ANNEXE

C

Guide de prise en main de JMulTi

Objectifs d'apprentissage

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

- préparer un fichier de données au format de JMulTi et l'importer ;
- enchaîner les étapes d'une analyse multivariée : racines unitaires, ordre du VAR, rang de cointégration, estimation du VECM, contrôle des résidus ;
- choisir la spécification déterministe d'un test de Johansen et en lire la sortie ;
- interpréter des fonctions de réponse impulsionnelle avec intervalles bootstrap, une décomposition de la variance et une prévision hors échantillon ;
- reproduire chaque étape avec R ou Gretl pour vérifier un résultat ;
- situer les modules ARIMA, GARCH et SVAR/SVEC de JMulTi.

JMulTi est un logiciel gratuit d'analyse économétrique des séries temporelles, écrit en Java par Markus Krätzig et diffusé avec le manuel de Lütkepohl et Krätzig (2004), dont il met en œuvre les procédures. Il s'utilise par menus et par modules : une analyse préliminaire (racines unitaires, cointégration), puis les modules VAR, VECM, ARIMA, GARCH et régression à transition lisse. Le tableau C.1 résume ce que chacun contient.

JMulTi, Gretl et R : trois outils complémentaires

JMulTi excelle dans la chaîne multivariée complète (tests avec ruptures, VECM, intervalles bootstrap des réponses impulsionnelles, modèles structurels) et guide l'analyste étape par étape. Gretl couvre le même terrain par menus ou par scripts et se prête à l'enseignement. R offre la plus grande souplesse mais demande d'écrire le code. Chaque section de cette annexe décrit donc la procédure de JMulTi, puis donne un équivalent court en R ou en Gretl. Les sorties numériques reproduites ici proviennent de ces équivalents ; JMulTi, qui applique les mêmes méthodes, doit livrer des valeurs identiques ou très proches (les critères d'information peuvent différer d'une constante selon la normalisation retenue).

Table C.1. Les modules de JMulTi et leurs procédures principales

Module	Procédures principales
`Initial Analysis`	Graphiques, statistiques descriptives, autocorrélations, spectre ; tests de racine unitaire ADF, KPSS, Schmidt-Phillips, racines saisonnières (HEGY) et tests avec rupture de niveau ; tests de cointégration de Johansen (trace) et de Saikkonen et Lütkepohl.
`VAR`	Spécification et critères d'information, estimation avec restrictions, analyse des résidus, stabilité (tests de Chow, CUSUM, coefficients récursifs), causalité, réponses impulsionnelles, décomposition de la variance, prévision, VAR structurels (modèle AB, Blanchard-Quah).
`VECM`	Estimation (procédure de Johansen, procédure en deux étapes), restrictions sur les vecteurs de cointégration, analyse des résidus, réponses impulsionnelles, décomposition de la variance, prévision, modèles structurels (SVEC).
`ARIMA`	Choix des retards par la procédure de Hannan-Rissanen, estimation avec régresseurs fixes, analyse des résidus, prévision.
`GARCH`	Modèles ARCH, GARCH et GARCH à seuil (T-GARCH), erreurs gaussiennes, GED ou de Student ; GARCH(1,1) multivarié.
`STR`	Régression à transition lisse (hors du champ de ce manuel).

Source : aide en ligne de JMulTi et liste de ses fonctions.

C.1 Installation et démarrage

Obtention. Le logiciel et son aide se téléchargent depuis la page du projet hébergée par l'Université Humboldt de Berlin (https://wiwi.hu-berlin.de/de/professuren/vwl/oe/software/jmulti). L'ancien site https://www.jmulti.de renvoie aux mêmes archives. Il suffit de décompresser l'archive dans un dossier de travail : il n'y a pas d'installation au sens habituel.

Java. JMulTi s'exécute sur la machine virtuelle Java, qu'il faut donc avoir installée (distribution libre, par exemple https://adoptium.net). Le logiciel étant ancien, le fichier d'aide de l'archive précise la version de Java recommandée. On le lance en suivant les indications de l'archive (double clic sur le fichier `.jar` ou, dans un terminal, `java -jar` suivi du nom de ce fichier).

Première fenêtre. La fenêtre principale comporte une barre de menus, une liste des variables chargées en mémoire, une zone d'affichage des résultats et, pour chaque analyse, un panneau de spécification organisé en onglets. L'interface est en anglais ; les intitulés des menus et des boutons sont donc cités dans cette langue, en caractères à `chasse fixe`, tels qu'ils figurent dans l'aide du logiciel. La disposition exacte des panneaux varie un peu d'une version à l'autre : les figures de cette annexe en donnent des schémas simplifiés, non des captures.

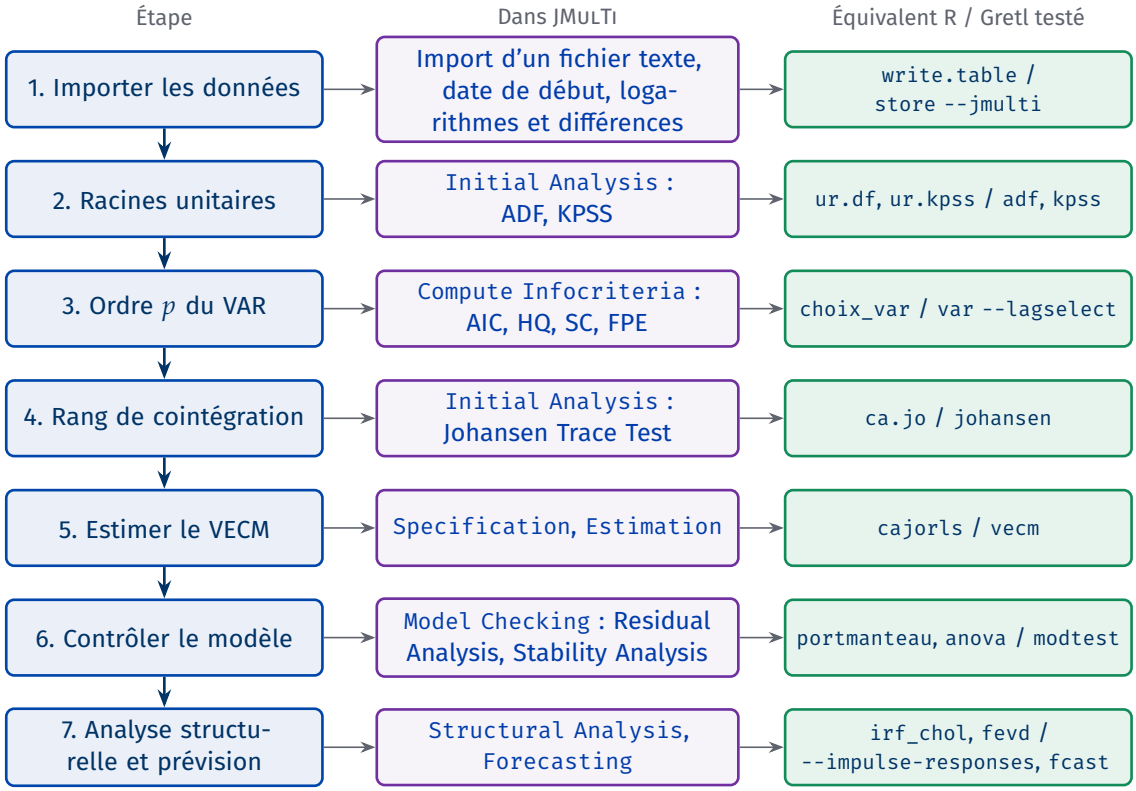


Figure C.1. Flux de travail d'une analyse de cointégration. On lit chaque ligne de gauche à droite : l'étape, l'endroit où JMulTi la réalise, l'instruction R ou Gretl qui permet de vérifier le résultat. Les flèches verticales donnent l'ordre à respecter : le test de cointégration suppose l'ordre p du VAR déjà choisi.

C.2 Choisir la démarche

Le modèle à estimer dépend de l'ordre d'intégration des séries, non de la préférence de l'analyste. Un VAR en niveaux ne convient qu'à des séries stationnaires. Si les séries sont intégrées d'ordre un ($I(1)$) et ne sont pas cointégrées, on travaille sur leurs différences premières ; si elles sont cointégrées, le modèle à correction d'erreur (VECM) conserve l'information de long terme que la différenciation détruirait. La figure C.2 résume ces choix ; les sections suivantes détaillent chaque case.

Attention : Un rang peut cacher une erreur de spécification

Le test de Johansen dépend de l'ordre p du VAR et des termes déterministes retenus. Un rang trouvé avec le mauvais p ou la mauvaise spécification n'a aucune valeur : c'est pourquoi l'ordre p se choisit avant le test et pourquoi on examine plusieurs spécifications déterministes, comme on le fait à la section C.6.

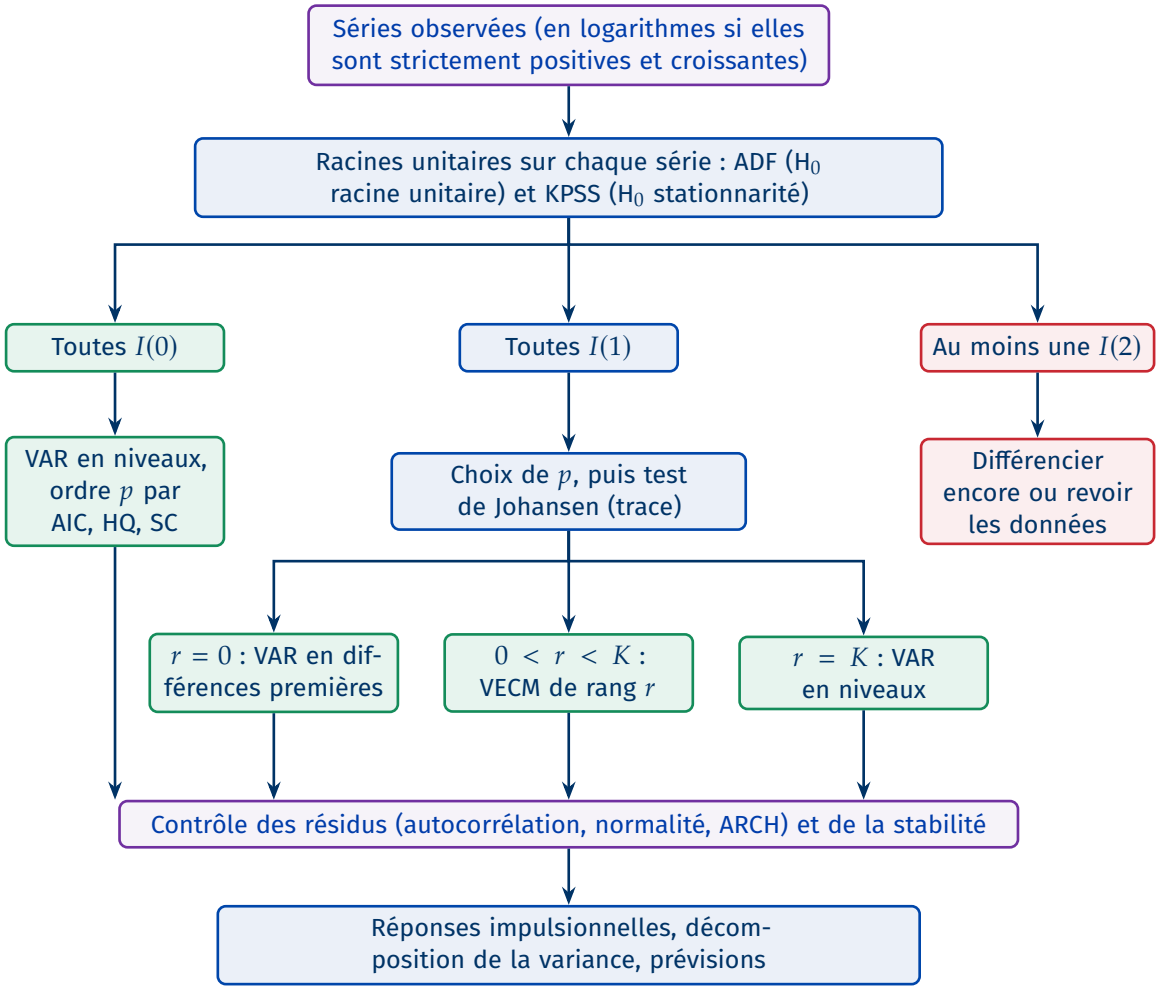


FIGURE C.2. Arbre de décision pour une analyse multivariée. Les cases bleues sont des tests ou des étapes de calcul, les cases vertes des modèles retenus, les cases rouges des situations à traiter avant d'aller plus loin. K est le nombre de séries et r le rang de cointégration. Le résultat des tests de racine unitaire n'est pas toujours net : lorsque ADF et KPSS se contredisent, on garde la décision la plus prudente et l'on vérifie la sensibilité des résultats.

