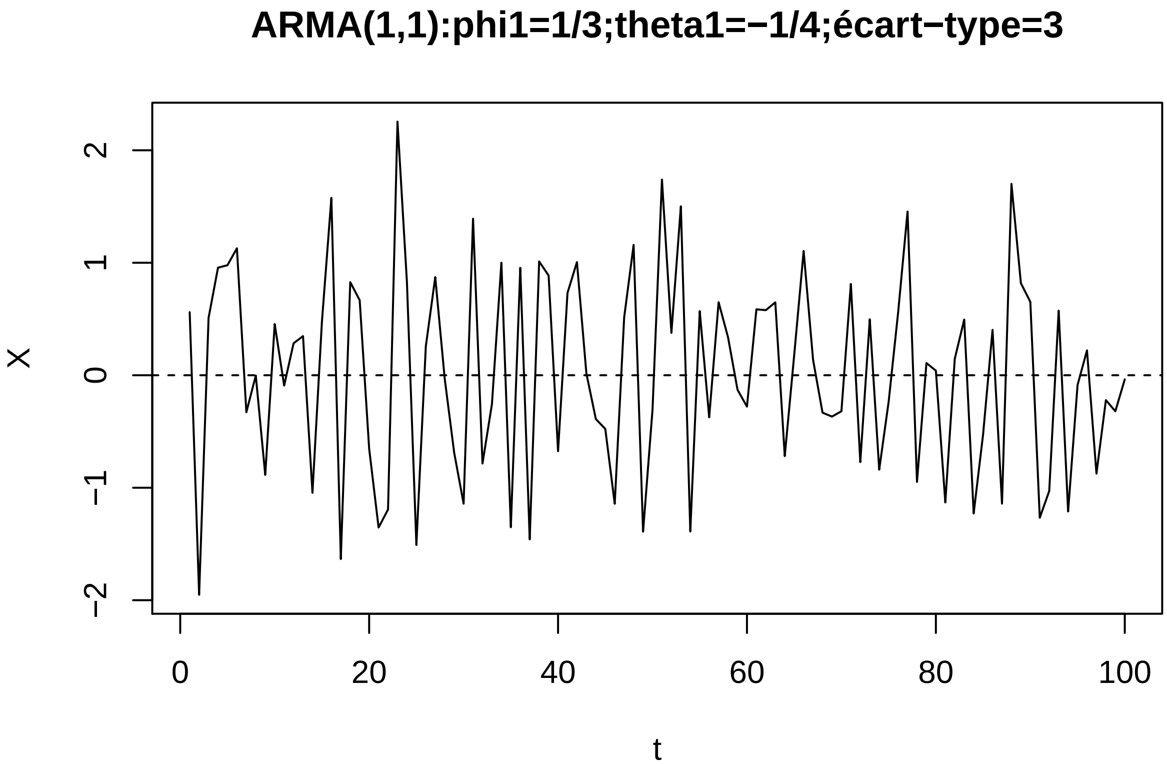



```
##       LAG          ACF1         PACF
##  [1,]   1 -0.4077016027 -0.407701603
##  [2,]   2 -0.1680966288 -0.400966040
##  [3,]   3  0.1802022434 -0.113308107
##  [4,]   4  0.0351840297  0.031592451
##  [5,]   5 -0.2130414353 -0.163896952
##  [6,]   6  0.1660284968  0.007098326
##  [7,]   7 -0.0763453576 -0.120808293
##  [8,]   8  0.0127427442 -0.012995746
##  [9,]   9 -0.0005057755 -0.037969853
## [10,]  10  0.0065274413 -0.037684001
## [11,]  11  0.0383589758  0.071428092
## [12,]  12  0.0257218529  0.079260628
## [13,]  13  0.0385396647  0.200994747
## [14,]  14 -0.0696419583  0.064934966
## [15,]  15  0.0141413165  0.053824092
## [16,]  16 -0.0323505479 -0.040333247
## [17,]  17 -0.0106079235 -0.066508253
## [18,]  18  0.1196145890  0.154072809
## [19,]  19 -0.1745999879 -0.113462570
## [20,]  20  0.0485391299 -0.020795026
```

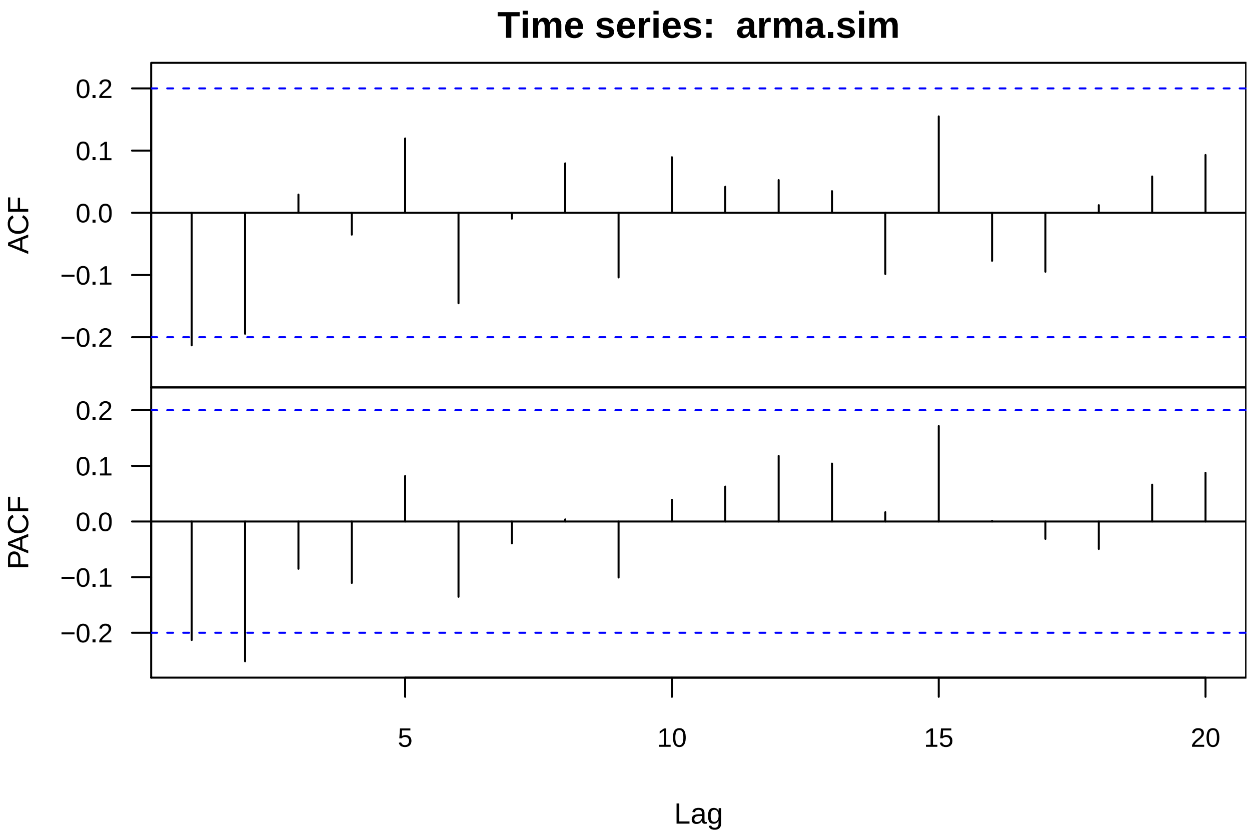
Autocorrélogrammes simple et partiel d'un processus ARMA

