Chapitre 4 : Prévisions de court terme

Olivier DARNÉ

Techniques de Prévision et Conjoncture (M1 EKAP)
2020-2021

Plan du cours de Prévisions Conjoncturelles de la Croissance

- 1. Analyse de la conjoncture
- 2. Modèles linéaires pour la prévision
- 3. Prévision par des modèles linéaires

L'indicateur favori des conjoncturistes pour appréhender lactivité économique est le produit intérieur brut (PIB) (voir Annexe pour sa définition)

Cet indicateur est publié avec une fréquence trimestrielle et avec un certain délai :

- Les "premiers résultats" sont publiés depuis janvier 2016 moins de 30 jours (45 jours avant) après la fin du trimestre.

  Ils donnent une première estimation de la croissance trimestrielle du PIB mais ne contient que des données encore très incomplètes, notamment en ce qui concerne le dernier mois du trimestre

- Les "résultats détaillés" sont publiés moins de 85 jours après la fin du trimestre.

  Ils mettent à jour la première estimation de la croissance trimestrielle du PIB.

Tableau 1 : estimations successives du PIB trimestriel pour l'année 2010

	2010 T1	2010 T2	2010 T3	2010 T4
Premiers résultats (J+45)	0,1%	0,6%	0,4%	0,3%
Résultats détaillés (J+90)	0,1%	0,7%	0,3%	0,4%
Compte annuel provisoire (A+1)	0,2%	0,5%	0,4%	0,3%
Compte annuel semi définitif (A+2)	0,3%	0,7%	0,4%	0,4%
Compte annuel définitif (A+3)	0,2%	0,6%	0,5%	0,5%

Source : Insee.

Les gouvernements et les banques centrales doivent disposer d'une évaluation précise et rapide du taux de croissance du PIB pour le trimestre en cours afin de fournir une analyse et plus précise et précoce de la situation économique.

Or, ce délai pose problème lorsque la décision de politique économique ou d'investissement doit être prise rapidement, voire dans l'urgence.

Tout particulièrement, la rapidité et l'ampleur de la crise économique associée à la crise sanitaire de la Covid-19 ont exposé le dilemme entre précision de la mesure de l'activité économique et rapidité de sa mise à disposition.

En effet, les décideurs publiques et politiques ont besoin d'avoir une estimation la plus précise possible et en temps réel des retombées économiques des mesures sanitaires et/ou de soutien à l'économie.

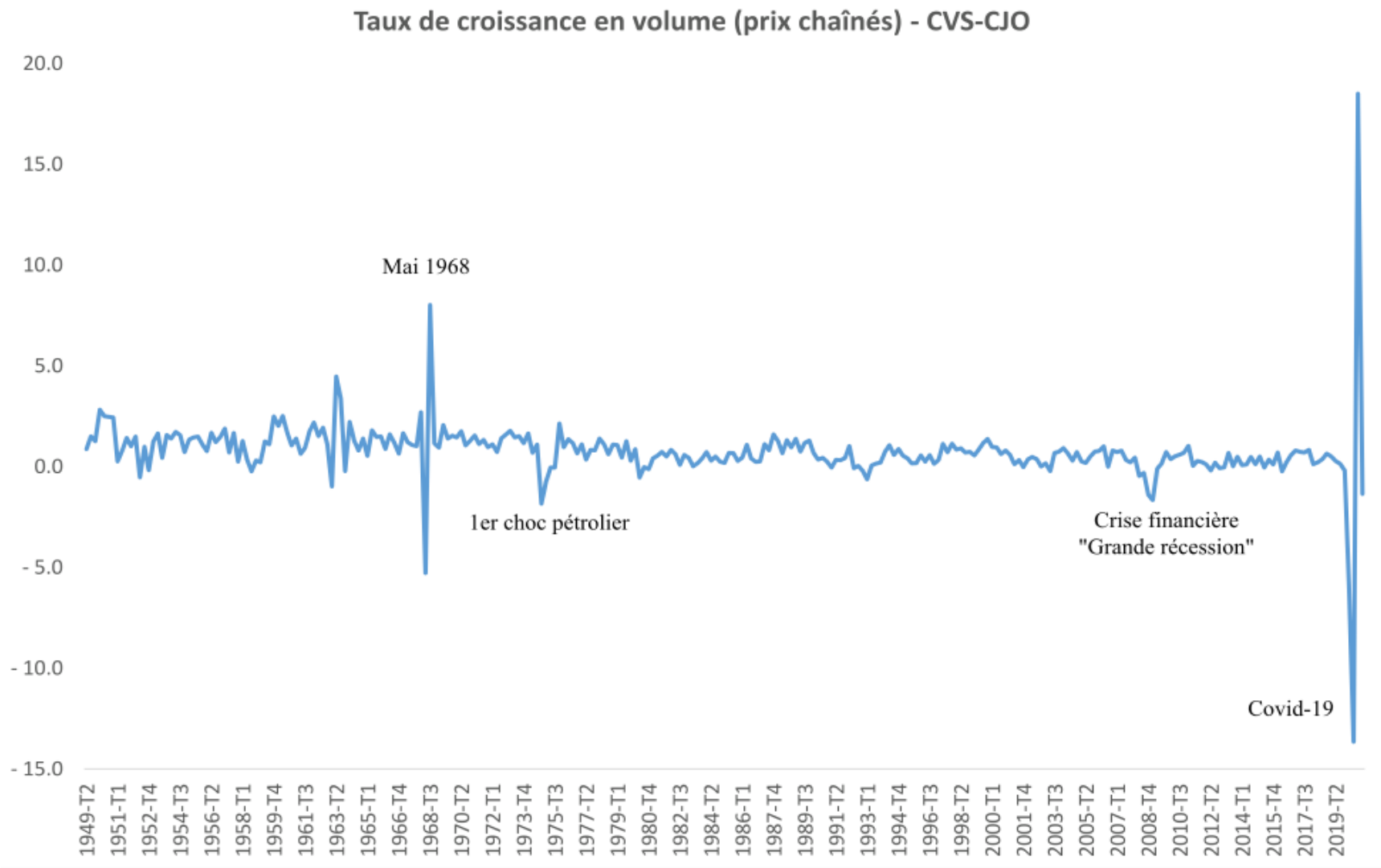
Taux de croissance du PIB

▶ Le taux de croissance du PIB pour la période t fait référence taux de croissance du PIB réel (ou PIB à prix constants ou en volume) :

$$gy_t = \frac{y_t - y_{t-1}}{y_{t-1}}$$

▶ Phases d'expansions et de récessions :

- Les périodes de croissance positives (au moins 2 trimestres consécutifs) sont appelées expansions
- Les périodes de croissance négatives sont appelées récessions

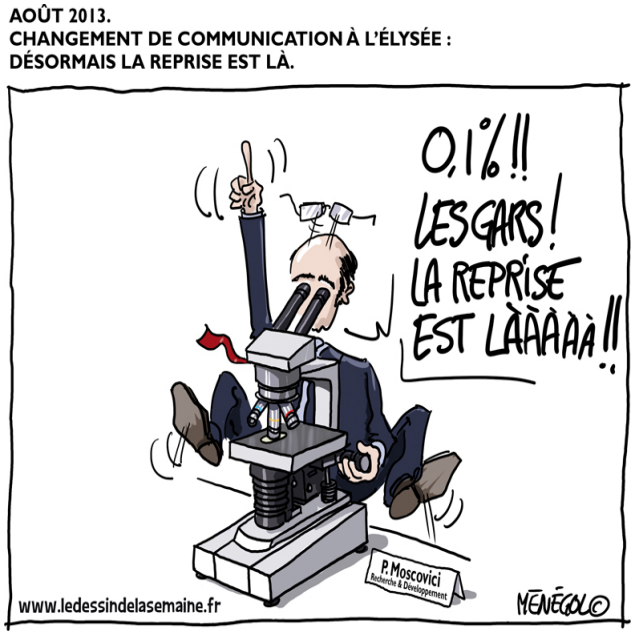


Source : Insee, 29 janvier 2021

1.2. La prévision de la croissance conjoncturelle

Les prévisions de la croissance se font

- soit à court terme, cad sur l'année en cours et l'année suivante (Direction Générale du Trésor, Ministère des Finances, FMI, OCDE, Commission Européenne, institutions privées françaises)
- soit à très court terme (prévisions conjoncturelles), cad mensuellement sur le trimestre en cours (*nowcasting*) et le trimestre suivant (*forecasting*) (Insee, Banque de France, OFCE)



Les prévisions de l'évolution du PIB français rendues publiques émanent aussi bien d'organismes internationaux ou nationaux, que d'institutions privées qui publient deux fois par an des prévisions de croissance pour l'année en cours et pour l'année à venir

- le FMI (World Economic Outlook)
- l'OCDE (Perspectives économiques de l'Organisation de Coopération et de Développement Économiques)
- la Commission Européenne (European Economic Forecast)
- la Direction Générale du Trésor
- l'INSEE (Point de conjoncture) **[site]**
- la Banque de France (Point sur la conjoncture française) **[site]**
- l'OFCE (Observatoire Français des Conjonctures Économiques)
- Institutions privées françaises et étrangères (BNP Paribas, Natixis, Crédit Agricole, Société Générale, HSBC, Citigroup, ING, . . .)

La seule prévision du gouvernement à s'inscrire dans un calendrier régulier est celle donnée chaque mois de septembre dans le projet de loi de finance (PLF).

Les données en prévision

Trois types de variables mensuelles (officielles) sont généralement utilisées en prévision :

- variables dites "hard" : **indicateurs quantitatifs** de l'activité réelle, souvent d'origine comptable : indice de production industrielle, consommation des ménages en produit manufacturés, etc.
- variables financières : disponible quotidiennement et **en temps réel** : cotation du CAC40, taux d'intérêt, prix des matières premières, taux de change, etc.
- variables dites "soft" : **indicateurs qualitatifs** comme les données d'enquêtes (de conjoncture) réalisées auprès des ménages et des chefs d'entreprise : Insee, BdF et PMI (*Purchasing Managers Index*)
  - rapidement disponibles
  - (très) peu révisées
  - indications précoces sur le passé récent et les perspectives d'évolution à court terme du comportement des acteurs économiques
- données alternatives $\Rightarrow$ Big data
  - découlant de données du web (Google Trends), des téléphones mobiles (Twitter) ...
  - très rapidement disponible (mensuelle voire hebdomadaire)
  - Problèmes de ces données : valeurs aberrantes, comportements saisonniers, fiabilité des données et des révisions ...

