ANNEXE

D

Guide de prise en main de R et RStudio

Cette annexe est un guide pratique de R pour l'économétrie. Elle part de zéro : installer le logiciel, créer ses premiers objets, importer des données, puis estimer, tester et diagnostiquer un modèle, avant d'aborder les séries temporelles, les modèles à variable dépendante limitée, l'inférence causale et l'apprentissage automatique. Chaque bloc de code est court, exécuté avec R 4.3.3, et suivi de sa sortie réelle et d'un commentaire qui explique ce que le logiciel a calculé.

Objectifs d'apprentissage

À l'issue de cette annexe, le lecteur sera capable de :

- installer R et RStudio, puis charger les paquets nécessaires ;
- créer et manipuler les objets de base (vecteurs, facteurs, matrices, listes, data frames) ;
- importer, inspecter, transformer et résumer un jeu de données ;
- estimer un modèle par les moindres carrés ordinaires (MCO), lire sa sortie, contrôler ses hypothèses et corriger les écarts-types ;
- retrouver les fonctions R des modèles de panel, de variables instrumentales, de séries temporelles, de choix discret et d'apprentissage automatique ;
- éviter les erreurs fréquentes et rendre une analyse reproductible.

Comment lire les blocs de code. Les codes sont présentés dans un cadre violet, avec le texte en bleu. Le symbole # introduit un commentaire, ignoré par R. La sortie qui suit chaque code est celle que R 4.3.3 produit réellement, avec une largeur de ligne de 72 caractères et des nombres arrondis pour la lisibilité. Les blocs d'une même section s'enchaînent dans une seule session : un bloc réutilise les objets créés par le précédent, et celui qui exécute un bloc isolé doit d'abord recréer les objets manquants.

Presque tous les exemples reposent sur des jeux de données réels livrés avec R ou avec ses paquets (tableau D.1). Le lecteur peut donc tout reproduire sans connexion à Internet et sans fichier à télécharger. Le jeu `GrowthSW`, qui décrit la croissance de 65 pays dont le Zaïre (nom de la République Démocratique du Congo de 1971 à 1997), sert de fil conducteur. Sept blocs n'ont pas pu être exécutés pour la rédaction de ce livre : quatre lisent un fichier de

l'annexe C du livre, trois autres exigent un paquet que le poste de l'auteur ne possédait pas. Chacun est précédé d'une note qui le signale ; leur syntaxe a été contrôlée, mais leurs résultats ne sont pas reproduits.

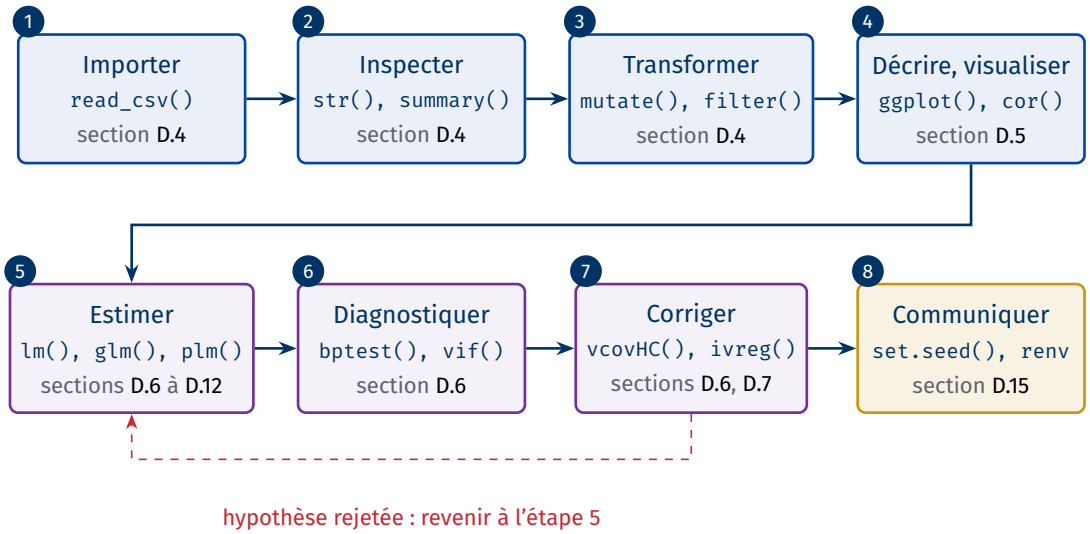


FIGURE D.1. Flux de travail d'une analyse économétrique sous R. Les quatre premières étapes (bleu) préparent les données, les trois suivantes (violet) estiment et contrôlent le modèle, la dernière (or) rend l'analyse reproductible. Sous chaque étape figurent les fonctions les plus utiles et la section de l'annexe qui la détaille. Le trait pointillé rappelle qu'un diagnostic défavorable oblige à revenir à l'estimation.

TABLE D.1. Jeux de données utilisés dans cette annexe

Jeu	Paquet	Contenu	Obs.	Sections
`GrowthSW`	`AER`	croissance moyenne du PIB de 65 pays, de 1960 à 1995, et déterminants	65	D.4 à D.6
`Grunfeld`	`plm`	investissement de 10 grandes entreprises américaines, de 1935 à 1954	200	D.4, D.7
`CigarettesSW`	`AER`	consommation de cigarettes par État américain (année 1995 retenue)	48	D.7
`USMacroG`	`AER`	macroéconomie américaine, trimestrielle, de 1950 à 2000	204	D.6
`AirPassengers`	`datasets`	passagers aériens mensuels, de 1949 à 1960	144	D.4

Jeu	Paquet	Contenu	Obs.	Sections
`EuStockMarkets`	`datasets`	indices boursiers européens quotidiens, de 1991 à 1998	1 860	D.8
`Nile`	`datasets`	débit annuel du Nil à Assouan, de 1871 à 1970	100	D.8
`MarkPound`	`fGarch`	rendements quotidiens du mark allemand en livres sterling	1 974	D.8
`denmark`	`urca`	demande de monnaie au Danemark, trimestrielle, de 1974 à 1987	55	D.8
`SwissLabor`	`AER`	participation de femmes suisses au marché du travail	872	D.10
`PSID1976`	`AER`	offre de travail de femmes mariées américaines (1975)	753	D.10
`NMES1988`	`AER`	visites chez le médecin de personnes âgées américaines	4 406	D.10
`engel`	`quantreg`	revenu et dépense alimentaire de ménages ouvriers belges	235	D.11
`CreditCard`	`AER`	demandes de carte de crédit	1 319	D.12
`CASchools`	`AER`	résultats scolaires de districts californiens	420	D.12
`nc.shp`	`sf`	comtés de Caroline du Nord (carte et morts subites du nourrisson)	100	D.13

Source : documentation des jeux de données de R. Les jeux `GrowthSW`, `CigarettesSW` et `CASchools` proviennent de Stock et Watson (2007) ; `USMacroG`, `CreditCard` et `PSID1976` de Greene (2003) ; `SwissLabor` de Gerfin (1996) ; `NMES1988` de Deb et Trivedi (1997) ; `engel` de Koenker et Bassett (1982) ; `MarkPound` de Bollerslev et Ghysels (1996) ; `denmark` de Johansen et Juselius (1990).

D.1 Pourquoi R pour l'économétrie africaine ?

R est un langage et un environnement libres pour le calcul statistique et les graphiques. Conçu au début des années 1990 par Ross Ihaka et Robert Gentleman à l'Université d'Auckland, il est diffusé sous licence libre GNU GPL depuis 1995 (Ihaka & Gentleman, 1996 ; R Core Team, 2024). Sa gratuité en fait un choix naturel pour les institutions africaines, où le coût des licences de logiciels propriétaires comme Stata ou SAS représente une contrainte budgétaire réelle.

Ce qui distingue R des autres environnements présentés dans ce manuel, c'est la richesse de son écosystème. Le dépôt CRAN (*Comprehensive R Archive Network*) héberge plus de 20 000 paquets, qui couvrent l'économétrie des panels (`plm`), l'inférence causale (`rdrobust`, `DoubleML`), l'apprentissage automatique (`xgboost`, `grf`), les réseaux de neurones (`keras`) et la cartographie (`sf`, `terra`). Aucun autre environnement libre ne réunit une telle profondeur dans les domaines abordés dans ce manuel. Pour une présentation plus longue, le lecteur peut consulter Chambers (2008) pour le langage lui-même, Kleiber et Zeileis (2008) et Hanck et al. (2018) pour l'économétrie, et Wickham et al. (2023) pour la manipulation de données.

Dans le contexte de la République Démocratique du Congo et de l'Afrique francophone, R offre des avantages pratiques. Les paquets `readr` et `haven` lisent les fichiers d'enquêtes (INS, MICS, DHS) quel que soit leur format. Le paquet `sf` lit les fichiers cartographiques des divisions administratives, comme ceux que publie le Bureau de la coordination des affaires humanitaires des Nations unies (OCHA), et permet de produire des cartes thématiques pour des rapports LaTeX ou R Markdown. Enfin, la reproductibilité de l'analyse, traitée à l'annexe F, découle de la conservation des scripts et d'outils comme `renv`.

Les parties I et II du manuel proposent surtout des mises en œuvre avec Gretl et JMulTi, et recourent à R pour les modèles à variable dépendante limitée (chapitre 8), les méthodes non paramétriques et bayésiennes (chapitre 9), l'inférence causale (chapitres 11 à 17 et 21) et l'économétrie spatiale (chapitre 18). Les chapitres d'apprentissage automatique (19, 20 et 22) reposent entièrement sur R ou Python.

D.2 Installation de R et RStudio

D.2.1 Installation de R

La procédure dépend du système d'exploitation. Sous Windows, on télécharge l'installeur depuis https://cran.r-project.org, rubrique `Download R for Windows > base` ; l'installation par défaut convient dans la grande majorité des cas. Sous macOS, on choisit la version qui correspond au processeur de la machine (Intel ou Apple Silicon). Sous Ubuntu ou Debian, deux commandes tapées dans un terminal suffisent :

```
sudo apt-get update
sudo apt-get install r-base r-base-dev
```

Les codes de cette annexe ont été exécutés avec R 4.3.3 (février 2024). Une version 4.2 ou plus récente convient. Le tube natif `|>`, utilisé plus loin pour enchaîner des opérations, exige R 4.1.0 ou plus.

D.2.2 Installation de RStudio

RStudio est un environnement de développement intégré (IDE) édité par la société Posit, anciennement RStudio PBC. Il ne remplace pas R : il s'y superpose et offre une interface organisée pour écrire des scripts, voir les résultats et gérer les graphiques. On le télécharge gratuitement sur https://posit.co/downloads/, version `RStudio Desktop`. Il faut installer R d'abord, puis RStudio.

Pour vérifier l'installation, on ouvre RStudio : la console affiche la version de R et un chevron > qui indique que le moteur attend une commande. Le code suivant interroge la version de R et celle d'un paquet :

Code D.1. Vérifier les versions installées

```
R.version.string                                  # version de R installée
packageVersion("AER")                             # version d'un paquet donné
```

```
[1] "R version 4.3.3 (2024-02-29)"
[1] '1.2.12'
```

La première ligne confirme la version de R ; la seconde la version du paquet `AER` (*Applied Econometrics with R*), qui fournit la plupart des jeux de données de l'annexe. Si `packageVersion()` signale que le paquet est absent, il faut l'installer comme indiqué ci-dessous.

D.2.3 Installation des paquets

Un *paquet* (*package*) est une extension de R qui regroupe des fonctions, de la documentation et souvent des données. On l'installe une seule fois par machine avec `install.packages("nom_du_paquet")`, ou par le menu `Tools > Install Packages` de RStudio. En revanche, chaque session de travail doit le charger avec `library(nom_du_paquet)`. Le bloc suivant installe d'un coup les vingt paquets utilisés par les codes exécutés de cette annexe, mais seulement ceux qui manquent :

Code D.2. Installer les paquets absents

```
# Paquets utilisés dans cette annexe : données, estimation, séries, apprentissage
paquets <- c("AER", "plm", "lmtest", "sandwich", "car", "dplyr", "readr", "haven",
             "ggplot2", "tseries", "forecast", "urca", "fGarch", "quantreg",
             "KernSmooth", "caret", "randomForest", "pROC", "glmnet", "sf")
manquants <- setdiff(paquets, rownames(installed.packages()))
if (length(manquants) > 0) install.packages(manquants)   # n'installe que l'absent
length(manquants)                                 # 0 si tout est déjà installé
```

```
[1] 0
```

La commande `setdiff()` compare la liste voulue à la liste des paquets déjà installés ; `install.packages()` n'est appelée que si la différence n'est pas vide. Le nombre affiché est celui des paquets qui manquaient avant l'installation : sur le poste de l'auteur, tous étaient présents (0). Les blocs non exécutés de cette annexe demandent d'autres paquets, à installer de la même façon :

Code D.3. Paquets des blocs non exécutés

```
install.packages(c("rdrobust", "rddensity", "did",       # inférence causale
                   "DoubleML", "mlr3", "mlr3learners",   # double machine learning
                   "ranger", "data.table",
                   "xgboost", "keras",                   # apprentissage automatique
                   "spdep", "spatialreg"))               # économétrie spatiale
```

Le paquet `keras` exige en plus un moteur TensorFlow, dont l'installation est décrite dans la documentation du paquet.

D.3 L'interface RStudio

L'interface de RStudio comporte quatre panneaux dont la disposition peut être modifiée par `Tools > Global Options > Pane Layout`. La figure D.2 montre la configuration par défaut, avec les résultats de l'exemple de la section D.4.