C.3 Importation des données

Le fil conducteur de cette annexe est un cas classique de la littérature, la demande de monnaie au Danemark. Johansen et Juselius (1990) étudient quatre séries trimestrielles de 1974T1 à 1987T3 (55 observations) : le logarithme de la monnaie réelle M2 (`LRM`), le logarithme du revenu réel (`LRY`), le taux des obligations (`IBO`) et le taux des dépôts bancaires (`IDE`), ces deux taux étant exprimés en décimales (0,10 pour 10 %). Les données sont livrées avec le package `urca` de R et avec GRETL (fichier `denmark.gdt`). Le cas se transpose à une économie africaine : une demande de monnaie reliant masse monétaire réelle, activité et taux d'intérêt, ou un taux de change reliant prix intérieurs et prix extérieurs.

Le format de fichier. JMULTI lit son format texte natif, qui se compose d'un commentaire placé entre `/*` et `*/`, de la date de début entre chevrons (par exemple `<1974 Q1>` pour une série trimestrielle), d'une ligne de noms de variables, puis des observations, une ligne par date. Il sait aussi lire des fichiers Excel et MATLAB, mais le fichier texte reste le plus sûr. Par prudence, on évite les accents dans le fichier.

Code C.1. Préparer le fichier de données au format de JMulTi

```
library(urca)
data(denmark)
d <- denmark[, c("LRM", "LRY", "IBO", "IDE")]
x <- ts(d, start = c(1974, 1), frequency = 4)       # série trimestrielle, T = 55
cat("/* Donnees danoises (Johansen et Juselius, 1990) */\n<1974 Q1>\n",
    paste(colnames(d), collapse = " "), "\n", file = "denmark.dat", sep = "")
write.table(format(d, digits = 6), "denmark.dat", append = TRUE,
            quote = FALSE, row.names = FALSE, col.names = FALSE)
cat(head(readLines("denmark.dat"), 5), sep = "\n")
```

```
/* Donnees danoises (Johansen et Juselius, 1990) */
<1974 Q1>
LRM LRY IBO IDE
11.6326 5.90366 0.154736 0.0940000
11.6042 5.87382 0.177991 0.0955000
```

Le script lit les quatre séries, les range en objet trimestriel, puis écrit le fichier `denmark.dat` avec les trois lignes d'en-tête attendues. La même opération s'obtient dans Gretl avec la commande suivante (option `--jmulti`) :

Code C.2. Export au format de JMulTi avec Gretl

```
open denmark.gdt
store "denmark_gretl.dat" LRM LRY IBO IDE --jmulti
```

Procédure dans JMulTi.

1. Lancer le programme et importer le fichier avec la commande d'importation de données de la fenêtre principale. Les variables apparaissent dans la liste des variables en mémoire.
2. Si le fichier ne contient pas la date de début, ouvrir la boîte de sélection de la période, saisir la date de début et choisir la fréquence (trimestrielle ici).
3. Pour créer des variables transformées, faire un clic droit sur la variable dans la liste et choisir la transformation (logarithme, différence première).
4. Choisir le module voulu, puis les variables à y inclure.

La figure C.3 montre les données. Les deux panneaux de gauche donnent l'allure typique de séries $I(1)$: pas de retour à une moyenne fixe, et une dérive commune de la monnaie réelle et du revenu réel. Le troisième panneau trace la combinaison $\hat{\beta}'\mathbf{y}_t$ estimée à la section C.7, qui oscille autour d'un niveau stable alors que chaque série prise isolément dérive : c'est l'image de la cointégration.

Source : package `urca` (jeu `denmark`), tiré de Johansen et Juselius (1990) ; calculs avec R.

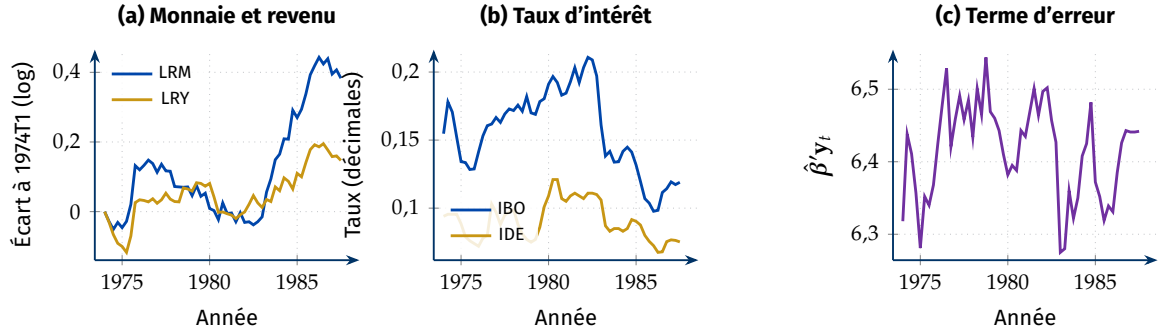


FIGURE C.3. Données danoises de Johansen et Juselius (1990), 55 trimestres. (a) Logarithmes de la monnaie réelle (`LRM`) et du revenu réel (`LRY`), mesurés en écart à leur valeur de 1974T1 pour que les deux courbes se lisent sur la même échelle. (b) Taux des obligations (`IBO`) et des dépôts (`IDE`). (c) Combinaison de cointégration estimée, LRM − 0,976 LRY + 5,409 IBO − 4,162 IDE (section C.7).

C.4 Tests de racine unitaire

Dans le module `Initial Analysis`, la rubrique des tests de racine unitaire propose le test de Dickey-Fuller augmenté (ADF), le test KPSS, le test de Schmidt-Phillips, les tests de racines saisonnières (HEGY pour données trimestrielles et mensuelles) et des tests qui admettent une rupture de niveau. Le test de Phillips-Perron n'y figure pas ; si l'on y tient, on l'obtient avec R (fonction `ur.pp` du package `urca`).

Principe. Le test ADF estime la régression

$$\Delta y_t = c + \gamma\, y_{t-1} + \sum_{i=1}^{k} \varphi_i\, \Delta y_{t-i} + e_t, \tag{C.1}$$

avec c remplacé par $c + \delta t$ si l'on ajoute une tendance, et teste $H_0 : \gamma = 0$ (racine unitaire) à l'aide de la statistique $\tau = \hat{\gamma}/\widehat{\text{s.e.}}(\hat{\gamma})$, dont la loi n'est pas celle d'un t de Student : il faut les valeurs critiques de Dickey et Fuller (1979, 1981), reproduites dans les tables statistiques (annexe A du livre). Les retards Δy_{t-i} absorbent l'autocorrélation des résidus. Le test KPSS (Kwiatkowski et al., 1992) inverse les hypothèses : H_0 est la stationnarité autour d'une constante ou d'une tendance. Quand ADF ne rejette pas et que KPSS rejette, la série est $I(1)$; quand les deux tests se contredisent ou ne concluent pas, la prudence s'impose. C'est l'analyse dite confirmatoire.

Procédure dans JMulTi. Après avoir choisi les variables, on coche les termes déterministes du test (cases `nonzero mean`, `time trend`, `seasonal dummies`), on fixe le nombre de retards (`actual number of lags`) ou le maximum à explorer par les critères d'information (`max number of lags`), puis on lance le calcul. L'option `residual analysis` affiche les diagnostics de la régression. Une série sans tendance visible, comme un taux d'intérêt, se teste avec une constante seule ; une série qui croît se teste avec constante et tendance, faute de quoi le test perd sa puissance.

Code C.3. Tests ADF et KPSS sur les quatre séries, en niveau et en différence

```
adf_aic <- function(y, kmax = 4) {
  a <- ur.df(y, type = "drift", lags = kmax, selectlags = "AIC")
  k <- nrow(coef(a@testreg)) - 2                    # nombre de retards retenus
  b <- ur.df(y, type = "drift", lags = k, selectlags = "Fixed")
  c(k = k, tau = b@teststat[1, 1], vc5 = b@cval[1, "5pct"])
}
res <- t(sapply(colnames(x), function(v) {
  n <- adf_aic(x[, v]); d1 <- adf_aic(diff(x[, v]))
  c(niv = n, dif = d1,
    kpss_niv = ur.kpss(x[, v], type = "mu", lags = "short")@teststat,
    kpss_dif = ur.kpss(diff(x[, v]), type = "mu", lags = "short")@teststat)
}))
cols <- c("niv.k", "niv.tau", "dif.k", "dif.tau", "kpss_niv", "kpss_dif")
print(round(res[, cols], 3))
vk <- ur.kpss(x[, 1], type = "mu")@cval[1, "5pct"]
cat("Valeurs critiques à 5 % : ADF", round(res[1, "niv.vc5"], 2),
    "; KPSS", vk, "\n")
```

```
    niv.k niv.tau dif.k dif.tau kpss_niv kpss_dif
LRM     4  -1.702     4  -2.772    0.781    0.204
LRY     1  -1.489     1  -4.811    0.879    0.067
IBO     1  -1.651     1  -4.388    0.489    0.141
IDE     1  -2.434     1  -4.767    0.247    0.071
Valeurs critiques à 5 % : ADF -2.89 ; KPSS 0.463
```

La fonction `adf_aic` procède en deux temps : elle choisit le nombre de retards k par le critère d'Akaike (au plus quatre), puis relance le test avec ce k fixé, afin d'utiliser tout l'échantillon disponible. Dans Gretl, les mêmes statistiques s'obtiennent avec les lignes suivantes, qui reproduisent à la décimale près les valeurs de `LRM` (extraits de la sortie ensuite) :

Code C.4. ADF et KPSS dans Gretl

```
open denmark.gdt
adf 4 LRM --c          # ADF avec constante et 4 retards
kpss 3 LRM             # KPSS avec 3 retards de troncature
```

```
  test statistic: tau_c(1) = -1.70189
  asymptotic p-value 0.4303
Test statistic = 0.780552
```

Lecture. Aucun test ADF en niveau ne rejette la racine unitaire (toutes les statistiques dépassent $-2{,}89$). Le test KPSS rejette la stationnarité pour `LRM`, `LRY` et `IBO` (0,78, 0,88 et 0,49 contre 0,463), ce qui confirme qu'elles sont $I(1)$. Pour `IDE` (0,25) les deux tests sont non concluants, signe d'une faible puissance sur un échantillon de 55 points : on la traite comme $I(1)$ par prudence. En différence première, l'ADF rejette la racine unitaire pour `LRY`, `IBO` et

Table C.2. Tests de racine unitaire sur les données danoises (constante, k retards choisis par l'AIC)

Série	k	ADF niveau	k	ADF diff.	KPSS niveau	KPSS diff.
LRM	4	−1,70	4	−2,77	0,78	0,20
LRY	1	−1,49	1	−4,81	0,88	0,07
IBO	1	−1,65	1	−4,39	0,49	0,14
IDE	1	−2,43	1	−4,77	0,25	0,07

Source : calculs avec R (code C.3). Valeurs critiques à 5 % : ADF −2,89 ; KPSS 0,463.

