Modèles de base en séries temporelles (compléments du Chapitre 4)

Yves Aragon*
Université Toulouse Capitole

28 janvier 2019

4.1 Stationnarité

Exercice 4.1 (Danone - test de blancheur)
Test de la blancheur du rendement de l'action Danone et celle de son carré. On suivra les étapes suivantes :

1. Calculer le carré du rendement centré.
2. Tester la blancheur du rendement sur toute la série (on pourra tester la blancheur aux retards 3, 6, 9 et 12). On obtient un résultat inattendu. Lequel ?
3. Après examen du chronogramme du rendement (fig. 1.4, chap. 1) on décide de se limiter à l'étude de la série des 600 premières valeurs. Pourquoi ce choix ? Qu'a-t-on observé sur le chronogramme ?
4. Tester la blancheur du rendement et du rendement centré au carré sur la série de ces 600 valeurs. Conclusion ?
5. Au vu de ces résultats, le rendement peut-il être un bruit blanc gaussien ?

Réponse.

Q-1. Le calcul du rendement s'effectue sans difficulté, voir le code ci-dessous.

Q-2. Si nous prenons toute la série, nous rejetons la blancheur du rendement, colonne `p-val. série compl.` du Tableau 1, or il est très rare qu'un rendement ne soit pas un bruit blanc. Nous examinons donc la série et constatons, fig. 1.4, chap. 1, qu'elle fluctue beaucoup autour de la 700^{e} observation. Aussi nous effectuons maintenant le test sur la sous-série des 600 premières valeurs.

```
> require("caschrono")
> require("timeSeries")
> data("csdl")
> aa <- returns(csdl, percentage = TRUE)
> aab <- aa[complete.cases(aa) == TRUE,]
> # in previous version we use package its which will not be maintaine
> # r.csdl = its(aab, as.POSIXct(row.names(aab)))
> r.csdl <- zoo(aab, as.POSIXct(row.names(aab)))
```

*yves.aragon@gmail.com

```
> r.danone <- r.csdl[, 3]
> rdt2 <- (r.danone-mean(r.danone))^2
> a0 <- Box.test.2(r.danone, nlag = c(3, 6, 9, 12),
+                  type = "Ljung-Box", decim = 4)
> a1 <- Box.test.2(r.danone[1:600], nlag = c(3, 6, 9, 12),
+                  type = "Ljung-Box", decim = 4)
> a2 <- Box.test.2(rdt2[1:600], nlag = c(3, 6, 9, 12),
+                  type = "Ljung-Box", decim = 4)
> a12 <- cbind(a0, a1[, 2], a2[, 2])
> colnames(a12) <- c("Retard", "p-val. serie compl.",
+                    "p-val. 600 obs.", "p-val. rdt carre")

> require(xtable)
> xtable(a12, caption = "Table a12 : test de blancheur du rende
+        de Danone et de son carre.", label = "danoblanc",
+        digits = 4)
```

	Retard	p-val. serie compl.	p-val. 600 obs.	p-val. rdt carre
1	3.0000	0.0005	0.2718	0.0000
2	6.0000	0.0007	0.2599	0.0000
3	9.0000	0.0012	0.3090	0.0000
4	12.0000	0.0039	0.1304	0.0000

Tableau 1 – Table a12 : test de blancheur du rendement de Danone et de son carre.

Q-3. On constate que le rendement est un bruit blanc, mais que son carré, dernière colonne de la table (1), n'en est pas un. Notons qu'on a centré le rendement avant de l'élever au carré, au cas où il serait de moyenne non nulle.

Q-4. Un bruit blanc gaussien (BBN) est une suite de v.a. *indépendantes* identiquement distribuées. Si le rendement de Danone était un BBN alors son carré serait une suite de v.a. indépendantes, identiquement distribuées, et donc un bruit blanc, ce qu'il n'est pas, donc le rendement de Danone ne peut pas être un BBN.