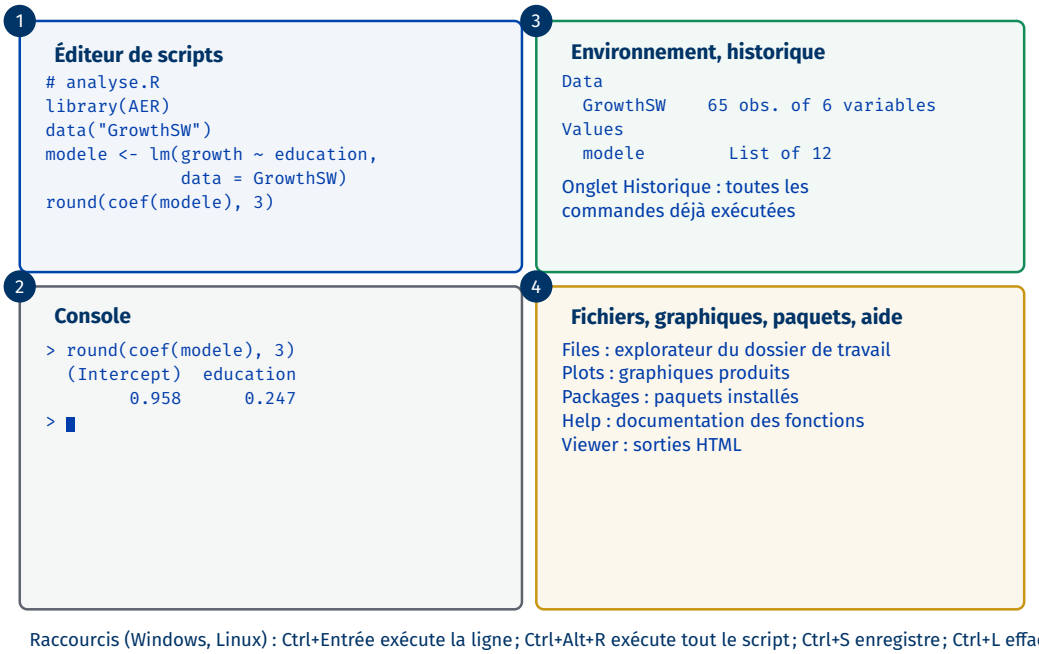


Figure D.2. Les quatre panneaux de l'interface RStudio dans sa configuration par défaut. L'éditeur (1) conserve le script, la console (2) affiche les résultats, l'environnement (3) liste les objets en mémoire et la quatrième zone (4) regroupe fichiers, graphiques, paquets et aide. Les contenus montrés sont ceux de l'exemple de la section D.4.

L'*éditeur* (1) sert à écrire et à enregistrer des scripts `.R`. On exécute la ligne ou la sélection courante avec `Ctrl+Entrée` (`Cmd+Entrée` sous macOS) et le script entier avec `Ctrl+Alt+R`. La *console* (2) affiche les résultats et permet un dialogue direct avec R ; `Ctrl+L` l'efface. Le

panneau *environnement* (3) liste les objets en mémoire (vecteurs, data frames, modèles), ce qui facilite la détection des erreurs, et son onglet *historique* conserve les commandes déjà exécutées. La quatrième zone (4) regroupe l'explorateur de fichiers, les graphiques, la liste des paquets et l'aide.

Travailler en projets RStudio

Il est fortement recommandé de travailler par *projets* (`File > New Project`) : le répertoire de travail est alors fixé automatiquement au dossier du projet, ce qui évite les chemins de fichiers absolus et facilite la reproductibilité. Sans projet, on fixe le répertoire au début du script avec `setwd("/Users/monnom/projets/rdc_eco")`. On vérifie le répertoire courant avec `getwd()` et son contenu avec `list.files()`.

D.4 Gestion des données sous R

D.4.1 Les objets de R

Tout ce que R manipule est un *objet* : un nombre, un vecteur, un tableau de données, un résultat d'estimation, ou même une fonction. On crée un objet avec l'opérateur d'affectation `<-`, puis on le désigne par son nom. Cinq familles d'objets suffisent pour commencer ; la figure D.3 montre comment elles se déduisent les unes des autres.

Code D.4. Les objets de base

```
x <- c(2, 4, 6, 8)                                  # vecteur numérique
x * 2                                               # opération terme à terme
statut <- factor(c("urbain", "rural", "urbain"))    # facteur (qualitative)
levels(statut)
M <- matrix(1:6, nrow = 2)                          # matrice 2 lignes, 3 colonnes
t(M) %*% M                                          # produit matriciel M'M
resultat <- list(methode = "MCO", coef = c(1.5, 0.8))  # liste hétérogène
resultat$coef
carre <- function(z) z^2                        # fonction définie par l'utilisateur
carre(x)
objets <- list(x = x, statut = statut, M = M, resultat = resultat, carre = carre)
sapply(objets, function(o) class(o)[1])             # classe de chaque objet
```

```
[1]  4  8 12 16
[1] "rural"  "urbain"
     [,1] [,2] [,3]
[1,]    5   11   17
[2,]   11   25   39
[3,]   17   39   61
[1] 1.5 0.8
[1]  4 16 36 64
         x     statut          M   resultat      carre
 "numeric"   "factor"   "matrix"     "list" "function"
```

Le vecteur x est multiplié terme à terme par 2. Le facteur `statut` range une variable qualitative en niveaux, classés par ordre alphabétique (`rural` avant `urbain`). L'opérateur `%*%` calcule le produit matriciel $\mathbf{M}'\mathbf{M}$, matrice carrée de dimension 3×3 et symétrique, qui est la brique de base de l'estimateur des MCO $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$. La liste `resultat` réunit un texte et un vecteur, que le signe $ permet d'extraire. La dernière ligne montre que chaque objet possède une *classe* : c'est elle qui détermine ce que font les fonctions génériques comme `summary()` ou `plot()`.

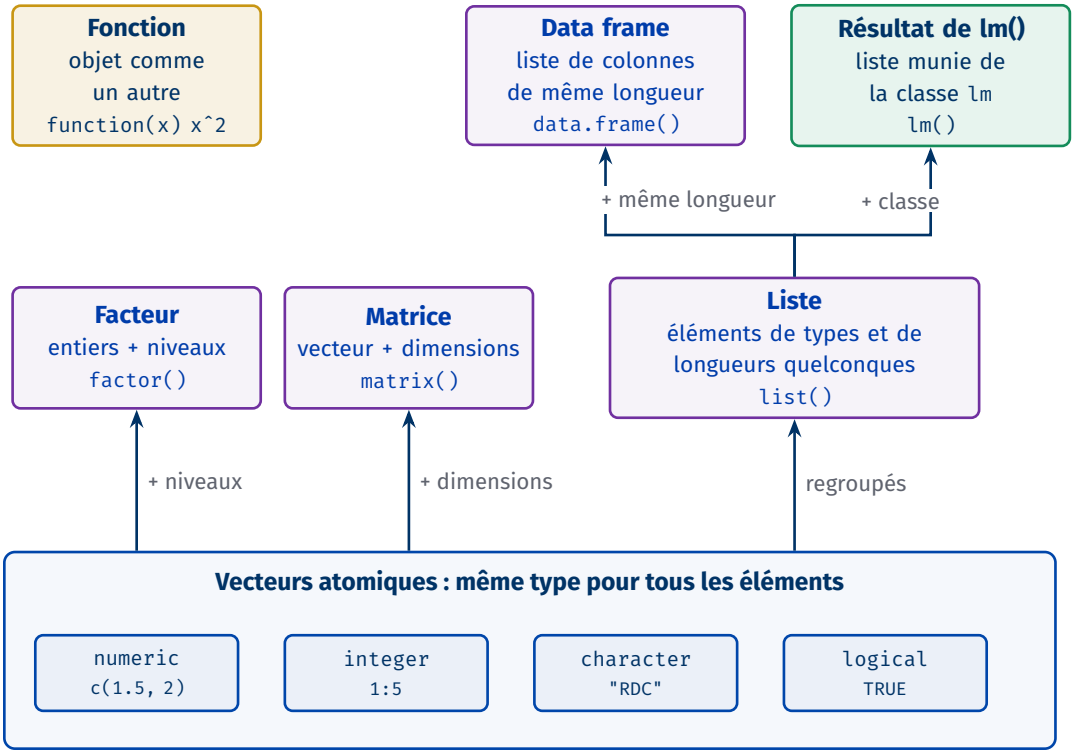


FIGURE D.3. Les objets de R et leurs liens de parenté. Tout part des vecteurs atomiques. Un facteur ajoute des niveaux, une matrice des dimensions, une liste regroupe des objets quelconques. Un data frame est une liste de colonnes de même longueur et le résultat d'une estimation par `lm()` est une liste munie de la classe `lm`. Les fonctions sont des objets comme les autres. La fonction `class()` indique le type d'un objet.

D.4.2 Le data frame, objet central de l'économétrie

Définition D.1 (Data frame)

Un *data frame* est un tableau rectangulaire dans lequel chaque colonne est une variable (un vecteur d'un seul type) et chaque ligne une observation. C'est la forme sous laquelle R lit presque toutes les données et la forme que les fonctions d'estimation attendent.

Le bloc suivant charge le jeu `GrowthSW` (Stock & Watson, 2007), qui décrit 65 pays : la croissance annuelle moyenne du PIB réel de 1960 à 1995 (`growth`, en %), le PIB par habitant de 1960 en dollars de 1960 (`rgdp60`), la part du commerce extérieur dans le PIB (`tradeshare`), le nombre moyen d'années de scolarité en 1960 (`education`), le nombre annuel moyen

de révolutions et de coups d'État (`revolutions`) et d'assassinats politiques par million d'habitants (`assassinations`).

Code D.5. Charger et inspecter un data frame

```
library(AER)                        # charge aussi lmtest, sandwich, car, survival
data("GrowthSW")                    # croissance de 65 pays, 1960-1995
croissance <- cbind(pays = rownames(GrowthSW), GrowthSW)  # noms de pays en colonne
rownames(croissance) <- NULL
dim(croissance)                     # (observations, variables)
str(croissance)                     # type de chaque colonne
croissance[croissance$pays == "Zaire", ]   # une ligne : [lignes, colonnes]
mean(croissance$growth)             # une colonne avec $
```

```
[1] 65  7
'data.frame': 65 obs. of  7 variables:
 $ pays          : chr  "India" "Argentina" "Japan" "Brazil" ...
 $ growth        : num  1.915 0.618 4.305 2.93 1.712 ...
 $ rgdp60        : num  766 4462 2954 1784 9895 ...
 $ tradeshare    : num  0.141 0.157 0.158 0.16 0.161 ...
 $ education     : num  1.45 4.99 6.71 2.89 8.66 ...
 $ revolutions   : num  0.133 0.933 0 0.1 0 ...
 $ assassinations: num  0.867 1.933 0.2 0.1 0.433 ...
    pays    growth   rgdp60 tradeshare education revolutions
15 Zaire -2.811944 488.9999  0.3523176      0.54   0.1481481
   assassinations
15     0.05555556
[1] 1.942715
```

Dans le jeu d'origine, les noms de pays sont les noms des lignes ; `cbind()` les place dans une vraie colonne `pays`, plus commode pour filtrer et fusionner. La fonction `dim()` confirme 65 lignes et 7 colonnes, et `str()` donne le type de chaque colonne. L'expression `croissance[ligne, colonne]` sélectionne une partie du tableau (une condition à gauche de la virgule choisit des lignes), tandis que `croissance$growth` extrait une colonne entière. Le Zaïre a vu son PIB par habitant reculer de 2,81 % par an en moyenne de 1960 à 1995, avec 0,54 année de scolarité moyenne en 1960, alors que la croissance moyenne de l'échantillon est de 1,94 % par an (figure D.4).

D.4.3 Importation de données

R lit pratiquement tous les formats usuels. Le tableau D.2 donne les fonctions les plus utiles en contexte africain, où les fichiers d'enquête (DHS, MICS, FinScope) arrivent souvent au format Stata.

Pour que l'exemple soit reproductible sans fichier externe, le bloc suivant écrit d'abord le data frame `croissance` dans un dossier temporaire, sous trois formes, puis le relit avec la fonction adaptée à chaque format.

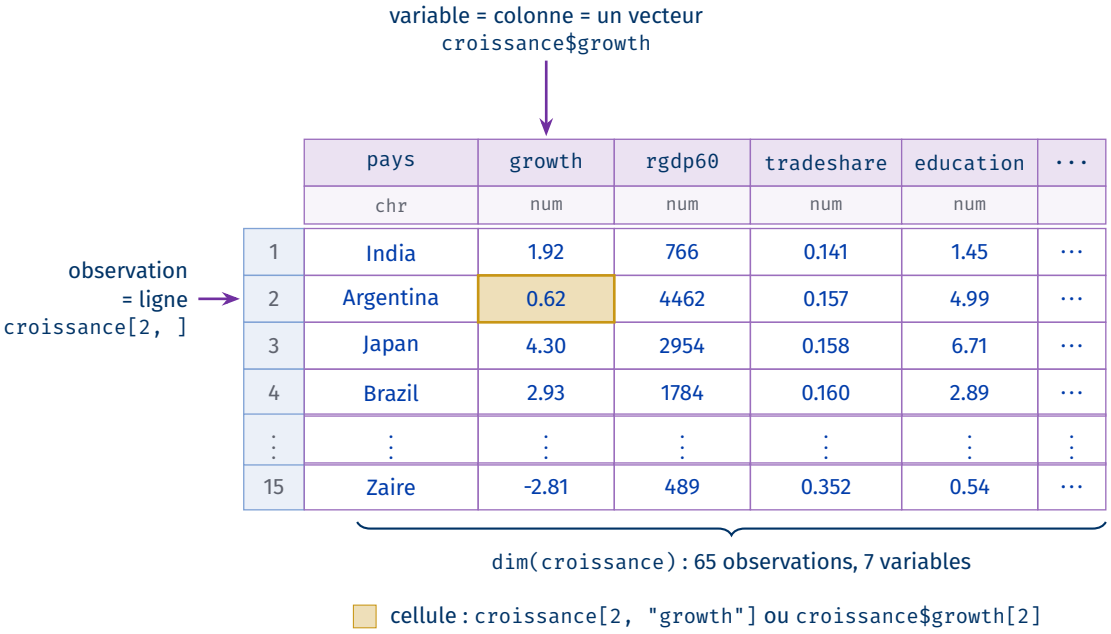


FIGURE D.4. Anatomie d'un data frame : extrait de l'objet `croissance` (65 pays, jeu `GrowthSW` du paquet `AER`, valeurs réelles arrondies). Chaque colonne est un vecteur dont le type figure sous le nom de la variable (`chr` pour du texte, `num` pour des nombres) ; chaque ligne est une observation, ici un pays, et la colonne de gauche donne son numéro. Le Zaïre, nom porté par la République Démocratique du Congo de 1971 à 1997, a la croissance moyenne la plus faible de l'échantillon. Le signe `$` extrait une colonne, les crochets `[ligne, colonne]` une ligne ou une cellule.