On obtient pour un processus $ARMA(1,1)$, $X_t = \frac{1}{3}X_{t-1} + \varepsilon_t - \frac{1}{4}\varepsilon_{t-1}$ où $\text{Var}\,(X_t) = 3^2$:

```r
arma.sim=arma.sim=arima.sim(n=100,list(ar=1/3,ma=-1/4),sd=1)
plot(arma.sim,xlab="t",ylab="X",main="ARMA(1,1):phi1=1/3;theta1=-1/4;écart-type=3")
abline(h=0,lty=2)
```




```r
acf2y(arma.sim,lag.max=20)
```




```
##        LAG         ACF1         PACF
##  [1,]    1 -0.213057431 -0.213057431
##  [2,]    2 -0.194496563 -0.251297288
##  [3,]    3  0.029352822 -0.085025633
##  [4,]    4 -0.035102082 -0.110434494
##  [5,]    5  0.119547410  0.081853641
##  [6,]    6 -0.145445633 -0.135370862
##  [7,]    7 -0.009414167 -0.039285044
##  [8,]    8  0.079484034  0.003862770
##  [9,]    9 -0.103902894 -0.100715357
## [10,]   10  0.089348162  0.039154416
## [11,]   11  0.041929337  0.062834961
## [12,]   12  0.052687092  0.118057453
## [13,]   13  0.034836513  0.104250908
## [14,]   14 -0.098467197  0.016791286
## [15,]   15  0.155034694  0.171752912
## [16,]   16 -0.077163047  0.001091475
## [17,]   17 -0.094720488 -0.031249456
## [18,]   18  0.012389377 -0.049438184
## [19,]   19  0.058436772  0.066342577
## [20,]   20  0.093022189  0.087635397
```

CHAPITRE 5 : MODELISATION SARIMA

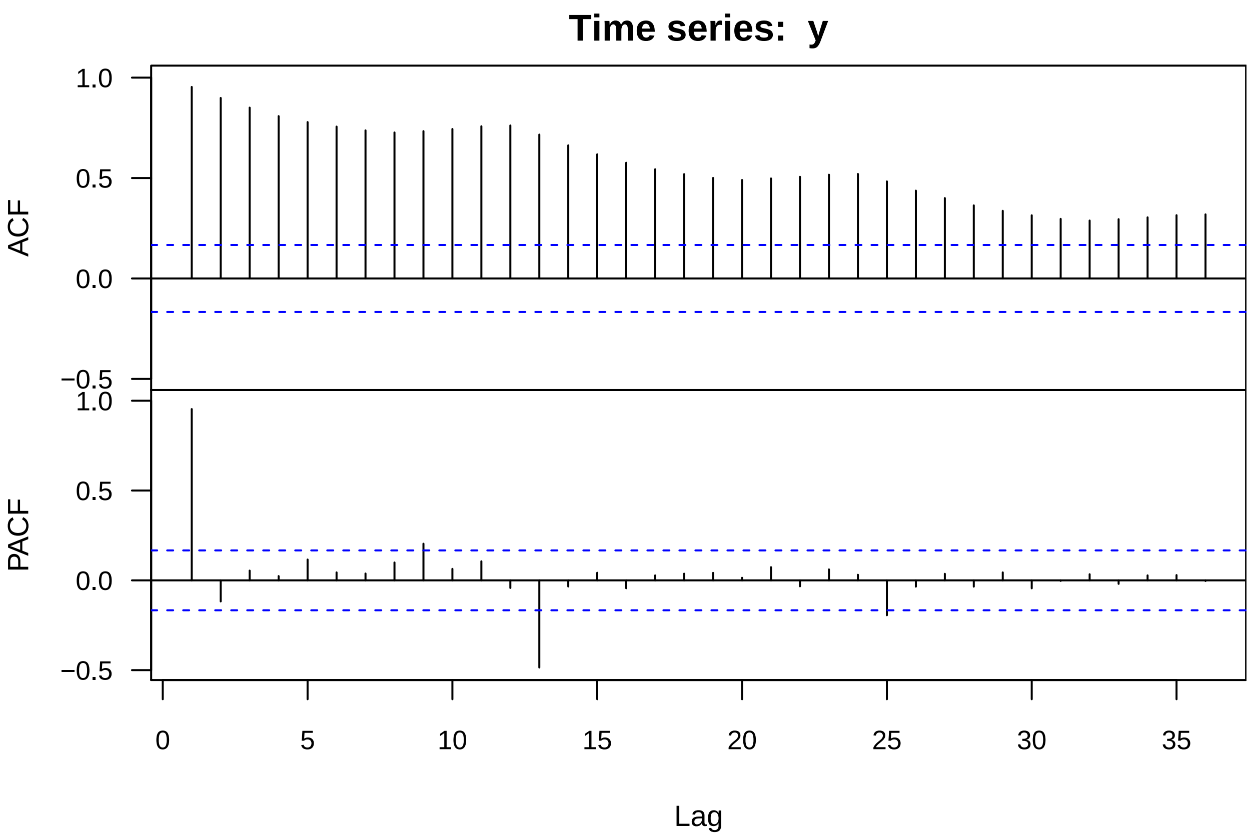
On utilise la librairie `caschrono`.

```r
library(caschrono)
```