IDE ; pour LRM, la statistique −2,77 ne franchit pas le seuil de 5 % (elle franchit celui de 10 %, soit −2,58), mais le KPSS (0,20) ne rejette pas la stationnarité. Les quatre séries sont donc retenues comme $I(1)$, ce qui ouvre la voie à l'étude de la cointégration.

C.5 Choix de l'ordre du VAR

Le VAR en niveaux d'ordre p s'écrit $\mathbf{y}_t = \boldsymbol{\nu} + \mathbf{A}_1\mathbf{y}_{t-1} + \cdots + \mathbf{A}_p\mathbf{y}_{t-p} + \mathbf{u}_t$, avec K variables. On choisit p en minimisant un critère d'information calculé sur le même échantillon pour tous les ordres candidats. Si $\tilde{\boldsymbol{\Sigma}}_u(m)$ est l'estimateur de la matrice de covariance des résidus d'un VAR d'ordre m estimé sur T observations et $n = Km + 1$ le nombre de régresseurs de chaque équation, les critères s'écrivent

$$\mathrm{AIC}(m) = \ln|\tilde{\boldsymbol{\Sigma}}_u(m)| + \frac{2Kn}{T}, \qquad \mathrm{HQ}(m) = \ln|\tilde{\boldsymbol{\Sigma}}_u(m)| + \frac{2\ln\ln T\; Kn}{T}, \tag{C.2}$$

$$\mathrm{SC}(m) = \ln|\tilde{\boldsymbol{\Sigma}}_u(m)| + \frac{\ln T\; Kn}{T}, \qquad \mathrm{FPE}(m) = \left(\frac{T+n}{T-n}\right)^K \left|\tilde{\boldsymbol{\Sigma}}_u(m)\right|. \tag{C.3}$$

Ces expressions ne diffèrent de celles de Lütkepohl (2005) que par une constante indépendante de m (le terme $2K/T$ lié à l'intercept), ce qui ne change pas l'ordre retenu. Le critère de Schwarz (SC) pénalise le plus la complexité et retient les ordres les plus petits ; celui d'Akaike (AIC) retient souvent un ordre plus élevé.

Procédure dans JMulTi. Dans le module VECM, le menu Specification → Specify VEC Model offre le bouton Compute Infocriteria, qui calcule les critères pour tous les ordres jusqu'au maximum demandé. Le module VAR propose le même calcul. On inscrit ensuite l'ordre retenu dans la spécification du modèle.

Code C.5. Critères d'information du VAR en niveaux (ordres 1 à 4)

```
choix_var <- function(x, pmax = 4) {
  x <- as.matrix(x); N <- nrow(x); K <- ncol(x); T <- N - pmax
  res <- t(sapply(1:pmax, function(m) {
    L <- lapply(1:m, function(i) x[(pmax + 1 - i):(N - i), ])
    Z <- cbind(1, do.call(cbind, L))
    U <- resid(lm.fit(Z, x[(pmax + 1):N, ]))         # résidus du VAR(m)
    ld <- log(det(crossprod(U) / T))
    n <- ncol(Z)                                      # régresseurs par équation
    c(AIC = ld + 2 * K * n / T, HQ = ld + 2 * log(log(T)) * K * n / T,
      SC = ld + log(T) * K * n / T, FPE = ((T + n) / (T - n))^K * exp(ld))
  }))
  rownames(res) <- 1:pmax
  res
}
ic <- choix_var(x)
print(cbind(round(ic[, 1:3], 3), FPE = signif(ic[, 4] * 1e15, 4)))  # FPE : 1e-15
apply(ic, 2, which.min)                               # ordre retenu par critère
```

```
      AIC      HQ      SC    FPE
1 -34.456 -34.166 -33.698 1.0900
2 -34.712 -34.191 -33.349 0.8533
3 -34.504 -33.751 -32.534 1.0840
4 -34.302 -33.317 -31.726 1.4090
AIC  HQ  SC FPE
  2   2   1   2
```

L'équivalent Gretl est la commande `var 4 LRM LRY IBO IDE --lagselect` ; elle retient elle aussi $p = 2$ pour l'AIC et le critère de Hannan-Quinn, $p = 1$ pour le critère de Schwarz. Les valeurs affichées diffèrent des nôtres car Gretl normalise les critères autrement, mais seul leur classement compte.

Table C.3. Critères d'information du VAR en niveaux, données danoises ($T = 51$)

Ordre m	AIC	HQ	SC	FPE ($\times 10^{-15}$)
1	−34,456	−34,166	**−33,698**	1,0900
2	**−34,712**	**−34,191**	−33,349	**0,8533**
3	−34,504	−33,751	−32,534	1,0840
4	−34,302	−33,317	−31,726	1,4090

Source : calculs avec R (code C.5). Le minimum de chaque critère est en gras.

Trois critères sur quatre choisissent $p = 2$, valeur que nous retenons : elle suffit à absorber l'autocorrélation (section C.10) sans consommer trop de degrés de liberté. Dans la suite, le VAR en niveaux est donc d'ordre 2, et le VECM comporte une différence retardée.

C.6 Test de cointégration de Johansen

Principe. Quand les séries sont $I(1)$, certaines combinaisons linéaires peuvent être stationnaires : on dit alors que les séries sont cointégrées. Le VAR d'ordre p en niveaux se réécrit sous forme à correction d'erreur,

$$\Delta \mathbf{y}_t = \mathbf{\Pi}\, \mathbf{y}_{t-1} + \sum_{i=1}^{p-1} \mathbf{\Gamma}_i\, \Delta \mathbf{y}_{t-i} + \boldsymbol{\mu}_t + \mathbf{u}_t, \qquad \mathbf{\Pi} = \boldsymbol{\alpha}\boldsymbol{\beta}', \tag{C.4}$$

où $\boldsymbol{\mu}_t$ regroupe les termes déterministes. Le rang r de la matrice $\mathbf{\Pi}$ ($K \times K$) est le nombre de relations de cointégration : $\boldsymbol{\beta}$ ($K \times r$) contient les vecteurs de cointégration, qui définissent les relations de long terme $\boldsymbol{\beta}'\mathbf{y}_t$, et $\boldsymbol{\alpha}$ ($K \times r$) les vitesses d'ajustement vers l'équilibre. Si $r = 0$, il n'y a pas de cointégration ; si $r = K$, toutes les séries sont stationnaires.

Johansen estime les valeurs propres $\hat{\lambda}_1 > \cdots > \hat{\lambda}_K$ d'un problème de corrélations canoniques, et teste $H_0 : \operatorname{rang}(\mathbf{\Pi}) \le r_0$ à l'aide de la statistique de la trace

$$\mathrm{LR}_{\mathrm{tr}}(r_0) = -T \sum_{i=r_0+1}^{K} \ln\bigl(1 - \hat{\lambda}_i\bigr), \tag{C.5}$$

où T est le nombre d'observations effectivement utilisées. La statistique de la valeur propre maximale, $-T \ln(1 - \hat{\lambda}_{r_0+1})$, teste $r = r_0$ contre $r = r_0 + 1$ (chapitre 7 du livre). On teste successivement $r_0 = 0, 1, \ldots$ et l'on retient le premier r_0 non rejeté.

Procédure dans JMulTi. Le test se trouve dans le module `Initial Analysis`, rubrique des tests de cointégration (`Johansen Trace Test`). On y sélectionne les variables endogènes et, si besoin, des variables déterministes (rupture de niveau, impulsion), puis on règle :

- `actual number of lags` : l'ordre p du VAR en niveaux (ici 2), à ne pas confondre avec le nombre de différences retardées, qui vaut $p - 1$;
- `max number of lags` : l'ordre maximal exploré par les critères d'information ;
- le terme déterministe : `nonzero mean`, `constant and trend` ou `orthogonal trend`, auquel s'ajoutent éventuellement des indicatrices saisonnières centrées (`seasonal dummies`).

La sortie donne, pour chaque rang testé r_0, la statistique de la trace, la valeur critique et la probabilité critique. D'après l'aide, les valeurs critiques proviennent d'approximations par surfaces de réponse. JMulTi propose aussi le test de Saikkonen et Lütkepohl, qui retire d'abord les termes déterministes par moindres carrés généralisés avant d'appliquer la procédure de Johansen aux séries corrigées ; il tolère des ruptures de niveau. L'aide ne documente que le test de la trace : pour la valeur propre maximale, on s'adresse à R ou Gretl.

Les trois spécifications déterministes. Le tableau C.4 associe chaque option de JMulTi au modèle correspondant et aux options de Gretl et `urca` ; nous l'avons vérifié numériquement en comparant les statistiques de R et de Gretl. Gretl propose deux cas de plus (aucun terme déterministe, tendance libre), que JMulTi ne propose pas.

Table C.4. Spécifications déterministes du test de Johansen

JMulTi	Interprétation	Gretl	urca (ecdet)
`nonzero mean`	Constante restreinte à la relation de cointégration ; pas de tendance linéaire dans les niveaux.	`--rc`	`"const"`
`constant and trend`	Constante libre, tendance linéaire restreinte à la relation de cointégration.	`--crt`	`"trend"`
`orthogonal trend`	Constante libre : tendance linéaire dans les niveaux, orthogonale aux relations de cointégration.	option par défaut	`"none"`

Code C.6. Statistiques de la trace pour les trois spécifications, VAR(2)

```
# Statistiques de la trace, VAR(2) en niveaux, trois cas déterministes
spec <- c("nonzero mean" = "const", "constant and trend" = "trend",
          "orthogonal trend" = "none")
for (s in names(spec)) {
  jo <- ca.jo(x, type = "trace", ecdet = spec[[s]], K = 2, spec = "transitory")
  cat("\n", s, "\n")
  print(round(rbind(trace = jo@teststat, vc5 = jo@cval[, "5pct"]), 2))
}
# Valeur propre maximale (lambda max) pour le dernier cas
jl <- ca.jo(x, type = "eigen", ecdet = "none", K = 2, spec = "transitory")
print(round(rbind(lambda = rev(jl@lambda[1:4]), lmax = jl@teststat,
                  vc5 = jl@cval[, "5pct"]), 3))
```

```
 nonzero mean
      r <= 3 | r <= 2 | r <= 1 | r = 0  |
trace     2.29     8.95    19.09    52.71
vc5       9.24    19.96    34.91    53.12

 constant and trend
      r <= 3 | r <= 2 | r <= 1 | r = 0  |
trace     2.13    10.75    26.64    59.51
vc5      12.25    25.32    42.44    62.99

 orthogonal trend
      r <= 3 | r <= 2 | r <= 1 | r = 0  |
trace     0.56     7.14    17.29    48.80
vc5       8.18    17.95    31.52    48.28
       r <= 3 | r <= 2 | r <= 1 | r = 0  |
lambda    0.010    0.117    0.174    0.448
lmax      0.556    6.589   10.145   31.514
vc5       8.180   14.900   21.070   27.140
```