TABLE D.2. Fonctions d'importation les plus courantes

Format du fichier	Fonction	Paquet
CSV (séparateur virgule, décimale point)	`read_csv()`	`readr`
CSV d'un tableur francophone (point-virgule, décimale virgule)	`read_csv2()`	`readr`
Texte séparé par des tabulations	`read_tsv()`	`readr`
Stata (`.dta`)	`read_dta()`	`haven`
SPSS (`.sav`) et SAS (`.sas7bdat`)	`read_sav()`, `read_sas()`	`haven`
Excel (`.xlsx`)	`read_excel()`	`readxl`

Code D.6. Écrire puis relire des fichiers CSV et Stata

```
library(readr)                  # lecture rapide de fichiers texte
library(haven)                  # formats Stata, SPSS et SAS
dossier <- tempdir()            # dossier temporaire pour l'exemple
# 1. CSV standard (séparateur ",", décimale ".")
write_csv(croissance, file.path(dossier, "croissance.csv"))
a <- read_csv(file.path(dossier, "croissance.csv"), show_col_types = FALSE)
# 2. CSV exporté par un tableur francophone (séparateur ";", décimale ",")
writeLines(c("pays;growth", "India;1,92", "Zaire;-2,81"), file.path(dossier, "fr.csv"))
b <- read_csv2(file.path(dossier, "fr.csv"), show_col_types = FALSE)
# 3. Fichier Stata (.dta)
write_dta(croissance, file.path(dossier, "croissance.dta"))
d <- read_dta(file.path(dossier, "croissance.dta"))
dim(a); dim(d)                  # 65 observations, 7 variables dans les deux cas
b
```

```
[1] 65  7
[1] 65  7
# A tibble: 2 x 2
  pays  growth
  <chr>  <dbl>
1 India   1.92
2 Zaire  -2.81
```

Les deux relectures (CSV standard et Stata) redonnent bien 65 lignes et 7 colonnes. Le troisième fichier imite un export d'un tableur en version française : le séparateur est le point-virgule et la décimale est la virgule. La fonction `read_csv()` l'aurait lu comme une seule colonne de texte ; `read_csv2()` sépare correctement les deux variables et convertit `1,92` en nombre (1.92).

Mise en garde D.1 (Étiquettes des fichiers Stata)

Les fichiers `.dta` d'enquête portent des étiquettes de variables et de valeurs. `read_dta()` les conserve dans des colonnes de classe `haven_labelled`. Avant d'estimer un modèle, on convertit les variables qualitatives avec `haven::as_factor()`, faute de quoi les codes numériques (par exemple 1 pour « urbain » et 2 pour « rural ») sont traités comme des quantités.

D.4.4 Manipulation des données avec `dplyr`

Le paquet `dplyr`, qui fait partie du *tidyverse* (Wickham et al., 2023), offre une grammaire cohérente pour transformer un data frame. Cinq verbes couvrent l'essentiel des besoins : `filter()` choisit des lignes, `select()` des colonnes, `mutate()` crée des variables, `group_by()` avec `summarise()` calcule des statistiques par groupe, et `arrange()` trie. On les enchaîne avec le tube `|>`, qui transmet le résultat de chaque étape à la suivante.

Code D.7. Transformer, résumer et fusionner avec dplyr

```
library(dplyr)
pays_afrique <- c("Zaire", "Niger", "Senegal", "Zimbabwe", "Togo", "South Africa",
                  "Sierra Leone", "Ghana", "Kenya", "Mauritius")
croissance <- croissance |>
  mutate(afrique = pays %in% pays_afrique,           # indicatrice Afrique
         log_rgdp60 = log(rgdp60))                   # niveau initial en logarithme
resume <- croissance |>
  group_by(afrique) |>                               # statistiques par groupe
  summarise(n = n(), croissance_moy = mean(growth),
            scolarite_moy = mean(education))
resume
croissance |>                                        # jointure : écart au groupe
  left_join(select(resume, afrique, croissance_moy), by = "afrique") |>
  mutate(ecart = growth - croissance_moy) |>
  arrange(ecart) |>
  select(pays, growth, ecart) |>
  head(3)
```

```
# A tibble: 2 x 4
  afrique     n croissance_moy scolarite_moy
  <lgl>   <int>          <dbl>         <dbl>
1 FALSE      55         2.31           4.47
2 TRUE       10        -0.0625         1.31
         pays     growth     ecart
1   Venezuela -0.8835508 -3.190844
2       Haiti -0.6579341 -2.965227
3 El Salvador -0.6075561 -2.914849
```

On crée d'abord l'indicatrice `afrique`, vraie pour les dix pays africains de l'échantillon (Zaïre, Niger, Sénégal, Zimbabwe, Togo, Afrique du Sud, Sierra Leone, Ghana, Kenya et Maurice), et le logarithme du PIB initial. Le résumé par groupe montre que les 55 autres pays ont crû de 2,31 % par an en moyenne, contre −0,06 % pour les dix pays africains, qui comptaient aussi moins d'années de scolarité (1,31 en moyenne contre 4,47). La jointure `left_join()` rattache à chaque pays la moyenne de son groupe, ce qui permet de calculer l'écart de chaque pays à son groupe, puis de trier : les trois pays les plus en retrait sont le Venezuela (−3,19 points), Haïti (−2,97) et le Salvador (−2,91).

D.4.5 Déclarer la structure d'une série temporelle ou d'un panel

Avant d'estimer un modèle de séries temporelles ou de panel, il faut déclarer la structure des données. C'est l'équivalent de la commande `setobs` de Gretl : on convertit les données en objet `ts` pour une série, en objet `pdata.frame` pour un panel.

Code D.8. Déclarer une série temporelle et un panel

```
library(plm)
window(AirPassengers, start = c(1949, 1), end = c(1949, 6))   # série mensuelle
c(frequency(AirPassengers), start(AirPassengers))             # fréquence et début
data("Grunfeld", package = "plm")                  # 10 entreprises, 1935-1954
panel <- pdata.frame(Grunfeld, index = c("firm", "year"))     # identifiant x temps
pdim(panel)                                        # panel cylindré ?
```

```
     Jan Feb Mar Apr May Jun
1949 112 118 132 129 121 135
[1]   12 1949    1
Balanced Panel: n = 10, T = 20, N = 200
```

`AirPassengers` est déjà un objet `ts` mensuel : `window()` en extrait le premier semestre de 1949, et la fonction `frequency()` renvoie 12 (observations par an), `start()` la date de début (1949, période 1). Pour le panel, `pdata.frame()` désigne l'identifiant de l'individu (`firm`) puis celui du temps (`year`), et `pdim()` confirme un panel cylindré : 10 entreprises observées sur 20 ans, soit 200 observations.

Le même principe s'applique à un fichier de l'annexe C du livre. Le bloc suivant construit la série mensuelle du taux de change CDF/USD de la Banque centrale du Congo (fichier BCC-01, de 2005 à 2023). Il n'a pas été exécuté, car le fichier n'est pas livré avec R ; la logique de conversion de la date a été vérifiée sur un fichier de même structure.

Code D.9. Série mensuelle du taux de change (non exécuté)

```
# Fichier de l'annexe E (BCC-01) : taux de change mensuel CDF/USD, 2005-2023
taux <- read.csv("bcc_taux_change_cdf_usd_2005_2023.csv")
taux$date <- as.Date(paste0(taux$date, "-01"))     # "2005-01" devient 2005-01-01
taux_ts <- ts(taux$cdf_usd, start = c(2005, 1), frequency = 12)   # série mensuelle
plot(taux_ts, ylab = "CDF pour 1 USD")
```

D.5 Statistiques descriptives, tests et graphiques

Toute analyse commence par regarder les données. Le bloc suivant résume les quatre variables principales du jeu `GrowthSW`, calcule leurs écarts-types et la matrice des corrélations, puis compte les valeurs manquantes.

Code D.10. Statistiques descriptives

```
vars <- c("growth", "rgdp60", "tradeshare", "education")
summary(croissance[, vars])                        # résumé de chaque variable
round(sapply(croissance[, vars], sd), 2)           # écarts types
round(cor(croissance[, c("growth", "log_rgdp60", "tradeshare", "education")]), 2)
sum(is.na(croissance))                             # nombre de valeurs manquantes
```

```
     growth            rgdp60       tradeshare       education
 Min.   :-2.8119   Min.   : 367   Min.   :0.1405   Min.   : 0.200
 1st Qu.: 0.8382   1st Qu.:1148   1st Qu.:0.3933   1st Qu.: 1.940
 Median : 1.9751   Median :2019   Median :0.5433   Median : 3.650
 Mean   : 1.9427   Mean   :3104   Mean   :0.5647   Mean   : 3.985
 3rd Qu.: 2.8803   3rd Qu.:5143   3rd Qu.:0.6816   3rd Qu.: 5.560
 Max.   : 7.1569   Max.   :9895   Max.   :1.9926   Max.   :10.070
    growth     rgdp60 tradeshare  education
      1.90    2512.66       0.29       2.54
           growth log_rgdp60 tradeshare education
growth       1.00       0.17       0.35      0.33
log_rgdp60   0.17       1.00      -0.10      0.84
tradeshare   0.35      -0.10       1.00     -0.01
education    0.33       0.84      -0.01      1.00
[1] 0
```

La croissance moyenne est de 1,94 % par an, pour une médiane de 1,98 % : la distribution est à peu près symétrique, avec un écart-type de 1,90 point. Le PIB par habitant initial est au contraire très asymétrique (moyenne de 3 104 dollars, médiane de 2 019), ce qui justifie de le prendre en logarithme dans les régressions. Les corrélations de la croissance avec la scolarité (0,33) et avec l'ouverture commerciale (0,35) sont positives et modérées. La corrélation de 0,84 entre la scolarité et le logarithme du PIB initial annonce une colinéarité à surveiller dans la régression. Le jeu ne contient aucune valeur manquante.

Le graphique le plus utile en économétrie est le nuage de points, accompagné de la droite des MCO. Avec `ggplot2`, on superpose des couches : les points, la droite, puis les étiquettes. Les graphiques de base de R donnent un résultat équivalent en deux lignes.

Code D.11. Nuage de points avec ggplot2

```
library(ggplot2)
ggplot(croissance, aes(x = education, y = growth)) +
  geom_point(aes(colour = afrique)) +                    # nuage de points
  geom_smooth(method = "lm", se = TRUE) +                # droite MCO et IC
  labs(x = "Années de scolarité en 1960",
       y = "Croissance annuelle moyenne (%)", colour = "Pays africain") +
  theme_minimal()
ggsave(file.path(tempdir(), "croissance_education.pdf"), width = 6, height = 4)
# Équivalent avec les graphiques de base
plot(growth ~ education, data = croissance)
abline(lm(growth ~ education, data = croissance))
```

Ce bloc ne produit pas de sortie texte : il dessine le graphique à l'écran, et `ggsave()` l'enregistre en PDF avec la taille voulue. La figure D.5 reproduit le même graphique à partir des mêmes données. La relation entre scolarité initiale et croissance est positive mais lâche : la droite a pour pente 0,247 (une année de scolarité de plus est associée à 0,25 point de croissance annuelle en plus) et la scolarité n'explique que 11 % de la variance de la croissance. Les pays africains se concentrent en bas à gauche, et le Zaïre est le point le plus bas.

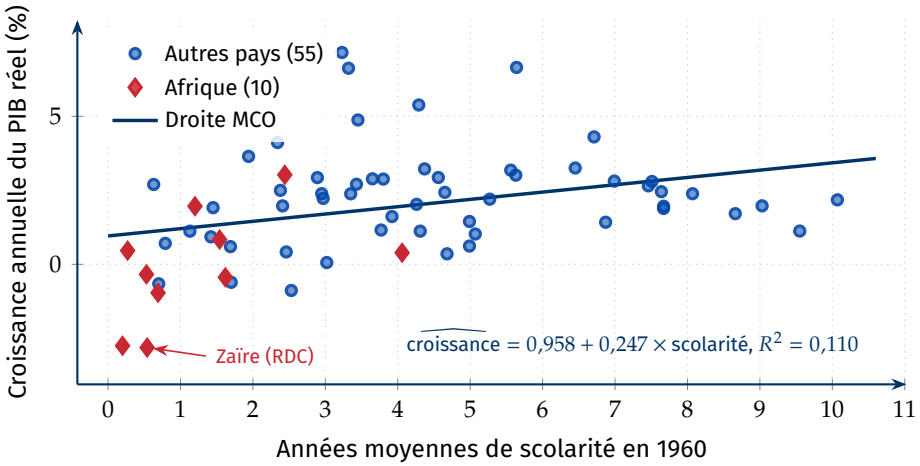


FIGURE D.5. Croissance économique et scolarité initiale dans 65 pays (`GrowthSW`). Chaque point est un pays ; les 10 pays africains de l'échantillon sont en rouge, dont le Zaïre, ancien nom de la République démocratique du Congo. La droite est celle de la régression simple estimée par les MCO ; la scolarité n'explique qu'environ 11 % de la variance de la croissance.

Pour tester des hypothèses simples, R offre `t.test()`, `wilcox.test()`, `shapiro.test()` et `cor.test()`. Le bloc suivant compare la croissance des pays africains et des autres, puis teste la normalité de la croissance et la corrélation avec la scolarité.

Code D.12. Tests d'hypothèses usuels

```
t_afr <- t.test(growth ~ afrique, data = croissance)     # test t de Welch, deux groupes
round(unlist(t_afr[c("estimate", "statistic", "parameter", "p.value")]), 4)
round(t_afr$conf.int[1:2], 3)                            # IC à 95 % de la différence
round(c(wilcoxon    = wilcox.test(growth ~ afrique, data = croissance)$p.value,
        shapiro     = shapiro.test(croissance$growth)$p.value,
        correlation = cor.test(~ growth + education, data = croissance)$p.value), 4)
```

```
estimate.mean in group FALSE  estimate.mean in group TRUE
                      2.3073                       -0.0625
                statistic.t                     parameter.df
                      3.7812                       11.8563
                     p.value
                      0.0027
[1] 1.002 3.737
   wilcoxon     shapiro correlation
     0.0009      0.0430      0.0071
```

Le test de Welch compare les deux moyennes sans supposer l'égalité des variances : la croissance moyenne est de 2,31 % hors d'Afrique contre −0,06 % en Afrique, soit un écart de 2,37 points, avec une statistique $t = 3{,}78$ (11,9 degrés de liberté) et une probabilité critique de 0,003. L'intervalle de confiance à 95 % de l'écart va de 1,00 à 3,74 point ; il exclut zéro, comme le test de Wilcoxon fondé sur les rangs ($p = 0{,}0009$). Le test de Shapiro-Wilk rejette de justesse la normalité de la croissance ($p = 0{,}043$), ce qui justifie ce second test, plus robuste.

La corrélation de 0,33 avec la scolarité est significative à 1 % ($p = 0{,}007$). Ces résultats décrivent des associations : l'indicatrice « Afrique » résume beaucoup de caractéristiques (niveau initial, institutions, histoire) et n'identifie aucun effet causal.

D.6 Régression MCO et diagnostics

La fonction `lm()` (*linear model*) est le point d'entrée de toute régression linéaire sous R. Elle reçoit une formule et un data frame, estime le modèle par les MCO et renvoie un objet de classe `lm`, une liste qui contient les coefficients, les résidus et les informations du modèle. Les autres fonctions de la figure D.6 ne réestiment rien : elles lisent cet objet pour résumer, prédire, contrôler les hypothèses ou corriger les écarts-types.