Stationnarisation de la série

On désigne par X_t la série *airpass*, et on considère $Y_t = \log(X_t)$. On travaille en effet sur le logarithme de la série afin de pallier l'accroissement de la saisonnalité. On passe ainsi d'un modèle multiplicatif à un modèle additif.

```r
acf2y(y,lag.max=36)
```

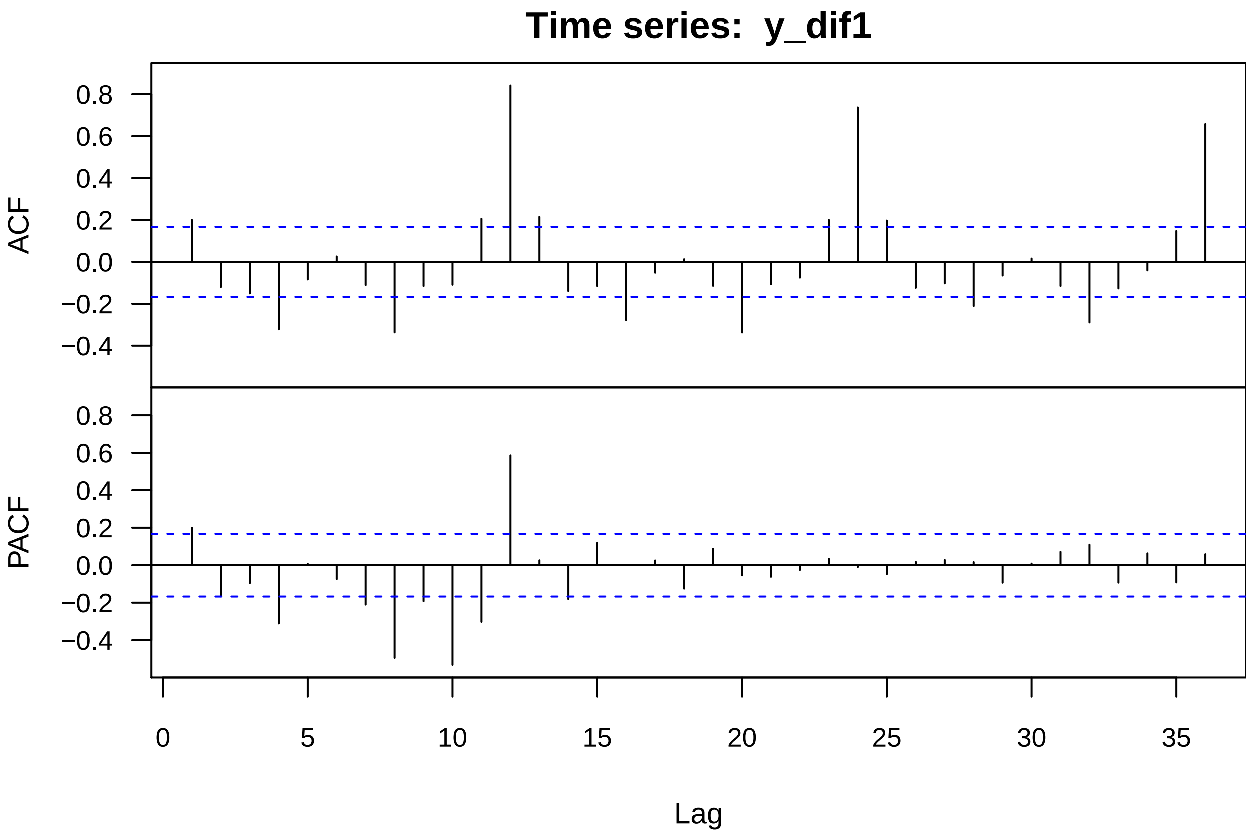



```
##       LAG      ACF1         PACF
##  [1,]   1 0.9537034  0.953703369
##  [2,]   2 0.8989159 -0.117569756
##  [3,]   3 0.8508025  0.054232738
##  [4,]   4 0.8084252  0.023756139
##  [5,]   5 0.7788994  0.115822195
##  [6,]   6 0.7564422  0.044367631
##  [7,]   7 0.7376017  0.038034142
##  [8,]   8 0.7271313  0.099622097
##  [9,]   9 0.7336487  0.204095742
## [10,]  10 0.7442552  0.063909253
## [11,]  11 0.7580266  0.106035483
## [12,]  12 0.7619429 -0.042466275
## [13,]  13 0.7165045 -0.485430132
## [14,]  14 0.6630428 -0.034350194
```

```
## [15,]  15 0.6183629  0.042224535
## [16,]  16 0.5762087 -0.044197224
## [17,]  17 0.5438013  0.027607922
## [18,]  18 0.5194561  0.037147942
## [19,]  19 0.5007029  0.041638426
## [20,]  20 0.4904028  0.014399904
## [21,]  21 0.4981819  0.073312465
## [22,]  22 0.5061666 -0.033395258
## [23,]  23 0.5167434  0.060996727
## [24,]  24 0.5204897  0.031077819
## [25,]  25 0.4835237 -0.194374014
## [26,]  26 0.4373983 -0.035075894
## [27,]  27 0.4004067  0.036454575
## [28,]  28 0.3641309 -0.035175307
## [29,]  29 0.3369823  0.044254783
## [30,]  30 0.3147227 -0.044544574
## [31,]  31 0.2967752 -0.003337424
## [32,]  32 0.2886164  0.034140566
## [33,]  33 0.2953547 -0.019607062
## [34,]  34 0.3045473  0.027721916
## [35,]  35 0.3150961  0.029354141
## [36,]  36 0.3192932 -0.003733930
```

La sortie *ACF* présente une décroissance lente vers 0, ce qui traduit un problème de non-stationnarité. On effectue donc une différenciation $(I - B)$.

```
y_dif1=diff(y,lag=1,differences=1)
acf2y(y_dif1,lag.max=36)
```