Tableau 1 : Date de parution des principaux indicateurs

	Date de parution	Enquêtes	Principales données « dures»
Mois 1	J0	Insee et PMI (mois 0)	
	J15	Banque de France (mois 0)	
	J30	Insee et PMI (mois 1)	Consommation en biens (mois 0)
Mois 2	J45	Banque de France (mois 1)	IPI (mois 0)
	J60	Insee et PMI (mois 2)	Consommation en biens (mois 1)
Mois 3	J75	Banque de France (mois 2)	IPI (mois 1)
	J90	Insee et PMI (mois 3)	Consommation en biens (mois 2)
Mois 4	J105	Banque de France (mois 3)	IPI (mois 2)
	J120	**Publication du PIB**	

Note de lecture : Outre les données déjà présentes dans la base J30, la base J45 contient les enquêtes de la Banque de France correspondant au 1er mois du trimestre ainsi que l'IPI du mois précédent le trimestre considéré.
Note : Ce tableau fait référence aux indicateurs conjoncturels « classiques » ; il ne mentionne pas les variables financières qui sont disponibles quotidiennement, en temps quasi réel.

INSEE : Note de conjoncture - octobre 2020. **[video]**

2. Modèles linéaires pour la prévision

Il existe trois grandes catégories de modèles linéaires pour la prévision d'une variable stationnaire, à un horizon $h > 0$:

1. Les modèles autorégressifs AR(p) et les modèles ARMA(p, q)
2. Les modèles AR et ARMA avec variables exogènes ou modèles ARX et ARMAX
3. Les modèles de régression linéaire multiple

Modèles autorégressifs AR(p) et ARMA(p, q)

Les modèles autorégressifs AR(p) et les modèles ARMA(p, q) comportent les valeurs retardées de la variable expliquée comme variable explicative, cad l'information est uniquement véhiculée par ses valeurs passées

▶ On appelle processus autorégressifs d'ordre p, noté AR(p), un processus stationnaire X_t vérifiant une relation du type :

$$Y_t = \phi_1 Y_{t-1} + \cdots + \phi_p Y_{t-p} + \varepsilon_t$$

$$\begin{aligned} Y_t - \phi_1 Y_{t-1} - \cdots - \phi_p Y_{t-p} &= \varepsilon_t \\ (1 - \phi_1 L - \cdots - \phi_p L^p) Y_t &= \varepsilon_t \\ \Phi(L) Y_t &= \varepsilon_t \end{aligned}$$

- ϕ_i : des réels, avec $|\phi_i| < 1$
- $\Phi(z) = I - \phi_1 z - \cdots - \phi_p z^p$: polynôme d'ordre p
- $\varepsilon_t \sim BB(0, \sigma_\varepsilon^2)$, cad ε_t suit un bruit blanc de moyenne nulle et d'écart-type σ_ε^2

▶ On considère un processus ARMA stationnaire d'ordre p et q, ARMA(p, q), de la forme suivante

$$\Phi(B)(X_t - \mu) = \Theta(B)\varepsilon_t \qquad \Leftrightarrow \qquad X_t = \frac{\Theta(B)}{\Phi(B)}\varepsilon_t = \Psi(B)\varepsilon_t$$

- μ : moyenne du processus
- B : opérateur de retard tel que, $\forall t$, $BX_t = X_{t-1}$, et $B^d X_t = X_{t-d}$, avec d un entier
- $\Phi(z) = I - \phi_1 z - \cdots - \phi_p z^p$: polynôme d'ordre p
- $\Theta(z) = I + \theta_1 z + \cdots + \theta_q z^q$: polynôme d'ordre q
- $\Psi(z) = I + \psi_1 z + \psi_2 z^2 + \ldots$: polynôme d'ordre infini : MA(∞)
- $(\varepsilon_t)_{t \in Z}$: processus bruit blanc centré de variance σ^2_ε

Estimation des paramètres

Il existe différentes méthodes pour estimer les paramètres des modèles linéaires AR, ARMA, ARIMA et SARIMA

- la méthode des MCO (OLS, *Ordinary Least Squares*) :
- l'algorithme de Burg : pas de formule analytique $\Rightarrow$ optimisation numérique via l'algorithme de Durbin-Levinson (méthode récursive)
- les équations de Yule-Walker
- l'estimation par le maximum de vraisemblance (MLE, *Maximum Likelihood Estimation*)
- la méthode de la somme conditionnelle des carrés (CSS, *Conditional Sum of Squares*)
- l'algorithme de Hannan-Rissanen ou l'algorithme de Hyndman et Khandakar
- la méthode des moindres carrés non linéaires (NLS, *Nonlinear Least Squares*)

Estimation & R packages

- Le package **stats** : la fonction `ar()` estime les modèles AR à partir des méthodes OLS, Yule-Walker, Burg et MLE
- Le package **stats** : la fonction `arima()` estime les modèles ARIMA (SARIMA et ARIMAX) à partir des méthodes MLE, CSS et CSS-MLE
- Le package **tseries** : la fonction `arma()` estime les modèles ARMA à partir de la méthode CSS et de l'algorithme de Hannan-Rissanen pour l'initialisation
- Le package **forecast** : la fonction `Arima()` estime les modèles AR et SARIMA à partir des méthodes MLE et CSS
- Le package **forecast** : la fonction `auto.arima()` estime les modèles AR et SARIMA à partir de l'algorithme de Hyndman et Khandakar

```
dyy <- diff(yy, differences = 1)
```

Modèle AR(1)

```
out1=ar(dyy,aic=FALSE,order.max=1, method="ols")
out1=ar(dyy,aic=FALSE,order.max=1, method="yule-walker")
out1=ar(dyy,aic=FALSE,order.max=1, method="burg")
out1=ar(dyy,aic=FALSE,order.max=1, method="mle")

library(forecast)
out5=Arima(dyy,order=c(1,0,0),seasonal=list(order=c(0,0,0),period=12),lambda=1)

library(tseries)
out6=arma(dyy,order=c(1,0))
```

Modèle ARMA(1,1)

```
out1=arima(dyy,order=c(1,0,1), method="CSS")
out2=arima(dyy,order=c(1,0,1), method="ML")
out3=arima(dyy,order=c(1,0,1), method="CSS-ML")

library(tseries)
out4=arma(dyy,order=c(1,1))

library(forecast)
out5=Arima(dyy,order=c(1,0,1),seasonal=list(order=c(0,0,0),period=12),lambda=1)
```

Table: Estimations de modèles AR(1) et ARMA(1,1).

Méthodes	OLS	Yule-Walker	Burg	MLE	MLE (forecast)	CSS (tseries)
AR model						
coef. AR(1)	0.5878	0.5878	0.5882	0.5863	0.5863	0.5878
σ^2	317.1	318.9	316.3	316.3	318.7	–
Méthodes	CSS	MLE	CSS-MLE	CSS (tseries)	MLE (forecast)	
ARMA model						
coef. AR(1)	0.9501	0.9480	0.9480	0.9501	0.9480	
s.e. AR(1)	0.0222	0.0216	0.0216	0.0222	0.0216	
coef. MA(1)	-0.6359	-0.6360	-0.6360	-0.6359	-0.6360	
s.e. MA(1)	0.0453	0.0454	0.0454	0.0453	0.0454	
σ^2	239.7	238.8	238.8		241.6	

Il est possible de sélectionner le nombre de retards des modèles AR(p) et ARMA(p, q) de manière automatique à partir de critères d'information AIC, AICc et BIC ou bien de l'algorithme de Hyndman et Khandakar

Modèle AR(p)

```
out1=ar(dyy, aic=TRUE, order.max=12)
```

Modèle ARMA(p, q)

```
library(forecast)
out2=auto.arima(dyy,d=0, D=0, max.P=0, max.Q=0)
```

Modèle SARIMA$(p, d, q)(P, D, Q)_S$

```
library(forecast)
out3=auto.arima(yy)
```

Les modèles ARX et ARMAX

Les modèles ARMAX (*ARMA model with eXogenous variables*) s'écrivent

$$Y_t = \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{i=0}^{k} \sum_{j=0}^{r} \delta_j X_{i,t-j} + \sum_{j=0}^{q} \varepsilon_{t-j}$$

- ε_t est un processus homoscédastique non autocorrélé
- p : retards de la variable endogène retardée (généralement $p = 1$ ou 4 ou 12)
- q : ordre du plynôme MA. Si $q = 0 \Rightarrow$ modèle ARX
- X : vecteur de variables explicatives ou régresseurs
- k : nombre de régresseurs
- r : retards des variables explicatives (généralement $r = 0$ ou 1)

Les packages R

- Le package **stats** permet d'estimer un modèle ARMAX avec la fonction `arima()` et l'option `xreg`
- Le package **forecast** permet d'estimer un modèle ARMAX ou SARIMAX avec les fonctions `Arima()` et `auto.arima()` puis l'option `xreg`
- Le package **gets** permet d'estimer un modèle ARX avec la fonction `arx()`

Le package **gets** permet d'estimer un modèle ARX avec la fonction `arx()`

```
library(readxl)
library(tidyverse)
library(lgarch)
library(gets)

dlbase <- read_excel("c://R//data//database.xlsx", sheet = "dlbase")
dlbase <- data.frame(dlbase)
training_dlbase <- dlbase
data.frame(training_dlbase)

# convert tibble in matrix for the function arx
class(dlbase[,2:4])      #tibble
mX = data.matrix(training_dlbase[,2:4])

# ARX model with AR(1)
Model01 <- arx(training_dlbase$Ddemand, mc = T, ar = 1, mxreg = mX[, 1:3],
vcov.type = "ordinary")
Model01
```

```
Date: Mon Feb 10 15:37:30 2020
Dependent var.: training_dlbase$Ddemand
Method: Ordinary Least Squares (OLS)
Variance-Covariance: Ordinary
No. of observations (mean eq.): 201
Sample: 2 to 202

Mean equation:

                              coef   std.error  t-stat   p-value
mconst                  0.00061978  0.00054801  1.1310   0.25944
ar1                     0.46776252  0.06063850  7.7140 5.935e-13
evolution.des.effectifs -0.00139739  0.00030294 -4.6127 7.145e-06
previsions.des.effectifs 0.00048663  0.00019267  2.5257   0.01233

Diagnostics and fit:

                   Chi-sq df    p-value
Ljung-Box AR(2)   20.5207  2 3.4994e-05
Ljung-Box ARCH(1)  9.1932  1 2.429le-03

SE of regression   0.00632
R-squared          0.46316
Log-lik.(n=201)  734.54272
```

Les modèles de régression linéaire multiple

Les modèles de régression linéaire multiple s'écrivent

$$Y_t = \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{i=0}^{k} \sum_{j=0}^{r} \delta_j X_{i,t-j} + \varepsilon_t$$

- ε_t est un processus homoscédastique non autocorrélé
- p : retards de la variable endogène retardée (généralement $p = 1$ ou 4 ou 12)
- X : vecteur de variables explicatives ou régresseurs
- k : nombre de régresseurs
- r : retards des variables explicatives (généralement $r = 0$ ou 1)

Le package **stats** permet d'estimer un modèle de régression linéaire multiple avec la fonction `lm()`

```
library(readxl)
dlbase <- read_excel("c://R//data//database.xlsx", sheet = "dlbase")
dlbase <- data.frame(dlbase)
training_dlbase <- dlbase
data.frame(training_dlbase)

model <- lm(Ddemand ~Evoleff + Preveff + Dprod, data = training_dlbase)
summary(model)
confint(model)
anova(model)
plot(model)
```

Critères d'information

Les critères d'information (*information criteria*, IC) permettent de déterminer le nombre de variables explicatives parmi k variables explicatives disponibles

Les IC proposent une estimation de la perte d'information lorsqu'on utilise le modèle considéré pour représenter le processus qui génère les données.

