Estimation de modèles autorégressifs à changement de régime markovien

Rynkiewicz J
SAMOS/MATISSE Université de Paris1

3 avril 2000

Résumé

Un modèle autorégressif à changement de régime markovien suppose que, conditionnellement à une suite de variables cachées, les observations forment un modèle autorégressif non-linéaire. Dans ce rapport, nous donnons un calcul analytique du gradient de la log-vraisemblance dans le cas gaussien. Nous étudions aussi un estimateur récursif qui donne de bons résultats sur des données simulées.

An autoregressive model with Markov-switching assume a sequence of random vector to be non linear autoregressive model given a sequence of non observed state variables wich forms a Markov chain. In this paper we give an analytical calculs of the gradient of the log-likelihood in the gaussian case. We also investigate a recursive estimator which works well on simulated data.

AMS classification : 62M10 ; 62L20

Keywords : Hidden Markov chains. Switching models, Maximum likelihood, Recursive estimation.

1 Introduction

Il s'agit de modéliser des séries non-stationnaires par morceaux, Hamilton a étudié de tels modèles afin de modéliser des séries temporelles sujettes à des changements discrets de régimes pour analyser la série GNP (gross national product) aux états-unis. On peut ainsi utiliser ce modèle pour des séries ayant un régime pour les périodes de croissance économique, un autre pour les périodes de récession. L'objet de cet article est d'estimer un processus autorégressif à changement de régimes markovien. Les fonctions autorégressives sont simplement supposées continûment dérivables mais pas forcément linéaires. Une façon classique de procéder est d'utiliser l'algorithme E.M. (ou G.E.M.) grâce à l'algorithme forward-backward de Baum et Welch (voir par exemple J. Rynkiewicz [9]) . Cette méthode est très coûteuse en temps de calcul car elle a l'inconvénient de demander des itérations pour optimiser les fonctions de régression lorsqu'elles ne sont pas linéaires (par exemple des MLP) à chaque pas de l'algorithme E.M. On propose donc un algorithme direct (sans utiliser l'algorithme E.M.) qui permet dans certains cas de diminuer le temps de calcul nécessaire, et d'en déduire aussi une méthode récursive d'estimation.

2 Le modèle

2.1 Les équations régissant le modèle

Soit (X_t), $1 \leq t \leq n$ une chaîne de Markov à valeurs dans un espace d'état fini $E = \{e_1, ..., e_N\}$. Soit $(Y_t) \in \mathbb{R}^d$, $1 \leq t \leq n$ la série des observations. on note Y_{t-p+1}^t le vecteur $(Y_{t-p+1}, \cdots, Y_t)$. On considère le modèle suivant à un instant t fixé : $Y_{t+1} = F_{X_{t+1}}(Y_{t-p+1}^t) + M_{X_{t+1}}\varepsilon_{t+1}$ avec

1. $F_{e_i} \in \{F_{e_1}, ..., F_{e_N}\}$une fonction paramétrique continûment différentiable de $\mathbb{R}^{d\times p} \longrightarrow \mathbb{R}^d$.
2. $M_{e_i} \in \{M_{e_1}, ..., M_{e_N}\}$ une matrice telle que $\Sigma_{e_i} = M_{e_i}M_{e_i}^T \in \mathbb{R}^{d\times d}$ soit définie positive.
3. (ε_t), $1 \leq t \leq T$ une suite i.i.d. gaussienne $\mathcal{N}(0, I_d)$ où I_d est la matrice identité de $\mathbb{R}^{d\times d}$
4. Sans perte de généralité, on identifie l'espace d'état $E = \{e_1, \cdots, e_N\}$ avec le simplexe de $\mathbb{R}^N$ où e_i est un vecteur unité de $\mathbb{R}^N$ avec 1 sur la i-ème composante zéro partout ailleurs.

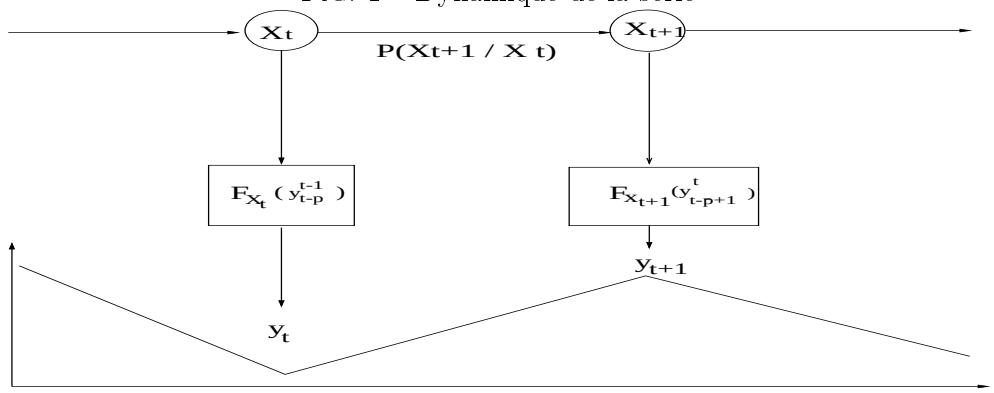
La chaîne X_t est caractérisée par sa matrice de transition $A = (a_{ij})$ qui est telle que [1] :
$P(X_{t+1} = e_i / X_t = e_j) = a_{ij}$
Ainsi si on définit : $V_{k+1} := X_{t+1} - AX_t$, on obtient les équations générales du modèle :

