FORMATION « MODÈLES DE PRÉVISION »

Insee
Mesurer pour comprendre

BCEAO
BANQUE CENTRALE DES ETATS DE L'AFRIQUE DE L'OUEST

Recherche de modèles de régression linéaires : l'approche *GEneral-To-Specific* (Gets)

INTERVENANTS

Morgane Glotain
Alain Quartier-la-Tente

29-31 mars 2017

Introduction

☺ **Avantage pour le conjoncturiste** : un très grand nombre de variables explicatives disponibles (soldes d'opinion, indicateurs quantitatifs, *etc.*)

Introduction

☺ **Avantage pour le conjoncturiste** : un très grand nombre de variables explicatives disponibles (soldes d'opinion, indicateurs quantitatifs, *etc.*)

☹ **Inconvénients pour le conjoncturiste** : un trop grand nombre de variables explicatives disponibles et un temps limité pour construire les modèles de prévision

Introduction

☺ **Avantage pour le conjoncturiste** : un très grand nombre de variables explicatives disponibles (soldes d'opinion, indicateurs quantitatifs, *etc.*)

☹ **Inconvénients pour le conjoncturiste** : un trop grand nombre de variables explicatives disponibles et un temps limité pour construire les modèles de prévision

Une solution : l'algorithme Gets qui recherche les meilleurs modèles admissibles qui respectent certains critères

Sommaire

Qu'est-ce qu'on cherche ?

Objectif : avoir des modèles de prévision **simples** avec une **interprétation économique rapide**

$$y_t = \beta_0 + \sum_{m=1}^{4} \beta_m y_{t-m} + \sum_{n=1}^{N} \zeta_n x_{n,t} + \varepsilon_t \qquad \begin{cases} x_{n,t} & \text{variables explicatives} \\ y_t & \text{variable à prévoir} \end{cases}$$

$$\varepsilon_t \overset{i.i.d}{\sim} \mathcal{N}(0, \sigma^2)$$

Hypothèses sur les résidus : gaussiens, indépendants et identiquement distribués

Point de départ de l'algorithme : un *general unrestricted model* – modèle général sans contrainte – (GUM), c'est le **modèle initial** avec toutes les variables explicatives

Cinq tests mobilisés

Cinq tests sont mobilisés :

- Trois tests de spécification :
  - test de Ljung-Box sur les résidus → (H_0) : résidus décorrélés (absence d'autocorrélation)
  - test de Ljung-Box sur le carré des résidus → (H_0) : homoscédasticité des résidus (absence d'hétéroscédasticité)
  - test de Jarque-Bera → (H_0) : résidus normaux
- Test de Student de nullité des coefficients (1 test par coefficient)
- *Parsimonious Encompassing Test* – test englobant parcimonieux – (PET ; également appelé *backtest*) : test de Wald pour vérifier si les régresseurs supprimés sont conjointement significativement nuls

Cinq tests mobilisés

Cinq tests sont mobilisés :

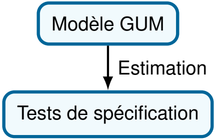
- Trois tests de spécification : (seuil : 1,7 %)
  - test de Ljung-Box sur les résidus → (H_0) : résidus décorrélés (absence d'autocorrélation)
  - test de Ljung-Box sur le carré des résidus → (H_0) : homoscédasticité des résidus (absence d'hétéroscédasticité)
  - test de Jarque-Bera → (H_0) : résidus normaux
- Test de Student de nullité des coefficients (1 test par coefficient) (seuil : 5 %)
- *Parsimonious Encompassing Test* – test englobant parcimonieux – (PET ; également appelé *backtest*) : test de Wald pour vérifier si les régresseurs supprimés sont conjointement significativement nuls (seuil : 5 %)

Pour chaque test il faut définir : les seuils et des ordres pour les tests de Ljung-Box (ordre 4 conseillé)

Description de l'algorithme

Algorithme en 3 étapes : estimation et test du modèle GUM, recherche des modèles « terminaux » et détermination du modèle final

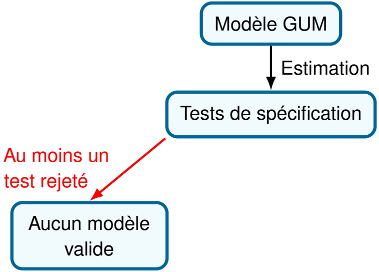
Étape 1 : Estimation et test du modèle général (GUM)



Description de l'algorithme

Algorithme en 3 étapes : estimation et test du modèle GUM, recherche des modèles « terminaux » et détermination du modèle final