Définition D.2 (Formule)

Une *formule* décrit un modèle : la variable expliquée est à gauche du signe `~`, les variables explicatives à droite, séparées par `+`. La constante est incluse par défaut. Le tableau D.3 donne les écritures les plus courantes.

TABLE D.3. Écritures usuelles d'une formule

Formule	Modèle estimé
`y ~ x1 + x2`	régression de `y` sur `x1` et `x2`, avec constante
`y ~ x1 + x2 - 1`	même modèle sans constante
`y ~ x1 * x2`	effets de `x1` et `x2` et de leur interaction (`x1 + x2 + x1:x2`)
`y ~ log(x1) + I(x1^2)`	transformations ; `I()` protège une opération arithmétique
`y ~ factor(g)`	indicatrices pour chaque modalité de `g`
`y ~ .`	toutes les autres colonnes du data frame comme régresseurs

Le bloc suivant estime l'équation de croissance de la section précédente : la croissance dépend du logarithme du PIB initial (convergence), de l'ouverture commerciale, de la scolarité et de l'instabilité politique.

Code D.13. Régression par les MCO

```
modele <- lm(growth ~ log_rgdp60 + tradeshare + education +
               revolutions + assassinations, data = croissance)
summary(modele)
```

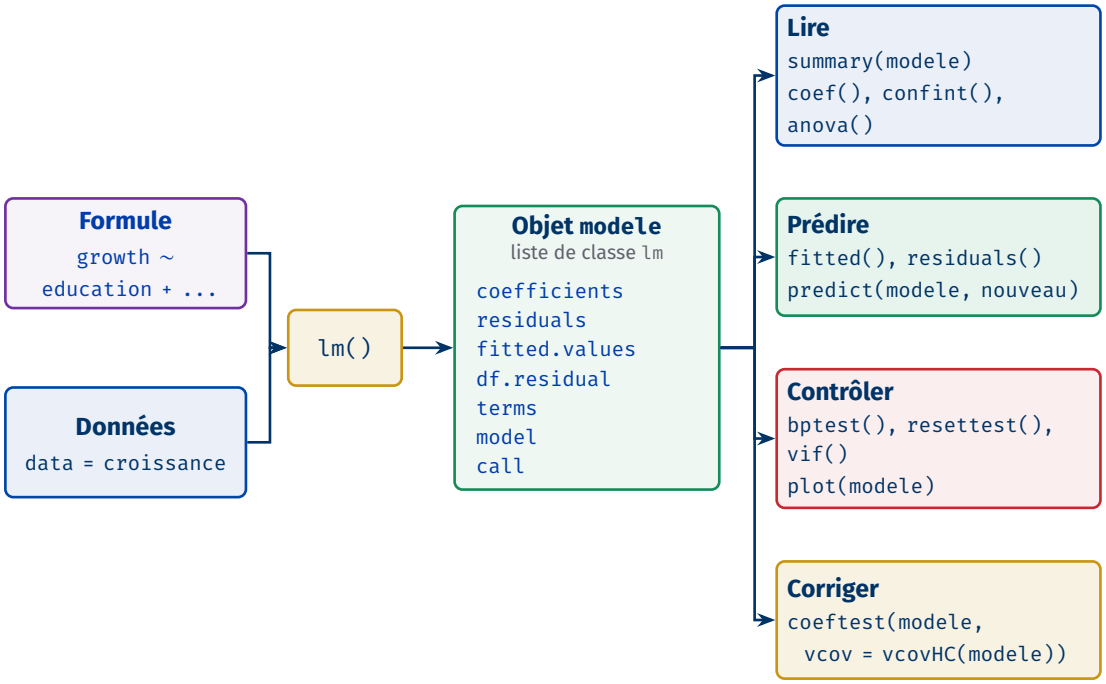


FIGURE D.6. Anatomie d'une estimation avec `lm()`. La fonction reçoit une formule et un data frame, et renvoie un objet, c'est-à-dire une liste de classe `lm` qui contient les coefficients, les résidus, les valeurs ajustées et les informations du modèle. Les fonctions de droite ne réestiment rien : elles lisent cet objet pour résumer, prédire, contrôler les hypothèses ou corriger les écarts-types.

```
Call:
lm(formula = growth ~ log_rgdp60 + tradeshare + education + revolutions +
    assassinations, data = croissance)

Residuals:
    Min      1Q  Median      3Q     Max
-3.9742 -0.7914 -0.3143  0.8840  5.3707

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)      5.3642     3.3926   1.581  0.11919
log_rgdp60      -0.7930     0.4837  -1.640  0.10640
tradeshare       2.0573     0.7770   2.648  0.01038 *
education        0.4277     0.1539   2.778  0.00732 **
revolutions     -1.8429     1.1947  -1.543  0.12830
assassinations   0.4697     0.5209   0.902  0.37083
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.674 on 59 degrees of freedom
Multiple R-squared:  0.2826,  Adjusted R-squared:  0.2218
F-statistic: 4.648 on 5 and 59 DF,  p-value: 0.00121
```

Le modèle explique 28,3 % de la variance de la croissance (R^2 ajusté de 22,2 %) et les régresseurs sont significatifs ensemble ($F = 4{,}65$ avec 5 et 59 degrés de liberté, $p = 0{,}001$). L'ouverture commerciale a un effet positif et significatif : une hausse de 0,1 de la part du

commerce dans le PIB est associée à 0,21 point de croissance annuelle en plus. Chaque année de scolarité initiale supplémentaire est associée à 0,43 point de plus ($p = 0{,}007$). Le coefficient du logarithme du PIB initial est négatif (−0,79), signe d'une convergence conditionnelle (les pays pauvres rattrapent les riches à scolarité et ouverture données), mais il n'est pas significatif à 5 % ($p = 0{,}106$), pas plus que ceux des révolutions et des assassinats. L'écart-type résiduel est de 1,67 point de croissance. Ces coefficients sont des associations conditionnelles, et non des effets causaux.

Les fonctions d'extraction lisent l'objet `modele` sans rien recalculer. Le bloc suivant récupère les coefficients, leurs intervalles de confiance, le R^2, l'écart-type résiduel, puis prédit la croissance d'un pays dont les caractéristiques sont choisies pour l'exemple.

Code D.14. Extraire les résultats et prédire

```
round(coef(modele), 3)                              # coefficients estimés
round(confint(modele, level = 0.95), 3)             # intervalles de confiance à 95 %
round(c(R2 = summary(modele)$r.squared, sigma = sigma(modele)), 3)
nobs(modele)                                        # nombre d'observations
# Prévision pour un pays fictif (valeurs choisies pour l'exemple)
nouveau <- data.frame(log_rgdp60 = log(1000), tradeshare = 0.5, education = 3,
                      revolutions = 0.2, assassinations = 0.3)
predict(modele, newdata = nouveau, interval = "prediction")
```

```
   (Intercept)     log_rgdp60     tradeshare      education
         5.364         -0.793          2.057          0.428
   revolutions assassinations
        -1.843          0.470
                2.5 % 97.5 %
(Intercept)    -1.424 12.153
log_rgdp60     -1.761  0.175
tradeshare      0.502  3.612
education       0.120  0.736
revolutions    -4.234  0.548
assassinations -0.573  1.512
   R2 sigma
0.283 1.674
[1] 65
       fit      lwr      upr
1 1.970142 -1.44915 5.389434
```

Les intervalles de confiance à 95 % de l'ouverture commerciale ([0,50 ; 3,61]) et de la scolarité ([0,12 ; 0,74]) excluent zéro, ce qui confirme leur significativité. Pour un pays fictif qui partirait de 1 000 dollars par habitant, avec une ouverture de 0,5, trois années de scolarité, 0,2 révolution et 0,3 assassinat par an, le modèle prédit 1,97 % de croissance annuelle. L'intervalle de prévision à 95 %, de −1,45 à 5,39 %, est très large : il tient compte non seulement de l'incertitude sur les coefficients, mais aussi de la dispersion des résidus (1,67 point), que le modèle n'explique pas.

D.6.1 Tester les hypothèses du modèle

Les résultats ci-dessus ne valent que si les hypothèses du modèle linéaire sont à peu près respectées. Le bloc suivant applique les contrôles usuels : le test de Breusch-Pagan pour l'hétéroscédasticité, le test RESET pour la forme fonctionnelle, les facteurs d'inflation de la variance (VIF) pour la colinéarité, le test de Shapiro-Wilk pour la normalité des résidus, et les mesures de levier et de distance de Cook pour repérer les observations influentes.

Code D.15. Diagnostics du modèle

```
library(lmtest)                                    # bptest, resettest
library(car)                                       # vif
bptest(modele)                                   # Breusch-Pagan : hétéroscédasticité
resettest(modele, power = 2:3, type = "fitted")    # RESET : forme fonctionnelle
vif(modele)                                      # facteurs d'inflation de la variance
shapiro.test(residuals(modele))                    # normalité des résidus
infl <- data.frame(levier = hatvalues(modele), cook = cooks.distance(modele),
                   row.names = croissance$pays)
round(head(infl[order(-infl$levier), ], 2), 3)     # fort levier (seuil 2k/n = 0,18)
round(head(infl[order(-infl$cook), ], 3), 3)       # pays les plus influents
```

```
  studentized Breusch-Pagan test

data:  modele
BP = 11.989, df = 5, p-value = 0.03494


  RESET test

data:  modele
RESET = 0.81373, df1 = 2, df2 = 57, p-value = 0.4483

    log_rgdp60     tradeshare      education    revolutions
      3.845668       1.154374       3.499384       1.646517
assassinations
      1.497769

  Shapiro-Wilk normality test

data:  residuals(modele)
W = 0.96293, p-value = 0.0488

          levier  cook
Malta      0.456 0.024
Guatemala  0.361 0.007
                  levier  cook
Korea, Republic of  0.060 0.118
Zaire               0.101 0.117
Niger               0.087 0.092
```

Le test de Breusch-Pagan rejette l'homoscédasticité au seuil de 5 % (BP = 11,99, 5 degrés de liberté, $p = 0{,}035$) : les écarts-types usuels sont donc suspects. Le test RESET ne détecte

pas d'erreur de forme fonctionnelle ($p = 0{,}45$). Tous les VIF sont inférieurs à 5 (le plus élevé est 3,85, pour le PIB initial), donc la colinéarité entre scolarité et PIB initial reste tolérable. La normalité des résidus est rejetée de justesse ($p = 0{,}049$). Malte a le levier le plus fort (0,456, contre un seuil indicatif de $2k/n = 0{,}18$), mais sa distance de Cook est faible (0,024) ; la Corée, le Zaïre et le Niger dépassent le seuil indicatif $4/n \approx 0{,}06$ sans qu'aucune distance n'approche 1. Aucun pays n'est donc assez influent pour exiger son retrait.

D.6.2 Corriger les écarts-types

Puisque l'homoscédasticité est rejetée, on remplace les écarts-types usuels par des écarts-types robustes à l'hétéroscédasticité. La version HC3, recommandée pour les petits échantillons (Long & Ervin, 2000), est disponible dans le paquet `sandwich` (Zeileis, 2004). On vérifie aussi la sensibilité des résultats en retirant les pays un à un.

Code D.16. Écarts types robustes et sensibilité

```
library(sandwich)                                  # matrices de covariance robustes
coeftest(modele, vcov = vcovHC(modele, type = "HC3"))   # écarts types robustes HC3
# Sensibilité : coefficient de tradeshare en retirant un pays à la fois
loo <- sapply(croissance$pays, function(p)
  coef(update(modele, data = subset(croissance, pays != p)))["tradeshare"])
round(range(loo), 2)
```

```
t test of coefficients:

               Estimate Std. Error t value Pr(>|t|)
(Intercept)     5.36416    4.14290  1.2948 0.200439
log_rgdp60     -0.79302    0.54115 -1.4654 0.148113
tradeshare      2.05726    0.72987  2.8187 0.006554 **
education       0.42767    0.14916  2.8673 0.005732 **
revolutions    -1.84287    1.04099 -1.7703 0.081842 .
assassinations  0.46972    0.41026  1.1449 0.256866
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] 1.74 2.28
```

Les coefficients sont identiques à ceux du tableau des MCO ; seuls les écarts-types, et donc les statistiques t, changent. Ici, la correction renforce les conclusions : l'écart-type de l'ouverture commerciale passe de 0,777 à 0,730 ($p = 0{,}007$), celui de la scolarité de 0,154 à 0,149 ($p = 0{,}006$), et le coefficient des révolutions devient significatif à 10 % ($p = 0{,}082$). La seconde partie du bloc retire chaque pays tour à tour : le coefficient de l'ouverture commerciale, de 2,06 sur l'échantillon complet, reste entre 1,74 et 2,28, preuve qu'aucun pays isolé ne commande le résultat.

D.6.3 Autocorrélation des erreurs

Avec des données de séries temporelles, les erreurs successives sont souvent corrélées entre elles. Le bloc suivant illustre le problème avec la macroéconomie américaine (jeu `USMacroG`, 204 trimestres de 1950 à 2000) : on régresse la consommation sur le revenu disponible, puis on applique les tests de Durbin-Watson et de Breusch-Godfrey, et on calcule des écarts-types robustes à l'autocorrélation et à l'hétéroscédasticité (HAC) de Newey et West (1987).

Code D.17. Autocorrélation et écarts types HAC

```
data("USMacroG")                                  # macroéconomie américaine, 1950-2000
usa <- as.data.frame(USMacroG)
conso <- lm(consumption ~ dpi, data = usa)        # consommation ~ revenu disponible
dwtest(conso)                                     # Durbin-Watson
bgtest(conso, order = 4)                    # Breusch-Godfrey, données trimestrielles
hac <- NeweyWest(conso, lag = 4, prewhite = FALSE)   # matrice de Newey-West (HAC)
round(cbind(MCO = sqrt(diag(vcov(conso))), HAC = sqrt(diag(hac))), 4)
```

```
  Durbin-Watson test

data:  conso
DW = 0.092048, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is greater than 0


  Breusch-Godfrey test for serial correlation of order up to 4

data:  conso
LM test = 186.6, df = 4, p-value < 2.2e-16

                MCO     HAC
(Intercept) 14.3059 31.1799
dpi          0.0039  0.0114
```

La statistique de Durbin-Watson vaut 0,09, très loin de la valeur 2 qui correspond à l'absence d'autocorrélation : les résidus sont fortement et positivement autocorrélés, ce que confirme le test de Breusch-Godfrey d'ordre 4 (statistique de 186,6, $p < 2{,}2 \times 10^{-16}$). Les écarts-types HAC sont environ trois fois plus grands que ceux des MCO (0,0114 contre 0,0039 pour le revenu) : les tests t usuels seraient gravement trop optimistes. Ce correctif ne règle pas tout, car consommation et revenu sont deux séries non stationnaires. La régression de niveaux de telles séries peut être fallacieuse (Granger & Newbold, 1974), problème que la section D.8 aborde avec les tests de racine unitaire et la cointégration.