Les IC ne fournissent pas un test de modèle dans le sens d'une hypothèse nulle, cad que ce test ne dit rien de la qualité absolue du modèle.

Le modèle M qui sera sélectionné est celui qui minimise

$$\begin{aligned} M_{IC} &= \arg\min_{M} IC(T,k) \\ IC(T,k) &= -2\frac{Log\ L}{T} + \frac{k.g(T)}{T} \end{aligned}$$

- T : taille de l'échantillon
- k : nombre de variables explicatives ($k = p+q$ si modèle ARMA(p,q))
- $g(T)$: fonction de pénalité
- L : la log-vraisemblance, définie par : $L = -(T/2)\left(1+\ln(2\pi)+\ln(\hat{\varepsilon}'\hat{\varepsilon}/T)\right)$

Le critère AIC (*Akaike IC*) tire son origine de la divergence de Kullback-Leibler (ou divergence K-L ou entropie relative) issue de la théorie de l'information et en théorie des probabilités, mesurant la dissimilarité entre deux distributions de probabilités F et G

Soient f et g deux densités de probabilités, et supposons que f est la vraie loi inconnue, g une approximation, alors la divergence, ou perte d'information pour utiliser g à la place de f est définie par le critère d'information de Kullback-Leibler (KLIC)

$$\begin{aligned} KLCI(f;g) &= D_{KL}(F\|G) = \int f(t) ln\left(\frac{f(t)}{g(t)}\right) dt \\ &= \int ln(f(t))f(t)dt - \int ln(g(t))f(t)dt = E_f[ln(f(t)] - E_f[ln(g(t)] \end{aligned}$$

On peut interpréter cette quantité comme étant une mesure du risque impliqué par la possibilité de choisir une fausse densité g au lieu de la bonne densité f.

Si les deux densités font partie d'une classe paramétrique et que g contient k paramètres, il est possible d'obtenir une approximation de $KLCI(f;g)$ en utilisant la méthode du maximum de vraisemblance dans un échantillon de taille T

Le critère d'information de Akaike (AIC) (Akaike, 1973)

L'AIC représente un compromis entre le biais, diminuant avec le nombre de paramètres k, et la parcimonie, volonté de décrire les données avec le plus petit nombre de paramètres possibles

$$AIC = -2\frac{Log\ L}{T} + 2\frac{k}{T} \qquad \text{avec } g(T) = 2$$

Lorsque le nombre de paramètres k est grand par rapport au nombre d'observations T, cad si $T/k < 40$, il est recommandé d'utiliser l'AIC corrigé (AIC_c) (Hurvich et Tsai, 1995)

$$AIC_c = AIC + \frac{2k(k+1)}{T-k-1}$$

Le critère d'information de Schwarz (SIC) ou Bayésien (BIC) (Schwarz, 1978) est issu de la théorie Bayésienne

$$SIC = BIC = -2\frac{Log\ L}{T} + k\frac{\ln(T)}{T} \qquad \text{avec } g(T) = \ln(T)$$

Il est plus parcimonieux que le critère AIC puisqu'il pénalise plus fortement le nombre de variables présentent de le modèle.

Le critère d'information de Hannan-Quinn (HQ) (Hannan et Quinn, 1979)

$$HQ = -2\frac{Log\ L}{T} + 2k\frac{\ln(\ln(T))}{T} \qquad \text{avec } g(T) = 2\ln(\ln(T))$$

Il peut être considéré comme un intermédiaire entre le AIC et le BIC.
Le critère HQ adoucit quelque peu la sévérité de la fonction de pénalité du BIC relativement à la croissance de la taille de l'échantillon T tout en maintenant une forte convergence dans l'identification de l'ordre réel du modèle k.

Sélection de variables

Il existe différentes méthodes algorithmiques de sélection de variables :

- Méthode de recherche exhaustive : nombre de sous-modèles possibles pour $N+1$ variables explicatives : $2^{N+1} \Rightarrow$ coût algorithmique prohibitif pour N grand, même modéré
- Méthode de recherche ascendante (*forward*) : on part du modèle constant (aucune variable explicative), et on ajoute de manière séquentielle les variables qui permettent d'optimiser le critère (maximiser le $\overline{R}^2$, minimiser les autres critères comme les critères d'information AIC ou BIC), ou d'accepter la validité du modèle (test)
- Méthode de recherche descendante (*backward*) : même principe, mais on part du modèle complet et on élimine une à une les variables.
- Méthode de recherche progressive ou pas-à-pas (*stepwise*) : algorithme mixte qui ajoute ou supprime une variable du modèle à chaque étape

Sélection et donc estimation finalement peu satisfaisantes, instables, surtout inadaptées au cas $p \geq N$

Big data

Il existe différentes approches (économétriques et statisques) pour gérer les **grandes bases de données** (*big data*) :

- La méthode GETS (*General-to-specific*) : méthode automatique de *backward selections*
- Les régressions pénalisées ou régularisées
  - les régressions Ridge
  - les régression LASSO (*Least Absolute Shrinkage and Selection Operator*)
  - les régressions Elastic-Net
- L'analyse en composantes principales (ACP) et l'ACP parcimonieuse (*sparse PCA*)
- Les modèles à facteurs dynamiques (DFM, *Dynamic Factor Models*)
- Les techniques d'apprentissage automatique (*machine learning*)
  - les forêts aléatoires (*random forests*)
  - les réseaux de neuronnes

Le package **MASS** permet de sélectionner les variables à partir des approches *forward*, *backward* et *stepwise* avec la fonction `stepAIC()`

```
library(readxl)
dlbase <- read_excel("c://R//data//database.xlsx", sheet = "dlbase")
dlbase <- data.frame(dlbase)
training_dlbase <- dlbase
data.frame(training_dlbase)

library(MASS)
fit <- lm(Ddemand ~Evoleff + Preveff + Dprod, data=training_dlbase)
step1 <- stepAIC(fit, direction="both")
step2 <- stepAIC(fit, direction="forward")
step3 <- stepAIC(fit, direction="backward")
```

Le package **olsrr** permet de sélectionner les variables à partir des approches *backward* et *stepwise*

```
model <- lm(Ddemand ~Evoleff + Preveff + Dprod, data=training_dlbase)
library(olsrr)
forward <- ols_step_backward_aic(model, details=TRUE)
stps <- ols_step_both_p(model, details=TRUE)
```

Subset Selection in Regression

Miller (1990) propose une approche alternative pour la sélection de variables avec la *Subset selection* ou *Best subsets regression*.

Cette méthode consiste à tester toutes les combinaisons possibles des variables prédictives, et sélectionne le meilleur modèle en fonction de critères statistiques (R^2, BIC, Cp de Mallows, ...)

Le package **leaps** permet de sélectionner les variables à partir de l'approche *best subsets* avec la fonction `regsubsets()`

```
library(leaps)
leaps <- regsubsets(Ddemand ~Evoleff + Preveff + Dprod, data=training_dlbase,
nbest=1, method=c("exhaustive"))
summary(leaps)

# Choosing the optimal model
res.sum <- summary(leaps)
data.frame(
Adj.R2 = which.max(res.sum$adjr2),
CP = which.min(res.sum$cp),
BIC = which.min(res.sum$bic))

# plot a table of models showing variables in each model
plot(leaps, scale="adjr2", main = "Adjusted R^2")

# nbest: Number of subsets of each size to report
# nvmax: maximum size of subsets to examine
# method=c("exhaustive","backward", "forward", "seqrep"): exhaustive search, forward selection, backward
selection or sequential replacement to search
```

La sélection de variables

```
Subset selection object
Call: regsubsets.formula(Ddemand ~ Evoleff + Preveff + Dprod, data = training_dlbase,
   nbest = 1, nvmax = 3, method = c("exhaustive"))
3 Variables  (and intercept)
      Forced in Forced out
Evoleff    FALSE     FALSE
Preveff    FALSE     FALSE
Dprod      FALSE     FALSE
1 subsets of each size up to 3
Selection Algorithm: exhaustive
      Evoleff Preveff Dprod
1  ( 1 ) "*"    " "     " "
2  ( 1 ) "*"    "*"     " "
3  ( 1 ) "*"    "*"     "*"
```

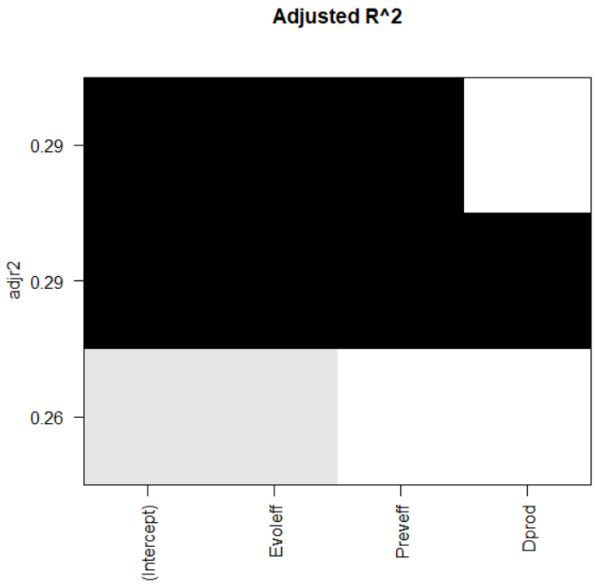
le meilleur modèle à 1 variable contient Evoleff

