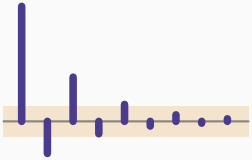
Modélisation ARMA des rentabilités financières

Chapitre 17 : Mise en œuvre sur R



Jaouad Madkour

Faculté d'Économie et de Gestion de Tanger · 27 août 2026

Plan du chapitre

Avant de commencer

Le jeu de données

Le fichier rentabilites.csv

2 000 rentabilités quotidiennes **simulées**, fournies avec le cours. Trois colonnes : `date`, `prix`, `rentabilite` (en pourcentage).

Pourquoi des données simulées

Parce qu'on sait ce qu'on y a mis. Le mécanisme générateur est connu : aucune mémoire en niveau, une forte persistance de la volatilité. On peut donc vérifier que la démarche **retrouve** ce qui s'y trouve — et ne trouve pas ce qui ne s'y trouve pas. Sur données réelles, on ne peut jamais faire cette vérification.

Les bibliothèques

```r
install.packages(c("forecast", "tseries", "FinTS"))
library(forecast)   # Acf, Pacf, auto.arima, forecast, checkresiduals
library(tseries)    # adf.test, kpss.test, jarque.bera.test
library(FinTS)      # ArchTest
```

Étape 1 — Lire et décrire

Charger la série

```r
don <- read.csv("rentabilites.csv", stringsAsFactors = FALSE)
don$date <- as.Date(don$date)
r <- don$rentabilite
T <- length(r)                        # 2000
```

```r
c(moyenne   = mean(r),
  ecart_type = sd(r),
  minimum   = min(r),
  maximum   = max(r))
```

```
   moyenne ecart_type    minimum    maximum
    0.0300     1.1764    -5.8412     4.9037
```

Les faits stylisés, en trois nombres

```r
library(moments)
skewness(r)     # -0.19
kurtosis(r)     #  6.60
jarque.bera.test(r)
```

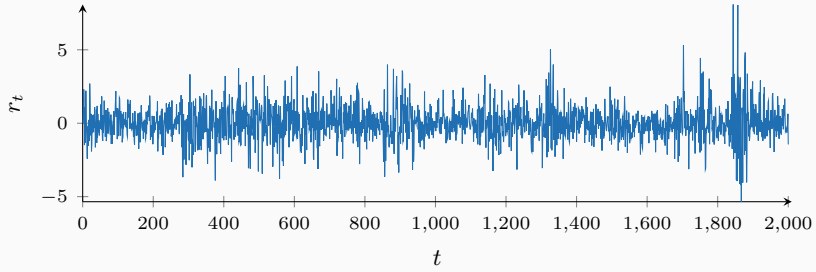
Ce que ces nombres disent

Un kurtosis de $6{,}6$ contre 3 pour une gaussienne : les valeurs extrêmes sont bien plus fréquentes que la loi normale ne l'autorise. Jarque-Bera rejette massivement.

Ce n'est **pas** un obstacle à la modélisation ARMA — le chapitre 14 l'a établi, l'estimateur reste convergent. C'est un obstacle aux **intervalles de prévision**.

Regarder la série

```r
plot(don$date, r, type = "l", col = "steelblue",
     xlab = "", ylab = "rentabilité (%)")
abline(h = 0, lty = 2, col = "grey40")
```



Ce qu'on voit avant tout calcul

Pas de tendance, pas de dérive : la moyenne semble stable. Mais l'amplitude, elle, ne l'est pas — des blocs calmes alternent avec des blocs agités. Retenez cette observation : le test formel viendra à la fin.

Étape 2 — Tester la stationnarité

```r
adf.test(r)                 # H0 : racine unitaire
pp.test(r)                  # idem, Phillips-Perron
kpss.test(r, null = "Level") # H0 : stationnarité
```

```
Augmented Dickey-Fuller Test
Dickey-Fuller = -12.517, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary

KPSS Test for Level Stationarity
KPSS Level = 0.1174, Truncation lag parameter = 8, p-value = 0.1
```

Croiser les conclusions

Lecture

ADF **rejette** la racine unitaire ($p < 0{,}01$). KPSS **ne rejette pas** la stationnarité ($p > 0{,}1$).

C'est la première ligne de la table de décision du chapitre 9 : conclusion **sans ambiguïté**, la série est $I(0)$.