4.2 Séries linéaires

Représentation MA(∞) des séries :

$$\begin{aligned} y_t &= -0.7 y_{t-1} + z_t, \text{ série } \texttt{y1} \\ y_t &= -0.7 y_{t-12} + z_t, \text{ série } \texttt{y2} \end{aligned}$$

Par `ARMAtoMA()` :

```
> ARMAtoMA(-.7, 0, 10)

 [1] -0.70000000  0.49000000 -0.34300000  0.24010000
 [5] -0.16807000  0.11764900 -0.08235430  0.05764801
 [9] -0.04035361  0.02824752

> ARMAtoMA(c(rep(0, 11), -.7), 0, 25)

 [1]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
[10]  0.00  0.00 -0.70  0.00  0.00  0.00  0.00  0.00  0.00
[19]  0.00  0.00  0.00  0.00  0.00  0.49  0.00
```

Par `ImpulseCoefficientsARMA()` de **FitARMA** :

```
> require(FitARMA)
> ImpulseCoefficientsARMA(-.7, 0, lag.max = 10)

 [1]  1.00000000 -0.70000000  0.49000000 -0.34300000
 [5]  0.24010000 -0.16807000  0.11764900 -0.08235430
 [9]  0.05764801 -0.04035361  0.02824752

> ImpulseCoefficientsARMA(c(rep(0, 11), -.7), 0, lag.max = 25)

 [1]  1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
[10]  0.00  0.00  0.00 -0.70  0.00  0.00  0.00  0.00  0.00
[19]  0.00  0.00  0.00  0.00  0.00  0.00  0.49  0.00
```

On appelle également fonction de réponse impulsionnelle la suite des ψ_i dans la représentation MA(∞) car ψ_i est la réponse de y_t à un choc de "1" sur le bruit en $t-i$.

4.3 Comparaison ACF et PACF théoriques et empiriques – exemples

Dans ces exemples on simule des trajectoires de 200 observations suivant des modèles particuliers. On trace l'ACF et la PACF théoriques, ainsi que leurs versions empiriques pour la trajectoire simulée. La comparaison des versions théoriques et empiriques permet de se convaincre que l'identification d'un modèle par les ACF et PACF empiriques n'est pas toujours chose aisée.

▶ Modèle MA(2) Considérons le modèle

$$y_t = (1 - 0.3\mathrm{B} + 0.6\mathrm{B}^2)z_t \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.2}$$

D'une part, `ARMAacf()` permet de calculer les ACF et PACF théoriques de cette série. D'autre part, nous en simulons une trajectoire par `arima.sim()` enfin par `acf()` appliquée à la trajectoire simulée, nous calculons des versions empiriques de l'ACF et de la PACF. Nous pouvons donc comparer les versions empiriques et théoriques obtenues.

```
> set.seed(951)
> ya <- arima.sim(n = 200, list(ma = c(-0.3, 0.6)), sd = sqrt(1.5))
```

Les différents résultats sont organisés en liste pour la commodité de la représentation graphique des quatre fonctions. Les bandes autour de 0 sont d'ordonnées $\pm 1.96/\sqrt{200}$ (fig. 1). Les deux ACF sont très ressemblantes et typiques d'un MA(2). On voit sur la figure que l'ACF empirique prend des valeurs non significativement différentes de 0 dès le retard 3 et, au retard 11, la valeur semble un peu significative. Les PACF ne sont pas simples à interpréter ici, mais leur examen suffit à comprendre que le modèle n'est pas un AR pur. Les graphiques de ces quatre fonctions, sans marges entre eux, sont obtenus par `plotacfthemp()` de **caschrono**.