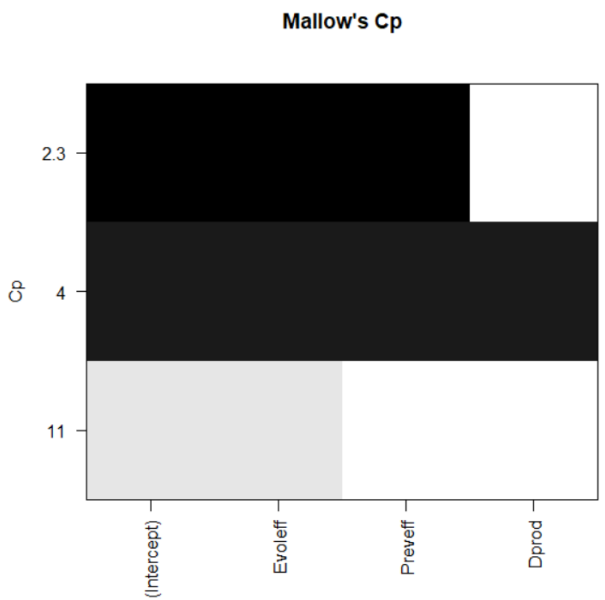
le meilleur modèle à 2 variables contient Evoleff+Preveff

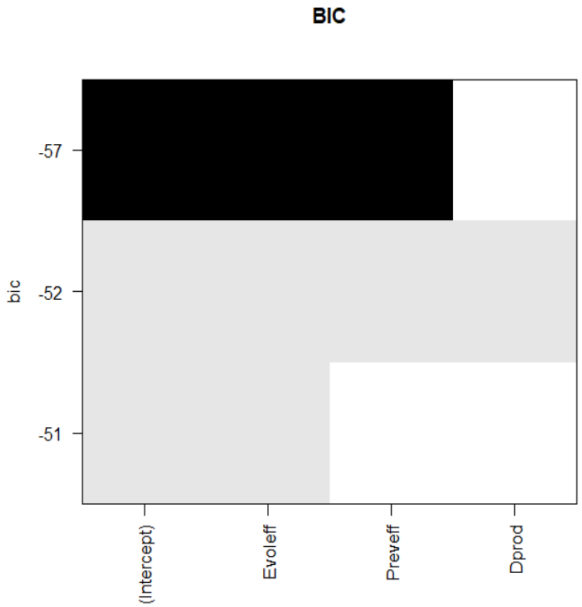
	Adj.R2	CP	BIC
1	2	2	2

les 3 critères concluent que le meilleur modèle est à 2 variables

La sélection de variables







Une approche alternative de sélection de variables est l'approche automatique de sélection de modèles basée sur les critères d'information (*IC-based model selection*.

Le package **glmulti** permet d'appliquer cette approche avec la fonction `glmulti()`.
Voir Calcagno et de Mazancourt (2010) **[online]**

```
library(rJava)          # install Java-64bits if you use R-64bits
library(glmulti)
multi <- glmulti(Ddemand ~Evoleff + Preveff + Dprod, data =
training_dlbase, level = 1, method = "h", crit = "aic", confsetsize = 2,
plotty = F, report = F, fitfunction = "lm")
summary(multi)

# method: "h" exhaustive approach - "g": genetic algorithm
# crit: IC: "aic", "aicc", "bic", "qaic", "qaicc"
```

Prévisions pour des modèles linéaires

Forecasting et Nowcasting

En prévision de court terme, l'horizon de prévision h permet de considérer 2 types d'approches :

- lorsque $h = 0 \Rightarrow$ Nowcasting ou prévision immédiate
- lorsque $h > 0 \Rightarrow$ Forecasting ou prévision (court terme : $h_{\max} = 6$)

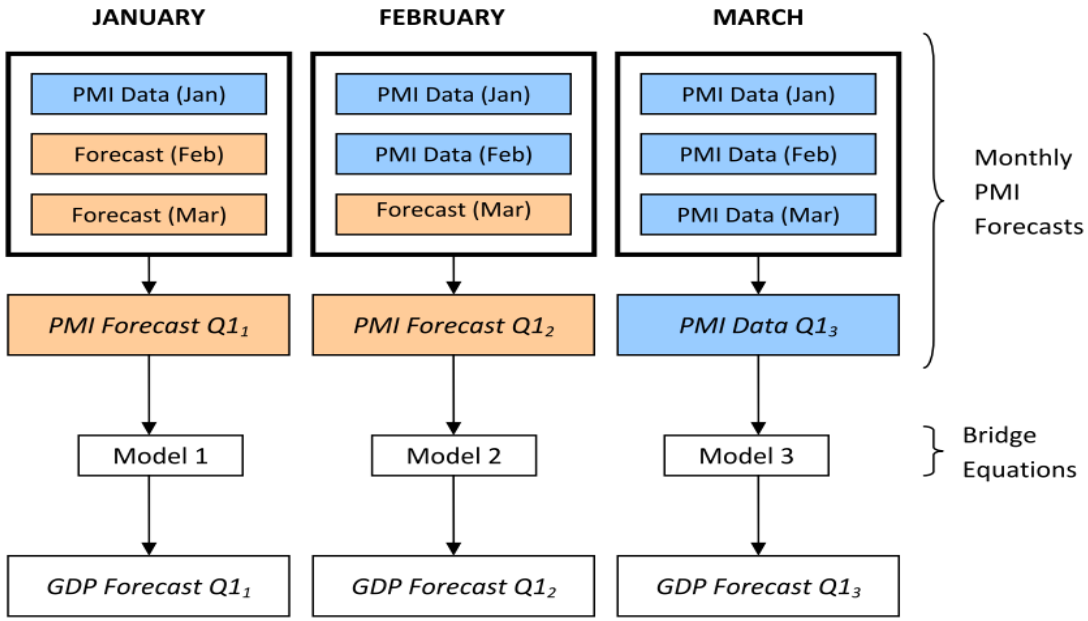
Nowcasting ou prévision immédiate

Le Nowcasting de la période en cours considère la prévision pour un horizon $h = 0$, principalement pour les modèles trimestriels

Les prévisions du trimestre en cours consistent à prévoir la croissance à la fin de chaque mois de ce trimestre (fin du mois 1, fin du mois 2 et fin du mois 3)

Il existe principalement 3 types de modèles dans le cadre du nowcasting pour traiter le problème des fréquences mixtes et des observations manquantes

- Les modèles bridges ou modèles d'étalonnage
- Les modèles par la méthode *blocking*
- Les modèles MIDAS (*MIxed DAta Sampling*) (en Annexe)



Les modèles bridges ou modèles d'étalonnage

Définition

Les **modèles bridges** ou **modèles d'étalonnage** lient une variable dépendante trimestrielle Y_t avec des variables obtenues par moyenne trimestrielle X_t^Q

$$Y_t = \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=0}^{k} \beta_j X_{t-j}^Q + \varepsilon_t$$

avec $X_t^Q = (1/3)(X_{M3}^Q + X_{M2}^Q + X_{M1}^Q)$

Lorsque les données de X_t sont manquantes pour le 2ème et/ou 3ème mois il existe 3 manières de faire

- Utiliser un modèle AR(p) mensuel pour prévoir les valeurs maqnuantes, pour un horizon $h = 1$ ou 2

$$X_t = \mu + \sum_{i=1}^{p} \phi_p X_{t-p} + \varepsilon_t$$

- Prendre la moyenne des 3 derniers mois connus

$$X_t^Q = (1/3)(X_{M2}^Q + X_{M1}^Q + X_{M3}^{Q-1}) \qquad \text{ou} \qquad X_t^Q = (1/3)(X_{M1}^Q + X_{M3}^{Q-1} + X_{M2}^{Q-1})$$

- Considérer que l'information du mois est celle du trimestre

$$X_t^Q = X_{M1}^Q \qquad \text{ou} \qquad X_t^Q = (1/2)(X_{M2}^Q + X_{M1}^Q)$$

Les modèles par la méthode *blocking*

Les modèles par la méthode *blocking* estiment pour chaque trimestre autant de modèles bridges qu'il y a de mois dans le trimestre, en utilisant à chaque fois la seule information disponible jusqu'alors.

Les séries mensuelles sont « trimestrialisées » en regroupant les données de chaque 1er, 2ème et 3ème mois de chaque trimestre :

$$X_t^{M1} = \text{données des mois de janvier, avril, juillet et octobre}$$

$$X_t^{M2} = \text{données des mois de février, mai, août et novembre}$$

$$X_t^{M3} = \text{données des mois de mars, juin, septembre et décembre}$$

On obtient ainsi des modèles pour chaque mois du trimestre :

$$\begin{aligned}
(\text{M1}) : Y_t &= \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} \\
&\quad + \sum_{s=1}^{k} \left(\beta_{1,s} X_t^{M1} + \beta_{2,s} X_{t-1}^{M3} + \beta_{3,s} X_{t-1}^{M2} + \beta_{4,s} X_{t-1}^{M1} \right) + \varepsilon_{1,t} \\
(\text{M2}) : Y_t &= \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} \\
&\quad + \sum_{s=1}^{k} \left(\beta_{1,s} X_t^{M2} + \beta_{2,s} X_t^{M1} + \beta_{3,s} X_{t-1}^{M3} + \beta_{4,s} X_{t-1}^{M2} \right) + \varepsilon_{2,t} \\
(\text{M3}) : Y_t &= \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} \\
&\quad + \sum_{s=1}^{k} \left(\beta_{1,s} X_t^{M3} + \beta_{2,s} X_t^{M2} + \beta_{3,s} X_t^{M1} + \beta_{4,s} X_{t-1}^{M3} \right) + \varepsilon_{3,t}
\end{aligned}$$

3. Prévision par processus linéaires

Les processus linéaires sont particulièrement bien adaptés pour la prévision des séries chronologiques car ils permettent d'utiliser de manière optimale l'information contenue dans le processus sous la forme d'autocorrélation linéaire.