D.7 Données de panel et variables instrumentales

D.7.1 Effets fixes et effets aléatoires avec `plm`

Le paquet `plm` (Croissant & Millo, 2008) estime les modèles de panel. Le bloc suivant ajuste l'équation d'investissement des 10 entreprises du jeu `Grunfeld` par quatre méthodes : les MCO empilés (`"pooling"`), les effets fixes individuels (`"within"`), les effets fixes à deux voies (individu et année) et les effets aléatoires. Il compare ensuite les modèles par le test de Hausman et par le test de présence d'effets individuels.

Code D.18. Modèles de panel

```r
library(plm)
data("Grunfeld", package = "plm")
panel <- pdata.frame(Grunfeld, index = c("firm", "year"))
formule <- inv ~ value + capital                  # investissement ~ valeur + capital
mco <- plm(formule, data = panel, model = "pooling")     # MCO empilés
fe  <- plm(formule, data = panel, model = "within")      # effets fixes individuels
ef2 <- plm(formule, data = panel, model = "within", effect = "twoways")   # + temps
re  <- plm(formule, data = panel, model = "random")      # effets aléatoires
modeles <- list(MCO = mco, EF = fe, EF2 = ef2, EA = re)
round(sapply(modeles, function(m) coef(m)[c("value", "capital")]), 4)
phtest(fe, re)                                    # test de Hausman : EF contre EA
pFtest(fe, mco)                                   # effets individuels significatifs ?
```

```
           MCO     EF    EF2     EA
value   0.1156 0.1101 0.1177 0.1098
capital 0.2307 0.3101 0.3579 0.3081

  Hausman Test

data:  formule
chisq = 2.3304, df = 2, p-value = 0.3119
alternative hypothesis: one model is inconsistent


  F test for individual effects

data:  formule
F = 49.177, df1 = 9, df2 = 188, p-value < 2.2e-16
alternative hypothesis: significant effects
```

L'effet de la valeur de marché de l'entreprise est stable d'une méthode à l'autre (de 0,110 à 0,118), tandis que celui du stock de capital varie davantage : 0,231 avec les MCO empilés, 0,310 avec les effets fixes et 0,358 avec les effets fixes à deux voies. Les MCO empilés sous-estiment donc l'effet du capital, car l'hétérogénéité non observée des entreprises est corrélée aux régresseurs. Le test F rejette l'absence d'effets individuels ($F = 49{,}18$, 9 et 188 degrés de liberté, $p < 2{,}2 \times 10^{-16}$), donc le modèle empilé est inadapté. Le test de Hausman ne rejette pas la cohérence des effets aléatoires ($\chi^2 = 2{,}33$, 2 degrés de liberté, $p = 0{,}31$) et les deux estimations sont très proches (0,308 et 0,310 pour le capital). Avec seulement 10 entreprises,

ce test a peu de puissance ; si l'effet individuel peut être corrélé aux régresseurs, les effets fixes restent le choix prudent. Le paquet `fixest`, plus rapide sur de gros panels, offre une alternative avec sa fonction `feols()` (non exécutée ici).

D.7.2 Variables instrumentales avec AER

Quand un régresseur est endogène, les MCO sont biaisés. L'estimateur des doubles moindres carrés (2SLS ou VI) utilise un instrument, c'est-à-dire une variable corrélée au régresseur endogène mais sans effet direct sur la variable expliquée. L'exemple est la demande de cigarettes des États américains en 1995 (Stock & Watson, 2007) : le prix réel est endogène, car il résulte de l'équilibre entre offre et demande, et la taxe de vente réelle, qui déplace l'offre, sert d'instrument.

Code D.19. Variables instrumentales (2SLS)

```
library(AER)
data("CigarettesSW")                              # consommation de cigarettes par État
cig <- subset(CigarettesSW, year == "1995")       # 48 États en 1995
cig$prix_reel   <- cig$price / cig$cpi            # prix réel
cig$revenu_hab  <- cig$income / cig$population / cig$cpi    # revenu réel par habitant
cig$taxe_reelle <- (cig$taxs - cig$tax) / cig$cpi # taxe de vente réelle : instrument
mco <- lm(log(packs) ~ log(prix_reel) + log(revenu_hab), data = cig)
iv  <- ivreg(log(packs) ~ log(prix_reel) + log(revenu_hab) |
               log(revenu_hab) + taxe_reelle, data = cig)
round(cbind(MCO = coef(mco), VI = coef(iv)), 3)   # comparaison des coefficients
round(summary(iv, diagnostics = TRUE)$diagnostics, 3)   # tests de diagnostic
```

```
                   MCO     VI
(Intercept)     10.342  9.431
log(prix_reel)  -1.407 -1.143
log(revenu_hab)  0.344  0.215
                 df1 df2 statistic p-value
Weak instruments   1  45    45.158     0.0
Wu-Hausman         1  44     1.102     0.3
Sargan             0  NA        NA      NA
```

L'élasticité-prix de la demande est estimée à −1,41 par les MCO et à −1,14 par les variables instrumentales : une hausse de 1 % du prix réel réduit la consommation d'environ 1,1 %. L'élasticité-revenu passe de 0,34 à 0,21. Le test d'instrument faible donne une statistique F de 45,2 (valeur p inférieure à 0,001), bien au-dessus du seuil habituel de 10 : l'instrument est suffisamment corrélé au prix. Le test de Wu-Hausman ($p = 0{,}30$) ne rejette pas l'exogénéité du prix, si bien que la correction ne change pas fondamentalement le résultat dans ce cas. La ligne du test de Sargan est vide, car le modèle est juste identifié (un instrument pour un régresseur endogène) : la validité de l'instrument, c'est-à-dire son absence d'effet direct sur la consommation, ne peut donc pas être testée et repose sur un argument économique.

D.8 Séries temporelles et cointégration

D.8.1 Racine unitaire, ARIMA et GARCH

Une série est non stationnaire si ses propriétés changent avec le temps ; la régression de séries non stationnaires est dangereuse. Le test de Dickey-Fuller augmenté (ADF) a pour hypothèse nulle l'existence d'une racine unitaire. La fonction `adf.test()` du paquet `tseries` choisit par défaut un nombre de retards de $\lfloor (T-1)^{1/3} \rfloor$ et inclut une constante et une tendance linéaire. Le bloc suivant l'applique au logarithme de l'indice boursier allemand DAX, puis à ses différences premières, c'est-à-dire aux rendements.

Code D.20. Test de racine unitaire (ADF)

```
library(tseries)
dax <- log(EuStockMarkets[, "DAX"])              # log de l'indice DAX, 1991-1998
adf.test(dax)                                    # niveau : racine unitaire ?
adf.test(diff(dax))                              # première différence (rendements)
```

```
  Augmented Dickey-Fuller Test

data:  dax
Dickey-Fuller = -1.3702, Lag order = 12, p-value = 0.8449
alternative hypothesis: stationary


  Augmented Dickey-Fuller Test

data:  diff(dax)
Dickey-Fuller = -11.105, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary
```

Pour le logarithme du DAX en niveau, la statistique de Dickey-Fuller vaut $-1{,}37$ ($p = 0{,}84$) : on ne rejette pas la racine unitaire, la série est non stationnaire. Pour les différences premières, la statistique est de $-11{,}1$ et la probabilité critique est inférieure à 0,01 (la fonction l'affiche sous forme tronquée à 0,01 et signale que la vraie valeur est plus petite) : on rejette la racine unitaire. Le logarithme du DAX est donc intégré d'ordre 1, et ses rendements sont stationnaires.

Pour modéliser une série stationnaire ou rendue stationnaire par différenciation, la fonction `auto.arima()` du paquet `forecast` choisit l'ordre de différenciation par un test de stationnarité, puis les ordres autorégressif et de moyenne mobile par le critère d'information d'Akaike (Hyndman & Khandakar, 2008). L'exemple est le débit annuel du Nil, de 1871 à 1970 (figure D.7).

Code D.21. Modèle ARIMA et prévision

```
library(forecast)
mod_nil <- auto.arima(Nile, seasonal = FALSE, stepwise = FALSE,
                      approximation = FALSE)      # débit annuel du Nil, 1871-1970
mod_nil                                           # ordre retenu par l'AIC
Box.test(residuals(mod_nil), lag = 10, type = "Ljung-Box", fitdf = 2)
# H0 : les résidus sont un bruit blanc
prev <- forecast(mod_nil, h = 5, level = 95)      # prévisions 1971-1975
round(as.data.frame(prev))
```

```
Series: Nile
ARIMA(1,1,1)

Coefficients:
         ar1      ma1
      0.2544  -0.8741
s.e.  0.1194   0.0605

sigma^2 = 20177:  log likelihood = -630.63
AIC=1267.25   AICc=1267.51   BIC=1275.04

  Box-Ljung test

data:  residuals(mod_nil)
X-squared = 9.7056, df = 8, p-value = 0.2863

     Point Forecast Lo 95 Hi 95
1971            816   538  1095
1972            836   538  1133
1973            840   536  1145
1974            842   533  1150
1975            842   530  1154
```

Le modèle retenu est un ARIMA(1,1,1) : $\Delta y_t = 0{,}254\,\Delta y_{t-1} + \varepsilon_t - 0{,}874\,\varepsilon_{t-1}$, avec une variance des erreurs de 20 177 et un critère d'Akaike de 1 267,25. Le test de Ljung-Box ne rejette pas l'hypothèse de résidus formant un bruit blanc ($X^2 = 9{,}71$, 8 degrés de liberté, $p = 0{,}29$), ce qui valide la spécification. Les prévisions pour 1971 à 1975 vont de 816 à 842 (en 10^8 m^3) et l'intervalle à 95 % s'élargit avec l'horizon, de [538 ; 1 095] à [530 ; 1 154]. Ce modèle ne prend toutefois pas en compte la chute du niveau du fleuve vers 1898, rupture décrite par Cobb (1978), visible sur la figure : un modèle avec rupture serait plus adapté à ces données.

Pour les rendements financiers, dont la variance change dans le temps, on utilise un modèle GARCH (Bollerslev, 1986). Dans un GARCH(1,1), la variance conditionnelle $\sigma_t^2 = \omega + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2$ dépend du carré du dernier choc et de la variance précédente. Le bloc suivant l'estime avec `fGarch` sur les rendements quotidiens du mark allemand en livres sterling (Bollerslev & Ghysels, 1996), avec une loi normale puis une loi de Student.

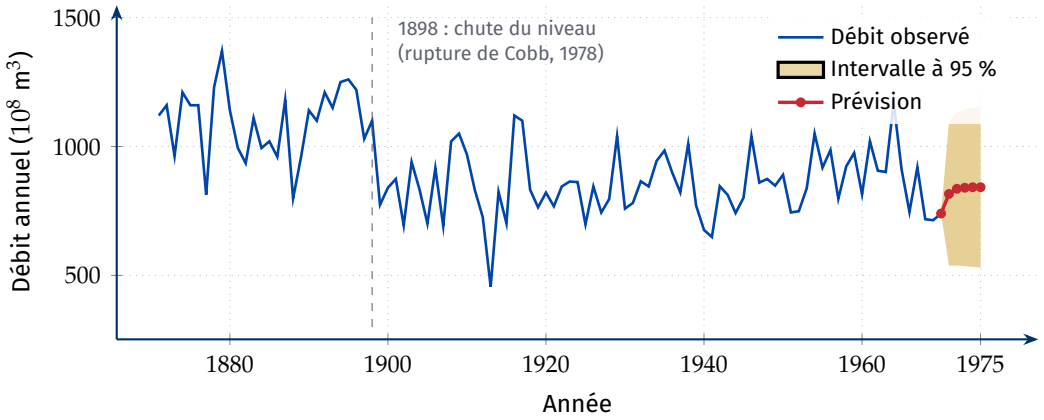


FIGURE D.7. Débit annuel du Nil à Assouan de 1871 à 1970 (jeu `Nile`), avec la prévision d'un modèle ARIMA(1,1,1) pour 1971 à 1975. La bande s'élargit avec l'horizon : à cinq ans, l'intervalle à 95 % va de 530 à 1 154. Le débit est exprimé en 10^8 m^3 (cent millions de mètres cubes). La chute du niveau vers 1898 n'est pas modélisée par l'ARIMA, ce qui est une limite du modèle.

Code D.22. Modèle GARCH(1,1)

```
library(fGarch)
data("MarkPound")                                  # rendements quotidiens DEM/GBP
g_norm <- garchFit(~ garch(1, 1), data = MarkPound, cond.dist = "norm", trace = FALSE)
g_std  <- garchFit(~ garch(1, 1), data = MarkPound, cond.dist = "std",  trace = FALSE)
round(g_norm@fit$matcoef, 4)                       # estimations, loi normale
c(AIC_normale = g_norm@fit$ics[["AIC"]], AIC_student = g_std@fit$ics[["AIC"]])
```

```
        Estimate  Std. Error  t value Pr(>|t|)
mu       -0.0062      0.0085  -0.7316   0.4644
omega     0.0108      0.0028   3.7925   0.0001
alpha1    0.1531      0.0264   5.7958   0.0000
beta1     0.8060      0.0334  24.1445   0.0000
AIC_normale AIC_student
   1.125236    1.007506
```

Les deux coefficients de variance sont significatifs : $\widehat{\alpha}_1 = 0{,}153$ et $\widehat{\beta}_1 = 0{,}806$. Leur somme, 0,959, mesure la persistance de la volatilité : un choc se dissipe lentement, mais la somme reste inférieure à 1, donc la variance inconditionnelle existe. La moyenne conditionnelle (μ) n'est pas significativement différente de zéro. Le critère d'Akaike par observation est de 1,125 avec la loi normale et de 1,008 avec la loi de Student : les queues épaisses des rendements sont mieux décrites par la seconde, qui est préférable.

D.8.2 Cointégration et modèle à correction d'erreur

Deux séries intégrées d'ordre 1 sont *cointégrées* si une combinaison linéaire d'elles est stationnaire : elles partagent alors une relation de long terme dont elles s'écartent temporairement. Le test de Johansen compte le nombre r de relations de cointégration ; le modèle à correction d'erreur (VECM) décrit ensuite l'ajustement vers l'équilibre. Le paquet `urca` fournit les deux. L'exemple est la demande de monnaie au Danemark (Johansen & Juselius, 1990) : logarithme

de la monnaie (`LRM`), logarithme du revenu (`LRY`), taux obligataire (`IBO`) et taux de dépôt (`IDE`).