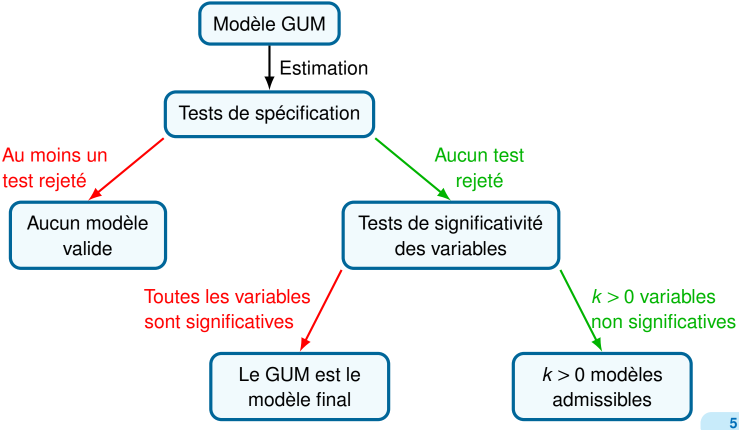
Étape 1 : Estimation et test du modèle général (GUM)



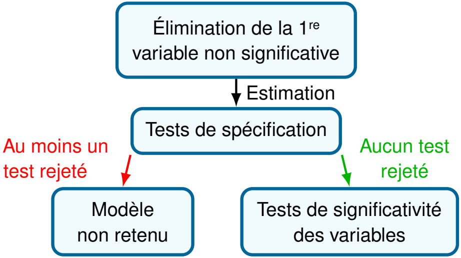
Description de l'algorithme

Algorithme en 3 étapes : estimation et test du modèle GUM, recherche des modèles « terminaux » et détermination du modèle final

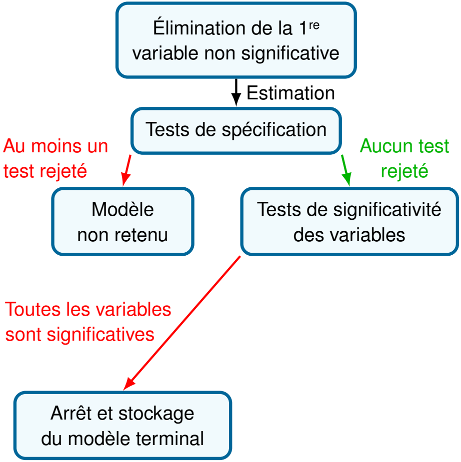
Étape 1 : Estimation et test du modèle général (GUM)



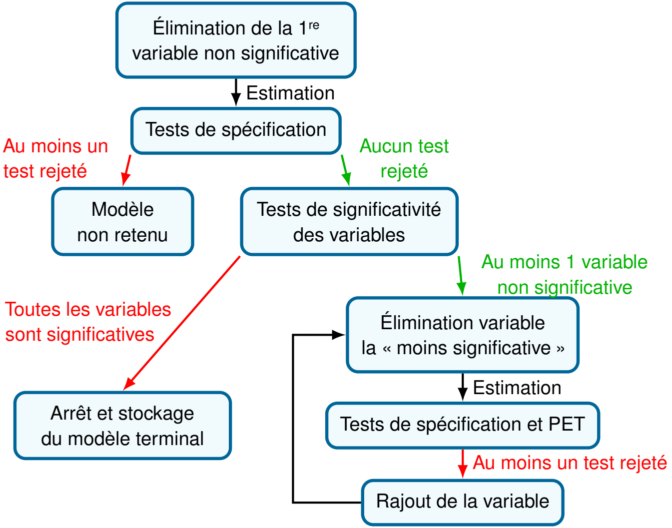
Étape 2 : Recherche des modèles terminaux



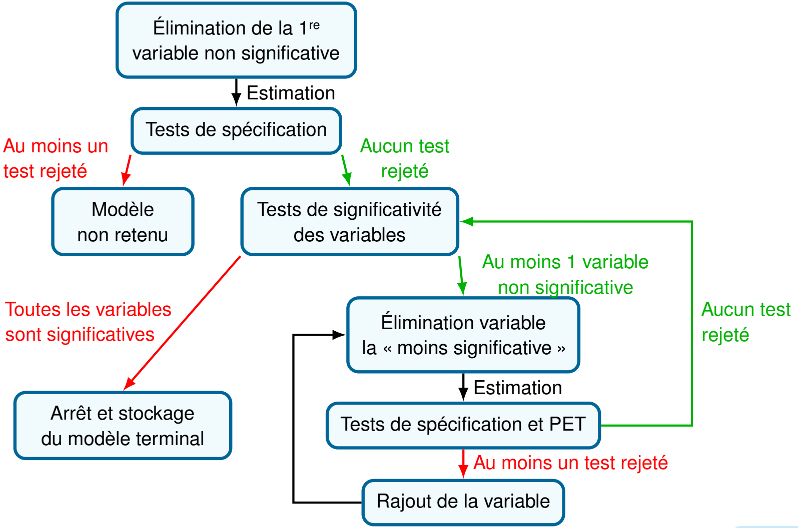
Étape 2 : Recherche des modèles terminaux