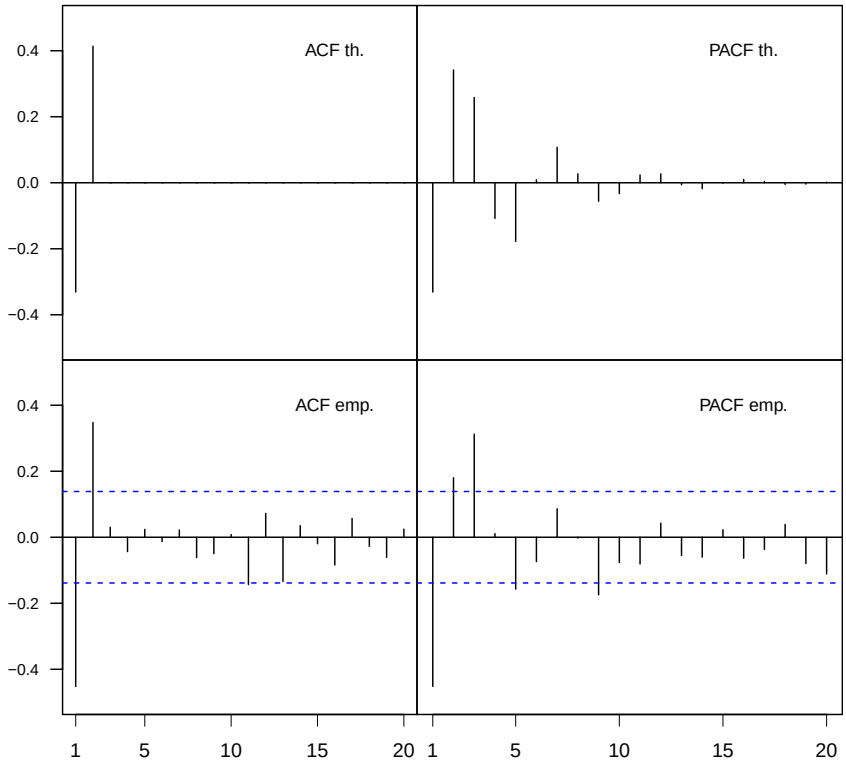

Fig. 1 – Fonctions d'autocorrélation d'un MA(2).

Si l'on veut estimer un MA(2) sur la série `ya`, on écrit la commande

```
> require("forecast")
> (mod.ma2 = Arima(ya, order = c(0, 0,2), include.mean = FALSE))
```

```
Series: ya
ARIMA(0,0,2) with zero mean

Coefficients:
          ma1     ma2
      -0.3986  0.5722
s.e.   0.0551  0.0632

sigma^2 estimated as 1.387:  log likelihood=-315.94
AIC=637.88   AICc=638   BIC=647.77
```

▶ Modèle AR(1) :

$$y_t = -10 + \frac{1}{1+0.8\mathrm{B}} z_t \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.3}$$

On en simule une trajectoire de 200 observations et on trace, fig. 2, les ACF et PACF théoriques et empiriques.

```
> set.seed(5419)
> n2 <- 210
> yb <- arima.sim(n = 200, list(ar = -0.8), sd = sqrt(1.5))
> yb <- yb - 10
```

Les PACF théorique et empirique sont proches. Sur le graphique de l'ACF théorique on vérifie qu'aucune autocorrélation n'est non nulle, alors que sur le graphique de l'ACF empirique, si on ne dispose pas simultanément de celui de la PACF empirique, on ne sait pas si les valeurs dans la bande autour de 0 sont des estimations de 0 ou bien des estimations de quantités faibles mais non nulles. Dans la pratique, il faut donc envisager les deux possibilités, estimation de 0 ou estimation de quantités faibles, tant que l'une ou l'autre de ces deux situations n'est pas très improbable.
L'estimation de (4.3), par exemple sur `yb`, s'obtient par

```
> (mod12 = Arima(yb, order = c(1, 0, 0)))

Series: yb
ARIMA(1,0,0) with non-zero mean

Coefficients:
          ar1      mean
      -0.7875  -10.0338
s.e.   0.0429    0.0462

sigma^2 estimated as 1.372:  log likelihood=-314.87
AIC=635.73   AICc=635.86   BIC=645.63
```

On voit que `Arima()` note `intercept`, ce qui est ici, la moyenne de la série. La notation est ambigue ; nous y reviendrons au chapitre 5 et dans ses exercices.