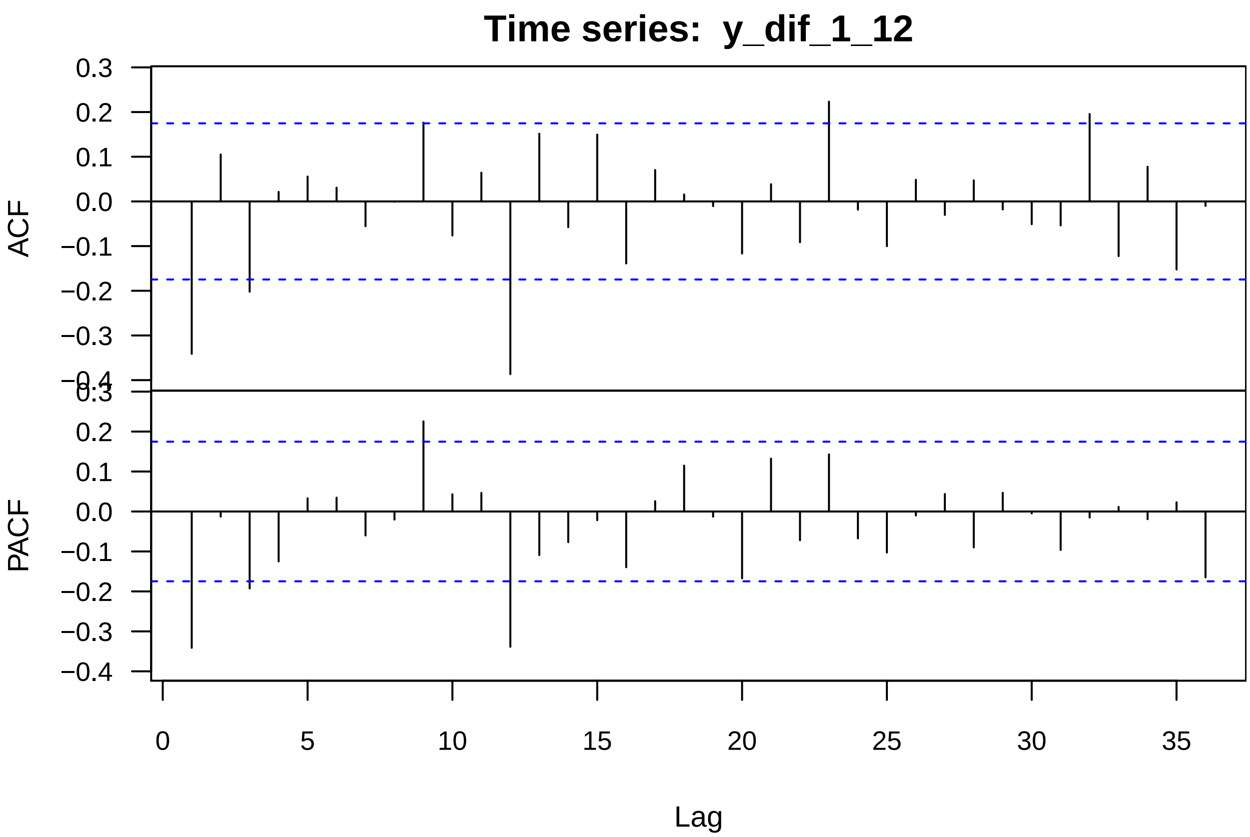



```
##        LAG        ACF1          PACF
##  [1,]   1  0.19975134  0.1997513367
##  [2,]   2 -0.12010433 -0.1666545451
##  [3,]   3 -0.15077204 -0.0958754158
##  [4,]   4 -0.32207432 -0.3108908559
##  [5,]   5 -0.08397453  0.0077849143
##  [6,]   6  0.02577843 -0.0745495301
##  [7,]   7 -0.11096075 -0.2102835666
##  [8,]   8 -0.33672146 -0.4947571488
##  [9,]   9 -0.11558631 -0.1922947064
## [10,]  10 -0.10926704 -0.5318752431
## [11,]  11  0.20585223 -0.3022925255
## [12,]  12  0.84142998  0.5860413494
## [13,]  13  0.21508704  0.0259783575
## [14,]  14 -0.13955394 -0.1811927039
## [15,]  15 -0.11599576  0.1200384625
## [16,]  16 -0.27894284  0.0004076076
## [17,]  17 -0.05170646  0.0252602791
## [18,]  18  0.01245814 -0.1249890976
## [19,]  19 -0.11435760  0.0874347248
## [20,]  20 -0.33717439 -0.0544710878
## [21,]  21 -0.10738490 -0.0618235781
## [22,]  22 -0.07521120 -0.0251952446
## [23,]  23  0.19947518  0.0333314860
## [24,]  24  0.73692070 -0.0096343848
## [25,]  25  0.19726236 -0.0480574740
## [26,]  26 -0.12388430  0.0184867064
```

```
## [27,]  27 -0.10269904  0.0279783480
## [28,]  28 -0.21099219  0.0163094446
## [29,]  29 -0.06535684 -0.0934574262
## [30,]  30  0.01572846  0.0084975849
## [31,]  31 -0.11537038  0.0714613376
## [32,]  32 -0.28925562  0.1095154908
## [33,]  33 -0.12688236 -0.0931830943
## [34,]  34 -0.04070684  0.0631344957
## [35,]  35  0.14741061 -0.0920556030
## [36,]  36  0.65743810  0.0584036191
```