Chaque tableau lit de droite à gauche : la dernière colonne teste $r_0 = 0$, l'avant-dernière $r_0 \leq 1$, etc. La fonction `ca.jo` ne renvoie pas de probabilité critique ; Gretl en donne, ce qui complète la lecture (extraits pour la constante libre, option par défaut) :

Code C.7. Test de Johansen dans Gretl

```
open denmark.gdt
johansen 2 LRM LRY IBO IDE --quiet   # constante libre ; ajouter --rc ou --crt pour les ↩
    autres cas
```

```
Rank Eigenvalue Trace test  p-value  Lmax test  p-value
   0    0.44821     48.804 [0.0389]     31.514 [0.0120]
   1    0.17421     17.290 [0.6274]     10.145 [0.7345]
   2    0.11690     7.1449 [0.5673]     6.5889 [0.5467]
   3   0.010436    0.55602 [0.4559]    0.55602 [0.4559]
```

Table C.5. Statistiques de la trace, données danoises, VAR(2), $T = 53$

	`nonzero mean`		`const. and trend`		`orth. trend`	
H_0	Trace	VC 5 %	Trace	VC 5 %	Trace	VC 5 %
$r_0 = 0$	52,71	53,12	59,51	62,99	48,80	48,28
$r_0 \leq 1$	19,09	34,91	26,64	42,44	17,29	31,52
$r_0 \leq 2$	8,95	19,96	10,75	25,32	7,14	17,95
$r_0 \leq 3$	2,29	9,24	2,13	12,25	0,56	8,18

Source : calculs avec R (code C.6). VC : valeur critique asymptotique à 5 %.

Lecture et décision. Les trois spécifications donnent des verdicts différents sur $r_0 = 0$. Avec `nonzero mean` (52,71 contre 53,12, probabilité critique 0,065) et `constant and trend` (59,51 contre 62,99, probabilité critique 0,109), l'absence de cointégration n'est pas rejetée à 5 % ; avec `orthogonal trend` (48,80 contre 48,28, probabilité critique 0,039), elle l'est, et $r_0 \leq 1$ n'est plus rejeté. Le test de la valeur propre maximale, calculé avec le code précédent, rejette $r = 0$ contre $r = 1$ (31,51 contre 27,14) et ne rejette pas $r = 1$ contre $r = 2$ (10,15 contre 21,07). Nous retenons $r = 1$ pour trois raisons : la valeur propre maximale, le verdict de la spécification `orthogonal trend`, qui est la plus naturelle pour des séries qui dérivent, et la théorie, qui prévoit une seule relation de demande de monnaie. La preuve reste modérée : la correction de Gretl pour petit échantillon porte la probabilité critique de la spécification `orthogonal trend` à 0,069. Un résultat aussi dépendant de la spécification se signale dans toute étude et se vérifie par des exercices de robustesse (autre ordre p, indicatrices saisonnières).

C.7 Estimation du VECM

Une fois r fixé, on estime le modèle (C.4). Les vecteurs de cointégration ne sont identifiés qu'à une normalisation près : par convention, on normalise le premier coefficient de chaque relation à 1, de sorte que la relation s'interprète comme une équation de long terme de la première variable (ici la monnaie réelle).

La figure C.4 schématise le parcours dans le module `VECM`. Les étapes sont les suivantes.

1. `Specification` → `Specify VEC Model` : choisir les variables, l'ordre des retards, le rang de cointégration r, puis, si l'on souhaite placer des termes déterministes dans la relation de cointégration, utiliser le panneau `Add Dets to EC Term`.
2. `Specification` → `Specify and Test Restrictions on Beta` (facultatif) : imposer des restrictions linéaires sur $\boldsymbol{\beta}$ et les tester. Le bouton `Delete Restrictions` les retire.
3. `Estimation` → `Estimate Model` : choisir la procédure d'estimation, soit celle de Johansen (régression de rang réduit), soit la procédure en deux étapes (S2S, ou deux étapes avec restrictions structurelles), puis lancer l'estimation. La sortie fournit $\hat{\boldsymbol{\alpha}}$, $\hat{\boldsymbol{\beta}}$, les matrices $\hat{\boldsymbol{\Gamma}}_i$ et les tests de significativité.
4. Tracer le terme à correction d'erreur $\hat{\boldsymbol{\beta}}'\mathbf{y}_t$ (rubrique `Plotting the EC term`) pour vérifier qu'il paraît stationnaire.

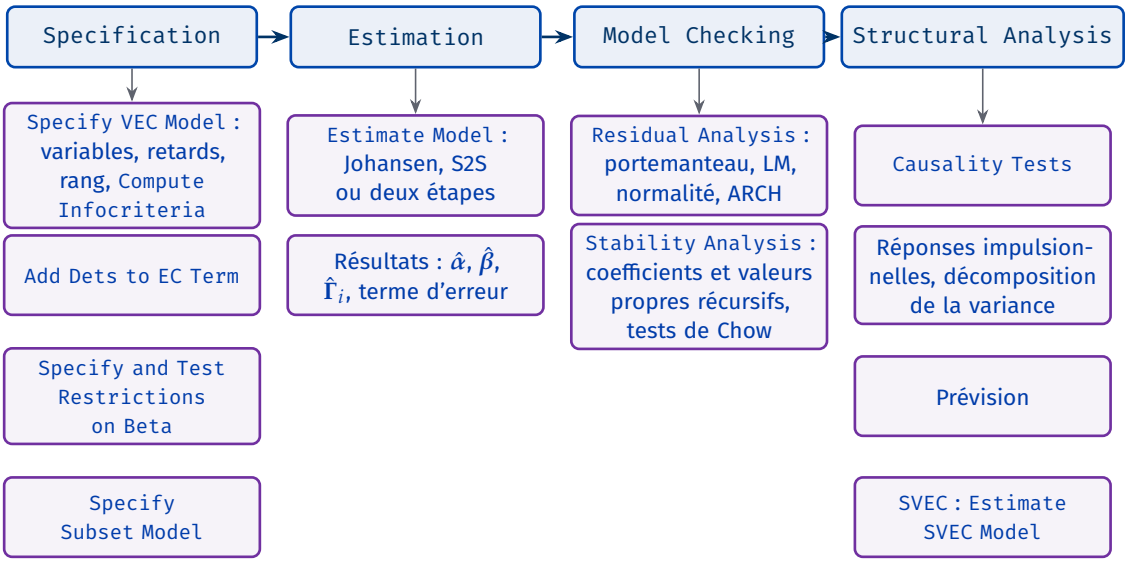


FIGURE C.4. Parcours dans le module `VECM`, d'après l'aide en ligne. Les quatre onglets se parcourent de gauche à droite ; sous chacun figurent les commandes principales. Schéma simplifié : la disposition exacte des panneaux varie selon la version du logiciel.

Code C.8. VECM de rang 1 et test de l'élasticité-revenu unitaire

```
jo  <- ca.jo(x, type = "trace", ecdet = "none", K = 2, spec = "transitory")
fit <- cajorls(jo, r = 1)                              # VECM de rang 1
print(round(rbind(beta = fit$beta[, 1], alpha = coef(fit$rlm)["ect1", ]), 4))
# Test LR : élasticité-revenu unitaire, beta = (1, -1, b3, b4)'
H  <- rbind(c(1, 0, 0), c(-1, 0, 0), c(0, 1, 0), c(0, 0, 1))
bt <- blrtest(jo, H = H, r = 1)
cat("LR =", round(bt@teststat, 4), "; p =", round(bt@pval[1], 4), "\n")
```

```
print(round(bt@V[, 1] / bt@V[1, 1], 4))              # beta restreint, normalisé
```

```
       LRM.l1  LRY.l1  IBO.l1  IDE.l1
beta   1.0000 -0.9757  5.4086 -4.1624
alpha -0.2815  0.0375 -0.0039  0.0200
LR = 0.0212 ; p = 0.8841
[1]  1.0000 -1.0000  5.3379 -4.1100
```

La fonction `cajorls` estime le VECM de rang $r = 1$ avec constante libre ; `blrtest` teste la restriction $\boldsymbol{\beta} = \mathbf{H}\boldsymbol{\varphi}$, ici $\beta_1 + \beta_2 = 0$ (élasticité-revenu unitaire). Les erreurs types de $\hat{\boldsymbol{\beta}}$ ne sont pas fournies par `urca` ; GRETL les donne, ainsi que le test de la restriction, avec les commandes suivantes :

Code C.9. VECM et test de restriction dans GRETL

```
open denmark.gdt
vecm 2 1 LRM LRY IBO IDE --quiet          # VECM(2), rang 1, constante libre
restrict --full                           # H0 : b[1,1] + b[1,2] = 0
  b[1,1] + b[1,2] = 0
end restrict
```

```
Unrestricted loglikelihood (lu) = 644.75421
Restricted loglikelihood (lr) = 644.74359
2 * (lu - lr) = 0.0212393
P(Chi-square(1) > 0.0212393) = 0.884129
```

TABLE C.6. VECM(2) de rang 1, données danoises (erreurs types entre parenthèses)

Variable	$\hat{\beta}$ (vecteur de cointégration)	$\hat{\alpha}$ (ajustement)	t de $\hat{\alpha}$
LRM	1 (normalisé)	−0,281 (0,075)	−3,74
LRY	−0,976 (0,144)	0,037 (0,065)	0,58
IBO	5,409 (0,551)	−0,004 (0,024)	−0,16
IDE	−4,162 (1,136)	0,020 (0,016)	1,27

Source : GRETL (code C.9) pour les erreurs types ; R (code C.8) donne les mêmes estimations.

Interprétation. La relation de long terme s'écrit LRM = 0,976 LRY − 5,409 IBO + 4,162 IDE (à une constante près), soit une demande de monnaie réelle croissante avec le revenu, décroissante avec le taux des obligations (coût d'opportunité) et croissante avec le taux des dépôts (rendement propre de la monnaie). Une hausse de un point de pourcentage du taux des obligations (0,01) abaisse la monnaie réelle de long terme d'environ 5,4 %. L'élasticité-revenu est proche de un et le test de la restriction ne la rejette pas (statistique 0,021, probabilité critique 0,88). Seule la monnaie corrige les écarts à l'équilibre : $\hat{\alpha}_{\text{LRM}} = -0{,}281$

(statistique t de −3,74) signifie qu'environ 28 % d'un écart se résorbe en un trimestre ; les autres coefficients d'ajustement ne sont pas significatifs, ce qui suggère, sans le démontrer, que revenu et taux sont faiblement exogènes. JMulTi teste les restrictions sur $\boldsymbol{\beta}$ par un test de Wald ; le test du rapport de vraisemblance ci-dessus est asymptotiquement équivalent, avec une loi χ^2 commune.