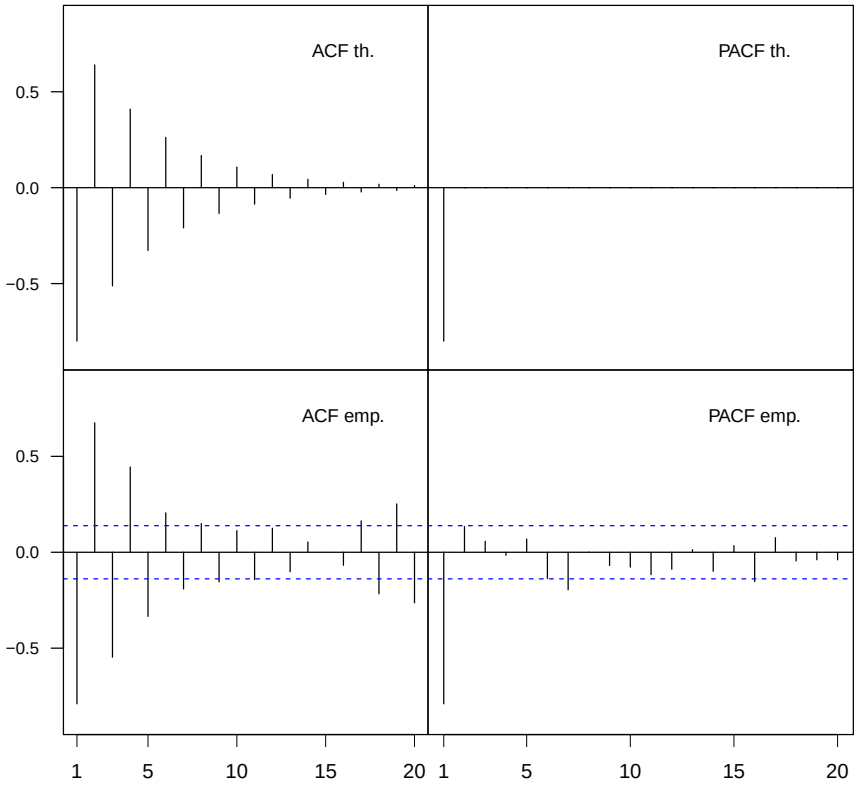

Fig. 2 – Fonctions d'autocorrélation d'un AR(1).

4.3.1 Fonction d'autocorrélation partielle

Exercice 4.2
Simuler une trajectoire de 200 valeurs d'un processus autorégressif obéissant à :

$$y_t = -0.7y_{t-1} + 0.2y_{t-2} + z_t \tag{4.4}$$

et calculer la PACF empirique jusqu'au retard 4. Comparer avec l'exemple précédent.

Réponse. Il nous faut fixer la variance du bruit. Choisissons $\sigma_z^2 = 2$. D'autre part nous effectuons `nsim` simulations pour comparer la précision de l'estimateur et nous superposons valeur théorique et quantile empirique d'ordres .25 et .75 pour chaque retard :

```
> require(FitARMA)
> set.seed(51)
> nsim <- 100
> nobs <- 200
> nsim <- 50
> nlag <- 20
> y.mat <- matrix(0, nrow = nobs, ncol = nsim)
> facp.mat <- matrix(0, nrow = nlag, ncol = nsim)
> y.mat <- matrix(0, nrow = 200, ncol = nsim)
> facp.mat <- matrix(0, nrow = nlag, ncol = nsim)
> for(isim in 1:nsim){
+  y.mat[, isim] <- arima.sim(n = nobs,
+                             list(ar = c(-0.7, 0.2)),
+                             sd = sqrt(2))
+  facp.mat[, isim] = pacf(y.mat[,isim], 20, plot = FALSE)$acf
+ }
> aa <- t(apply(facp.mat,1, "quantile", probs = c(0.25, .75)))
> # pacf theo
> theo <- TacvfARMA(phi = c(-.7, .2), lag.max = 20)
> pacf.theo <- PacfDL(theo/theo[1], LinearPredictor = TRUE)$Pacf
> # intervalle autour de 0 \`a 50%
> binf <- qnorm(.25)/nobs^.5
> bsup <- qnorm(.75)/nobs^.5
> aaa <- cbind(aa, pacf.theo, binf, bsup)
```

`binf` et `bsup` définissent la bande de confiance asymptotique, valable évidemment à partir du retard 3, voir la propriété 4.6, alors que les quantiles d'ordres .25 et .75, matrice `aa`, donnent les quantiles empiriques d'après les 50 simulations. On obtient le graphique (fig. 3) pour 50 simulations par

```
> matplot(1:20, aaa, type = "l", ylab = "PACF",
+         xlab = "retard", col = "black")
> legend("topright", paste("nombre de simulations : ",
+                          as.character(nsim)))
```

Il est instructif de faire le graphique pour différents nombres de simulations ; on se fait ainsi une idée de la vitesse de convergence vers la loi asymptotique de la propriété 4.6.