La sortie ACF de la série ainsi différenciée présente encore une décroissance lente vers 0 pour les multiples de 12. On effectue cette fois la différenciation $\left(I - B^{12}\right)$.

```
y_dif_1_12=diff(y_dif1,lag=12,differences=1)
acf2y(y_dif_1_12,lag.max=36)
```




```
##       LAG          ACF1         PACF
##  [1,]   1 -0.3411237983 -0.341123798
##  [2,]   2  0.1050467496 -0.012809250
##  [3,]   3 -0.2021386642 -0.192662435
##  [4,]   4  0.0213592288 -0.125028366
##  [5,]   5  0.0556543435  0.033089658
##  [6,]   6  0.0308036696  0.034677379
##  [7,]   7 -0.0555785695 -0.060186934
##  [8,]   8 -0.0007606578 -0.020223154
```

```
##  [9,]   9  0.1763686815  0.225576717
## [10,]  10 -0.0763581912  0.043070773
## [11,]  11  0.0643839399  0.046588236
## [12,]  12 -0.3866128596 -0.338694805
## [13,]  13  0.1516020121 -0.109178652
## [14,]  14 -0.0576067980 -0.076839449
## [15,]  15  0.1495652202 -0.021750781
## [16,]  16 -0.1389421819 -0.139545243
## [17,]  17  0.0704823385  0.025891863
## [18,]  18  0.0156307241  0.114821992
## [19,]  19 -0.0106106130 -0.013162286
## [20,]  20 -0.1167285978 -0.167430139
## [21,]  21  0.0385542023  0.132403960
## [22,]  22 -0.0913645276 -0.072038705
## [23,]  23  0.2232689055  0.142854473
## [24,]  24 -0.0184181674 -0.067331874
## [25,]  25 -0.1002881161 -0.102667592
## [26,]  26  0.0485657567 -0.010065593
## [27,]  27 -0.0302396339  0.043783534
## [28,]  28  0.0471343505 -0.089951024
## [29,]  29 -0.0180304684  0.046904263
## [30,]  30 -0.0510696473 -0.004895487
## [31,]  31 -0.0537672361 -0.096380590
## [32,]  32  0.1957284827 -0.015278257
## [33,]  33 -0.1224193885  0.011500315
## [34,]  34  0.0777498102 -0.019159019
## [35,]  35 -0.1524548378  0.023034543
## [36,]  36 -0.0099950101 -0.164879721
```

La sortie ACF de la série doublement différenciée semble pouvoir être interprétée comme un autocorrélogramme simple empirique.

On identifiera donc un modèle ARMA sur la série :

$$(I-B)\left(I-B^{12}\right)\log\left(X_t\right).$$

Identification, estimation et validation de modèles

Tous les tests sont effectués au niveau de test 5%.

Modèle 1

On estime en premier lieu un modèle $SARIMA(1,1,1)(1,1,1)_{12}$ au vu des autocorrélogrammes empiriques simples et partiels.

Ce modèle s'écrit :

$$(I-\varphi_1 B)\left(I-\varphi_1' B^{12}\right)(I-B)\left(I-B^{12}\right)\log\left(X_t\right)=(I+\theta_1 B)\left(I+\theta_1' B^{12}\right)\varepsilon_t.$$

```
model1=Arima(y,order=c(1,1,1),list(order=c(1,1,1),period=12),include.mean=FALSE,method="CSS-ML")
summary(model1)
```

```
## Series: y
## ARIMA(1,1,1)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1    sar1     sma1
##       0.1666  -0.5615  -0.099  -0.4973
## s.e.  0.2459   0.2115   0.154   0.1360
##
## sigma^2 estimated as 0.001336:  log likelihood=245.16
## AIC=-480.31   AICc=-479.83   BIC=-465.93
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE      MAPE
## Training set 0.0006239395 0.03489259 0.02595463 0.01199887 0.4696646
##                   MASE       ACF1
## Training set 0.2144266 0.07971558
```

```
t_stat(model1)
```

```
##             ar1       ma1      sar1      sma1
## t.stat 0.677738 -2.654214 -0.642984 -3.657670
## p.val  0.497938  0.007949  0.520235  0.000255
```

```
Box.test.2(model1$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.64051
## [2,]     12 0.81959
## [3,]     18 0.82768
## [4,]     24 0.59646
## [5,]     30 0.75443
## [6,]     36 0.65902
```

Ce modèle ayant des paramètres non significatifs, on en teste un second.

Modèle 2

Ce modèle s'écrit :

$$\left(I - \varphi_1' B^{12}\right)(I - B)\left(I - B^{12}\right)\log\left(X_t\right) = (I + \theta_1 B)\left(I + \theta_1' B^{12}\right)\varepsilon_t.$$

```
model2=Arima(y,order=c(0,1,1),list(order=c(1,1,1),period=12),include.mean=FALSE,method="CSS-ML")
summary(model2)
```

```
## Series: y
## ARIMA(0,1,1)(1,1,1)[12]
##
## Coefficients:
##           ma1     sar1     sma1
##       -0.4143  -0.1116  -0.4817
## s.e.   0.0899   0.1547   0.1363
```

```
##
## sigma^2 estimated as 0.001341:  log likelihood=244.96
## AIC=-481.91   AICc=-481.6   BIC=-470.41
##
## Training set error measures:
##                       ME       RMSE        MAE        MPE      MAPE
## Training set 0.000590882 0.03496264 0.02632396 0.01124103 0.4763403
##                   MASE       ACF1
## Training set 0.2174779 0.02342844
```

```r
t_stat(model2)
```

```
##              ma1      sar1      sma1
## t.stat -4.606259 -0.721477 -3.534060
## p.val   0.000004  0.470616  0.000409
```

```r
Box.test.2(model2$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.52132
## [2,]     12 0.75737
## [3,]     18 0.79542
## [4,]     24 0.55073
## [5,]     30 0.71886
## [6,]     36 0.65343
```

Ce modèle ayant des paramètres non significatifs, on en teste un troisième.

Modèle 3

Ce modèle s'écrit :

$$(I-B)\left(I-B^{12}\right)\log\left(X_t\right)=\left(I+\theta_1 B\right)\left(I+\theta_1' B^{12}\right)\varepsilon_t.$$

```r
model3=Arima(y,order=c(0,1,1),list(order=c(0,1,1),period=12),include.mean=FALSE,method="CSS-ML")
summary(model3)
```

```
## Series: y
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##           ma1     sma1
##       -0.4018  -0.5569
## s.e.   0.0896   0.0731
##
## sigma^2 estimated as 0.001348:  log likelihood=244.7
## AIC=-483.4   AICc=-483.21   BIC=-474.77
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE      MAPE
## Training set 0.0005730622 0.03504883 0.02626034 0.01098898 0.4752815
##                   MASE       ACF1
## Training set 0.2169522 0.02352795
```

```r
t_stat(model3)
```

```
##              ma1      sma1
## t.stat -4.482494 -7.618978
## p.val   0.000007  0.000000
```

```r
Box.test.2(model3$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.51519
## [2,]     12 0.72613
## [3,]     18 0.77822
## [4,]     24 0.50077
## [5,]     30 0.68838
## [6,]     36 0.65352
```

Les tests de significativité des paramètres et de blancheur du résidu sont validés au niveau 5%.

```r
shapiro.test(model3$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model3$residuals
## W = 0.98637, p-value = 0.1674
```

Le test de normalité est également validé pour ce modèle.