Définition

Un processus $(X_t)_{t\in Z}$ est un processus linéaire s'il admet une décomposition de la forme suivante, $\forall t \in Z$

$$X_t = \sum_{i=-\infty}^{\infty} = \psi_i \varepsilon_{t-i}$$

- ψ_i : coefficients sommables, cad $\sum_{i=-\infty}^{\infty} |\psi_i| < \infty$
- ε_t : processus bruit blanc

La justification de l'utilisation extensive en prévision des processus linéaires provient du théorème de Wold (1938) qui montre que tout processus fortement stationnaire peut s'écrire sous la forme d'un processus linéaire.

Si on observe une trajectoire $(x_1, \ldots, x_t)$ que l'on suppose engendrée par un processus linéaire $(X_t)_{t \in Z}$, on connait alors le meilleur prédicteur $\widehat{X}_t(h)$, au sens de la plus faible erreur quadratique moyenne.

Le prédicteur $\widehat{X}_T(h)$ minimisant l'erreur quadratique moyenne (MSE) est le prédicteur des moindres carrés défini par

$$\widehat{X}_t(h) = \widehat{X}_{t+h} = E(X_{t+h}|I_t)$$

- $E(X_{t+h}|I_t)$: espérance conditionnelle de X prise au temps $(t+h)$, conditionnellement à l'information I_t
- I_t : ensemble d'information au temps t apporté par les variables $(X_1, \ldots, X_t)$
- La prévision pour un processus bruit blanc de moyenne nulle est zéro :

$$E(\varepsilon_{t+h}|I_t) = 0 \qquad \forall h > 0$$

Prévision modèle AR(1)

Définition

On considère un **processus autoregressif stationnaire d'ordre 1**, AR(1), de moyenne nulle, de la forme suivante

$$X_t = \phi X_{t-1} + \varepsilon_t \qquad \Leftrightarrow \qquad X_t - \phi X_{t-1} = \varepsilon_t$$

- ε_t : processus bruit blanc
- ϕ : paramètre autorégressif tel que $|\phi| < 1$

Le prédicteur à l'horizon h, noté $\widehat{X}_t(h)$, $\forall t$, est donné par

$$\widehat{X}_t(h) = \widehat{X}_{t+h} = E(X_{t+h}|I_t) \qquad \text{avec } I_t = (X_1, X_2, \ldots, X_t, \varepsilon_t)$$

$$
\begin{array}{lll}
\text{pour h = 1} & X_{t+1} = \phi X_t + \varepsilon_{t+1} \\
& \widehat{X}_{t+1} = \widehat{X}_t(1) = E(X_{t+1}|I_t) = \phi X_t \\
\\
\text{pour h = 2} & X_{t+2} = \phi X_{t+1} + \varepsilon_{t+2} \\
& \widehat{X}_{t+2} = \widehat{X}_t(2) = E(X_{t+2}|I_t) = \phi \widehat{X}_{t+1} = \phi^2 X_t \\
\\
\text{pour h > 1} & \widehat{X}_{t+h} = \phi \widehat{X}_{t+h-1} = \phi^h X_t
\end{array}
$$

Lorsque l'horizon $h > 1$, on remplace :

- les valeurs inconnues de la série par les valeurs prédites aux pas précédents, et
- les valeurs inconnues des résidus par leur moyenne conditionelle, à savoir zéro.

Définition

La **prévision** pour un **processus AR(1)** avec $h > 1$ est donnée par

$$\widehat{X}_{t+h} = \phi \widehat{X}_{t+h-1} = \phi^h X_t$$

Lorsque $h \to \infty \Rightarrow \widehat{X}_t(h)$ converge vers son espérance non conditionelle $E(X_t)$ (égale à 0 ici). La vitesse de convergence est ici inversement proportionnelle à la valeur du paramètre autorégressif ϕ.

Définition

L'**erreur de prévision** pour un **processus AR(1)** avec $h > 1$ est donnée par (voir Annexe)

$$\widehat{e}_{t+h} = \varepsilon_{t+h} + \phi \varepsilon_{t+h-1}$$

Prévision modèle ARMA(p, q)

Définition

On considère un **processus ARMA stationnaire** d'ordre p et q, ARMA(p, q), de la forme suivante

$$\Phi(B)(X_t - \mu) = \Theta(B)\varepsilon_t \qquad \Leftrightarrow \qquad X_t = \frac{\Theta(B)}{\Phi(B)}\varepsilon_t = \Psi(B)\varepsilon_t$$

- μ : moyenne du processus
- B : opérateur de retard tel que, $\forall t$, $BX_t = X_{t-1}$, et $B^d X_t = X_{t-d}$, avec d un entier
- $\Phi(z) = I - \phi_1 z - \cdots - \phi_p z^p$: polynôme d'ordre p
- $\Theta(z) = I + \theta_1 z + \cdots + \theta_q z^q$: polynôme d'ordre q
- $\Psi(z) = I + \psi_1 z + \psi_2 z^2 + \ldots$: polynôme d'ordre infini : MA(∞)
- $(\varepsilon_t)_{t \in Z}$: processus bruit blanc centré de variance σ_ε^2

Le prédicteur à l'horizon h, noté $\widehat{X}_t(h)$, $\forall t$, est donné par

$$\widehat{X}_t(h) = \widehat{X}_{t+h} = E(X_{t+h}|I_t) \qquad \text{avec } I_t = (X_1, X_2, \ldots, X_t, \varepsilon_1, \varepsilon_2, \ldots \varepsilon_t)$$

$$\begin{aligned}
\text{pour h = 1} \qquad X_{t+1} &= \varepsilon_{t+1} + \psi_1\varepsilon_t + \psi_2\varepsilon_{t-1} + \ldots \\
\widehat{X}_{t+1} &= E(X_{t+1}|I_t) = \psi_1\varepsilon_t + \psi_2\varepsilon_{t-1} + \ldots \\
\\
\text{pour h = 2} \qquad X_{t+2} &= \varepsilon_{t+2} + \psi_1\varepsilon_{t+1} + \psi_2\varepsilon_t + \psi_3\varepsilon_{t-1} + \ldots \\
\widehat{X}_{t+2} &= E(X_{t+2}|I_t) = \psi_2\varepsilon_t + \psi_3\varepsilon_{t-1} + \ldots \\
\\
\text{pour h > 1} \qquad \widehat{X}_{t+h} &= \sum_{i=0}^{\infty} \psi_{h+i}\varepsilon_{t-i}
\end{aligned}$$

Définition

La **prévision** pour un **processus ARMA(p, q)** avec $h > 1$ est donnée par

$$\widehat{X}_{t+h} = \sum_{i=0}^{\infty} \psi_{h+i} \varepsilon_{t-i}$$

Définition

L'**erreur de prévision** pour un **processus ARMA(p, q)** avec $h > 1$ est donnée par

$$\widehat{e}_{t+h} = \sum_{i=0}^{h-1} \psi_i \varepsilon_{t+h-i} \qquad \text{avec } \psi_0 = 1 \text{ et } \Psi(B) = \frac{\Theta(B)}{\Phi(B)}$$

On montre alors facilement que l'erreur de prévision $\widehat{e}_{t+h}$ a les propriétés suivantes :

- espérance nulle : $E[\widehat{e}_{t+h}] = 0$
- variance telle que

$$\begin{aligned} E[\widehat{e}_{t+h}]^2 &= E\left[\sum_{i=0}^{h-1} \psi_i \varepsilon_{t+h-i}\right]^2 \\ &= \left(\sum_{i=0}^{h-1} \psi_i E[\varepsilon_{t+h-i}]\right)^2 \\ &= \sigma_\varepsilon^2 \sum_{i=0}^{h-1} \psi_i^2 = \sigma_\varepsilon^2 \left(1 + \sum_{i=1}^{h-1} \psi_i^2\right) \end{aligned}$$

Evolution de la variance pour les processus linéaires (AR et ARMA) :

- la variance d'erreur de prévision croît avec l'horizon h
- la valeur σ_ε^2 obtenue pour $h = 1$ est la plus petite
- lorsque $h \to \infty$, la variance d'erreur tend vers

$$E(e_{T+h}^2)_\infty = \sigma_\varepsilon^2 \lim_{h\to\infty} (1 + a_1^2 + \cdots + a_{h-1}^2 = \sigma_\varepsilon^2 \left(1 + \sum_{i=1}^{\infty} a_i^2\right) = \sigma_\varepsilon^2(0)$$

Sous l'hypothèse supplémentaire de connaissance de la loi du processus d'erreur de prévision, on peut calculer un intervalle de confiance pour la prévision.

Par exemple, dans la cas d'un processus Gaussien, on obtient l'intervalle de confiance suivant pour X_{T+h}, au niveau de confiance $1-\alpha$

$$\widehat{X}_{t+h} \in \left[\widehat{X}_T(h) \pm t_{1-\alpha/2} \sigma_\varepsilon \sqrt{\sum_{i=0}^{h-1} \psi_i^2} \right]$$

- avec $t_{1-\alpha/2}$: quantile de la loi d'ordre $(1-\alpha)$ de la loi Normale centrée réduite

On suppose ici que les paramètres du processus sont connus mais en pratique on utilise les valeurs des paramètres estimés, sans toutefois rajouter d'incertitude sur le prédicteur due à la variabilité des estimateurs.

Importance de l'horizon de prévision

Quel horizon de prévision ? Court, moyen ou long terme ?

Si le processus est stationnaire, on montre que la loi de distribution conditionnelle converge vers la loi de distribution non conditionnelle lorsque l'horizon tend vers l'infini

$$\mathcal{L}(X_{T+h}|I_T) \underset{h\to\infty}{\to} \mathcal{L}(X_1)$$

Seule la vitesse de convergence différe en fonction de la mémoire de processus : plus la mémoire d'un processus est courte, plus la vitesse de convergence est grande, et inversement

Importance de l'horizon de prévision

Lorsqu'on utilise en prévision un processus ARMA, il est particulièrement recommandé que l'horizon soit de très court terme : $h = 1$ ou $h = 2$.