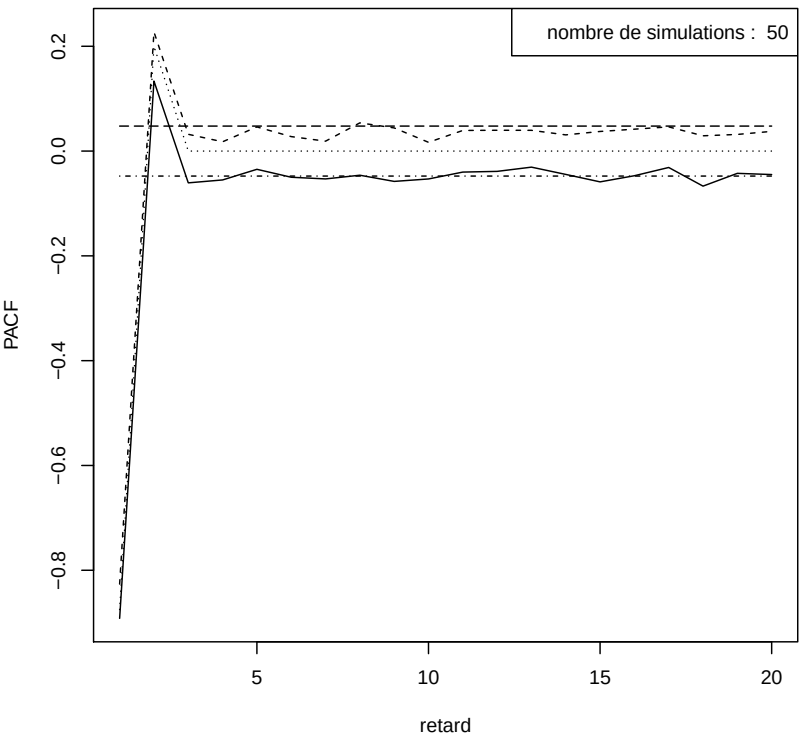

Fig. 3 – Estimations de PACF.

Exercice 4.3
Ajuster un MA(1) à y_t simulé suivant :

$$y_t = -0.7y_{t-1} + z_t, \text{ série } \texttt{y1} \tag{4.5a}$$
$$y_t = -0.7y_{t-12} + z_t, \text{ série } \texttt{y2} \tag{4.5b}$$

où z_t est un bruit blanc gaussien de variance 4, c'est-à-dire un modèle incorrect. Effectuer un test montrant que ce modèle ne convient pas.

Réponse. Nous ajustons un MA(1) à `y1` qui suit en réalité un AR(1).

```
> (mod2 <- Arima(y1, order = c(0, 0, 1), include.mean = FALSE))

Series: y1
ARIMA(0,0,1) with zero mean

Coefficients:
          ma1
      -0.5235
s.e.   0.0686

sigma^2 estimated as 3.502:  log likelihood=-204.22
AIC=412.43   AICc=412.56   BIC=417.64
```

On n'observe rien de particulier, il n'y a pas d'avertissement sur une éventuelle mauvaise convergence de l'algorithme d'estimation. Testons maintenant la blancheur du résidu.

```
> aa <- Box.test.2(residuals(mod2), nlag = c(3, 6, 9, 12),
+                  type = "Ljung-Box",
+                  decim = 4, fitdf = 1)
> colnames(aa) <- c("Retard", "p-val.")
> t(aa)

         [,1]   [,2]   [,3]    [,4]
Retard 3.0000 6.0000 9.0000 12.0000
p-val. 0.0049 0.0389 0.1041  0.1234
```

On voit que le résidu de l'ajustement n'est pas un bruit blanc car, au moins parmi les autocorrélations d'ordre 1 à 6 certaines ne peuvent être considérées comme nulles. L'ajustement est mauvais et il faut chercher autre chose. C'est l'examen de l'ACF et de la PACF de la série qui guide vers un bon modèle. Cette démarche, appelée *identification* de la série, est détaillée à la section 4.6.

Exercice 4.4 (Modèle MA(2))
On considère le modèle MA(2)

$$y_t = (1 - 0.3\mathrm{B} + 0.6\mathrm{B}^2)z_t \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.6}$$

1. Calculer les ACF et PACF théoriques de cette série par `ARMAacf()`.
2. Simuler une trajectoire de 200 observations de (4.6) par `arima.sim()`.
3. Par `acf()` appliquée à la trajectoire simulée, calculer des versions empiriques de l'ACF et de la PACF.