C.8 Fonctions de réponse impulsionnelle

Une fonction de réponse impulsionnelle (FRI) mesure l'effet, h périodes plus tard, d'un choc unitaire sur une variable. Le VECM estimé se réécrit comme un VAR en niveaux dont les matrices sont $\mathbf{A}_1 = \mathbf{I} + \boldsymbol{\Pi} + \boldsymbol{\Gamma}_1$ et $\mathbf{A}_2 = -\boldsymbol{\Gamma}_1$ pour $p = 2$. La représentation en moyenne mobile, $\mathbf{y}_t = \sum_{i \geq 0} \boldsymbol{\Phi}_i \mathbf{u}_{t-i}$, s'obtient par la récurrence

$$\boldsymbol{\Phi}_0 = \mathbf{I}_K, \qquad \boldsymbol{\Phi}_i = \sum_{j=1}^{\min(i,p)} \boldsymbol{\Phi}_{i-j}\, \mathbf{A}_j. \tag{C.6}$$

Les résidus $\mathbf{u}_t$ étant corrélés entre eux, on les orthogonalise par la décomposition de Cholesky $\tilde{\boldsymbol{\Sigma}}_u = \mathbf{P}\mathbf{P}'$ ($\mathbf{P}$ triangulaire inférieure) : les réponses orthogonalisées sont $\boldsymbol{\Theta}_i = \boldsymbol{\Phi}_i \mathbf{P}$, et l'élément (k, j) de $\boldsymbol{\Theta}_i$ est la réponse de la variable k, i périodes après un choc d'un écart-type sur la variable j.

Attention : L'ordre des variables compte

Avec la décomposition de Cholesky, la première variable réagit seule au premier choc, la deuxième aux deux premiers, etc. Changer l'ordre des variables change les réponses lorsque les résidus sont corrélés (ici, la corrélation entre les résidus de `LRM` et de `LRY` est de 0,59). L'ordre doit résulter d'un argument économique et se tester par un ordre alternatif.

Procédure dans JMulTi. Dans le menu `Structural Analysis`, la rubrique des réponses impulsionnelles demande le nombre de périodes, le type d'intervalle de confiance et, pour un intervalle bootstrap, le nombre de répétitions, le niveau de confiance et la graine du générateur aléatoire. Les intervalles asymptotiques sont proposés par défaut ; trois méthodes de bootstrap sont disponibles : l'intervalle percentile d'Efron, l'intervalle percentile de Hall et l'intervalle studentisé de Hall. On peut aussi demander les effets cumulés. Le premier intervalle prend les quantiles des réponses simulées ; les deux autres utilisent les quantiles de l'écart entre réponse simulée et réponse estimée.

Code C.10. Du VECM à la forme VAR en niveaux et réponses orthogonalisées

```
# Du VECM estimé au VAR(2) en niveaux : A1 = I + Pi + Gamma1, A2 = -Gamma1
estime <- function(y) {
  g <- ca.jo(y, type = "trace", ecdet = "none", K = 2, spec = "transitory")
  f <- cajorls(g, r = 1); cf <- coef(f$rlm)
  Pi <- cf["ect1", ] %o% f$beta[, 1]
  G <- t(cf[grep("dl1$", rownames(cf)), ]); U <- resid(f$rlm)
  list(A = list(diag(4) + Pi + G, -G), cst = cf["constant", ],
       Sigma = crossprod(U) / nrow(U), U = U)
```

```
}
# Représentation MA (récurrence), puis réponses orthogonalisées (Cholesky)
phi_ma <- function(A, H) {
  Phi <- vector("list", H); Phi[[1]] <- diag(nrow(A[[1]]))
  for (s in 2:H) Phi[[s]] <- Reduce('+', lapply(seq_along(A), function(j)
    if (s - j >= 1) Phi[[s - j]] %*% A[[j]] else 0))
  Phi
}
irf_chol <- function(A, Sigma, H = 20)
  lapply(phi_ma(A, H), function(M) M %*% t(chol(Sigma)))
```

Code C.11. Réponses à un choc sur `IBO` et décomposition de la variance de `LRM`

```
fevd <- function(Theta, h) {        # part de chaque choc dans la variance à h
  num <- Reduce('+', lapply(Theta[1:h], function(M) M^2)); num / rowSums(num)
}
m <- estime(x); Theta <- irf_chol(m$A, m$Sigma)
s <- c(1, 2, 5, 9, 13, 17)                          # période 1 = impact
rep <- sapply(s, function(i) Theta[[i]][, 3])       # réponses à un choc IBO
dimnames(rep) <- list(colnames(x), s - 1)           # colonnes : horizon
print(round(rep, 4))
hh <- c(1, 4, 8, 12, 20)                            # FEVD de LRM, en %
dv <- 100 * sapply(hh, function(h) fevd(Theta, h)[1, ])
dimnames(dv) <- list(colnames(x), hh)
print(round(dv, 1))
```

```
         0       1       4       8      12      16
LRM 0.0000 -0.0107 -0.0286 -0.0314 -0.0320 -0.0321
LRY 0.0000 -0.0012 -0.0098 -0.0100 -0.0101 -0.0101
IBO 0.0074  0.0102  0.0098  0.0088  0.0088  0.0088
IDE 0.0015  0.0046  0.0064  0.0060  0.0060  0.0060
      1    4    8   12   20
LRM 100 64.2 41.5 35.0 30.5
LRY   0  1.4  4.1  4.9  5.5
IBO   0 32.9 48.7 52.5 55.1
IDE   0  1.5  5.7  7.5  8.9
```

La commande `vecm 2 1 LRM LRY IBO IDE --impulse-responses --variance-decomp` de Gretl produit, à la quatrième décimale, les mêmes nombres (la période 1 de Gretl correspond à l'horizon 0 ci-dessus). Un choc d'un écart-type sur le taux des obligations (environ 0,74 point de pourcentage) abaisse la monnaie réelle d'environ 3,2 % à long terme, et cet effet s'installe en un à deux ans. Il est permanent : les séries étant $I(1)$, un choc déplace durablement leur niveau, et la relation de cointégration reste respectée à l'horizon 16, car $0{,}976 \times (-0{,}0101) - 5{,}409 \times 0{,}0088 + 4{,}162 \times 0{,}0060 \approx -0{,}032$, soit la réponse de long terme de `LRM`.

Pour les intervalles de confiance, nous mettons en œuvre le bootstrap résiduel : on rééchantillonne les résidus centrés du VECM, on reconstruit une série artificielle à partir du modèle estimé, on réestime le VECM sur cette série puis on recalcule les réponses, et l'on répète

500 fois. L'intervalle à 95 % est formé des quantiles 2,5 % et 97,5 % des réponses simulées (méthode d'Efron).

Code C.12. Intervalles bootstrap à 95 % de la réponse de `LRM` à un choc sur `IBO`

```
set.seed(2024); B <- 500; N <- nrow(x); Y0 <- as.matrix(x)
Uc <- scale(m$U, scale = FALSE)                       # résidus centrés
boot <- replicate(B, {
  e  <- Uc[sample(nrow(Uc), replace = TRUE), ]        # rééchantillonnage
  ys <- Y0                                            # 2 valeurs initiales
  for (t in 3:N)                                      # puis récurrence du VAR(2)
    ys[t, ] <- m$cst + m$A[[1]] %*% ys[t - 1, ] + m$A[[2]] %*% ys[t - 2, ] +
      e[t - 2, ]
  mb <- estime(ys)                                    # VECM ré-estimé
  sapply(irf_chol(mb$A, mb$Sigma), c)                 # matrice 16 x 20
})
idx <- (3 - 1) * 4 + 1                                # LRM (i = 1), IBO (j = 3)
ic <- apply(boot[idx, , ], 1, quantile, probs = c(0.025, 0.975))
res <- rbind(estim = sapply(Theta, function(M) M[1, 3]), ic)[, s]
colnames(res) <- s - 1
print(round(res, 4))
```

```
      0       1       4       8      12      16
estim 0 -0.0107 -0.0286 -0.0314 -0.0320 -0.0321
2.5%  0 -0.0160 -0.0415 -0.0491 -0.0532 -0.0548
97.5% 0 -0.0037 -0.0079 -0.0072 -0.0075 -0.0076
```

L'intervalle exclut zéro à partir de l'horizon 1 (par exemple $[-0{,}0415\,;\,-0{,}0079]$ à l'horizon 4) et reste étroit à long terme : l'effet négatif du taux d'intérêt sur la monnaie est donc statistiquement net. La figure C.5 le montre, à côté de la réponse de la monnaie à un choc de revenu, dont l'intervalle contient zéro à tous les horizons : l'effet estimé n'est pas distinguable de zéro avec 53 observations.

C.9 Décomposition de la variance

La décomposition de la variance de l'erreur de prévision (FEVD) répond à la question : quelle part de l'incertitude sur la variable k à l'horizon h provient des chocs sur la variable j ? Avec les notations précédentes,

$$\omega_{kj}(h) = \frac{\sum_{i=0}^{h-1} \left(\mathbf{e}_k' \boldsymbol{\Theta}_i \mathbf{e}_j\right)^2}{\sum_{i=0}^{h-1} \mathbf{e}_k' \boldsymbol{\Phi}_i \tilde{\boldsymbol{\Sigma}}_u \boldsymbol{\Phi}_i' \mathbf{e}_k}, \qquad \sum_{j=1}^{K} \omega_{kj}(h) = 1, \tag{C.7}$$

où $\mathbf{e}_k$ est le k-ième vecteur de la base canonique. Dans JMulTi, la rubrique de décomposition de la variance du menu `Structural Analysis` s'utilise comme celle des réponses impulsionnelles : on choisit la variable puis l'horizon, et le logiciel affiche un diagramme en barres. Le tableau C.7 donne la décomposition de la monnaie réelle (code C.11).

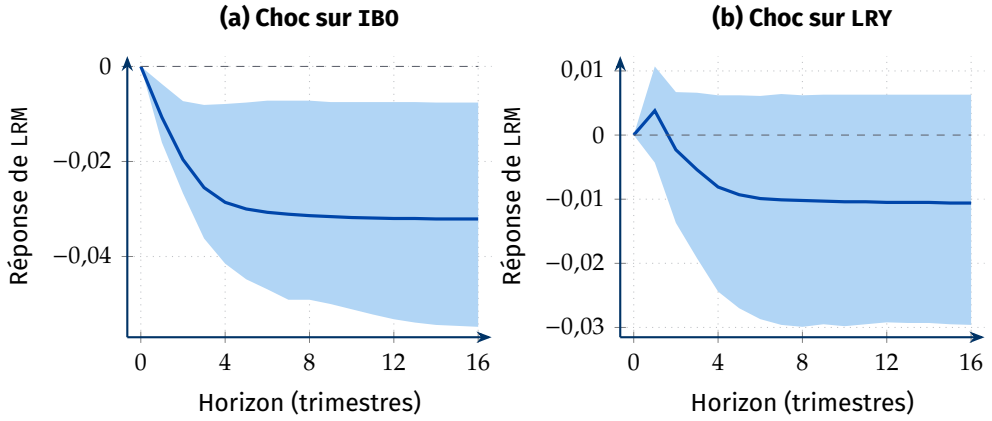


FIGURE C.5. Réponses orthogonalisées (Cholesky, ordre `LRM`, `LRY`, `IBO`, `IDE`) de la monnaie réelle à un choc d'un écart-type. La courbe bleue est l'estimation, la bande bleu clair l'intervalle bootstrap à 95 % (méthode d'Efron, 500 répétitions, graine 2024). Une bande qui exclut la ligne pointillée de zéro indique un effet significatif : c'est le cas en (a), pas en (b).

TABLE C.7. Décomposition de la variance de l'erreur de prévision de `LRM` (en %)

Horizon (trimestres)	Choc LRM	Choc LRY	Choc IBO	Choc IDE
1	100,0	0,0	0,0	0,0
4	64,2	1,4	32,9	1,5
8	41,5	4,1	48,7	5,7
12	35,0	4,9	52,5	7,5
20	30,5	5,5	55,1	8,9

Source : calculs avec R (code C.11), identiques à ceux de GRETL.