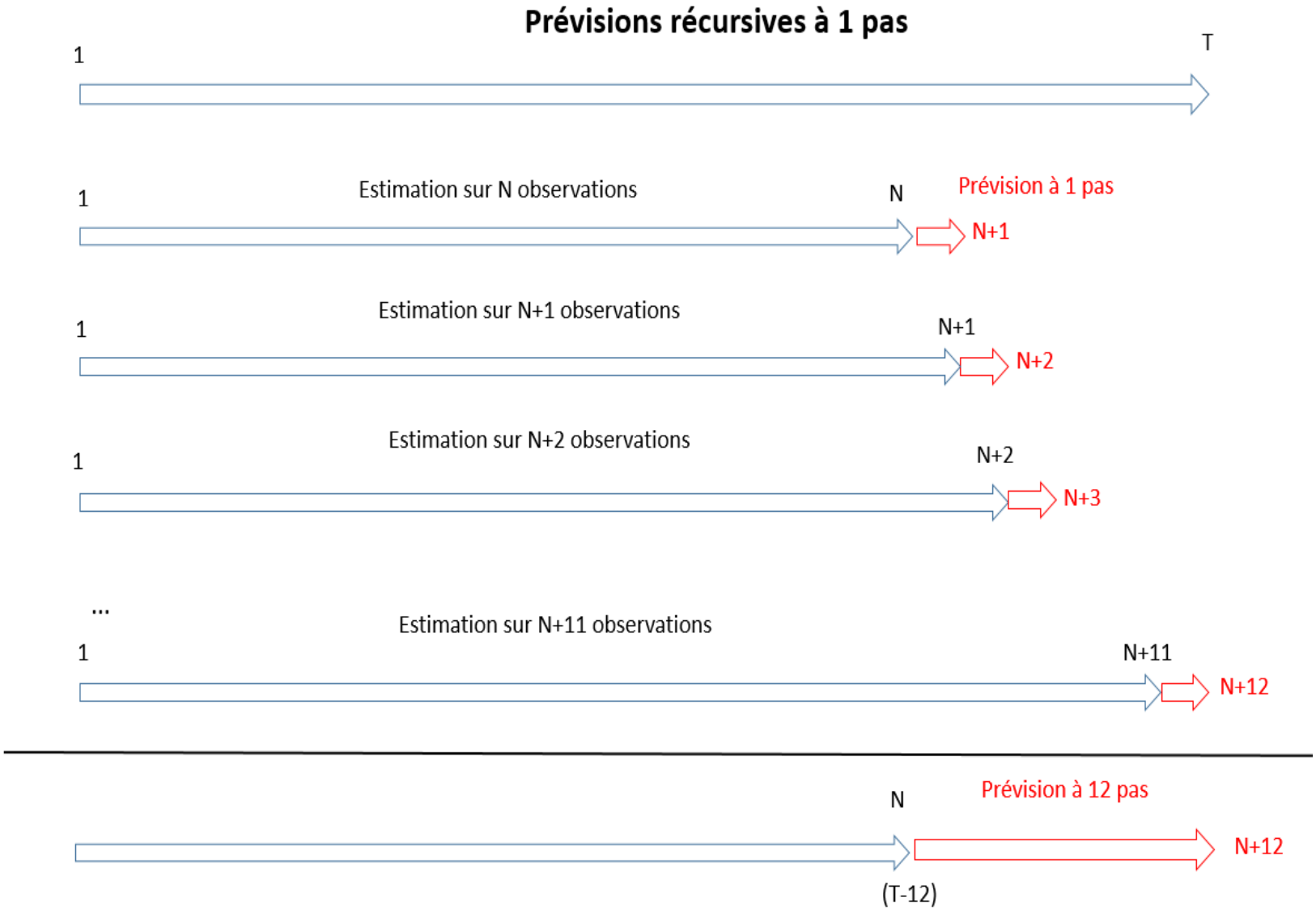
En effet, ce type de processus étant à mémoire courte, au bout de quelques pas le prédicteur va être égal à la moyenne non conditionnelle de la série, ce qui est très peu informatif et toujours décevant pour un praticien.

Ainsi, le prédicteur retourne vers la moyenne non conditionnelle très rapidement.

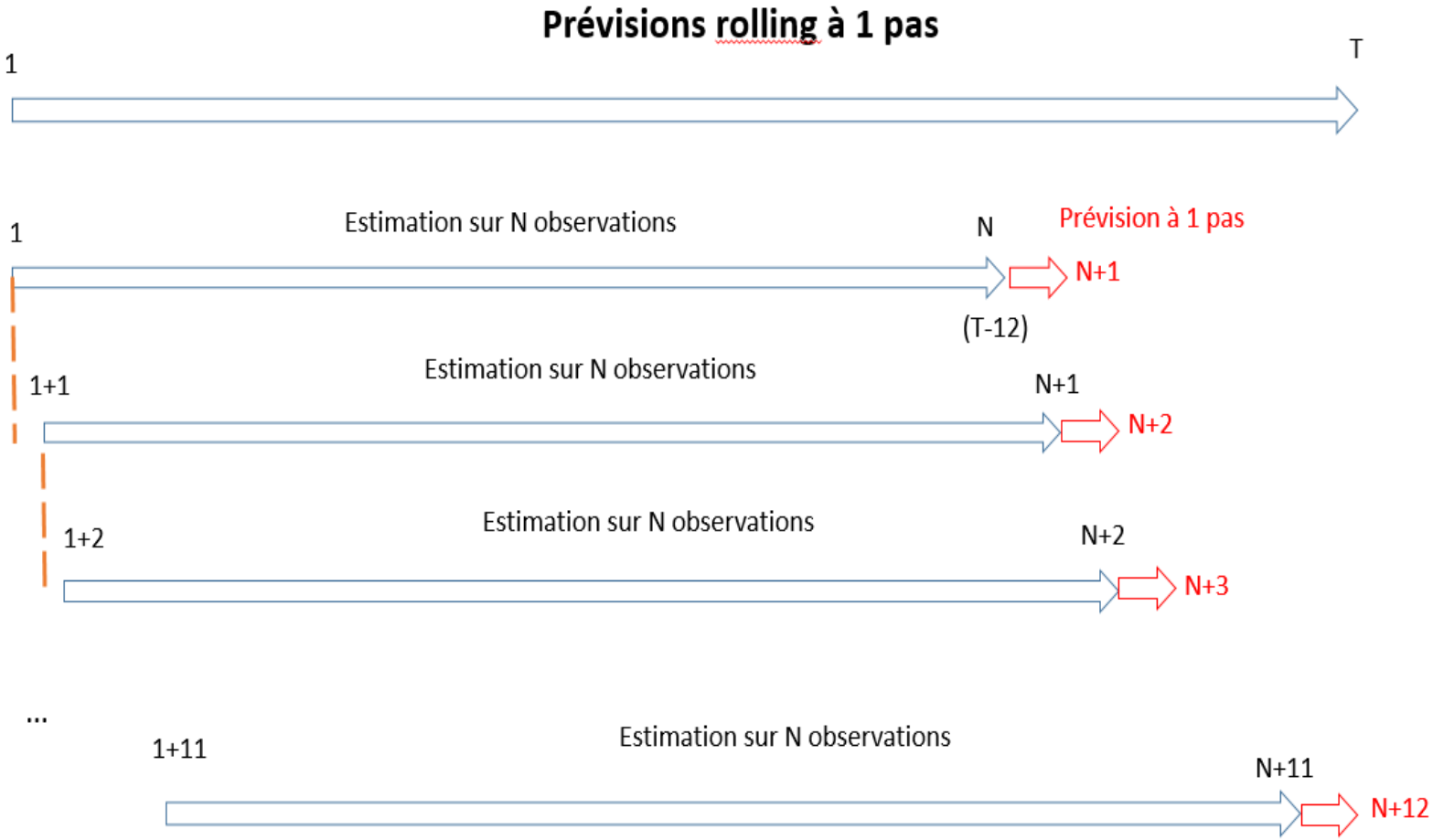
Si l'on désire effectuer des prévisions sur un horizon de plus long terme, les processus à mémoire longue fournissent une alternative plus intéressante (évidemment si la persistance est présente dans la série).

Il existe deux principales manières de faire des prévisions en fonction de l'horizon h et du pas i

- les prévisions récursives (*recursive forecasting*)
- les prévisons sur fenêtre roulante (*rolling forecating*)


Prévisions récursives à 1 pas
1
T
1
Estimation sur N observations
N
Prévision à 1 pas
N+1
1
Estimation sur N+1 observations
N+1
N+2
1
Estimation sur N+2 observations
N+2
N+3
...
1
Estimation sur N+11 observations
N+11
N+12
N
Prévision à 12 pas
N+12
(T-12)

Prévisions sur fenêtre roulante



Le package R forecast permet d'estimer et de prévoir à partir d'un modèle AR, ARIMA ou SARIMA avec les fonctions `Arima`, `auto.arima` et `forecast`

 # estimation d'un SARIMA $(0,1,1)(0,1,1)_{12}$
```
out=Arima(yy,order=c(0,1,1),seasonal=list(order=c(0,1,1),period=12),lambda=0)
show(out)
```

 # Box Cox transformation
 # lambda=0 : level / lambda=1 : log

 # estimation automatique d'un SARIMA
```
out=auto.arima(yy)
```

 # estimation automatique d'un ARIMA
```
out=auto.arima(yy, seasonal=FALSE)
```

 # Prevision
```
plot(forecast(out,h=10))
```

 # Pour récupérer les "point forecast"
```
out=auto.arima(yy)
fc=forecast(yy,model=out,h=1)
prev=as.numeric(fc$mean)
```

Comment caractériser la qualité d'une prévision ?

All models are false, but some are useful (G. Box, 1979).

En statistique, les propriétés qui caractérisent un « bon » estimateur d'une valeur est le fait d'être sans biais et de variance minimale.

Dans le cas de la variable $\widehat{X}_t(h)$, le prédicteur à la date t pour l'horizon h ($h > 0$) de X_{t+h}, on introduit la variable d'erreur de prévision à l'horizon h définie par

$$e_{t+h} = X_{t+h} - \widehat{X}_t(h)$$

La caractéristique principale d'un « bon » prédicteur $\widehat{X}_T(h)$ est de minimiser cette erreur de prévision au sens d'une fonction de perte.