4. Comparer graphiquement les versions théorique et empirique de chaque fonction.

Réponse.

```
> theo <- TacvfARMA(theta = c(.3, -.6), lag.max = 20)
> (acf.theo <- theo[-1]/theo[1])

 [1] -0.3310345  0.4137931  0.0000000  0.0000000  0.0000000
 [6]  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
[11]  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
[16]  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000

> (pacf.theo <- PacfDL(theo/theo[1], LinearPredictor = TRUE)$Pacf)

 [1] -0.331034483  0.341648416  0.258464738 -0.107800685
 [5] -0.177796528  0.008869934  0.107363621  0.026964063
 [9] -0.055940840 -0.032871596  0.023626742  0.026775283
[13] -0.006130026 -0.017895212 -0.001691987  0.010227663
[17]  0.004083332 -0.004911226 -0.003923235  0.001769693

> set.seed(12)
> y <- arima.sim(n = 200, list(ma = c(-0.3, .6)), sd = sqrt(1.5))
> (acf.emp <- acf(y, 20, plot = FALSE)$acf[-1])

 [1]  0.467609888 -0.053867386  0.050203331 -0.052461522
 [5]  0.042538546 -0.111115234  0.056911396 -0.058921853
 [9] -0.062354154  0.002589215 -0.141154640  0.049879337
[13] -0.080060148  0.033042598 -0.007376373  0.015161314
[17]  0.000108792 -0.017187224 -0.033206065

> (pacf.emp <- pacf(y, 20, plot = FALSE)$acf)

, , 1

              [,1]
 [1,] -0.367623075
 [2,]  0.384415684
 [3,]  0.262227005
 [4,] -0.114402582
 [5,] -0.233375690
 [6,]  0.025826224
 [7,]  0.029313781
 [8,]  0.011307431
 [9,] -0.002173825
[10,] -0.132381196
[11,] -0.066497305
[12,] -0.094200841
[13,]  0.052820111
[14,]  0.068290583
[15,]  0.007313172
[16,] -0.029683977
[17,] -0.035758730
```

```
[18,]  0.003764168
[19,] -0.040858358
[20,] -0.063321776
```

`acf()` fournit les autocorrélations à partir du retard 0, tandis que `pacf()` fournit les autocorrélations partielles à partir du retard 1. Noter également les changements de convention pour les paramètres ϕ et θ entre `arima()` et les fonctions de **FitAR** ou **FitARMA**

Exercice 4.5 (Modèle AR(1))
On considère le modèle AR(1) :

$$y_t = -10 + \frac{1}{1+0.8\mathrm{B}} z_t \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.7}$$

Répondre pour ce modèle, aux questions de l'exercice précédent.

Réponse.

```
> theo <- TacvfARMA(phi = -.8, lag.max = 20)
> (acf.theo <- theo[-1]/theo[1])

 [1] -0.80000000  0.64000000 -0.51200000  0.40960000
 [5] -0.32768000  0.26214400 -0.20971520  0.16777216
 [9] -0.13421773  0.10737418 -0.08589935  0.06871948
[13] -0.05497558  0.04398047 -0.03518437  0.02814750
[17] -0.02251800  0.01801440 -0.01441152  0.01152922

> (pacf.theo <- PacfDL(theo/theo[1], LinearPredictor = TRUE)$Pacf)

 [1] -8.000000e-01 -3.083953e-16 -6.853229e-17  3.083953e-17
 [5] -6.167906e-17 -6.167906e-17  6.167906e-17  1.541976e-17
 [9] -3.083953e-17  7.709882e-18  2.312965e-17  5.396917e-17
[13]  2.698459e-17 -1.156482e-17 -7.709882e-18  1.927471e-18
[17] -3.854941e-18 -3.854941e-18  3.854941e-18  9.637353e-19

> set.seed(23)
> y <- arima.sim(n = 200, list(ar = -.8),
+                sd = sqrt(1.5)) - 10
> (acf.emp <- acf(y, 20, plot = FALSE)$acf[-1])

 [1]  0.65980412 -0.57744654  0.46447446 -0.37815454
 [5]  0.31819409 -0.28396716  0.24067700 -0.16848600
 [9]  0.11836334 -0.07103647  0.01241517  0.02174048
[13] -0.05019808  0.08184075 -0.07943445  0.07794366
[17] -0.06185864  0.04575964 -0.03761195

> (pacf.emp <- pacf(y, 20, plot = FALSE)$acf)

, , 1

              [,1]
 [1,] -0.802096305
 [2,]  0.046112504
```

```
 [3,] -0.100137863
 [4,] -0.098953087
 [5,] -0.009535549
 [6,]  0.011118743
 [7,] -0.064007456
 [8,] -0.026238694
 [9,]  0.100950172
[10,] -0.016090384
[11,]  0.033308665
[12,] -0.048546608
[13,] -0.002283685
[14,] -0.026744241
[15,]  0.035697359
[16,]  0.057397743
[17,] -0.004376272
[18,]  0.051427477
[19,] -0.007360111
[20,]  0.002049935
```