$$\begin{cases} X_{t+1} = AX_t + V_{t+1} \\ Y_{t+1} = F_{X_{t+1}}(Y_{t-p+1}^t) + M_{X_{t+1}}\varepsilon_{t+1} \end{cases} \tag{1}$$

On remarquera que l'ordre de régression des fonctions F_{e_i} est toujours $p \in \mathbb{N}^*$. En fait c'est l'ordre maximal des N modèles de régression (un modèle avec un ordre de régression plus petit peut toujours être vu comme un modèle de régression d'ordre p avec certains paramètres toujours nuls).

On peut schématiser le modèle de la façon suivante :

FIG. 1 – Dynamique de la série



2.2 Paramétrisation du modèle

A priori les paramètres du modèle sont donc
– Les coefficients (a_{ij}) de la matrice de transition A

[1] La notation traditionnelle d'un matrice de transition est plutôt $a_{ij} = P(X_{t+1} = e_j / X_t = ei)$ cependant la notation transposée utilisée ici et empruntée à Elliott, permet une notation du modèle complet plus confortable.

– Les matrices de covariance Σ_{e_i}
– Les paramètres ω_{e_i} des fonctions de régression F_{e_i}

Cependant cette paramétrisation ne facilite pas les calculs, et on utilisera plutôt la paramétrisation décrite ci-après.

2.2.1 Paramétrisation de la matrice A

La matrice A est stochastique, c'est-à-dire que la somme d'une colonne quelconque de A est 1, on a donc $N-1$ paramètres libres par colonne. Pour traduire cette contrainte on pose $v_{ij} = \ln \frac{a_{ij}}{a_{Nj}}$, on remarquera alors que $v_{Nj} = 0$, et $(v_{1j}, \cdots, v_{N-1,j}) \in \mathbb{R}^{N-1}$, l'avantage de cette paramétrisation est de pouvoir optimiser la matrice A sans contrainte.

Expression de A en fonction de (v_{ij}), $1 \leq i \leq N-1$, $1 \leq j \leq N$ Notons A_j la colonne j de A, alors on a :

$$A_j = \left(\frac{e^{v_{ij}}}{1 + e^{v_{1j}} + \cdots + e^{v_{N-1j}}} \right)_{1 \leq i \leq N}$$

On en déduit une forme agréable pour dériver A par rapport aux paramètres (v_{ij}) :

$$\frac{\partial a_{ij}}{\partial v_{ij}} = \frac{\partial}{\partial v_{ij}} \frac{e^{v_{ij}}}{1 + e^{v_{1j}} + \cdots + e^{v_{N-1j}}} = \frac{e^{v_{ij}}}{1 + e^{v_{1j}} + ... + e^{v_{N-1j}}} \left(1 - \frac{e^{v_{ij}}}{1 + e^{v_{1j}} + \cdots + e^{v_{N-1j}}} \right)$$

d'où

$$\frac{\partial a_{ij}}{\partial v_{ij}} = a_{ij}(1 - a_{ij}) \tag{2}$$

Et pour $l \neq i$

$$\frac{\partial a_{ij}}{\partial v_{lj}} = \frac{e^{v_{ij}}}{1 + \cdots + e^{v_{N-1j}}} \times \left(-\frac{e^{v_{lj}}}{1 + \cdots + e^{v_{N-1j}}} \right) == -a_{ij} a_{lj} \tag{3}$$

On a, par ailleurs, si $k \neq j$:

$$\frac{\partial a_{ij}}{\partial v_{lk}} = 0$$

2.2.2 Paramétrisation des matrices de covariance du bruit

Puisque Σ_{e_i} est supposée définie positive, on choisit alors de prendre pour paramètres les termes de son inverse $\Sigma_{e_i}^{-1}$. De plus cette matrice étant symétrique, on ne prendra que les éléments sous la diagonale (diagonale incluse.). On verra par la suite que cela simplifie l'expression de la dérivée de la vraisemblance par rapport aux paramètres.

2.2.3 Paramétrisation des fonctions de régression

On supposera seulement que la fonction F_{e_i} est continûment dérivable par rapport à chaque composante de son vecteur paramètre ω_{e_i} .