La perte associée avec la prévision h est supposée être une fonction de l'erreur de prévision e_{t+h} et est notée $g(e_{t+h})$

Cette fonction $g(.)$ est une fonction de perte telle que

- $g(.) = 0$ quand aucune erreur n'est faite
- $g(.) \geq 0$
- elle est croissante en taille lorsque les erreurs deviennent plus importantes en magnitude

Généralement, 3 critères d'erreur de prévision à l'horizon h sont retenus en fonction de leur fonction de perte :

- l'erreur moyenne (ME, *Mean Error*)

$$ME = E[g(e_{t+h})] = E[e_{T+h}] = e_{T+h}$$

- l'erreur absolue moyenne (MAE, *Mean Absolute Error*)

$$MAE = E[g(e_{t+h})] = E[|e_{T+h}|] = |e_{T+h}|$$

- l'erreur quadratique moyenne (MSE, *Mean Squared Error*)

$$MSE = E[g(e_{t+h})] = E[e^2_{T+h}] = e^2_{T+h}$$

Benchmark ou référence

Il est d'usage de comparer le modèle de prévision proposé à une prévision ou un modèle de prévision qui servira de benchmark (référence).

L'idée est de savoir si le modèle a des capacités prédictives supérieures à celle d'une prévision obtenue par un modèle très simple

En général, on retient 3 types de prévision "benchmark" ($h > 0$) :

- prévision naïve : $\widehat{X}_{t+h} = X_t$
- prévision naïve saisonnière : $\widehat{X}_{t+h} = X_{t-s}$ avec s la période saisonnière
- prévision modèle AR(1) : $\widehat{X}_{t+h} = \phi X_t$
- prévision Holt-Winters (si tendance)

Fonctions sous R

- prévision naïve : naive(x,h=10) ou rwf(x,h=10)
- prévision naïve saisonnière : snaive(x,h=10)
- prévision méthode de tendance : rwf(x,drift=TRUE,h=10)
- prévision modèle AR(1) : predict(arima(x,order=c(1,0,0)),10)
- prévision modèle SARIMA : forecast(auto.arima(x),h=10)
- prévision modèle BATS : forecast(tbats(x),h=10)
- prévision modèle STL : forecast(stlf(x),lambda=0,h=10)
- prévision modèle ETS (*exponential smoothing state space model*) : forecast(ets(x),h=10)

▶ Comment savoir lorsque $MSE_1 > MSE_2$ si les prévisions du modèle 1 sont ou pas statistiquement différentes de celles du modèle 2 ?

Diebold et Mariano (1995) ont donc proposé un test (DM) pour répondre à cette question basé sur l'hypothèse d'égalité prédictive :

$$MSE_1 = MSE_2$$

Le test DM est basé sur le différentiel de pertes d'erreurs de prévision (ponctuelle) :

$$d_{12,t} = g(e_{1,t}) - g(e_{2,t})$$

- $g(.)$: une fonction de perte (MSE, MAE, ...)
- $(e_{1,t})$ et $(e_{2,t})$: erreurs de prévision

L'hypothèse nulle de capacité prédictive égale (*equal predictive ability*, EPA) correspond à

$$\begin{aligned} H_0 &: E(d_{12,t}) = E\left[g(e_{1,t})\right] - E\left[g(e_{2,t})\right] = 0 \\ H_1 &: E(d_{12,t}) \neq 0 \end{aligned}$$

Le test statistique DM est

$$DM_{12} = \frac{\overline{d}_{12}}{\widehat{\sigma}_{\overline{d}_{12}}} \rightarrow N(0,1)$$

avec

- $\overline{d}_{12} = T^{-1} \sum_{t=1}^{T} d_{12,t}$: moyenne empirique des différentiels de pertes
- $\widehat{\sigma}^2_{\overline{d}_{12}}$: estimation de la variance de long-terme de $\overline{d}_{12}$
- sous l'hypothèse que le différentiel de pertes soit stationnaire en covariance
- le test n'est applicable que pour des **modèles non emboîtés** (*nested models*, voir Clark et MacCracken, 2001)

Puisque les erreurs de prévision, et donc les différentiels de pertes, peuvent être sériellement corrélées, DM ont proposé une version robuste de leur test :

$$DM_{12} = \frac{\overline{d}_{12}}{\widehat{\sigma}_{\overline{d}}} \rightarrow N(0,1)$$

avec

- $\widehat{\sigma}_{\overline{d}} = \sqrt{\widehat{g}_d(0)T^{-1}}$
- $\widehat{g}_d(0)$: estimateur du spectre à la fréquence zéro du différentiel de pertes

$$\widehat{g}_d(0) = \sum_{j=-(h-1)}^{h-1} \widehat{\gamma}_d(j) = \widehat{\gamma}_d(0) + 2\sum_{j=1}^{h-1} \widehat{\gamma}_d(j)$$

- $\widehat{\gamma}_d(j)$: autocovariance empirique d'ordre j du processus $d_{12,t}$

$$\widehat{\gamma}_d(j) = (1/T) \sum_{i=j+1}^{m} (d_{12,i} - \overline{d}_{12})(d_{12,i-j} - \overline{d}_{12})$$

- T : nombre de prévisions

Règle de décision

$$\begin{aligned} H_0 &: E(d_{12,t}) = 0 \\ H_1 &: E(d_{12,t}) \neq 0 \end{aligned}$$

L'hypothèse nulle EPA sera rejetée au seuil α si

$$|DM_{12}| > z_{\alpha/2}$$

avec $z_{\alpha/2}$ la valeur d'une loi Normale centrée réduite.

Diebold et Mariano proposent également de tester l'hypothèse nulle EPA contre les alternatives $H_1 : E(d_{12,t}) < 0$ ou $H_1 : E(d_{12,t}) > 0$

Harvey, Leybourne et newbold (1997) (HLN) montrent que le test DM est biasé en petits échantillons et proposent d'améliorer ses propriétés en petits échantillons en

- rajoutant une correction du biais à la statistique de test DM
- comparant la statistique corrigée avec une loi de Student à $(T-1)$ ddl plutôt qu'une loi Normale

La statistique corrigée est définie par

$$HLN-DM=\sqrt{\frac{T+1-2h+h(h-1)}{T}}DM$$

Le package R forecast implémente le test de Diebold-Mariano avec la fonction `dm.test`

```
# Test on in-sample (IS) one-step forecasts
out1=Arima(yy,order=c(0,1,1),seasonal=list(order=c(0,1,1),period=12),lambda=0)
accuracy(out1)
out2=auto.arima(yy)
accuracy(out2)
dm.test(residuals(out),residuals(out2),h=10)
```

```
# DM test
# par défaut : "two.sided"
dm.test(e1, e2, alternative = c("two.sided", "less", "greater"), h = 1, power = 2)
```

```
# Test on out-of-sample (OOS) one-step forecasts for horizon = 100
# Multi-step forecasts are computed recursively
out1=Arima(window(yy,
end=100),order=c(0,1,1),seasonal=list(order=c(0,1,1),period=12),lambda=0)
out2=auto.arima(window(yy, end=100))
fc1=forecast(yy,model=out1,h=1)
fc2=forecast(yy,model=out2,h=1)
accuracy(fc1)
accuracy(fc2)
dm.test(residuals(out),residuals(out2),h=1)
```

```
# ou bien
out1=Arima(yy[1:100],order=c(0,1,1),seasonal=list(order=c(0,1,1),period=12),lambda=0)
out2=auto.arima(yy[1:100])
fc1=forecast(yy[101:200],model=out1,h=1)
fc2=forecast(yy[101:200],model=out2,h=1)
accuracy(fc1)
accuracy(fc2)
dm.test(residuals(out),residuals(out2),h=1)
```

Combinaison de prévisions

La combinaison de prévisions est une méthode de prévision proposée par Bates et Granger (1969). L'idée consiste à penser qu'il est possible de réduire l'incertitude de prévision en combinant les prévisions de différents modèles de prévision

Des études montrent que la combinaison de prévisions améliore l'exactitude des prévisions par rapport aux prévisions des modèles individuels

$$f_{C,t} = \sum_{i=1}^{n} \omega_i f_{i,t}$$

- $f_{i,t}$: prévisions individuelles des n modèles
- ω_i : pondérations telles que $\sum_{i=1}^{n} \omega_i = 1$

Les techniques de combinaison les plus simples (utilisant la moyenne des prévisions des modèles individuels) produisent des prévisions de meilleure qualité que les techniques de combinaison plus complexes.

Fonctions sous R

- Package *opera* (*Online Prediction by ExpeRt Aggregation*)

  La fonction mixture calcule les poids lors de la combinaison des prévisions en fonction de la qualité des résultats obtenus.

- Package *forecastHybrid* associé au package *forecast*

  La combinaison se fait avec soient des poids égaux soient des poids fondés sur une erreur dans l'échantillon (*in-sample error*).

  Par défaut, elle combine les prévisions issues des fonctions auto.arima, ets, nnetar (*Neural Network Time Series Forecasts*), stlm et tbats