Exercice 4.6
Simuler une trajectoire de 200 observations du modèle ARMA(1,2) combinant les composantes autorégressive et moyenne mobile des modèles précédents (4.7 et 4.6) et comparer les ACF et PACF théoriques et empiriques.

Réponse. (Simuler une trajectoire de 200 ...) Le modèle à simuler est

$$y_t = -10 + \frac{1 - 0.3\mathrm{B} + 0.6\mathrm{B}^2}{1 + 0.8\mathrm{B}} z_t, \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.8}$$

La simulation ne pose pas de difficultés :

```
> set.seed(4123)
> yc <- arima.sim(n = 200, list(ar = -0.8, ma = c(-0.3, 0.6)),
+                 sd = sqrt(1.5)) - 10
```

On calcule d'autre part les ACF et PACF théroriques

```
> acf.th <- ARMAacf(ar = -0.8, ma = c(-0.3, 0.6),
+                   lag.max = 20, pacf = FALSE)
> pacf.th <- ARMAacf(ar = -0.8, ma = c(-0.3, 0.6),
+                    lag.max = 20, pacf = TRUE)
```

Enfin on représente ces quatre fonctions sur un même graphique, fig. 4 :

```
> plotacfthemp(yc, ar = -0.8, ma = c(-0.3, 0.6),
+              lag.max = 20)
```

L'estimation de (4.8), par exemple sur `yc`, s'obtient par

```
> (mod12 = Arima(yc, order=c(1, 0, 2)))
```