Code D.23. Test de Johansen et VECM

```
library(urca)
data(denmark)                                        # demande de monnaie au Danemark
y <- denmark[, c("LRM", "LRY", "IBO", "IDE")]        # monnaie, revenu, taux
jo <- ca.jo(y, type = "trace", ecdet = "const", K = 2)  # test de la trace, 2 retards
round(cbind(statistique = jo@teststat, jo@cval), 2)  # statistiques et valeurs critiques
vecm <- cajorls(jo, r = 1)                 # VECM avec une relation de cointégration
round(vecm$beta[, 1], 3)                           # relation de long terme normalisée
round(vecm$rlm$coefficients["ect1", ], 3)          # vitesses d'ajustement
```

```
         statistique 10pct  5pct  1pct
r <= 3 |        2.29  7.52  9.24 12.97
r <= 2 |        8.95 17.85 19.96 24.60
r <= 1 |       19.09 32.00 34.91 41.07
r = 0  |       52.71 49.65 53.12 60.16
  LRM.l2   LRY.l2   IBO.l2   IDE.l2 constant
   1.000   -0.969    5.403   -4.140   -6.478
 LRM.d  LRY.d  IBO.d  IDE.d
-0.300  0.027  0.004  0.020
```

Le test de la trace compare chaque statistique à ses valeurs critiques. Pour $r = 0$, la statistique de 52,71 est inférieure à la valeur critique à 5 % (53,12) mais supérieure à celle à 10 % (49,65) : on ne rejette pas l'absence de cointégration à 5 %, mais on la rejette à 10 %. Le cas est donc limite. On retient ici $r = 1$ pour illustrer l'estimation du VECM, appuyé par la théorie de la demande de monnaie, mais une conclusion ferme exigerait de tester la robustesse au nombre de retards. La relation de long terme normalisée s'écrit $\text{LRM} - 0{,}969\,\text{LRY} + 5{,}403\,\text{IBO} - 4{,}140\,\text{IDE} - 6{,}478 = 0$, soit une élasticité de la demande de monnaie au revenu proche de 1 et des signes conformes à la théorie : un taux obligataire élevé (coût d'opportunité de la monnaie) la réduit, un taux de dépôt élevé (rendement de la monnaie) l'augmente. Le coefficient d'ajustement de la monnaie, $-0{,}30$, indique qu'environ 30 % de l'écart à l'équilibre se résorbe à chaque trimestre ; les trois autres coefficients, proches de zéro, montrent que le revenu et les taux ne réagissent presque pas à cet écart. Le paquet `vars` complète ces outils avec `VARselect()` pour choisir le nombre de retards, `VAR()` et `irf()` pour les fonctions de réponse impulsionnelle (non exécutés ici).

D.9 Inférence causale et méthodes d'identification

Les trois méthodes de cette section utilisent des fichiers de l'annexe C du livre et des paquets qui ne sont pas installés par défaut. Leurs blocs n'ont donc pas été exécutés pour la rédaction de ce livre : la syntaxe a été contrôlée, mais aucune sortie n'est reproduite. Les arguments sont ceux des versions actuelles des paquets ; la page d'aide de chaque fonction (`?rdrobust`, par exemple) donne le détail.

D.9.1 Régression par discontinuité avec `rdrobust`

Dans le fichier de l'annexe C du livre consacré au programme de scolarisation du Kivu, les écoles dont le score d'éligibilité est inférieur à 50 sont éligibles. Or `rdrobust` considère comme traitées les observations dont la variable de score est supérieure ou égale au seuil. Pour que le coefficient estimé ait le bon signe, on construit une variable de distance qui est positive du côté éligible. La fonction choisit elle-même la largeur de bande (option `bwselect = "mserd"`) et propose des intervalles de confiance robustes (Calonico et al., 2014). Le test de densité de `rddensity` vérifie l'absence de manipulation du score autour du seuil (McCrary, 2008).

Code D.24. Régression par discontinuité (non exécuté)

```
library(rdrobust); library(rddensity)
ecoles <- read.csv("epsp_scolarisation_kivu_2014_2022.csv")   # annexe E
ecoles$distance <- 50 - ecoles$score_eligibilite  # > 0 pour les écoles éligibles (score
    < 50)
rd <- rdrobust(y = ecoles$scolarisation, x = ecoles$distance, c = 0,
               kernel = "triangular", bwselect = "mserd", vce = "hc1")
summary(rd)                                       # effet au seuil, IC robustes
dens <- rddensity(X = ecoles$distance, c = 0)     # test de manipulation du score
summary(dens)
rdplotdensity(dens, ecoles$distance)
```

La sortie de `summary(rd)` donne l'effet estimé au seuil avec trois lignes (estimation conventionnelle, avec correction du biais, et robuste), la largeur de bande retenue et le nombre d'observations de chaque côté du seuil. Un test de densité dont la probabilité critique est grande indique que le score n'a pas été manipulé.

D.9.2 Différences en différences avec `did`

Quand les unités sont traitées à des dates différentes, l'estimateur usuel en double différence peut être biaisé. Callaway et Sant'Anna (2021) estiment un effet moyen pour chaque groupe de traitement et chaque période, puis les agrègent. Dans le fichier de la subvention agricole du Kivu et du Kasaï, la colonne `annee_traitement` indique l'année d'accès à la subvention, avec 0 pour les ménages jamais traités ; le paquet exige un identifiant de ménage numérique.

Code D.25. Différences en différences échelonnées (non exécuté)

```
library(did)
menages <- read.csv("fao_subvention_kivu_kasai_2015_2022.csv")   # annexe E
menages$id <- as.integer(factor(menages$menage_id))   # identifiant numérique exigé
att <- att_gt(yname = "rendement_agricole", tname = "annee", idname = "id",
              gname = "annee_traitement",           # 0 pour les ménages jamais traités
              data = menages, control_group = "nevertreated", est_method = "dr")
effets <- aggte(att, type = "dynamic")              # effets selon la durée d'exposition
summary(effets)
ggdid(effets)                                       # graphique d'événements
```

L'agrégation `type = "dynamic"` donne l'effet selon le nombre d'années écoulées depuis l'accès à la subvention. Les effets avant le traitement, proches de zéro s'il n'y a pas de tendance différentielle, servent de test de plausibilité.

D.9.3 Double machine learning avec `DoubleML`

Le double machine learning (Chernozhukov et al., 2018) estime l'effet d'un traitement D sur un résultat Y en présence de nombreuses variables de contrôle $\mathbf{X}$. Il utilise un algorithme d'apprentissage pour prédire Y et D à partir de $\mathbf{X}$, puis régresse les résidus de Y sur ceux de D, avec validation croisée pour éviter le surajustement. Ici, le traitement est l'usage du mobile money et le résultat est le logarithme du revenu mensuel des clients du fichier FINCA-RDC.

Code D.26. Double machine learning (non exécuté)

```
library(DoubleML); library(mlr3); library(mlr3learners); library(data.table)
finca <- as.data.table(read.csv("finca_rdc_clients_2022.csv"))   # annexe E
dml_data <- DoubleMLData$new(finca, y_col = "log_revenu", d_cols = "acces_mobile_money",
                             x_cols = c("age", "education", "actif_total",
                                        "zone_rurale", "menage_taille"))
set.seed(2025)
dml <- DoubleMLPLR$new(dml_data,
                       ml_l = lrn("regr.ranger", num.trees = 200),     # E[Y | X]
                       ml_m = lrn("classif.ranger", num.trees = 200,
                                  predict_type = "prob"),              # E[D | X]
                       n_folds = 5)
dml$fit()
dml$summary()                                          # effet estimé de D sur Y
```

La sortie de `dml$summary()` donne le coefficient de `acces_mobile_money`, son écart-type, sa statistique t et son intervalle de confiance. Il s'interprète comme un effet causal seulement si les contrôles $\mathbf{X}$ suffisent à neutraliser tous les facteurs qui influencent à la fois l'accès au mobile money et le revenu.

D.10 Modèles à variable dépendante limitée

D.10.1 Logit et probit

Quand la variable expliquée est binaire, on modélise une probabilité avec `glm()` et une loi binomiale dont le lien est logit ou probit. L'exemple est la participation de 872 femmes suisses au marché du travail (Gerfin, 1996) : la variable `income` est le logarithme du revenu hors travail et `age` est exprimé en décennies. Les coefficients d'un logit ne sont pas des effets sur la probabilité : on calcule les *effets marginaux moyens*, ici par la formule $\frac{1}{n}\sum_i f(\mathbf{x}_i'\widehat{\boldsymbol{\beta}})\,\widehat{\beta}_k$, où f est la densité de la loi logistique.

Code D.27. Logit, effets marginaux et comparaison avec le probit

```
data("SwissLabor")                    # participation des femmes, 872 observations
logit <- glm(participation ~ income + age + education + youngkids + oldkids + foreign,
             family = binomial(link = "logit"), data = SwissLabor)
probit <- update(logit, family = binomial(link = "probit"))
round(summary(logit)$coefficients, 3)             # coefficients du logit
```

```
ame <- mean(dlogis(predict(logit, type = "link"))) * coef(logit)  # effets marginaux
round(ame[-1], 3)
round(coef(logit) / coef(probit), 2)[-1]          # rapport logit / probit
```

```
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   10.374      2.167   4.788    0.000
income        -0.815      0.206  -3.966    0.000
age           -0.510      0.091  -5.638    0.000
education      0.032      0.029   1.093    0.275
youngkids     -1.331      0.180  -7.386    0.000
oldkids       -0.022      0.074  -0.298    0.766
foreignyes     1.310      0.200   6.560    0.000
    income        age  education  youngkids    oldkids foreignyes
    -0.170     -0.106      0.007     -0.277     -0.005      0.273
    income        age  education  youngkids    oldkids foreignyes
      1.62       1.64       1.55       1.70       1.63       1.63
```

Le revenu hors travail (−0,82), l'âge (−0,51) et le nombre d'enfants de moins de 7 ans (−1,33) réduisent significativement la probabilité de participation, tandis que la nationalité étrangère l'augmente (+1,31). L'éducation et le nombre d'enfants plus âgés ne sont pas significatifs. Les effets marginaux moyens sont plus parlants : chaque enfant de moins de 7 ans supplémentaire réduit en moyenne la probabilité de participer de 27,7 points de pourcentage, et dix ans de plus de 10,6 points ; une femme étrangère participe, à caractéristiques égales, avec 27,3 points de plus. Pour une variable indicatrice, cet effet calculé par la dérivée est une approximation de la variation discrète de la probabilité. Enfin, les coefficients du logit sont de 1,55 à 1,70 fois ceux du probit, ce qui correspond à la règle empirique usuelle (entre 1,6 et 1,8) : les deux modèles livrent des effets marginaux quasi identiques.

D.10.2 Modèle Tobit

Le modèle Tobit s'applique quand la variable expliquée est censurée, comme les heures de travail, qui valent 0 pour les femmes qui ne travaillent pas. Il se compose d'une variable latente $y_i^* = \mathbf{x}_i'\boldsymbol{\beta} + u_i$, observée seulement quand elle est positive. Le bloc suivant utilise la fonction `tobit()` de `AER` sur les 753 femmes mariées du jeu `PSID1976` (Mroz, 1987), dont 325 ont travaillé 0 heure.

Code D.28. Modèle Tobit censuré à gauche

```
data("PSID1976")                          # offre de travail de 753 femmes mariées
f <- hours ~ age + education + youngkids + oldkids + hwage
tob <- tobit(f, left = 0, data = PSID1976)        # Tobit censuré à gauche en 0
mco_h <- lm(f, data = PSID1976)                   # MCO sur toutes les observations
round(cbind(Tobit = coef(tob)[1:6], MCO = coef(mco_h)), 1)
c(censurees = sum(PSID1976$hours == 0), total = nrow(PSID1976))
```

```
              Tobit    MCO
(Intercept)  1259.7 1418.4
age           -39.0  -21.5
education     121.9   55.1
youngkids   -1065.3 -512.7
oldkids      -123.2  -90.2
hwage         -51.0  -26.4
censurees     total
      325       753
```

Les coefficients du Tobit sont plus grands en valeur absolue que ceux des MCO : par exemple, un enfant de moins de 6 ans de plus réduit les heures désirées de 1 065 heures (Tobit) contre 513 heures (MCO), et une année d'éducation de plus augmente les heures de 122 contre 55. Les MCO sont biaisés vers zéro, car les 325 zéros écrasent la relation. Les deux colonnes ne sont toutefois pas strictement comparables : les coefficients du Tobit mesurent l'effet sur les heures désirées (la variable latente), et l'effet sur les heures effectivement observées est plus faible. Une règle empirique l'obtient en multipliant le coefficient par la part des observations non censurées, soit $428/753 \approx 0{,}57$; le coefficient des enfants de moins de 6 ans devient alors environ -605 heures, proche de celui des MCO.

D.10.3 Modèles de comptage

Pour une variable de comptage (nombre de visites, d'accidents), on part d'une régression de Poisson, qui suppose que la variance est égale à la moyenne. Cette hypothèse est souvent fausse : on teste la surdispersion avec `dispersiontest()` (Cameron & Trivedi, 1990) et, si elle est présente, on passe à une loi binomiale négative avec `glm.nb()` du paquet `MASS`. L'exemple porte sur le nombre de visites chez le médecin de 4 406 personnes âgées américaines (Deb & Trivedi, 1997).

Code D.29. Poisson, surdispersion et binomiale négative

```
library(MASS)
data("NMES1988")                                  # visites médicales, 4 406 individus
pois <- glm(visits ~ hospital + health + chronic + gender + school + insurance,
            family = poisson, data = NMES1988)
dispersiontest(pois)                              # surdispersion ? (Cameron-Trivedi)
nb <- glm.nb(visits ~ hospital + health + chronic + gender + school + insurance,
             data = NMES1988)
round(summary(nb)$coefficients[, 1:3], 3)
round(exp(coef(nb))[c("hospital", "chronic")], 2) # rapports de taux d'incidence
c(AIC_poisson = AIC(pois), AIC_binomiale_negative = AIC(nb))
```

```
  Overdispersion test

data:  pois
z = 11.509, p-value < 2.2e-16
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  6.706192

                Estimate Std. Error z value
(Intercept)        0.929      0.055  17.022
hospital           0.218      0.020  10.793
healthpoor         0.305      0.049   6.288
healthexcellent   -0.342      0.061  -5.610
chronic            0.175      0.012  14.466
gendermale        -0.126      0.031  -4.052
school             0.027      0.004   6.103
insuranceyes       0.224      0.039   5.686
hospital  chronic
    1.24     1.19
           AIC_poisson AIC_binomiale_negative
              35959.23               24359.11
```

Le test de surdispersion rejette l'égalité de la variance et de la moyenne ($z = 11{,}5$, $p < 2{,}2 \times 10^{-16}$) : la variance est estimée à 6,7 fois la moyenne, ce qui invalide les écarts-types du modèle de Poisson. Le modèle binomial négatif est préféré, et son critère d'Akaike est bien plus faible (24 359 contre 35 959). Les coefficients s'interprètent en rapports de taux d'incidence, obtenus en prenant l'exponentielle : chaque séjour hospitalier supplémentaire multiplie le nombre de visites par 1,24 (+24 %) et chaque maladie chronique de plus par 1,19. Une assurance privée est associée à 25 % de visites en plus ($e^{0{,}224} \approx 1{,}25$), une mauvaise santé perçue à 36 % de plus, et une excellente santé perçue à 29 % de moins, par rapport à une santé moyenne.