Application

Figure: Séries en niveau

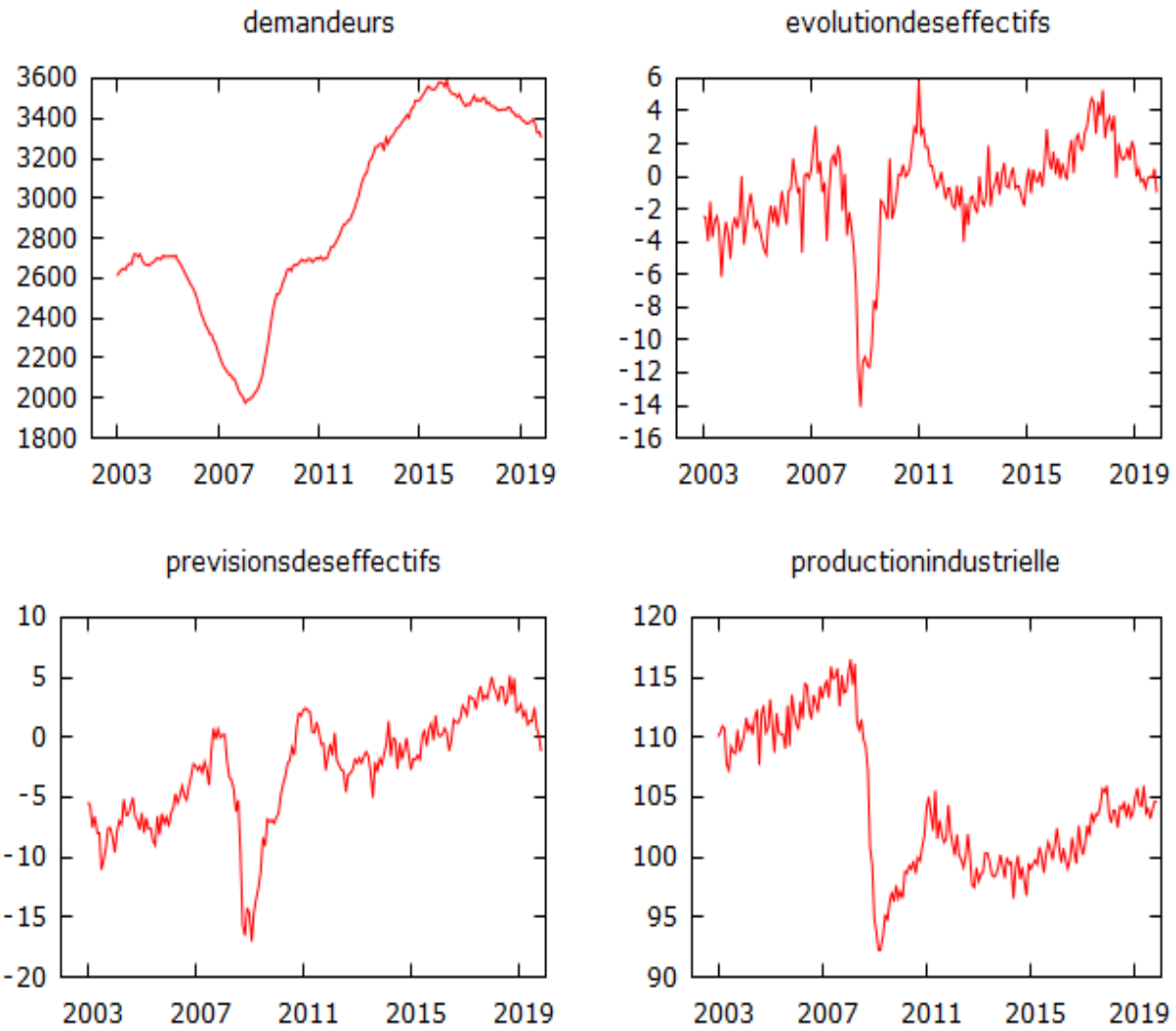


Figure: Séries en différence première

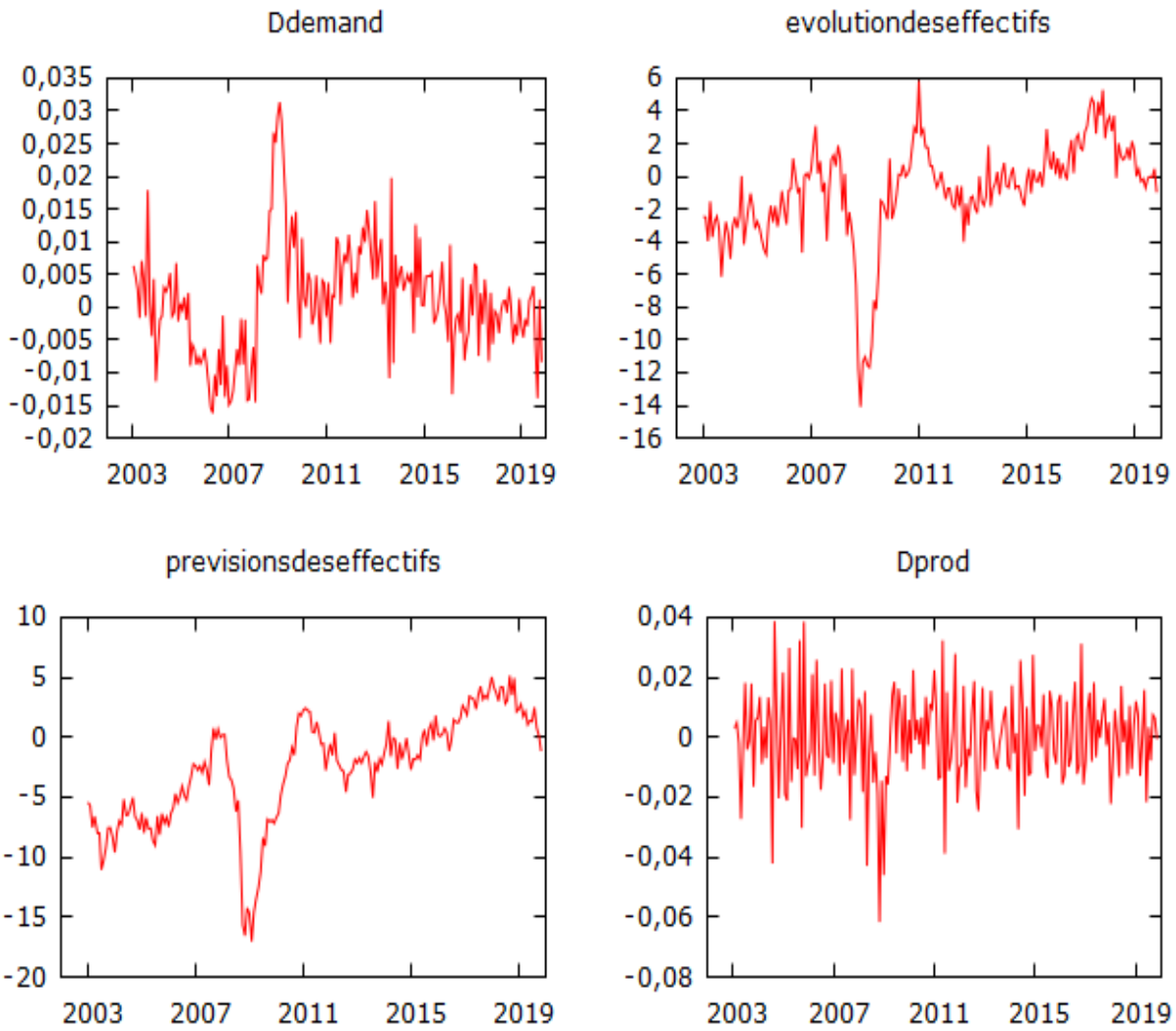


Figure: Matrice des corrélations

Coefficients de corrélation, utilisant les observations 2003:02 - 2019:11
5% valeur critique (bilatéral) = 0,1381 pour n = 202

Ddemand	evolutiondesef~	previsionsdese~	Dprod	
1,0000	-0,5142	-0,3419	-0,2089	Ddemand
	1,0000	0,8574	0,2706	evolutiondesef~
		1,0000	0,1274	previsionsdese~
			1,0000	Dprod

Le package **gets** permet d'estimer un modèle ARX et faire des prévisions avec des variables explicatives avec les fonctions `arx` et `predict`

```
library(readxl)
library(tidyverse)

# Gets modelling
library(lgarch)
library(gets)

# database
dlbase <- read_excel("c://R//data//database.xlsx", sheet = "dlbase")
dlbase <- data.frame(dlbase)
summary(dlbase)
str(dlbase)
training_dlbase <- dlbase
data.frame(training_dlbase)
```

```
# Gets modelling
# convert tibble in matrix for the function arx
class(dlbase[,2:4])        #tibble
mX = data.matrix(training_dlbase[,2:4])
# ARX model with AR(1)
Model01 <- arx(training_dlbase$Ddemand, mc = T, ar = 1, mxreg = mX[, 1:3], vcov.type =
"ordinary")
Model01
```

```
Date: Mon Feb 10 15:36:19 2020
Dependent var.: training_dlbase$Ddemand
Method: Ordinary Least Squares (OLS)
Variance-Covariance: Ordinary
No. of observations (mean eq.): 201
Sample: 2 to 202

Mean equation:

                              coef   std.error  t-stat   p-value
mconst                 0.00058459  0.00054925  1.0643   0.28848
ar1                    0.47038385  0.06070442  7.7488 4.894e-13
evolution.des.effectifs -0.00130122  0.00031853 -4.0850 6.420e-05
previsions.des.effectifs  0.00044522  0.00019729  2.2567   0.02513
Dprod                  -0.02965198  0.03032048 -0.9780   0.32930

Diagnostics and fit:

                  Chi-sq df    p-value
Ljung-Box AR(2)   21.455  2 2.1932e-05
Ljung-Box ARCH(1) 10.461  1 1.2193e-03

SE of regression   0.00632
R-squared          0.46577
Log-lik.(n=201)  735.02048
```

```
Date: Mon Feb 10 15:37:30 2020
Dependent var.: training_dlbase$Ddemand
Method: Ordinary Least Squares (OLS)
Variance-Covariance: Ordinary
No. of observations (mean eq.): 201
Sample: 2 to 202

Mean equation:

                              coef   std.error  t-stat   p-value
mconst                  0.00061978  0.00054801  1.1310   0.25944
ar1                     0.46776252  0.06063850  7.7140 5.935e-13
evolution.des.effectifs -0.00139739  0.00030294 -4.6127 7.145e-06
previsions.des.effectifs 0.00048663  0.00019267  2.5257   0.01233

Diagnostics and fit:

                   Chi-sq df    p-value
Ljung-Box AR(2)   20.5207  2 3.4994e-05
Ljung-Box ARCH(1)  9.1932  1 2.429le-03

SE of regression   0.00632
R-squared          0.46316
Log-lik.(n=201)  734.54272
```

```
# # Forecasting
# Loss functions MSE, MAE, MdSE and MdAE
# MSE
mse <- function(error) {
mean(error^2)
}

# MAE
mae <- function(error) {
mean(abs(error))
}

# MdSE
mdse <- function(error) {
median(error^2)
}

# MdAE
mdae <- function(error) {
median(abs(error))
}
```

```
# Initialization
training_dlbase <- dlbase[1:180,]              # estimation sample
testing_dlbase <- dlbase[181:202, 1]          # observed sample for forecast testing
mX = data.matrix(training_dlbase[,2:4])
predicted_dlbase2 <- 0

# One-step ahead forecasts for h=22
# forecast horizon h=22
for (i in 1:22) {
mode01 <- arx(training_dlbase$Ddemand, mc = T, ar = 1, mxreg = mX[, 1:2], vcov.type =
"ordinary")
# one-step ahead forecast
predicted_dlbase <- predict(mod01, n.ahead = 1, newmxreg = training_dlbase[1,1])
training_dlbase <- dlbase[1:180 + i, ]
mX = data.matrix(training_dlbase[,2:4])
predicted_dlbase2 <- c(predicted_dlbase2, predicted_dlbase)
}

# Compute forecast errors
predicted_dlbase <- matrix(predicted_dlbase2[-1])
error <- testing_dlbase - predicted_dlbase
mse(error)
mae(error)
mdse(error)
mdae(error)
```