2.2.4 Notation du vecteur paramètre du modèle

Le vecteur paramètre θ est donc (en le notant en ligne)

$$\theta = (\omega_{e_i}^T, \cdots, \omega_{e_N}^T, \upsilon_{11}, \cdots, \upsilon_{(N-1)N}, (\Sigma_{e_1}^{-1})_{11}, \cdots, (\Sigma_{e_1}^{-1})_{dd}, \cdots, (\Sigma_{e_N}^{-1})_{dd})$$

où $(\Sigma_{e_i}^{-1})_{lk}$ est le coefficient de la ligne l et de la colonne k ($k \leq l$) de la matrice $\Sigma_{e_i}^{-1}$et $\omega_{e_i}^T$ représente les paramètres de la fonction F_{e_i} écrit en ligne.

3 Calcul de la vraisemblance

Afin de simplifier les notations on notera $L(y_1, \cdots, y_n)$ la vraisemblance des observations $(y_1, \cdots, y_n)$ bien que celle-ci dépende du paramètre θ. De même la dépendance aux paramètres sera implicite dans l'expression des probabilités et des densités conditionnelles. Il serait possible de calculer la vraisemblance grâce à l'algorithme forward de Baum et Welch, néanmoins cette méthode requiert une stratégie de normalisation pour ne pas dépasser les capacités numériques de l'ordinateur, c'est pourquoi on utilise ici une forme plus agréable qui permet de calculer récursivement le *logarithme* de la vraisemblance par une méthode que l'on retrouve dans la thèse de Mevel [5]dans le cas de chaîne de Markov cachée à observations indépendantes.

3.1 Calcul préliminaire

On suppose les observations (y_{-p+1}, y_0) connues, la fonction de vraisemblance s'écrit :

$$L(y_1, ..., y_n) = \prod_{k=1}^{n} L(y_k \,|y_{-p+1}, \cdots, y_{k-1}) = L(y_n \,|y_{-p+1}, \cdots, y_{n-1}) \times \prod_{k=1}^{n-1} L(y_k \,|y_{-p+1}, \cdots, y_{k-1})$$

$$= \sum_{i=1}^{N} L(y_n \,|X_n = e_i, y_{-p+1}, \cdots, y_{n-1}) P(X_n = e_i \,|y_{-p+1}, \cdots, y_{n-1}) \times \prod_{k=1}^{n-1} L(y_k \,|y_{-p+1}, \cdots, y_{k-1})$$

Si on note

- p_n le vecteur dont la i-ème composante est : $p_n(i) = P(X_n = e_i \,|y_{-p+1}, \cdots, y_{n-1})$
- b_n le vecteur dont la i-éme composante est :$b_n(i) = L(y_n \,|X_n = e_i, y_{-p+1}, \cdots, y_{n-1})$, c'est-à-dire la densité conditionnelle de y_n sachant $X_n = e_i$ et $(y_{-p+1}, \cdots, y_{n-1})$.
- $B_n = diag\,(b_n)$ la matrice diagonale ayant pour diagonale le vecteur b_n

on aura :

$$L(y_1, \cdots, y_n) = b_n^T p_n \times \prod_{k=1}^{n-1} L(y_k \,|y_{-p+1}, \cdots, y_{k-1}) = \prod_{k=1}^{n} b_k^T p_k$$

On en déduit une forme pratique de la log-vraisemblance :

$$\ln(L(y_1, \cdots, y_n)) = \sum_{k=1}^{n} \ln(b_k^T p_k)$$

Il suffit donc de calculer p_k pour $k = 1, \cdots, n$, pour pouvoir calculer la log-vraisemblance, car :

$$b_k(i) = L(y_k \,|X_k = e_i, y_{k-1}, \cdots, y_{-p+1}) := \Phi_{e_i}(y_k - F_{e_i}(y_{k-p+1}^{k-1}))$$

où

$$\Phi_{e_i}(y_k - F_{e_i}(y_{k-1})) = \frac{1}{\left(\sqrt{2\pi}\right)^d \sqrt{\det\left(\Sigma_{e_i}\right)}} \exp\left(-\frac{1}{2}\left((y_k - F_{e_i}(y_{k-p+1}^{k-1}))^T \Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p+1}^{k-1}))\right)\right)$$

est la densité conditionnelle de y_k sachant $X_k = e_i$.