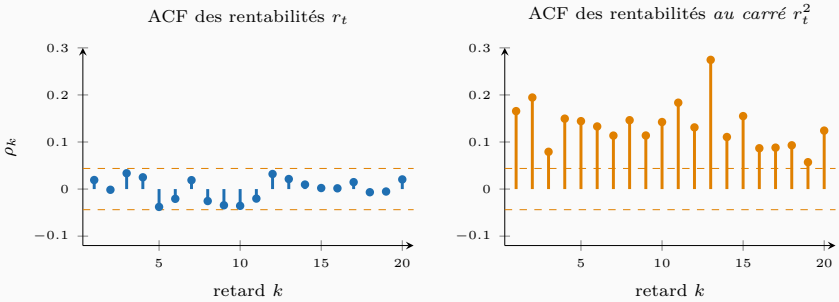
Un avertissement de R à ne pas mal lire

R affiche `p-value smaller than printed p-value`. Cela signifie que la p-valeur est **hors table**, donc encore plus petite que $0{,}01$ — pas qu'il y a un problème.

Étape 3 — Identifier

Les corrélogrammes

```
par(mfrow = c(1, 2))
Acf(r,  lag.max = 20, main = "ACF des rentabilités")
Pacf(r, lag.max = 20, main = "PACF des rentabilités")
```



Le diagnostic, et il est décevant

Les 20 autocorrélations sont **toutes** à l'intérieur des bandes $\pm 1{,}96/\sqrt{2000} = \pm 0{,}044$. La plus forte vaut $0{,}034$ au retard 3.

```r
Box.test(r, lag = 20, type = "Ljung-Box")
```

```
Box-Ljung test
X-squared = 20.351, df = 20, p-value = 0.4368
```

Conclusion

$Q_{\mathrm{LB}}(20) = 20{,}4$ contre une valeur critique de $31{,}4$. On ne rejette pas : **les rentabilités sont un bruit blanc au sens linéaire**.

C'est l'efficience faible, mesurée. Le cours pourrait s'arrêter là — et c'est précisément pour cela qu'il ne s'arrête pas là.

La sélection automatique

```
auto.arima(r, max.p = 4, max.q = 4, d = 0,
           seasonal = FALSE, ic = "bic", stepwise = FALSE)
```

```
Series: r
ARIMA(0,0,0) with non-zero mean

sigma^2 = 1.3841:  log likelihood = -3157.62
AIC=6319.24   AICc=6319.25   BIC=6330.44
```

Ce que le BIC retient

Le modèle vide, avec une simple constante. La procédure du chapitre 13 confirme la lecture des corrélogrammes.

Comparer les candidats à la main

```r
grille <- expand.grid(p = 0:3, q = 0:3)
res <- t(apply(grille, 1, function(o) {
  m <- try(arima(r, order = c(o[1], 0, o[2])), silent = TRUE)
  if (inherits(m, "try-error")) c(NA, NA)
  else c(AIC = AIC(m), BIC = BIC(m))
}))
cbind(grille, res)[order(res[, 2]), ][1:4, ]
```

```
  p q      AIC      BIC
1 0 0 6319.240 6330.442
2 1 0 6320.518 6337.321
3 0 1 6320.531 6337.334
5 1 1 6321.921 6344.325
```

La lecture

Le minimum du BIC est atteint en $(0, 0)$, et les écarts avec les voisins sont de l'ordre de 7 points — c'est net. Ajouter un paramètre coûte $\ln(2000) = 7{,}6$ et ne rapporte presque rien.

Étape 4 — Estimer et valider

Estimer, pour la démonstration

```r
mod <- arima(r, order = c(1, 0, 0))
mod
```

```
Coefficients:
         ar1  intercept
      0.0192     0.0300
s.e.  0.0224     0.0269
```

Comment lire ce tableau

$\hat{\alpha}_1 = 0{,}019$ avec un écart-type de $0{,}022$: le rapport de Student vaut $0{,}86$, très loin de $1{,}96$. Le coefficient n'est **pas significatif**.

Et même s'il l'avait été : un α de $0{,}02$ signifie que la rentabilité d'hier explique $0{,}04\,\%$ de celle d'aujourd'hui. Aucun coût de transaction ne le laisserait exploiter.