À l'horizon d'un trimestre, la monnaie ne dépend que de son propre choc, par construction de l'ordre de Cholesky. Au bout de trois ans, plus de la moitié de la variance provient du taux des obligations, et seulement 30 % de la monnaie elle-même : la dynamique de la demande de monnaie est dominée par le coût d'opportunité. Le revenu pèse peu (5 %), ce qui s'accorde avec la corrélation des résidus de la monnaie et du revenu, absorbée ici par la première position de la monnaie dans l'ordre.

C.10 Diagnostic du modèle

Avant toute interprétation, on vérifie que les résidus se comportent comme un bruit blanc gaussien et que les paramètres sont stables. Dans JMULTI, le menu `Model Checking` → `Residual Analysis` rassemble les tests des résidus et le sous-menu `Stability Analysis` les tests de stabilité.

Autocorrélation. Le test portemanteau utilise les autocovariances $\hat{\mathbf{C}}_j = T^{-1}\sum_{t=j+1}^{T}\hat{\mathbf{u}}_t\hat{\mathbf{u}}_{t-j}'$ des résidus :

$$Q_h = T\sum_{j=1}^{h}\operatorname{tr}\!\left(\hat{\mathbf{C}}_j'\hat{\mathbf{C}}_0^{-1}\hat{\mathbf{C}}_j\hat{\mathbf{C}}_0^{-1}\right), \qquad Q_h^{*} = T^2\sum_{j=1}^{h}\frac{1}{T-j}\operatorname{tr}\!\left(\hat{\mathbf{C}}_j'\hat{\mathbf{C}}_0^{-1}\hat{\mathbf{C}}_j\hat{\mathbf{C}}_0^{-1}\right). \tag{C.8}$$

Sous H_0 (pas d'autocorrélation jusqu'à l'ordre h), les deux statistiques suivent approximativement une loi χ^2 à $K^2(h-p)$ degrés de liberté pour un VAR(p) en niveaux, et à $K^2h - n^*$ degrés de liberté pour un VECM, avec $n^* = Kr + K^2(p-1)$ le nombre de paramètres estimés des coefficients autorégressifs. La version Q_h^* est préférable sur petit échantillon. Le test LM de type Breusch-Godfrey régresse les résidus sur leurs h retards et sur les régresseurs d'origine ; il suit approximativement une loi $\chi^2(hK^2)$, et sa version F se comporte mieux sur petit échantillon.

Code C.13. Tests d'autocorrélation des résidus du VECM

```
U <- resid(fit$rlm); Tn <- nrow(U); K <- ncol(U)     # résidus du VECM (T = 53)
# 1. Portemanteau : Q_h et Q*_h ; ddl = K^2 h - n*, n* = K r + K^2 (p - 1)
portmanteau <- function(U, h, nstar) {
  Tn <- nrow(U); Ci <- solve(crossprod(U) / Tn)
  tr <- sapply(1:h, function(j) {
    Cj <- crossprod(U[(j + 1):Tn, ], U[1:(Tn - j), ]) / Tn
    sum(diag(t(Cj) %*% Ci %*% Cj %*% Ci)) })
  Q <- Tn * sum(tr); Qs <- Tn^2 * sum(tr / (Tn - 1:h))
  df <- ncol(U)^2 * h - nstar
  c(Q = Q, p = pchisq(Q, df, lower.tail = FALSE), Qstar = Qs,
    pstar = pchisq(Qs, df, lower.tail = FALSE), df = df)
}
nstar <- K * 1 + K^2 * (2 - 1)                       # r = 1, p = 2
print(round(rbind(h8 = portmanteau(U, 8, nstar),
                  h12 = portmanteau(U, 12, nstar)), 4))
# 2. Test LM d'autocorrélation (F de Rao) ; retards manquants mis à zéro
Z <- model.matrix(fit$rlm)
for (h in c(1, 4)) {
  L <- do.call(cbind, lapply(1:h, function(i)
    rbind(matrix(0, i, K), U[1:(Tn - i), ])))
  a <- anova(lm(U ~ Z + L - 1), lm(U ~ Z - 1), test = "Wilks")
  cat("LM, h =", h, ": F =", round(a$approx[2], 3), "ddl", round(a$num[2]),
      round(a$den[2], 1), "; p =", round(a[2, "Pr(>F)"], 4), "\n")
}
```

```
          Q      p    Qstar  pstar  df
h8  108.7116 0.4627 120.5559 0.1926 108
h12 153.0714 0.8472 175.8933 0.4035 172
LM, h = 1 : F = 1.241 ddl 16 122.8 ; p = 0.2469
LM, h = 4 : F = 0.94 ddl 64 111.9 ; p = 0.601
```

Normalité et hétéroscédasticité conditionnelle. Le test de Jarque-Bera multivarié standardise les résidus par la racine de Cholesky $\mathbf{P}$ de leur matrice de covariance, $\mathbf{w}_t = \mathbf{P}^{-1}\hat{\mathbf{u}}_t$, puis compare l'asymétrie et l'aplatissement des composantes à ceux de la loi normale ; chaque statistique suit une loi $\chi^2(K)$ et leur somme une loi $\chi^2(2K)$. Le test ARCH-LM multivarié régresse $\operatorname{vech}(\hat{\mathbf{u}}_t\hat{\mathbf{u}}_t')$ sur ses q retards ; il suit approximativement une loi $\chi^2\big(qK^2(K+1)^2/4\big)$.

Code C.14. Normalité (Jarque-Bera multivarié) et effets ARCH multivariés

```
# 3. Jarque-Bera multivarié (Lütkepohl) : résidus standardisés par Cholesky
jb_multi <- function(U) {
  Tn <- nrow(U); P <- t(chol(crossprod(U) / Tn)); W <- t(solve(P, t(U)))
  c(asym = Tn * sum(colMeans(W^3)^2) / 6,
    aplat = Tn * sum((colMeans(W^4) - 3)^2) / 24)
}
jb <- jb_multi(U); jt <- sum(jb)
cat("JB: asymétrie", round(jb[1], 2), "; aplatissement", round(jb[2], 2),
    "; total", round(jt, 2), "; p =",
    signif(pchisq(jt, 2 * K, lower.tail = FALSE), 2), "\n")
# 4. ARCH-LM multivarié : LM = (T - q) K (K + 1) R2m / 2
arch_multi <- function(U, q) {
  Tn <- nrow(U); K <- ncol(U); bas <- lower.tri(diag(K), diag = TRUE)
  V <- t(apply(U, 1, function(u) c(u %o% u)[bas]))   # vech(u u')
  Y <- V[(q + 1):Tn, ]
  Xl <- do.call(cbind, lapply(1:q, function(i) V[(q + 1 - i):(Tn - i), ]))
  e <- resid(lm(Y ~ Xl)); e0 <- scale(Y, scale = FALSE)
  R2 <- 1 - 2 / (K * (K + 1)) * sum(diag(crossprod(e) %*% solve(crossprod(e0))))
  lm <- (Tn - q) * K * (K + 1) * R2 / 2; df <- q * K^2 * (K + 1)^2 / 4
  c(LM = lm, ddl = df, p = pchisq(lm, df, lower.tail = FALSE))
}
print(round(rbind(q1 = arch_multi(U, 1), q2 = arch_multi(U, 2)), 4))
```

```
JB: asymétrie 16.07 ; aplatissement 14.16 ; total 30.23 ; p = 0.00019
         LM ddl      p
q1  82.3696 100 0.8998
q2 194.9589 200 0.5874
```

Dans Gretl, les commandes `modtest --autocorr 4`, `modtest --normality` et `modtest --arch 2` appliquées après `vecm` fournissent les tests équivalents (test de Rao pour l'autocorrélation, test de Doornik-Hansen pour la normalité, ARCH multivarié). Le test de Rao de Gretl donne $F(16;125) = 1{,}263$ pour $h = 1$ et $F(64;115) = 0{,}966$ pour $h = 4$, avec un petit ajustement des degrés de liberté différent du nôtre ; la conclusion est la même. De même, Gretl rapporte 83,0 (ordre 1) et 193,0 (ordre 2) pour l'ARCH, contre 82,4 et 195,0 ici.

Stabilité. Les tests de Chow (rupture à une date donnée, sur échantillon scindé, ou test de prévision) et les coefficients récursifs détectent un changement structurel ; JMulTi peut calculer leurs probabilités critiques par bootstrap. Pour le VECM s'ajoutent les valeurs propres récursives, disponibles uniquement sans restrictions ni variables exogènes. Une vérification équivalente dans R utilise le package `strucchange` sur l'équation de `LRM` :

Code C.15. Test CUSUM et test de rupture à date inconnue (équation de ΔLRM)

```
suppressMessages(library(strucchange))
dLRM <- model.response(model.frame(fit$rlm))[, 1]    # équation de d(LRM)
st <- sctest(efp(dLRM ~ Z - 1, type = "OLS-CUSUM"))  # CUSUM des résidus MCO
cat("OLS-CUSUM : S =", round(unname(st$statistic), 3),
    "; p =", round(st$p.value, 3), "\n")
sc <- sctest(Fstats(dLRM ~ Z - 1, from = 0.2), type = "supF")  # date inconnue
cat("sup-F : ", round(unname(sc$statistic), 2),
    "; p =", round(sc$p.value, 3), "\n")
```

```
OLS-CUSUM : S = 0.626 ; p = 0.828
sup-F :  7.18 ; p = 0.921
```

Table C.8. Résumé des diagnostics du VECM(2) de rang 1 ($T = 53$, $K = 4$)

Test	Statistique	Ddl	Probabilité critique
Portemanteau Q^*_{12}	175,9	172	0,40
Portemanteau Q_8	108,7	108	0,46
LM de Rao, $h = 4$ (version F)	0,94	64 ; 111,9	0,60
Jarque-Bera, asymétrie	16,07	4	0,003
Jarque-Bera, aplatissement	14,16	4	0,007
Jarque-Bera, total	30,23	8	0,0002
ARCH-LM multivarié, $q = 2$	195,0	200	0,59
CUSUM (équation de ΔLRM)	0,63		0,83
Rupture à date inconnue (sup-F)	7,18		0,92

Source : calculs avec R (codes C.13, C.14 et C.15).

Lecture. Il n'y a ni autocorrélation résiduelle, ni effet ARCH, ni instabilité détectable : la dynamique est bien captée par une différence retardée. En revanche, la normalité est nettement rejetée (30,2, probabilité critique 0,0002), asymétrie et aplatissement y contribuant à parts comparables. Ce rejet ne remet pas en cause l'estimation (les estimateurs restent convergents), mais il affaiblit les intervalles asymptotiques et justifie le recours aux intervalles bootstrap de la section précédente. Pour une étude appliquée, on chercherait d'éventuelles valeurs aberrantes (un indicateur d'impulsion se spécifie dans JMulTi).