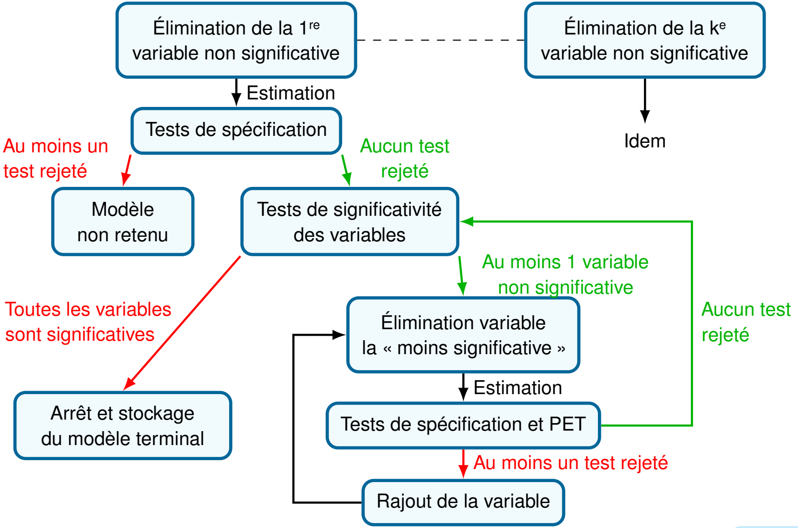
Étape 2 : Recherche des modèles terminaux



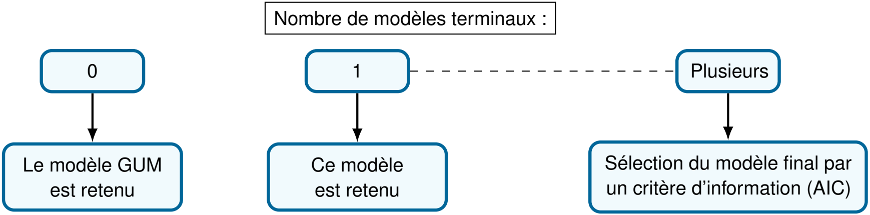
Étape 2 : Recherche des modèles terminaux



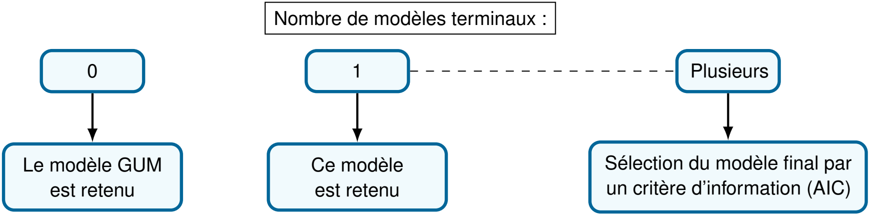
Étape 2 : Recherche des modèles terminaux



Étape 3 : Détermination du modèle final



Étape 3 : Détermination du modèle final



Théoriquement les modèles terminaux sont dits *mutuellement englobants* (la prévision d'un modèle n'apporte aucune information conditionnellement à la prévision des autres modèles) :

- les critères d'information auront des valeurs proches (son choix n'est donc pas important)
- en pratique, le choix du modèle final ne changera pas fondamentalement la qualité du modèle

Exemple et décryptage de la sortie R (1/2)

En lançant la procédure `getsm` les résultats des trois étapes sont indiquées :

Étape 1 :

```
GUM mean equation:

                     reg.no keep      coef std.error   t-stat     p-value
mconst                    1    0  2.560827  2.701316  0.94799 3.4579e-01
acquis_ipi1_c1            2    0  0.464746  0.054964  8.45552 6.3100e-13
facind_c1_m1              3    0 -0.047610  0.030984 -1.53660 1.2806e-01
facind_c1_m2              4    0  0.026290  0.019273  1.36407 1.7611e-01
ind_manuf_oscd_c1_m1      5    0  0.019280  0.021371  0.90213 3.6951e-01

Diagnostics:

                     Chi-sq df p-value
Ljung-Box AR(1)   1.6152372  1 0.20376
Ljung-Box ARCH(1) 0.9026844  1 0.34206
Jarque-Bera       0.0064816  2 0.99676
```