D.11 Régression quantile et méthodes non paramétriques

D.11.1 Régression quantile

La régression quantile (Koenker & Bassett, 1978) estime l'effet d'une variable non pas sur la moyenne, mais sur un quantile de la variable expliquée : la médiane, le dixième ou le neuvième décile. Elle montre si l'effet est le même pour tous. Le jeu `engel` du paquet `quantreg` contient le revenu et la dépense alimentaire annuels, en francs belges, de 235 ménages ouvriers (Koenker & Bassett, 1982). Le bloc estime la relation aux quantiles 0,1, 0,5 et 0,9 et la compare à celle des MCO.

Code D.30. Régression quantile

```
library(quantreg)
data(engel)                        # 235 ménages belges : revenu et dépense alimentaire
rq_engel <- rq(foodexp ~ income, tau = c(0.10, 0.50, 0.90), data = engel)
round(coef(rq_engel), 3)                            # une colonne par quantile
round(coef(lm(foodexp ~ income, data = engel)), 3)   # MCO pour comparaison
se <- sapply(summary(rq_engel, se = "nid"), function(s) s$coefficients["income", 2])
round(setNames(se, c("tau=0.1", "tau=0.5", "tau=0.9")), 3)  # écarts types de la pente
```

```
            tau= 0.1 tau= 0.5 tau= 0.9
(Intercept)  110.142   81.482   67.351
income         0.402    0.560    0.686
(Intercept)      income
    147.475       0.485
tau=0.1 tau=0.5 tau=0.9
  0.040   0.028   0.028
```

La pente, c'est-à-dire la part d'un franc de revenu supplémentaire consacrée à l'alimentation, augmente avec le quantile : 0,40 au quantile 0,1, 0,56 à la médiane et 0,69 au quantile 0,9, alors que les MCO résument tout par une seule pente de 0,49. L'écart de 0,28 entre les quantiles extrêmes est grand au regard des erreurs types des pentes (de 0,03 à 0,04), ce qui suggère des pentes différentes ; la fonction `anova()` appliquée à des modèles quantiles permet un test formel. La dispersion de la dépense alimentaire croît avec le revenu, signe d'hétéroscédasticité que la régression des MCO ne montre pas.

D.11.2 Régression non paramétrique

Quand la forme de la relation est inconnue, on peut estimer la moyenne conditionnelle sans l'imposer. L'estimateur de Nadaraya-Watson calcule en chaque point une moyenne pondérée des observations voisines, avec des poids décroissants avec la distance. Sa qualité dépend de la *largeur de bande* h : trop petite, la courbe suit le bruit ; trop grande, elle lisse jusqu'à gommer la structure. Le paquet `KernSmooth` fournit un choix automatique (`dpill()`) et l'estimateur lui-même (`locpoly()` avec `degree = 0`). Le paquet `np`, non exécuté ici, choisit h par validation croisée.

Code D.31. Estimateur de Nadaraya-Watson et largeur de bande

```
library(KernSmooth)
h <- dpill(engel$income, engel$foodexp)          # fenêtre plug-in
revenus <- c(500, 1000, 2000)
nw <- function(fenetre) approx(locpoly(engel$income, engel$foodexp, degree = 0,
                                       bandwidth = fenetre), xout = revenus)$y
res <- rbind(plugin = nw(h), h100 = nw(100), h300 = nw(300),     # Nadaraya-Watson
             MCO = predict(lm(foodexp ~ income, data = engel),
                           data.frame(income = revenus)))
colnames(res) <- paste("revenu", revenus)
round(res)
round(h, 1)
```

```
       revenu 500 revenu 1000 revenu 2000
plugin        352         651        1322
h100          371         636        1171
h300          455         599        1072
MCO           390         633        1118
[1] 29.5
```

La largeur de bande automatique vaut 29,5 francs. Avec une petite largeur (29,5 ou 100), l'estimation de la dépense alimentaire d'un ménage de revenu 2 000 francs est de 1 322 ou 1 171 francs ; avec une grande largeur (300), elle tombe à 1 072, proche de la droite des MCO (1 118). L'estimation varie donc de plus de 20 % selon la largeur de bande, parce que peu de ménages ont un revenu élevé : c'est le compromis entre biais et variance. À 500 francs, où les données sont plus nombreuses, les estimations sont plus proches (de 352 à 455 francs) ; la grande largeur de bande tire pourtant l'estimation vers le haut, car le point se trouve près du revenu minimal de l'échantillon (377 francs) et la moyenne locale inclut surtout des ménages plus riches (biais de bord).

D.12 Apprentissage automatique et réseaux de neurones

D.12.1 Forêt aléatoire et *boosting*

Une *forêt aléatoire* (Breiman, 2001) moyenne les prévisions de nombreux arbres de décision construits sur des échantillons *bootstrap* et des sous-ensembles aléatoires de variables. L'exemple prédit si une demande de carte de crédit est acceptée (jeu `CreditCard`, 1 319 demandes). On exclut les variables `share` et `expenditure`, car elles ne sont observées qu'après l'octroi de la carte : les inclure donnerait une fuite d'information. On sépare les données en un échantillon d'apprentissage (70 %) et un échantillon de test (30 %) avec une graine fixée par `set.seed()`, puis on compare la forêt à un logit par l'aire sous la courbe ROC (AUC), calculée avec `pROC` (Robin et al., 2011).

Code D.32. Forêt aléatoire contre logit

```
library(caret); library(randomForest); library(pROC)
data("CreditCard")                              # 1 319 demandes de carte de crédit
set.seed(2025)
idx <- createDataPartition(CreditCard$card, p = 0.70, list = FALSE)   # 70 % / 30 %
app <- CreditCard[idx, ]; test <- CreditCard[-idx, ]
f <- card ~ reports + age + income + owner + selfemp + dependents +
  months + majorcards + active                  # sans share ni expenditure
logit_cc <- glm(f, family = binomial, data = app)
rf <- randomForest(f, data = app, ntree = 500)
p_logit <- predict(logit_cc, test, type = "response")
p_rf    <- predict(rf, test, type = "prob")[, "yes"]
roc_logit <- roc(test$card, p_logit, levels = c("no", "yes"), direction = "<")
roc_rf    <- roc(test$card, p_rf,    levels = c("no", "yes"), direction = "<")
round(c(AUC_logit = auc(roc_logit), AUC_foret = auc(roc_rf)), 3)
```

```
AUC_logit AUC_foret
    0.866     0.879
```

L'AUC mesure la capacité à classer les demandes acceptées avant les refusées : 0,5 correspond au hasard et 1 à un classement parfait. La forêt obtient 0,879 et le logit 0,866 sur les 394 demandes de l'échantillon de test. L'écart de 0,013 est faible : le gain d'une méthode plus flexible est modeste ici, et il est mesuré sur un seul découpage. Pour conclure, il faudrait répéter la comparaison par validation croisée.

Le *boosting* construit les arbres l'un après l'autre, chacun corrigeant les erreurs du précédent ; `xgboost` (Chen & Guestrin, 2016) en est l'implémentation la plus utilisée. Le bloc suivant prolonge le précédent (il réutilise `f`, `idx` et `CreditCard`). Il n'a pas été exécuté, le paquet n'étant pas installé sur le poste de l'auteur.

Code D.33. XGBoost (non exécuté)

```
library(xgboost)                              # suite du bloc précédent : f, idx, roc
X <- model.matrix(f, data = CreditCard)[, -1]       # matrice des régresseurs
y <- as.integer(CreditCard$card == "yes")           # variable cible 0/1
dtrain <- xgb.DMatrix(X[idx, ], label = y[idx])
dtest  <- xgb.DMatrix(X[-idx, ], label = y[-idx])
params <- list(objective = "binary:logistic", eval_metric = "auc",
               eta = 0.05, max_depth = 3, subsample = 0.8, colsample_bytree = 0.8)
set.seed(2025)
xgb <- xgb.train(params = params, data = dtrain, nrounds = 200)
p_xgb <- predict(xgb, dtest)                        # probabilités prédites
auc(roc(y[-idx], p_xgb, direction = "<"))           # AUC sur l'échantillon test
```

Le nombre d'arbres (`nrounds`) est ici fixé à 200 ; on le choisit normalement par validation croisée avec `xgb.cv()`. La dernière ligne affiche l'AUC sur l'échantillon de test, comparable à celles de la forêt et du logit.

D.12.2 LASSO, ridge et elastic net

Quand les variables explicatives sont nombreuses ou fortement corrélées, la régression pénalisée stabilise l'estimation. Le *LASSO* (Tibshirani, 1996) minimise la somme des carrés des résidus plus une pénalité $\lambda \sum_k |\beta_k|$, qui ramène à zéro les coefficients les moins utiles ; la régression ridge pénalise la somme des carrés des coefficients ; l'*elastic net* combine les deux. Le paquet `glmnet` (Friedman et al., 2010) les estime avec l'argument `alpha` (1 pour le LASSO, 0 pour la régression ridge, une valeur intermédiaire pour l'elastic net) et choisit λ par validation croisée. L'exemple explique le score moyen en lecture et en mathématiques de 420 districts scolaires californiens.

Code D.34. LASSO avec validation croisée

```
library(glmnet)
data("CASchools")                                  # 420 districts scolaires californiens
CASchools$score <- (CASchools$read + CASchools$math) / 2
f_e <- score ~ students + teachers + calworks + lunch + computer +
  expenditure + income + english + grades
Xe <- model.matrix(f_e, data = CASchools)[, -1]   # matrice des régresseurs
ye <- CASchools$score
set.seed(2025)
cv <- cv.glmnet(Xe, ye, alpha = 1, nfolds = 10)   # LASSO, validation croisée
round(c(lambda_min = cv$lambda.min, lambda_1se = cv$lambda.1se), 3)
lasso <- as.numeric(coef(cv, s = "lambda.1se"))   # coefficients au lambda 1 ET
round(cbind(MCO = coef(lm(f_e, data = CASchools)), lasso_1se = lasso), 3)
```

```
lambda_min lambda_1se
     0.019      1.773
                MCO lasso_1se
(Intercept) 659.181   665.671
students     -0.001     0.000
teachers      0.004     0.000
calworks     -0.101     0.000
lunch        -0.356    -0.391
computer      0.004     0.000
expenditure   0.002     0.000
income        0.612     0.526
english      -0.223    -0.131
gradesKK-08  -3.644     0.000
```

La validation croisée propose deux valeurs de λ : 0,019 (celle qui minimise l'erreur) et 1,773 (la plus grande valeur dont l'erreur reste à un écart-type du minimum, plus parcimonieuse). À cette seconde valeur, le LASSO ne retient que trois variables : la part d'élèves qui bénéficient de repas à prix réduit (`lunch`, −0,39), le revenu du district (`income`, 0,53) et la part d'élèves apprenant l'anglais (`english`, −0,13). Les coefficients conservés sont plus petits que ceux des MCO (−0,36, 0,61 et −0,22) : la pénalité les rétrécit vers zéro, au prix d'un biais. Les nombres d'élèves et d'enseignants, corrélés à 0,997, ont des coefficients minuscules et instables avec les MCO ; le LASSO écarte les deux. Le choix de λ dépend d'un découpage aléatoire, d'où le `set.seed()`. Les coefficients pénalisés ne sont pas des estimations sans biais : ils servent à sélectionner des variables et à prédire, et l'inférence exige des méthodes spécifiques.

D.12.3 Réseaux de neurones avec `keras`

Un réseau de neurones enchaîne des couches de neurones, chacun combinant linéairement ses entrées puis appliquant une fonction d'activation. Le bloc suivant prolonge celui du *boosting* (il réutilise `X`, `y` et `idx`) : il standardise les variables avec les moyennes et écarts-types de l'échantillon d'apprentissage seulement, construit un réseau à deux couches cachées (16 et 8 neurones) avec un abandon (*dropout*) de 30 %, et arrête l'entraînement quand l'erreur de validation cesse de baisser. Il n'a pas été exécuté ici ; il est écrit pour le paquet `keras`, et la nouvelle interface `keras3` demande quelques ajustements décrits dans sa documentation.

Code D.35. Perceptron multicouche avec keras (non exécuté)

```
library(keras)                                  # suite : X, y, idx du bloc précédent
X_app  <- scale(X[idx, ])                          # standardisation (apprentissage)
X_test <- scale(X[-idx, ], center = attr(X_app, "scaled:center"),
                scale = attr(X_app, "scaled:scale"))  # mêmes moyennes et écarts types
modele <- keras_model_sequential() |>
  layer_dense(units = 16, activation = "relu", input_shape = ncol(X_app)) |>
  layer_dropout(rate = 0.3) |>
  layer_dense(units = 8, activation = "relu") |>
  layer_dense(units = 1, activation = "sigmoid")
modele |> compile(optimizer = optimizer_adam(learning_rate = 0.001),
                  loss = "binary_crossentropy", metrics = "AUC")
fit_nn <- modele |> fit(X_app, y[idx], epochs = 100, batch_size = 32,
                        validation_split = 0.2, verbose = 0,
                        callbacks = callback_early_stopping(patience = 10,
                                                            restore_best_weights = TRUE)↩
    )
modele |> evaluate(X_test, y[-idx], verbose = 0)
```

L'étape de standardisation applique à l'échantillon de test les moyennes et écarts-types de l'échantillon d'apprentissage : recalculer ces statistiques sur le test introduirait une fuite d'information. La fonction `evaluate()` donne la perte et l'AUC sur l'échantillon de test, à comparer à celles de la forêt et du logit. Sur un jeu de cette taille (1 319 observations), un réseau n'a aucune raison de battre un logit : il faut s'attendre à un résultat comparable.