C.11 Prévision

Pour un VAR en niveaux, la prévision à l'horizon h se calcule par récurrence, $\hat{\mathbf{y}}_{T+h|T} = \hat{\boldsymbol{\nu}} + \sum_{j=1}^{p} \hat{\mathbf{A}}_j \, \hat{\mathbf{y}}_{T+h-j|T}$, avec $\hat{\mathbf{y}}_{T+s|T} = \mathbf{y}_{T+s}$ pour $s \leq 0$. La matrice des erreurs quadratiques moyennes de prévision, qui ignore l'incertitude d'estimation, vaut

$$\boldsymbol{\Sigma}_y(h) = \sum_{i=0}^{h-1} \boldsymbol{\Phi}_i \, \tilde{\boldsymbol{\Sigma}}_u \, \boldsymbol{\Phi}_i', \tag{C.9}$$

et un intervalle à 95 % pour la variable k est $\hat{y}_{k,T+h|T} \pm 1{,}96\sqrt{[\boldsymbol{\Sigma}_y(h)]_{kk}}$. Parce que les séries sont $I(1)$, cette variance croît avec h : l'incertitude de long terme est grande, et c'est un résultat, non un défaut. Dans JMulTi, la rubrique de prévision du module `VECM` demande l'horizon et le niveau de confiance ; elle prévoit les niveaux des variables, à partir de la forme VAR en niveaux, avec des intervalles asymptotiques.

Pour juger la qualité d'un modèle, on réserve les dernières observations. Nous estimons le VECM sur 1974T1 à 1985T3 (47 observations) et prévoyons les huit trimestres suivants, comparés aux valeurs réalisées.

Code C.16. Prévision hors échantillon du VECM sur huit trimestres

```
x1 <- window(x, end = c(1985, 3)); m1 <- estime(x1)  # estimation 1974T1-1985T3
h <- 8; y <- rbind(tail(as.matrix(x1), 2), matrix(0, h, 4))
for (i in 1:h)
  y[2 + i, ] <- m1$cst + m1$A[[1]] %*% y[1 + i, ] + m1$A[[2]] %*% y[i, ]
prev <- y[-(1:2), ]                                  # prévisions 1985T4-1987T3
reel <- as.matrix(window(x, start = c(1985, 4)))
naif <- matrix(tail(as.matrix(x1), 1), h, 4, byrow = TRUE)  # sans changement
rmse <- function(p) sqrt(colMeans((reel - p)^2))
print(round(rbind(VECM = rmse(prev), naif = rmse(naif)), 4))
Phi <- phi_ma(m1$A, h)                               # MSE de prévision à h = 8
mse <- Reduce('+', lapply(Phi, function(P) P %*% m1$Sigma %*% t(P)))
print(round(sqrt(diag(mse)), 4))
```

```
       LRM    LRY    IBO    IDE
VECM 0.0615 0.0297 0.0083 0.0042
naif 0.0772 0.0370 0.0090 0.0075
[1] 0.1036 0.0722 0.0336 0.0231
```

La première ligne de la sortie compare l'erreur quadratique moyenne (RMSE) du VECM à celle d'une prévision « sans changement » qui répète la dernière valeur observée ; la dernière ligne donne l'écart-type de prévision des quatre variables à l'horizon 8. Le VECM bat la prévision naïve pour les quatre séries (par exemple 0,0615 contre 0,0772 pour `LRM`), avec un gain notable pour le taux des dépôts. La commande `fcast 1985:4 1987:3` de Gretl, appliquée après `smpl 1974:1 1985:3` et `vecm`, donne la même prévision de `LRM` (12,0186 au premier horizon, 12,1358 au huitième) et le même écart-type (0,1036 à l'horizon 8).

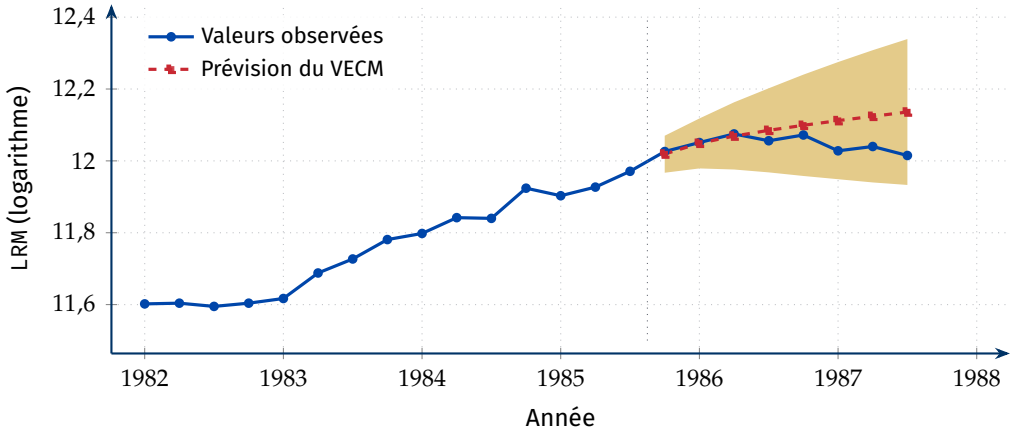


FIGURE C.6. Prévision hors échantillon de la monnaie réelle (`LRM`). Le modèle est estimé sur 1974T1-1985T3 (à gauche de la ligne pointillée verticale) et prévoit 1985T4-1987T3. La bande orangée est l'intervalle à 95 %, qui s'élargit avec l'horizon parce que la série est $I(1)$. Les valeurs réalisées (points bleus) restent dans la bande.

C.12 Modèles ARIMA

Le module `ARIMA` traite une série à la fois. Il propose de choisir les retards autorégressifs et de moyenne mobile par la procédure de Hannan-Rissanen, d'estimer le modèle (avec des régresseurs fixes éventuels, comme des indicatrices), d'analyser les résidus, de modéliser leur éventuelle hétéroscédasticité conditionnelle et de prévoir. Le modèle saisonnier « airline » de Box et Jenkins, très utilisé pour des séries mensuelles à saisonnalité stable, s'écrit

$$(1-L)(1-L^{12})\,y_t = (1+\theta L)(1+\Theta L^{12})\,e_t, \tag{C.10}$$

où L est l'opérateur retard. Nous l'estimons sur le logarithme du nombre mensuel de passagers aériens (jeu `AirPassengers`, livré avec R, 144 mois de 1949 à 1960).

Code C.17. Modèle ARIMA saisonnier avec `stats::arima`

```
f <- arima(log(AirPassengers), order = c(0, 1, 1), seasonal = c(0, 1, 1))
print(f)                                              # modèle « airline »
print(Box.test(resid(f), lag = 24, type = "Ljung-Box", fitdf = 2))
```

```
Call:
arima(x = log(AirPassengers), order = c(0, 1, 1), seasonal = c(0, 1, 1))

Coefficients:
          ma1     sma1
      -0.4018  -0.5569
s.e.   0.0896   0.0731

sigma^2 estimated as 0.001348:  log likelihood = 244.7,  aic = -483.4

  Box-Ljung test

data:  resid(f)
X-squared = 26.446, df = 22, p-value = 0.233
```

Les deux coefficients de moyenne mobile sont fortement significatifs ($\hat{\theta} = -0{,}402$ et $\hat{\Theta} = -0{,}557$, avec des erreurs types de 0,090 et 0,073) et le test de Ljung-Box sur 24 retards ne détecte pas d'autocorrélation résiduelle (probabilité critique 0,23). La commande équivalente de Gretl est `arima 0 1 1 ; 0 1 1 ; lg --nc`, appliquée au logarithme `lg` de la série `g` du fichier `bjg.gdt` ; elle donne −0,4018 et −0,5569, soit les mêmes valeurs à la quatrième décimale.

C.13 Modèles ARCH et GARCH

Les séries financières (rendements, taux de change) ont une variance qui change dans le temps, avec des périodes calmes et des périodes agitées. Le modèle GARCH(1,1) de Bollerslev (1986), qui généralise le modèle ARCH d'Engle (1982), le décrit par

$$r_t = \mu + \varepsilon_t, \qquad \varepsilon_t = \sigma_t z_t, \qquad \sigma_t^2 = \omega + \alpha\, \varepsilon_{t-1}^2 + \beta\, \sigma_{t-1}^2, \tag{C.11}$$

avec z_t de moyenne nulle et de variance un, $\omega > 0$, $\alpha, \beta \geq 0$. La somme $\alpha + \beta$ mesure la persistance de la volatilité : proche de un, les chocs de volatilité se dissipent lentement. Dans JMulTi, le module `GARCH` (ouvert depuis le panneau des modules) propose des modèles ARCH, GARCH et GARCH à seuil (T-GARCH), avec des erreurs gaussiennes, de loi des erreurs généralisée (GED) ou de Student. Dans l'onglet de spécification (`Specify model`), on choisit le modèle, la loi des erreurs et les ordres ; l'estimation utilise l'algorithme BHHH, et l'onglet de diagnostics affiche le test de Jarque-Bera et le graphique des autocorrélations des résidus. Dans l'aide, les coefficients ARCH sont notés γ et les coefficients GARCH β.

Nous estimons un GARCH(1,1) sur les rendements quotidiens (en pourcentage) de l'indice boursier allemand DAX, du jeu `EuStockMarkets` livré avec R (1 859 rendements). Le test ARCH-LM précède l'estimation : il régresse le carré des rendements centrés sur ses q retards, et $(T - q)R^2$ suit approximativement une loi $\chi^2(q)$.

Code C.18. Test ARCH et GARCH(1,1) gaussien avec `fGarch`

```
suppressMessages(library(fGarch))
r <- 100 * diff(log(EuStockMarkets[, "DAX"]))        # rendements du DAX, en %
e2 <- (r - mean(r))^2; q <- 5; Xl <- embed(e2, q + 1)
LM <- (length(r) - q) * summary(lm(Xl[, 1] ~ Xl[, -1]))$r.squared
cat("ARCH-LM(5) =", round(LM, 2), "; p =",
    signif(pchisq(LM, q, lower.tail = FALSE), 2), "\n")
g <- garchFit(~ garch(1, 1), data = r, cond.dist = "norm", trace = FALSE)
print(round(g@fit$matcoef, 4))
cat("alpha + beta =", round(sum(coef(g)[c("alpha1", "beta1")]), 4), "\n")
write.csv(data.frame(r = as.numeric(r)), "dax.csv", row.names = FALSE)  # Gretl
```

```
ARCH-LM(5) = 69.71 ; p = 1.2e-13
        Estimate  Std. Error  t value Pr(>|t|)
mu        0.0654      0.0216   3.0289   0.0025
omega     0.0475      0.0126   3.7601   0.0002
alpha1    0.0684      0.0148   4.6299   0.0000
beta1     0.8876      0.0236  37.6768   0.0000
alpha + beta = 0.956
```

L'effet ARCH est massif (69,7 pour un seuil de 11,1 à 5 %). Les estimations donnent $\hat{\alpha} = 0{,}068$ et $\hat{\beta} = 0{,}888$, soit une persistance de 0,956 : un choc de volatilité met plusieurs semaines à s'estomper. L'équivalent Gretl est `garch 1 1 ; r`, appliqué au fichier `dax.csv` que le code précédent écrit ; il fournit les mêmes valeurs (0,0654, 0,0475, 0,0684 et 0,8876). Avec des erreurs de Student, `fGarch` estime un nombre de degrés de liberté d'environ 6 : les queues des rendements sont plus épaisses que celles d'une loi normale, ce qu'illustre l'exercice C.6.

C.14 VAR et VECM structurels

Un VAR en forme réduite décrit des corrélations ; pour interpréter des chocs comme une demande ou une offre, il faut des restrictions d'identification. Les résidus structurels $\boldsymbol{\varepsilon}_t$ (de matrice de covariance $\mathbf{I}_K$) sont liés aux résidus réduits par $\mathbf{A}\mathbf{u}_t = \mathbf{B}\boldsymbol{\varepsilon}_t$, d'où $\boldsymbol{\Sigma}_u = \mathbf{A}^{-1}\mathbf{B}\mathbf{B}'\mathbf{A}^{-1\prime}$. On distingue :

- le modèle A ($\mathbf{B} = \mathbf{I}_K$) et le modèle B ($\mathbf{A} = \mathbf{I}_K$), qui exigent chacun $K(K-1)/2$ restrictions ;
- le modèle AB, qui exige $K^2 + K(K-1)/2$ restrictions ;
- le modèle à restrictions de long terme de Blanchard et Quah (1989), où l'on impose que certains chocs n'aient pas d'effet cumulé sur certaines variables : la matrice des effets de long terme $\boldsymbol{\Xi} = (\mathbf{I} - \mathbf{A}_1 - \cdots - \mathbf{A}_p)^{-1}\mathbf{B}$ est alors triangulaire.