Exemple et décryptage de la sortie R (2/2)

Pour les étapes 2 et 3 :

```
Paths searched:

path 1 : 1 5 3 4
path 2 : 3 1 4 5
path 3 : 4 5 3 1
path 4 : 5 1 3 4

Terminal models:

spec 1 : 1 2 3 4 5
spec 2 : 2

              info(aic)      logl  n  k
spec 1 (gum):    2.3877 -103.6411 91  5
spec 2:          2.3430 -105.6074 91  1
```

L'*Impulse-Indicator Saturation* (IIS)

Généralement : le nombre, la date et l'ampleur des ruptures dans un modèle sont inconnus

L'*Impulse-Indicator Saturation* (IIS)

Généralement : le nombre, la date et l'ampleur des ruptures dans un modèle sont inconnus
→ la méthode IIS permet de les détecter et de les corriger en rajoutant des indicatrices

L'*Impulse-Indicator Saturation* (IIS)

Généralement : le nombre, la date et l'ampleur des ruptures dans un modèle sont inconnus
→ la méthode IIS permet de les détecter et de les corriger en rajoutant des indicatrices

On rajoute dans les variables explicatives une indicatrice temporelle pour chaque date et on utilise l'algorithme Gets pour sélectionner les indicatrices

L'*Impulse-Indicator Saturation* (IIS)

Généralement : le nombre, la date et l'ampleur des ruptures dans un modèle sont inconnus
→ la méthode IIS permet de les détecter et de les corriger en rajoutant des indicatrices

On rajoute dans les variables explicatives une indicatrice temporelle pour chaque date et on utilise l'algorithme Gets pour sélectionner les indicatrices

On a plus de variables explicatives que d'observations, l'estimation est donc impossible !

L'*Impulse-Indicator Saturation* (IIS)

Généralement : le nombre, la date et l'ampleur des ruptures dans un modèle sont inconnus
→ la méthode IIS permet de les détecter et de les corriger en rajoutant des indicatrices

On rajoute dans les variables explicatives une indicatrice temporelle pour chaque date et on utilise l'algorithme Gets pour sélectionner les indicatrices

On a plus de variables explicatives que d'observations, l'estimation est donc impossible !

On partitionne pour appliquer l'algorithme Gets sur plusieurs blocs de variables

Exemple de l'algorithme IIS dans le cas de 2 blocs (1/2)

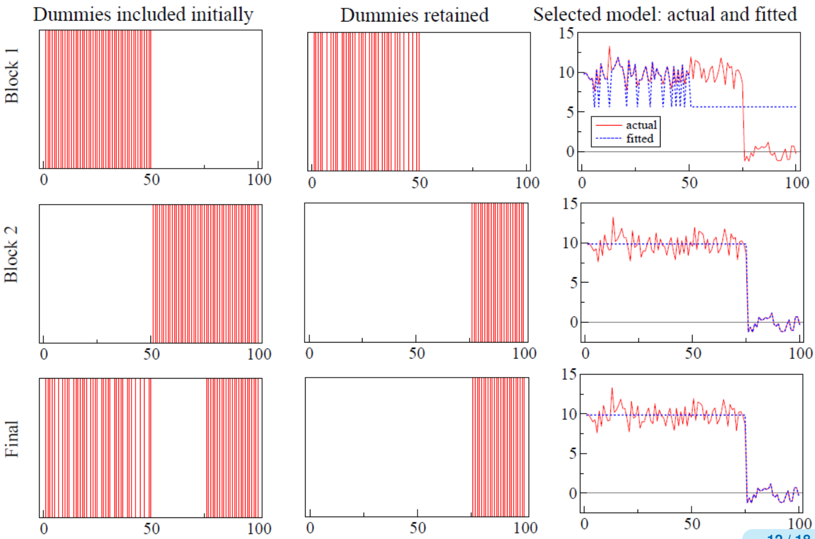
On utilise l'algorithme Gets sur les deux modèles suivants :

$$y_t = \underbrace{\beta_0 + \sum_{m=1}^{4} \beta_m y_{t-m} + \sum_{n=1}^{N} \zeta_n x_{n,t}}_{\text{On garde obligatoirement ces variables}} + \underbrace{\sum_{i=1}^{T/2} \delta_i \mathbb{1}_{i=t}}_{\substack{\text{On retire les ind.} \\ \text{non significatives} \\ \text{avec Gets}}} + \varepsilon_t$$

$$y_t = \overbrace{\beta_0 + \sum_{m=1}^{4} \beta_m y_{t-m} + \sum_{n=1}^{N} \zeta_n x_{n,t}} + \overbrace{\sum_{i=T/2+1}^{T} \delta_i \mathbb{1}_{i=t}} + \varepsilon_t$$