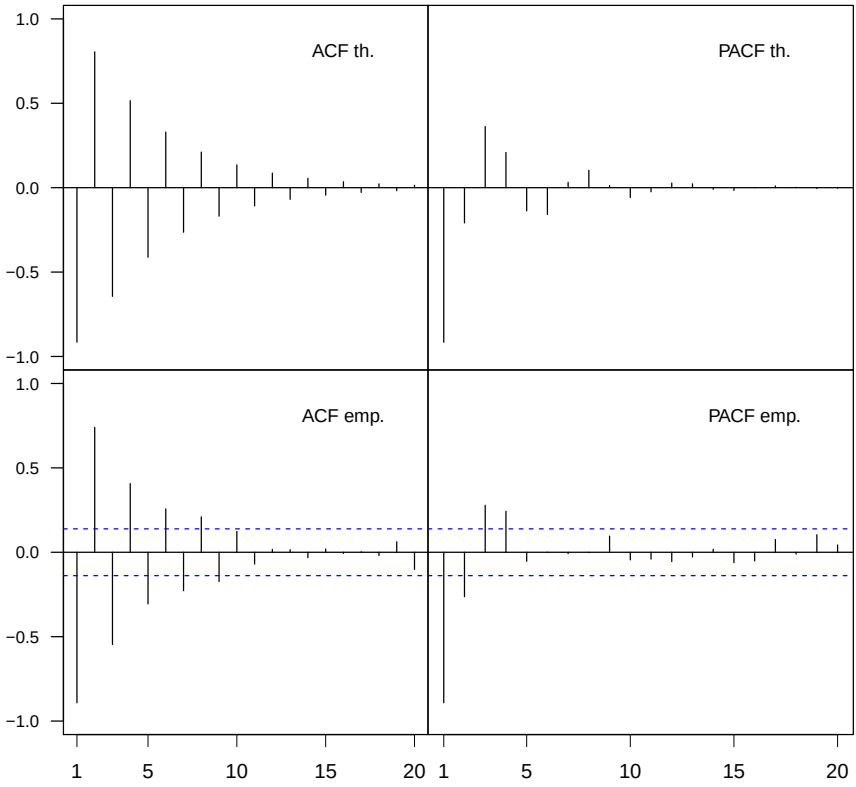

Fig. 4 – Fonctions d'autocorrélation d'un ARMA(1,2).

```
Series: yc
ARIMA(1,0,2) with non-zero mean

Coefficients:
          ar1      ma1     ma2      mean
      -0.7743  -0.3980  0.6121  -10.0162
s.e.   0.0503   0.0606  0.0619    0.0599

sigma^2 estimated as 1.569:  log likelihood=-328.27
AIC=666.53   AICc=666.84   BIC=683.03
```

On voit que `Arima()` note `intercept`, ce qui est ici, la moyenne de la série.

Exercice 4.7
`armasubsets()` de **TSA** aide à l'identification des modèles ARMA suivant le même principe que `armaselect()` mais reconnaît les trous (les plages de coefficients nuls) dans les ordres de régression. Utiliser `armasubsets()` pour identifier le modèle de `yd` simulé suivant

$$y_t = 4 + \frac{1+0.6\mathrm{B}^2}{(1+0.8\mathrm{B})(1-0.7\mathrm{B}^4)} z_t, \qquad z_t \sim \mathrm{BBN}(0, 1.5). \tag{4.9}$$

Réponse.

```
> set.seed(951)
> ya <- arima.sim(n = 200, list(ma = c(-0.3, 0.6)), sd = sqrt(1.5))
> set.seed(7392)
> require("polynom")
> autopol <- polynomial(c(1, 0.8)) * polynomial(c(1, 0, 0, 0, -0.7))
> yd <- arima.sim(n = 200, list(ar = -autopol[-1],
+                               ma = c(0, 0.6)), sd = sqrt(1.5))
> yd <- yd + 4
```

```
> require("TSA")
> res <- armasubsets(y = yd, nar = 10, nma = 10, y.name = "yd",
+                    ar.method = "ols")
> plot(res)
```

Les zones les plus sombres du graphique (fig. 5) suggèrent les ordres à retenir mais l'interprétation n'est pas simple.

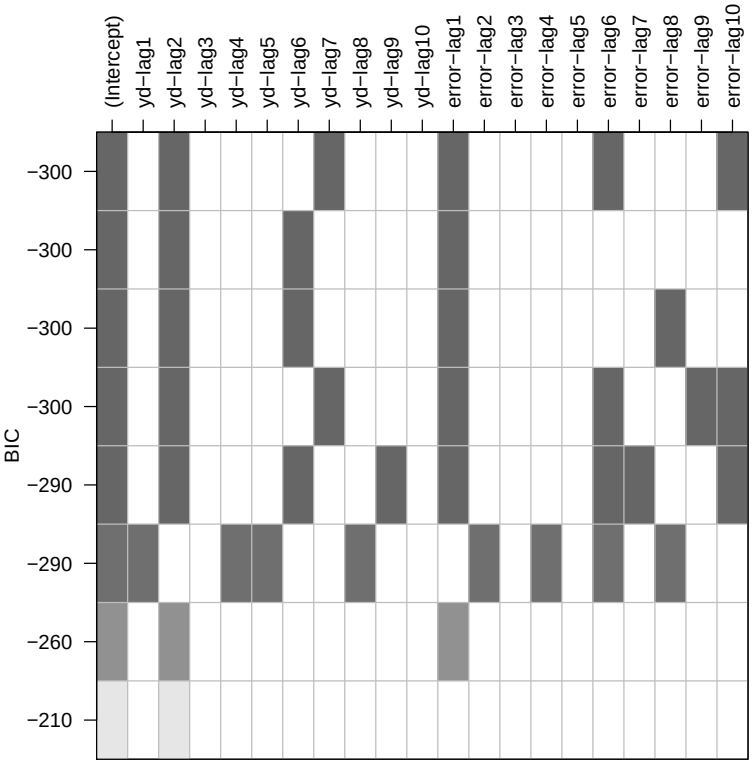

Fig. 5 – Identification de yd par armasubsets().