Pour un système cointégré, JMulTi propose le VECM structurel (SVEC), où r colonnes de la matrice des effets de long terme sont nulles ($r(K-r)$ restrictions indépendantes) et où les restrictions restantes portent sur les chocs permanents, au nombre de $(K-r)(K-r-1)/2$, et transitoires, au nombre de $r(r-1)/2$. L'estimation avec erreurs types bootstrap s'obtient par `SVEC` → `Estimate SVEC Model`, avec les boutons `Point Estimates Only` ou `Estimate with Boot. Std. Err.`.

À titre d'illustration de la mécanique de Blanchard-Quah, le code suivant impose qu'un choc sur la deuxième variable (IBO) n'ait aucun effet cumulé sur la première (LRY), dans un VAR(2) estimé sur les variations (ΔLRY, ΔIBO). Il s'agit d'un exemple de calcul, non d'une identification économique défendue.

Code C.19. Identification de long terme à la Blanchard-Quah (illustration)

```
dx <- as.matrix(diff(x[, c("LRY", "IBO")])); n <- nrow(dx)   # (dLRY, dIBO)
Zr <- cbind(1, dx[2:(n - 1), ], dx[1:(n - 2), ]); Yr <- dx[3:n, ]  # VAR(2)
Bh <- solve(crossprod(Zr), crossprod(Zr, Yr))        # MCO
Ur <- Yr - Zr %*% Bh; S <- crossprod(Ur) / nrow(Ur)
Asum <- t(Bh[2:3, ]) + t(Bh[4:5, ])                  # A1 + A2
LR <- solve(diag(2) - Asum)                          # effets cumulés
Xi <- t(chol(LR %*% S %*% t(LR)))                    # long terme
B <- solve(LR) %*% Xi                                # matrice d'impact
cat("B (impact) :\n"); print(round(B, 4))
cat("Xi (long terme) :\n"); print(round(Xi, 4))
cat("max |B B' - Sigma| =", signif(max(abs(B %*% t(B) - S)), 2), "\n")
```

```
B (impact) :
        LRY    IBO
LRY  0.0215 0.0094
IBO -0.0035 0.0072
Xi (long terme) :
        LRY    IBO
LRY  0.0254 0.0000
IBO -0.0017 0.0125
max |B B' - Sigma| = 2.2e-19
```

La matrice Ξ est triangulaire inférieure par construction (zéro en position $(1,2)$), et $\mathbf{BB}'$ restitue la covariance des résidus à l'erreur d'arrondi près : les $K(K-1)/2 = 1$ restriction suffit à identifier le système. Pour un SVAR de type A, B ou AB, le package `vars` de R propose la fonction `SVAR` ; elle n'est pas disponible dans notre environnement de test, et nous ne l'avons donc pas exécutée.

C.15 Récapitulatif

Le tableau C.9 associe chaque procédure de JMulTi à son équivalent en R et en Gretl. Les fonctions dont le nom est suivi de « (maison) » sont celles écrites dans cette annexe.

TABLE C.9. Correspondance entre JMulTi, R et Gretl

Procédure	JMulTi (module, rubrique)	R	Gretl
Import	Import de fichier texte	`write.table`	`store --jmulti`
ADF, KPSS	`Initial Analysis`, racines unitaires	`ur.df`, `ur.kpss`	`adf`, `kpss`
Ordre du VAR	`Compute Infocriteria`	`choix_var` (maison)	`var --lagselect`
Johansen	`Initial Analysis`, `Johansen Trace Test`	`ca.jo`	`johansen`
VECM	`VECM`, `Estimate Model`	`cajorls`	`vecm`
Restrictions sur $\boldsymbol{\beta}$	`Specify and Test Restrictions on Beta`	`blrtest`	`restrict`
Résidus	`Residual Analysis`	`portmanteau` (maison), `anova`	`modtest`
Stabilité	`Stability Analysis`	`strucchange`	non couvert
Réponses impulsionnelles	`Structural Analysis`	`irf_chol` (maison)	`--impulse-responses`
FEVD	`Structural Analysis`	`fevd` (maison)	`--variance-decomp`
Prévision	Prévision (VAR, VECM)	boucle sur $\hat{\mathbf{A}}_j$	`fcast`
ARIMA	Module `ARIMA`	`stats::arima`	`arima`
GARCH	Module `GARCH`	`fGarch::garchFit`	`garch`
SVAR, SVEC	`SVEC`, modèles A, B, AB	`vars::SVAR` (non testé)	non couvert

Points clés à retenir

- JMulTi enchaîne les étapes d'une analyse multivariée : données, racines unitaires, ordre p, rang de cointégration, VECM, contrôle des résidus, analyse structurelle, prévision.
- Le test de Johansen dépend de p et de la spécification déterministe (`nonzero mean`, `constant and trend`, `orthogonal trend`) ; examiner plusieurs spécifications, et ne pas décider sur un seul résultat limite.
- JMulTi documente le test de la trace mais ni le test de Phillips-Perron ni la valeur propre maximale ; R et Gretl comblent ces lacunes.
- Dans le VECM, $\boldsymbol{\beta}$ donne la relation de long terme et $\boldsymbol{\alpha}$ la vitesse d'ajustement ; une restriction sur $\boldsymbol{\beta}$ se teste par un rapport de vraisemblance (χ^2).
- Les réponses orthogonalisées dépendent de l'ordre des variables ; les intervalles bootstrap sont préférables aux intervalles asymptotiques quand les résidus ne sont pas gaussiens.
- Avant d'interpréter, tester les résidus (portemanteau, LM, Jarque-Bera, ARCH) et la stabilité ; le rejet de la normalité n'invalide pas l'estimation mais justifie le bootstrap.
- Chaque résultat de JMulTi se vérifie avec une commande de R ou de Gretl : ce double contrôle est une bonne pratique de reproductibilité.

C.16 Exercices

Exercice C.1 : Choisir la spécification déterministe

Pour chaque situation, indiquez l'option du test de Johansen de JMulTi (`nonzero mean`, `constant and trend` ou `orthogonal trend`) et justifiez-la : (a) des taux d'intérêt de court et de long terme, sans tendance visible, dont l'écart a une moyenne non nulle ; (b) des logarithmes de prix à la consommation de deux pays voisins, qui croissent tous deux avec une dérive linéaire ; (c) des séries qui croissent et dont la relation d'équilibre contient elle-même une tendance. Expliquez ensuite pourquoi JMulTi propose moins de cas que Gretl.

Exercice C.2 : Statistique de la trace à la main

Un analyste de la Banque centrale du Congo étudie trois séries mensuelles (logarithme du taux de change CDF/USD, de l'indice des prix de Kinshasa et de la masse monétaire M2). Il estime un VAR(2) sur 120 mois (donc $T = 118$ observations utilisées) avec la spécification `orthogonal trend` et obtient les valeurs propres, purement illustratives, $\hat{\lambda}_1 = 0{,}20$, $\hat{\lambda}_2 = 0{,}08$, $\hat{\lambda}_3 = 0{,}02$. (a) Calculez la statistique de la trace pour $r_0 = 0$, 1 et 2 avec la formule (C.5). (b) Les valeurs critiques à 5 % sont 31,52, 17,95 et 8,18 pour $r_0 = 0, 1, 2$. Quel rang retient-il ? (c) Vérifiez votre méthode en retrouvant la valeur 48,80 du tableau C.5 avec les valeurs propres de la sortie de Gretl et $T = 53$.

Exercice C.3 : Critères d'information

Pour le VAR en niveaux des données danoises ($K = 4$, $T = 51$), les logarithmes des déterminants des covariances résiduelles sont $\ln|\tilde{\boldsymbol{\Sigma}}_u(1)| = -35{,}240$ et $\ln|\tilde{\boldsymbol{\Sigma}}_u(2)| = -36{,}124$. (a) Calculez $\mathrm{AIC}(m)$ et $\mathrm{SC}(m)$ pour $m = 1$ et $m = 2$ avec $n = Km + 1$ régresseurs par équation, et comparez au tableau C.3. (b) Expliquez pourquoi le critère de Schwarz retient $m = 1$ et celui d'Akaike $m = 2$ en comparant les pénalités des deux critères.

Exercice C.4 : Test d'une restriction sur le vecteur de cointégration

Reprenez le VECM de rang 1 des données danoises. Testez avec Gretl (commande `restrict`) puis avec `urca` (fonction `blrtest`, matrice $\mathbf{H}$ de dimensions 4×3) l'hypothèse que les coefficients de `IBO` et de `IDE` sont opposés, $\beta_3 + \beta_4 = 0$, c'est-à-dire que seul l'écart entre les deux taux intervient dans la demande de monnaie. Comparez les deux résultats et interprétez la conclusion.

Exercice C.5 : Taux de change CDF/USD (RDC)

La série BCC-01 de l'annexe C du livre (fichier `bcc_taux_change_cdf_usd_2005_2023.csv`, colonnes `date` et `cdf_usd`) donne le taux de change mensuel. (a) Calculez les variations logarithmiques mensuelles et vérifiez par un test ADF que la série du taux est $I(1)$ tandis que ses variations sont stationnaires. (b) Estimez un modèle ARIMA sur le logarithme du taux avec `stats::arima` ou Gretl, choisissez les ordres par le critère d'Akaike et testez les résidus. (c) Testez la présence d'effets ARCH sur les résidus et, le cas échéant, estimez un GARCH(1,1). (d) Écrivez le fichier au format de JMulTi comme au code C.1 et indiquez les

menus à suivre pour refaire (a) et (b) dans ce logiciel. Ce fichier n'est pas joint au livre : il se construit à partir de la source indiquée à l'annexe C du livre.

Exercice C.6 : GARCH à erreurs de Student

Avec les rendements du DAX du code C.18, estimez un GARCH(1,1) avec `cond.dist = "std"` dans `garchFit`. (a) Comparez le critère d'information (`g@fit$ics`) à celui du modèle gaussien. (b) Interprétez le paramètre de forme `shape` (degrés de liberté de la loi de Student) et la persistance $\alpha + \beta$. (c) Quel intérêt présente, pour la gestion des risques d'une banque commerciale de Kinshasa, le choix d'une loi à queues épaisses ?

Références

Blanchard, O. J., & Quah, D. (1989). The dynamic effects of aggregate demand and supply disturbances. *American Economic Review*, *79*(4), 655–673.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327.

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, *74*(366), 427–431.

Dickey, D. A., & Fuller, W. A. (1981). Likelihood ratio statistics for autoregressive time series with a unit root. *Econometrica*, *49*(4), 1057–1072.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, *50*(4), 987–1007.

Johansen, S. (1995). *Likelihood-based inference in cointegrated vector autoregressive models*. Oxford University Press.

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.

Kwiatkowski, D., Phillips, P. C. B., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1–3), 159–178.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer.

Lütkepohl, H., & Krätzig, M. (Éds.). (2004). *Applied time series econometrics*. Cambridge University Press.