Modèle 4

On tente également d'estimer un quatrième modèle avec des polynômes Φ et Φ_s, ainsi que Θ et Θ_s, réunifiés.

Ce modèle s'écrit :

$$\left(I-\varphi_1 B-\varphi_{12} B^{12}\right)(I-B)\left(I-B^{12}\right)\log\left(X_t\right)=\left(I+\theta_1 B+\theta_{12} B^{12}\right)\varepsilon_t .$$

```r
model4=Arima(y,order=c(12,1,12),fixed=c(NA,0,0,0,0,0,0,0,0,0,0,NA,NA,0,0,0,0,0,0,0,0,0,0,NA),list(order=
summary(model4)
```

```
## Series: y
## ARIMA(12,1,12)(0,1,0)[12]
##
## Coefficients:
##           ar1  ar2  ar3  ar4  ar5  ar6  ar7  ar8  ar9  ar10  ar11     ar12
##       -0.2732    0    0    0    0    0    0    0    0     0     0  -0.1151
## s.e.   0.1866    0    0    0    0    0    0    0    0     0     0   0.2170
##           ma1  ma2  ma3  ma4  ma5  ma6  ma7  ma8  ma9  ma10  ma11     ma12
##       -0.0902    0    0    0    0    0    0    0    0     0     0  -0.4625
## s.e.   0.2246    0    0    0    0    0    0    0    0     0     0   0.2145
##
```

```
## sigma^2 estimated as 0.001367:  log likelihood=243.9
## AIC=-477.8   AICc=-477.32   BIC=-463.42
##
## Training set error measures:
##                        ME       RMSE        MAE         MPE      MAPE
## Training set 0.0004817889 0.03528818 0.02675162 0.009240302 0.4839335
##                  MASE        ACF1
## Training set 0.221011 -0.07306001
```

```
t_stat(model4)
```

```
##              ar1      ar12       ma1      ma12
## t.stat -1.464195 -0.530264 -0.401488 -2.155729
## p.val   0.143141  0.595929  0.688061  0.031105
```

```
Box.test.2(model4$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.26230
## [2,]     12 0.48362
## [3,]     18 0.51529
## [4,]     24 0.28378
## [5,]     30 0.45565
## [6,]     36 0.40236
```

Ce modèle ayant des paramètres non significatifs, on en teste un cinquième.

Modèle 5

Ce modèle s'écrit :

$$\left(I-\varphi_1 B-\varphi_{12} B^{12}\right)(I-B)\left(I-B^{12}\right) \log \left(X_t\right)=\left(I+\theta_{12} B^{12}\right) \varepsilon_t .$$

```
model5=Arima(y,order=c(12,1,12),fixed=c(NA,0,0,0,0,0,0,0,0,0,0,NA,0,0,0,0,0,0,0,0,0,0,0,NA),list(order=
summary(model5)
```

```
## Series: y
## ARIMA(12,1,12)(0,1,0)[12]
##
## Coefficients:
##           ar1  ar2  ar3  ar4  ar5  ar6  ar7  ar8  ar9  ar10  ar11     ar12
##       -0.3405    0    0    0    0    0    0    0    0     0     0  -0.0423
## s.e.   0.0817    0    0    0    0    0    0    0    0     0     0   0.1272
##       ma1  ma2  ma3  ma4  ma5  ma6  ma7  ma8  ma9  ma10  ma11     ma12
##         0    0    0    0    0    0    0    0    0     0     0  -0.5350
## s.e.    0    0    0    0    0    0    0    0    0     0     0   0.1137
##
## sigma^2 estimated as 0.001366:  log likelihood=243.8
## AIC=-479.6   AICc=-479.28   BIC=-468.1
##
## Training set error measures:
```

```
##                       ME       RMSE        MAE         MPE      MAPE
## Training set 0.0004553055 0.03527657 0.02664365 0.008906562 0.4818932
##                  MASE        ACF1
## Training set 0.220119 -0.09152048
```

```r
t_stat(model5)
```

```
##              ar1      ar12      ma12
## t.stat -4.165473 -0.332246 -4.703712
## p.val   0.000031  0.739703  0.000003
```

```r
Box.test.2(model5$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.23099
## [2,]     12 0.41831
## [3,]     18 0.44120
## [4,]     24 0.22790
## [5,]     30 0.37751
## [6,]     36 0.34766
```

Ce modèle ayant des paramètres non significatifs, on en teste un sixième.

Modèle 6

Ce modèle s'écrit :

$$(I-\varphi_1 B)(I-B)\left(I-B^{12}\right)\log(X_t)=\left(I+\theta_{12}B^{12}\right)\varepsilon_t.$$

```r
model6=Arima(y,order=c(1,1,12),fixed=c(NA,0,0,0,0,0,0,0,0,0,0,0,NA),list(order=c(0,1,0),period=12),inclu
summary(model6)
```

```
## Series: y
## ARIMA(1,1,12)(0,1,0)[12]
##
## Coefficients:
##          ar1  ma1  ma2  ma3  ma4  ma5  ma6  ma7  ma8  ma9  ma10  ma11
##      -0.3395    0    0    0    0    0    0    0    0    0     0     0
## s.e.  0.0822    0    0    0    0    0    0    0    0    0     0     0
##         ma12
##      -0.5619
## s.e.  0.0748
##
## sigma^2 estimated as 0.001367:  log likelihood=243.74
## AIC=-481.49   AICc=-481.3   BIC=-472.86
##
## Training set error measures:
##                       ME       RMSE        MAE         MPE      MAPE
## Training set 0.0004500154 0.03529899 0.02662601 0.008828412 0.4816646
##                   MASE        ACF1
## Training set 0.2199733 -0.08828148
```

```r
t_stat(model6)
```

```
##              ar1      ma12
## t.stat -4.129480 -7.510894
## p.val   0.000036  0.000000
```

```r
Box.test.2(model6$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

```
##      Retard p-value
## [1,]      6 0.24413
## [2,]     12 0.43316
## [3,]     18 0.45925
## [4,]     24 0.23920
## [5,]     30 0.39768
## [6,]     36 0.38129
```

Les tests de significativité des paramètres et de blancheur du résidu sont validés au niveau 5%.