On utilise une nouvelle fois l'algorithme Gets avec les indicatrices retenues dans les deux blocs précédents

Exemple de l'algorithme IIS dans le cas de 2 blocs (2/2)



Les paramètres de Gets dans IIS

L'algorithme IIS de R (fonction `isat`) utilise l'algorithme Gets avec des paramètres spécifiques. Par défaut :

- on ne force pas les résidus à être normaux, hétéroscédastiques et décorrélés : on supprime les indicatrices uniquement par rapport à leur statistique de Student
- le seuil utilisé par le test de significativité des coefficients est de 0,1 %

Il faut d'abord utiliser l'algorithme IIS et ensuite rechercher les modèles de prévision avec Gets : cela notamment permet de gérer les cas où les problèmes de spécification sont dûs à des *outliers*

Sommaire

Comment modifier les paramètres sous R ?

Il faut rajouter des options dans `isat` et `getsm` :

- Tests de spécification :
  - test de Ljung-Box sur les résidus → **par défaut** : `ar.LjungB=list(lag=NULL, pval=0.025)` sous `getsm` et `ar.LjungB=NULL` sous `isat`
  - test de Ljung-Box sur le carré des résidus → **par défaut** : `arch.LjungB=list(lag=NULL, pval=0.025)` sous `getsm` et `arch.LjungB=NULL` sous `isat`
  - test de Jarque-Bera → **par défaut** : `normality.JarqueB=NULL` sous `getsm` et `isat`
- Test de Student de nullité des coefficients → **par défaut** : `t.pval=0.05` sous `getsm` et `t.pval=0.001` sous `isat`
- *Backtest* → **par défaut** : `wald.pval=t.pval` sous `getsm` et `isat`

⚠ **Par défaut** sous R on ne force pas les résidus à être normaux

Paramètres de la fonction `isat`

Par défaut sous R, la fonction `isat` n'utilise pas l'algorithme *Impulse-Indicator Saturation* (IIS) mais l'algorithme *Step-Indicator Saturation* (SIS ; pour détecter les ruptures en niveau). On utilise donc la fonction `isat` avec les paramètres :

```
isat(y = ipi, mxreg = soldes, iis = TRUE, sis = FALSE,
ar = 1)
```

Lorsque la fonction est lancée, le nombre de blocs est indiqué :

```
IIS block 1 of 4:
```

Quelques conseils pratiques (1/2)

- Définir l'ensemble des paramètres au début du programme et pour utiliser les mêmes à chaque fois :

```
ordreLJ<-4
pval<-0.017
nb_retards<-4
ic<-"aic"

gum <- isat(y = ipi, mxreg = soldes, iis = TRUE, sis = FALSE,
            ar = 1:nb_retards,info.method = ic)
getsm(arx(y = gum$aux$y, mxreg = gum$aux$mX),info.method = ic,
      normality.JarqueB=pval,
      ar.LjungB=list(lag=ordreLJ, pval=pval),
      arch.LjungB=list(lag=ordreLJ, pval=pval)
      )
```

- Faire la recherche de modèles avec les soldes contemporains et retardés puis tâtonner pour savoir s'il faut mettre des variables en différence

Quelques conseils pratiques (2/2)

- Ne faire la recherche de modèles qu'avec des variables qui ont du sens
- Chercher (si possible) des modèles sans retard de la variable endogène (attention aux retournements)
- Avoir plusieurs modèles de prévision de la même variable (pas de meilleur modèle)
- Retenir les fonctions de base (à appliquer aux sorties de `isat`, `getsm` ou `arx`) :
    - `coef(objet)` pour avoir l'estimation des coefficients du modèle associé à `objet`
    - `residuals(objet)` pour avoir les résidus du modèle associé à `objet`
    - `fitted(objet)` pour avoir les valeurs prévues par le modèle associé à `objet`

Merci de votre attention !

Morgane Glotain	morgane.glotain@insee.fr
Alain Quartier-la-Tente	alain.quartier-la-tente@insee.fr