D.13 Économétrie spatiale sous R

L'économétrie spatiale prend en compte la proximité géographique. On commence par définir les voisins de chaque unité : deux zones sont voisines si elles se touchent (contiguïté « reine »). On en tire une matrice de poids $\mathbf{W}$ normalisée par ligne. L'*indice de Moran* mesure l'autocorrélation spatiale d'une variable x : $I = \frac{n}{S_0} \frac{\sum_i \sum_j w_{ij}(x_i-\bar{x})(x_j-\bar{x})}{\sum_i (x_i-\bar{x})^2}$ avec $S_0 = \sum_i \sum_j w_{ij}$. Une valeur positive signale que des zones voisines ont des valeurs semblables. Deux modèles prennent le relais : le modèle à retard spatial (SAR), $\mathbf{y} = \rho \mathbf{W}\mathbf{y} + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, où le résultat d'une zone dépend de celui de ses voisines, et le modèle à erreur spatiale (SEM), $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{u}$ avec $\mathbf{u} = \lambda \mathbf{W}\mathbf{u} + \boldsymbol{\varepsilon}$. Pour un modèle SAR, l'effet d'une variable explicative sur la zone elle-même (effet direct) diffère de son effet total, car il se propage aux voisines (effet indirect) : la fonction `impacts()` sépare les deux (Bivand et al., 2013).

Le fichier cartographique des territoires de la République Démocratique du Congo n'étant pas livré avec R, le bloc suivant utilise un jeu fourni par le paquet `sf` : les 100 comtés de Caroline du Nord, avec le taux de mort subite du nourrisson (pour 1 000 naissances) et la part des naissances non blanches (Pebesma & Bivand, 2023). Pour un fichier du Bureau de la coordination des affaires humanitaires (OCHA), il suffit de remplacer l'appel à `st_read()`. La lecture de la carte et le calcul des deux variables ont été vérifiés séparément ; le reste du bloc exige les paquets `spdep` et `spatialreg`, non installés sur le poste de l'auteur, et n'a pas été exécuté.

Code D.36. Moran, tests du multiplicateur de Lagrange et SAR (non exécuté)

```
library(sf); library(spdep); library(spatialreg)
nc <- st_read(system.file("shape/nc.shp", package = "sf"), quiet = TRUE)  # 100 comtés
nc$taux <- 1000 * nc$SID74 / nc$BIR74                # décès par 1 000 naissances
nc$part_nb <- nc$NWBIR74 / nc$BIR74                  # part des naissances non blanches
voisins <- poly2nb(nc, queen = TRUE)                 # contiguïté (reine)
poids <- nb2listw(voisins, style = "W")              # poids standardisés par ligne
moran.test(nc$taux, listw = poids)                   # indice de Moran global
mco <- lm(taux ~ part_nb, data = nc)
lm.RStests(mco, listw = poids, test = c("RSerr", "RSlag", "adjRSerr", "adjRSlag"))
sar <- lagsarlm(taux ~ part_nb, data = nc, listw = poids)   # modèle à retard spatial
summary(sar)
impacts(sar, listw = poids)                          # effets direct, indirect et total
```

La fonction `moran.test()` teste l'absence d'autocorrélation spatiale du taux. `lm.RStests()` applique, aux résidus des MCO, les tests du multiplicateur de Lagrange de Anselin : `RSerr` et `RSlag` pour une erreur ou un retard spatial, et leurs versions robustes `adjRSerr` et `adjRSlag`. Si le test robuste du retard est significatif et pas celui de l'erreur, on retient le modèle SAR. Les anciennes versions du paquet utilisaient le nom `lm.LMtests`.

D.14 Récapitulatif des paquets par chapitre

Le tableau D.4 dresse l'inventaire des principaux paquets R associés aux méthodes du manuel, avec le chapitre où chacune est présentée.

Table D.4. Principaux paquets R du manuel

Méthode	Paquets	Chapitre(s)
MCO et diagnostics	`lmtest`, `sandwich`, `car`	5
Panel (effets fixes, aléatoires)	`plm`, `fixest`	6
Variables instrumentales	`AER` (`ivreg`)	6, 13
Panels dynamiques (GMM)	`plm` (`pgmm`)	6
Sélection de Heckman	`sampleSelection`	6, 8
Séries temporelles (ADF, ARIMA)	`tseries`, `forecast`	7
GARCH	`fGarch`, `rugarch`	7
VAR, cointégration, VECM	`vars`, `urca`	7
Logit, probit, modèles linéaires généralisés	`stats` (`glm`), `marginaleffects`	8
Tobit	`AER` (`tobit`)	8
Comptage (Poisson, binomiale négative)	`MASS` (`glm.nb`), `AER` (`dispersiontest`)	8
Durées de survie	`survival`	8

Méthode	Paquets	Chapitre(s)
Régression quantile	`quantreg`	8
Régression non paramétrique	`KernSmooth`, `np`, `mgcv`	9
Régression bayésienne (MCMC)	`MCMCpack`	9
Différences en différences	`did`	15
Régression par discontinuité	`rdrobust`, `rddensity`	14
Double machine learning	`DoubleML`, `mlr3`	21
Forêts causales	`grf`	21
Données spatiales, économétrie spatiale	`sf`, `terra`, `spdep`, `spatialreg`	18
Grands volumes de données	`data.table`, `exactextractr`	18
Arbres de décision	`rpart`	19
Forêts aléatoires	`randomForest`, `ranger`	19
Boosting	`xgboost`, `gbm`	19
Machines à vecteurs de support	`e1071`, `kernlab`	19
Validation croisée, courbes ROC	`caret`, `pROC`	19
Interprétabilité (valeurs de Shapley)	`shapviz`	19
Régularisation (ridge, LASSO, elastic net)	`glmnet`	19, 20
Analyse en composantes principales, classification	`FactoMineR`, `factoextra`	20
Réseaux de neurones multicouches	`nnet`, `keras`	22
Réseaux récurrents (LSTM, GRU)	`keras`, `torch`	22
Reproductibilité	`renv`, `rmarkdown`	chapitre 24

D.15 Erreurs fréquentes et bonnes pratiques

Cinq erreurs fréquentes sous R

1. *Objet introuvable* (`Error: object 'x' not found`). L'objet n'existe pas encore en mémoire, souvent parce qu'une ligne du script n'a pas été exécutée, ou parce que son nom est mal orthographié : R distingue majuscules et minuscules. On consulte le panneau Environnement et on exécute le script depuis le début avec `Ctrl+Alt+R`.
2. *Paquet non installé* (`there is no package called 'x'`). On installe d'abord le paquet avec `install.packages("x")`, puis on appelle `library(x)`.
3. *Variable qualitative codée par des nombres.* Un code numérique (par exemple 1 à 26 pour 26 provinces) est traité par `lm()` comme une quantité, ce qui donne des résultats absurdes. On le convertit avec `factor()`, par exemple `donnees$province <- factor(donnees$province)`. Une colonne de texte est en revanche convertie automatiquement.
4. *Valeurs manquantes.* Les statistiques descriptives comme `mean()` renvoient `NA` s'il manque une valeur, sauf si l'on ajoute `na.rm = TRUE`. Les fonctions d'estimation comme `lm()` éliminent au contraire sans avertissement les lignes incomplètes (réglage `na.omit`) : l'échantillon rétrécit sans bruit. On compare donc toujours `nobs(modele)` et `nrow(donnees)`.
5. *Chemin de fichier incorrect.* Sous Windows, les barres obliques inversées d'un chemin doivent être doublées (`"C:\\donnees\\fichier.csv"`) ou remplacées par des barres obliques simples (`"C:/donnees/fichier.csv"`). Un projet RStudio évite le problème en travaillant avec des chemins relatifs.

Bonnes pratiques de programmation sous R

- *Fixer la graine aléatoire* avant toute opération stochastique (découpage apprentissage et test, validation croisée, *bootstrap*, MCMC), par exemple avec `set.seed(2025)` : les résultats sont alors identiques à chaque exécution.
- *Structurer le script* en sections : chargement des paquets, importation, préparation, estimation, graphiques. Les commentaires (lignes qui commencent par `#`) documentent les choix méthodologiques.
- *Figer les versions* des paquets avec `renv` : `renv::snapshot()` écrit l'état de l'environnement dans un fichier `renv.lock`, que d'autres chercheurs restaurent avec `renv::restore()`.
- *Noter la configuration* avec `sessionInfo()`, qui liste la version de R et celle des paquets chargés.
- *Exporter les résultats* sans recopie manuelle : `modelsummary` ou `stargazer` produisent des tableaux de régression directement en LaTeX.

Ces habitudes reviennent à chaque étape du flux de travail de la figure D.1 : importer et inspecter, décrire, estimer et diagnostiquer, corriger si nécessaire, puis communiquer de façon reproductible. Un diagnostic défavorable renvoie à l'estimation, et non à la rédaction.

Points clés à retenir

- R est libre, gratuit et extensible par des milliers de paquets ; RStudio en facilite l'usage mais ne le remplace pas.
- Tout est un objet ; le data frame, tableau de variables en colonnes et d'observations en lignes, est l'objet central de l'économétrie.
- Une régression s'écrit `lm(y ~ x1 + x2, data = donnees)` ; le résultat est un objet que l'on lit (`summary()`, `confint()`), contrôle (`bptest()`, `vif()`) et corrige (`vcovHC()`).
- Les tests de diagnostic orientent le choix de la méthode : écarts-types robustes en cas d'hétéroscédasticité ou d'autocorrélation, effets fixes pour un panel, variables instrumentales pour un régresseur endogène, binomiale négative pour un comptage surdispersé.
- En séries temporelles, on teste la racine unitaire avant de régresser des niveaux, puis on recourt à la cointégration si les séries sont intégrées.
- Un résultat d'apprentissage automatique se juge sur un échantillon de test, avec une graine fixée ; sur de petits échantillons, un modèle flexible ne bat pas forcément un modèle simple.
- La reproductibilité exige des scripts commentés, une graine fixée et la liste des versions de paquets.

Références

Bivand, R. S., Pebesma, E., & Gómez-Rubio, V. (2013). *Applied spatial data analysis with R* (2e éd.). Springer. https://doi.org/10.1007/978-1-4614-7618-4

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1

Bollerslev, T., & Ghysels, E. (1996). Periodic autoregressive conditional heteroscedasticity. *Journal of Business & Economic Statistics*, *14*(2), 139–151. https://doi.org/10.1080/07350015.1996.10524640

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Callaway, B., & Sant'Anna, P. H. C. (2021). Difference-in-differences with multiple time periods. *Journal of Econometrics*, *225*(2), 200–230. https://doi.org/10.1016/j.jeconom.2020.12.001

Calonico, S., Cattaneo, M. D., & Titiunik, R. (2014). Robust nonparametric confidence intervals for regression-discontinuity designs. *Econometrica*, *82*(6), 2295–2326. https://doi.org/10.3982/ECTA11757

Cameron, A. C., & Trivedi, P. K. (1990). Regression-based tests for overdispersion in the Poisson model. *Journal of Econometrics*, *46*(3), 347–364. https://doi.org/10.1016/0304-4076(90)90014-K

Chambers, J. M. (2008). *Software for data analysis: Programming with R*. Springer. https://doi.org/10.1007/978-0-387-75936-4

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Dans *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (p. 785–794). ACM. https://doi.org/10.1145/2939672.2939785

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W., & Robins, J. (2018). Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, *21*(1), C1–C68. https://doi.org/10.1111/ectj.12097

Cobb, G. W. (1978). The problem of the Nile: Conditional solution to a changepoint problem. *Biometrika*, *65*(2), 243–251. https://doi.org/10.1093/biomet/65.2.243

Croissant, Y., & Millo, G. (2008). Panel data econometrics in R: The plm package. *Journal of Statistical Software*, *27*(2), 1–43. https://doi.org/10.18637/jss.v027.i02

Deb, P., & Trivedi, P. K. (1997). Demand for medical care by the elderly: A finite mixture approach. *Journal of Applied Econometrics*, *12*(3), 313–336.

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, *33*(1), 1–22. https://doi.org/10.18637/jss.v033.i01

Gerfin, M. (1996). Parametric and semi-parametric estimation of the binary response model of labour market participation. *Journal of Applied Econometrics*, *11*(3), 321–339.

Granger, C. W. J., & Newbold, P. (1974). Spurious regressions in econometrics. *Journal of Econometrics*, *2*(2), 111–120. https://doi.org/10.1016/0304-4076(74)90034-7

Greene, W. H. (2003). *Econometric analysis* (5e éd.). Prentice Hall.

Hanck, C., Arnold, M., Gerber, A., & Schmelzer, M. (2018). *Introduction to econometrics with R*. Université de Duisbourg-Essen. https://www.econometrics-with-r.org

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, *27*(3), 1–22. https://doi.org/10.18637/jss.v027.i03

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, *5*(3), 299–314. https://doi.org/10.1080/10618600.1996.10474713

Johansen, S., & Juselius, K. (1990). Maximum likelihood estimation and inference on cointegration, with applications to the demand for money. *Oxford Bulletin of Economics and Statistics*, *52*(2), 169–210. https://doi.org/10.1111/j.1468-0084.1990.mp52002003.x

Kleiber, C., & Zeileis, A. (2008). *Applied econometrics with R*. Springer. https://doi.org/10.1007/978-0-387-77318-6

Koenker, R., & Bassett, G. (1978). Regression quantiles. *Econometrica*, *46*(1), 33–50. https://doi.org/10.2307/1913643

Koenker, R., & Bassett, G. (1982). Robust tests for heteroscedasticity based on regression quantiles. *Econometrica*, *50*(1), 43–61. https://doi.org/10.2307/1912528

Long, J. S., & Ervin, L. H. (2000). Using heteroscedasticity consistent standard errors in the linear regression model. *The American Statistician*, *54*(3), 217–224. https://doi.org/10.1080/00031305.2000.10474549

McCrary, J. (2008). Manipulation of the running variable in the regression discontinuity design: A density test. *Journal of Econometrics*, *142*(2), 698–714. https://doi.org/10.1016/j.jeconom.2007.05.005

Mroz, T. A. (1987). The sensitivity of an empirical model of married women's hours of work to economic and statistical assumptions. *Econometrica*, *55*(4), 765–799. https://doi.org/10.2307/1911029

Newey, W. K., & West, K. D. (1987). A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. *Econometrica*, *55*(3), 703–708. https://doi.org/10.2307/1913610

Pebesma, E., & Bivand, R. (2023). *Spatial data science: With applications in R*. Chapman and Hall/CRC. https://doi.org/10.1201/9780429459016

R Core Team. (2024). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C., & Müller, M. (2011). pROC: An open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, *12*, Article 77. https://doi.org/10.1186/1471-2105-12-77

Stock, J. H., & Watson, M. W. (2007). *Introduction to econometrics* (2e éd.). Pearson Addison-Wesley.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, *58*(1), 267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

Wickham, H., Çetinkaya-Rundel, M., & Grolemund, G. (2023). *R for data science* (2e éd.). O'Reilly Media. https://r4ds.hadley.nz

Zeileis, A. (2004). Econometric computing with HC and HAC covariance matrix estimators. *Journal of Statistical Software*, *11*(10), 1–17. https://doi.org/10.18637/jss.v011.i10