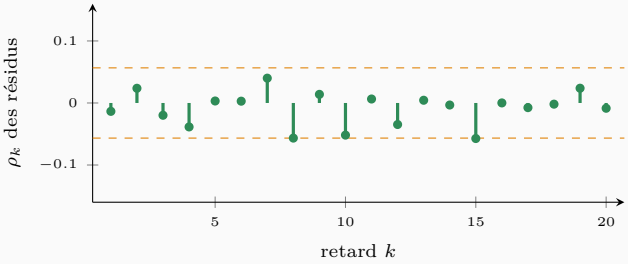
Valider

```
checkresiduals(mod, lag = 20)
```

```
Ljung-Box test
Q* = 20.06, df = 19, p-value = 0.3907
```

```
res <- residuals(mod)
jarque.bera.test(res)   # rejeté : queues épaisses
```

Valider (suite)



Valider (suite 2)

Verdict

Résidus non autocorrélés, mais non gaussiens. Le modèle est **valide au sens linéaire**, et ses intervalles de prévision sont suspects.

Étape 5 — Prévoir

Prévision ponctuelle et intervalle

```
p <- forecast(mod, h = 10, level = c(80, 95))
p
```

```
     Point Forecast     Lo 95    Hi 95
2001         0.0300   -2.2762   2.3362
2002         0.0300   -2.2766   2.3366
2005         0.0300   -2.2766   2.3366
```

Le chapitre 16, illustré en trois lignes

La prévision ponctuelle est **constante et égale à la moyenne** dès le premier pas — puisque $\alpha \simeq 0$, il n'y a rien à propager.

L'intervalle est lui aussi constant : $\pm 2{,}31$, soit $1{,}96 \times 1{,}176$. C'est la volatilité **moyenne** de tout l'échantillon, appliquée indifféremment à demain.

Étape 6 — Ce que le modèle a laissé passer

Regarder les résidus au carré

```r
Acf(res^2, lag.max = 20, main = "ACF des résidus au carré")
Box.test(res^2, lag = 20, type = "Ljung-Box")
```

```
Box-Ljung test
X-squared = 814.02, df = 20, p-value < 2.2e-16
```

Le contraste, en deux nombres

Série testée	$Q_{\mathrm{LB}}(20)$	p-valeur
résidus $\widehat{\varepsilon}_t$	20,1	0,39
résidus **au carré** $\widehat{\varepsilon}_t^2$	814,0	$< 10^{-16}$

```
ArchTest(res, lags = 12)
```

```
ARCH LM-test; Null hypothesis: no ARCH effects
Chi-squared = 274.53, df = 12, p-value < 2.2e-16
```

La conclusion de tout le cours, obtenue en une commande

Les rentabilités sont **non corrélées** — Ljung-Box ne rejette pas. Elles ne sont **pas indépendantes** — le test ARCH rejette avec une force écrasante.

Bruit blanc faible, pas bruit blanc fort. La distinction du chapitre 3 se lit ici dans deux p-valeurs.

Ce qu'il faut modéliser maintenant

La suite, en R

```r
library(rugarch)
spec <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
  mean.model     = list(armaOrder = c(0, 0), include.mean = TRUE),
  distribution.model = "std")     # Student : queues épaisses
fit <- ugarchfit(spec, data = r)
```

L'`armaOrder` est ce que ce cours a déterminé. Le `variance.model` est l'objet du cours suivant.

Synthèse

Ce qu'il faut retenir

L'essentiel du chapitre 17

- `adf.test` et `kpss.test` se lisent **ensemble** ; ici, série $I(0)$ sans ambiguïté.
- `Acf` / `Pacf` : rien hors des bandes. `Box.test` : $Q = 20{,}4$, non rejeté.
- `auto.arima` avec `ic = "bic"` retient ARMA(0,0).
- `forecast` donne une prévision constante et un intervalle constant — la volatilité moyenne.
- `Box.test` sur les **carrés** : $Q = 814$. `ArchTest` : rejet massif.
- Le diagnostic complet tient en six commandes, et il désigne les GARCH.

Vers le chapitre 18

La même démarche, la même série, les mêmes conclusions — en Python. L'intérêt n'est pas la répétition : c'est de constater que la méthode ne dépend pas de l'outil.