```r
shapiro.test(model6$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model6$residuals
## W = 0.98611, p-value = 0.1569
```

Le test de normalité est également validé pour ce modèle.

Procédure de sélection automatique de modèles

Nous pouvons constater que la sélection automatique testée ici n'est pas concluante.

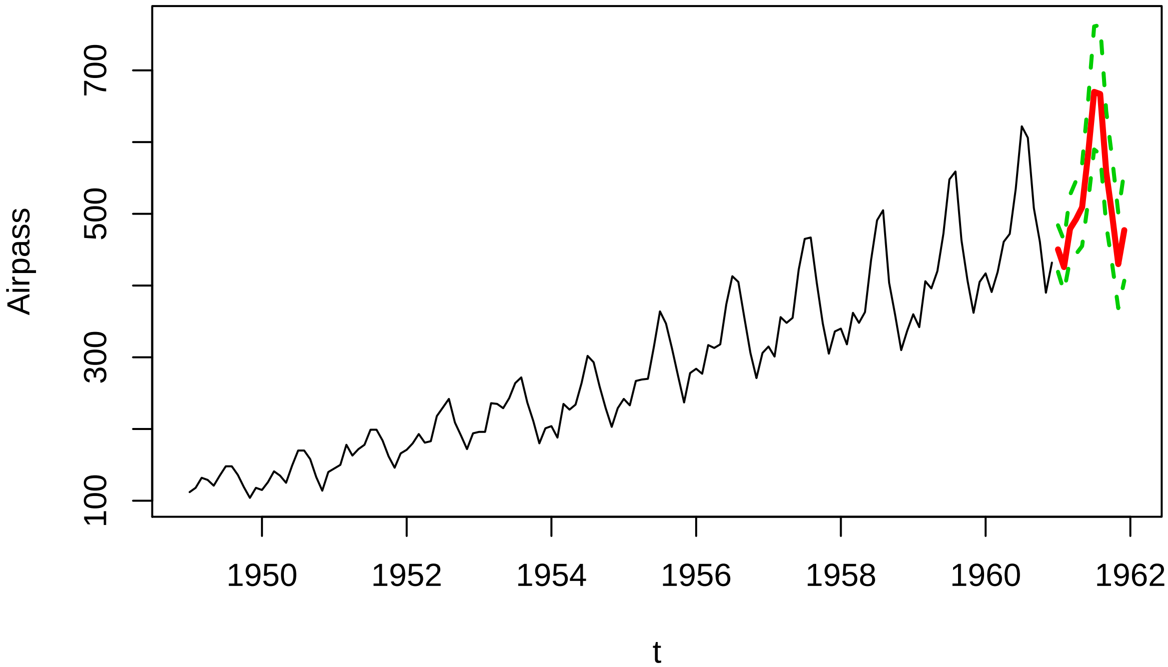
```r
armaselect(y_dif_1_12,max.p=20,max.q=20,nbmod=10)
```

```
##       p  q       sbc
##  [1,] 1  1 -829.6517
##  [2,] 0  1 -827.8830
##  [3,] 1  2 -826.4204
##  [4,] 2  1 -824.7766
##  [5,] 0  2 -824.5979
##  [6,] 3  1 -824.1854
##  [7,] 0 12 -823.8556
##  [8,] 1 20 -822.1977
##  [9,] 2  2 -822.1365
## [10,] 1  3 -822.1087
```

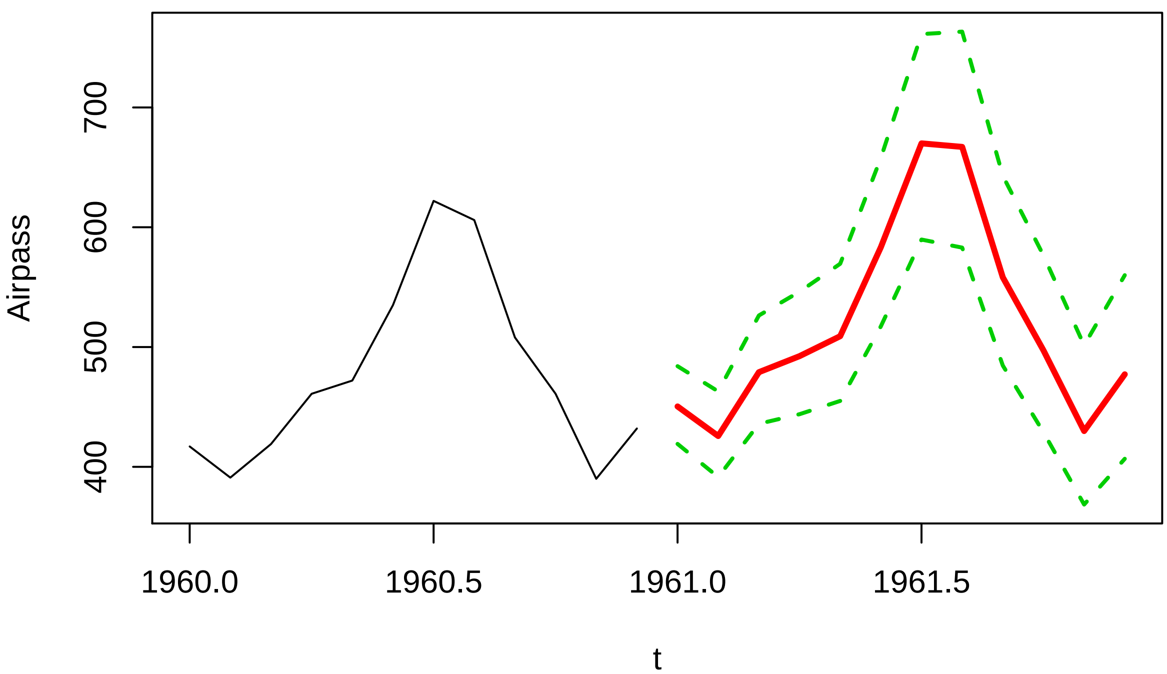
Prévision à l'aide du modèle retenu (3) de l'année 1961

Le BIC du troisième modèle vaut -474.77, contre -472.86 pour le sixième modèle, on retient donc le modèle 3.

```
pred_model3=forecast(model3,h=12,level=95)
pred=exp(pred_model3$mean)
pred_l=ts(exp(pred_model3$lower),start=c(1961,1),frequency=12)
pred_u=ts(exp(pred_model3$upper),start=c(1961,1),frequency=12)
ts.plot(x,pred,pred_l,pred_u,xlab="t",ylab="Airpass",col=c(1,2,3,3),lty=c(1,1,2,2),lwd=c(1,3,2,2))
```




```
ts.plot(window(x,start=c(1960,1)),pred,pred_l,pred_u,xlab="t",ylab="Airpass",col=c(1,2,3,3),lty=c(1,1,2
```



Analyse a posteriori

On tronque la série de l'année 1960, qu'on cherche ensuite à prévoir à partir de l'historique 1949-1959.

```
x_tronc=window(x,end=c(1959,12))
y_tronc=log(x_tronc)
x_a_prevoir=window(x,start=c(1960,1))
```

On vérifie que le modèle 3 sur la série tronquée est toujours validé.

```
model3tronc=Arima(y_tronc,order=c(0,1,1),list(order=c(0,1,1),period=12),include.mean=FALSE,method="CSS-
summary(model3tronc)
```

```
## Series: y_tronc
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##           ma1     sma1
##       -0.3484  -0.5623
## s.e.   0.0943   0.0774
##
## sigma^2 estimated as 0.001313:  log likelihood=223.63
## AIC=-441.26   AICc=-441.05   BIC=-432.92
##
## Training set error measures:
##                      ME       RMSE        MAE        MPE      MAPE
## Training set 0.00104934 0.03443221 0.02590904 0.01899277 0.4738142
##                   MASE       ACF1
## Training set 0.2113963 0.04394741
```

```
t_stat(model3tronc)
```

```
##              ma1      sma1
## t.stat -3.695894 -7.262873
## p.val   0.000219  0.000000
```

```
Box.test.2(model3tronc$residuals,nlag=c(6,12,18,24,30,36),type="Ljung-Box",decim=5)
```

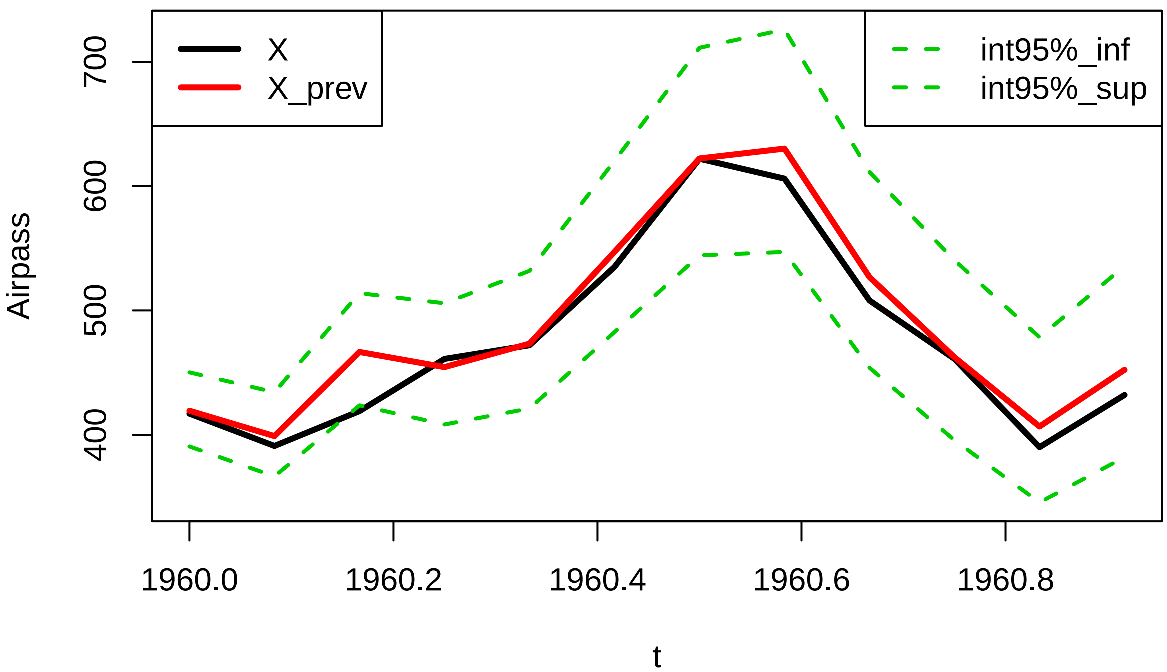
```
##      Retard p-value
## [1,]      6 0.52539
## [2,]     12 0.85631
## [3,]     18 0.87341
## [4,]     24 0.78327
## [5,]     30 0.90181
## [6,]     36 0.84635
```

```
shapiro.test(model3tronc$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model3tronc$residuals
## W = 0.988, p-value = 0.3065
```

On constate que la réalisation 1960 est dans l'intervalle de prévision à 95% (basé sur les données antérieures à 1959).

```
pred_model3tronc=forecast(model3tronc,h=12,level=95)
pred_tronc=exp(pred_model3tronc$mean)
pred_l_tronc=ts(exp(pred_model3tronc$lower),start=c(1960,1),frequency=12)
pred_u_tronc=ts(exp(pred_model3tronc$upper),start=c(1960,1),frequency=12)
ts.plot(x_a_prevoir,pred_tronc,pred_l_tronc,pred_u_tronc,xlab="t",ylab="Airpass",col=c(1,2,3,3),lty=c(1
legend("topleft",legend=c("X","X_prev"),col=c(1,2,3,3),lty=c(1,1),lwd=c(3,3))
legend("topright",legend=c("int95%_inf","int95%_sup"),col=c(3,3),lty=c(2,2),lwd=c(2,2))
```



On calcule les RMSE et MAPE.

```
rmse=sqrt(mean((x_a_prevoir-pred_tronc)^2))
rmse
```

```
## [1] 18.59359
```

```
mape=mean(abs(1-pred_tronc/x_a_prevoir))*100
mape
```

```
## [1] 2.904473
```

L'interprétation des critères d'erreur dépend de la série et de la qualité de prévision exigée. Dans le cas présent, un MAPE de 2.9% semble satisfaisant a priori.