3.2 Calcul de p_k

Soit $p_k(i)$ la i-ème coordonnée ($1 \leq i \leq N$) du vecteur p_k . On a $p_k(i) = P(X_k = e_i \,|y_{-p+1}, \cdots, y_{k-1})$. Calculons $p_{k+1}(i)$:

$$p_{k+1}(i) = P(X_{k+1} = e_i \,|y_{-p+1}, \cdots, y_k) = \sum_{j=1}^{N} P(X_{k+1} = e_i, X_k = e_j \,|y_{-p+1}, \cdots, y_k)$$

$$= \sum_{j=1}^{N} P(X_{k+1} = e_i \,|X_k = e_j, y_{-p+1}, \cdots, y_k) \times P(X_k = e_j \,|y_{-p+1}, \cdots, y_k)$$

Mais comme (X_k) est une chaîne de Markov homogène :

$$P(X_{k+1} = e_i \,|X_k = e_j, y_{-p+1}, \cdots, y_k) = P(X_{k+1} = e_i \,|X_k = e_j) = a_{ij}$$

d'où

$$p_{k+1}(i) = \sum_{j=1}^{N} a_{ij} P(X_k = e_j \,|y_{-p+1}, \cdots, y_k)$$

de plus par la définition des densités conditionnelles :

$$P(X_k = e_j \,|y_{-p+1}, \cdots, y_k) = \frac{L(e_j, y_k \,|y_{-p+1}, \cdots, y_{k-1})}{L(y_k \,|y_{-p+1}, \cdots, y_{k-1})}$$

soit

$$P(X_k = e_j \,|y_{-p+1}, \cdots, y_k) = \frac{L(y_k \,|X_k = e_j, y_{-p+1}, \cdots, y_{k-1}) \times P(X_k = e_j \,|y_{-p+1}, \cdots, y_{k-1})}{L(y_k \,|y_{-p+1}, \cdots, y_{k-1})}$$

$$P(X_k = e_j \,|y_{-p+1}, \cdots, y_k) = \frac{b_k(j) \times p_k(j)}{b_k^T p_k}$$

donc

$$p_{k+1}(i) = \frac{\sum a_{ij} b_k(j) \times p_k(j)}{b_k^T p_k}$$

finalement on en déduit :

$$p_{k+1} = \frac{A B_k p_k}{b_k^T p_k} \tag{4}$$

On supposera que p_1 suit la distribution uniforme sur $\{1, \cdots, N\}$ et on pourra ainsi calculer p_k, $k = 1, \cdots n$ par récurrence.

4 Dérivée de la log-vraisemblance

On rappelle que l'on a :

$$\ln(L(y_1, \cdots, y_n)) = \sum_{k=1}^{n} \ln(b_k^T p_k)$$

donc si on note θ_j , le j-ième paramètre du modèle on a :

$$\frac{\partial \ln(L(y_1, \cdots, y_n))}{\partial \theta_j} = \sum_{k=1}^{n} \frac{\frac{\partial b_k^T p_k}{\partial \theta_j}}{b_k^T p_k}$$

4.1 Calcul de $\frac{\partial b_k^T p_k}{\partial \theta_j}$

on a :

$$\frac{\partial b_k^T p_k}{\partial \theta_j} = \frac{\partial b_k}{\partial \theta_j}^T p_k + b_k^T \frac{\partial p_k}{\partial \theta_j} \tag{5}$$

4.1.1 Calcul de $\frac{\partial b_k}{\partial \theta_j}$ suivant θ_j

Comme $b_k(i)$ ne s'annule jamais on peut utiliser la formule :

$$\frac{\partial b_k(i)}{\partial \theta_j} = b_k(i) \times \frac{\partial \ln(b_k(i))}{\partial \theta_j}$$

la dérivée du logarithme de $b_k(i)$ étant plus simple à exprimer.

Si θ_j est un coefficient de la matrice A

$$\frac{\partial b_k}{\partial \theta_j} = 0$$

Si θ_j est un coefficient de la matrice de covariance inverse $\Sigma_{e_i}^{-1}$: $\theta_j = (\Sigma_{e_i}^{-1})_{kl}$

$$\frac{\partial b_k}{\partial \theta_j} = u_i$$

où u_i est le vecteur de $\mathbb{R}^N$ dont tous les éléments sont nuls sauf la i-ème coordonnée qui vaut :

$$b_k(i) \times \frac{\partial[-\frac{1}{2}(d\ln(2\pi) - \ln(\det(\Sigma_{e_i}^{-1})) + Tr((y_k - F_{e_i}(y_{k-p}^{k-1}))^T \Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p}^{k-1}))]}{\partial \theta_j}$$

On utilise alors les formules classiques :

– Si A (de coefficients a_{ij}) est une matrice constante et X une matrice de coefficients x_{ij} :

$$\frac{\partial}{\partial x_{ij}} Tr(AX) = a_{ji} \tag{6}$$

– En supposant maintenant X inversible et en notant x_{ij}^{-1} les coefficients de X^{-1} on a :

$$\frac{\partial}{\partial x_{ij}} \ln(\det(X)) = x_{ji}^{-1} \tag{7}$$

– Si A, B, C sont trois matrices de tailles convenables, la trace de leur produit est invariante par permutation circulaire :

$$Tr(ABC) = Tr(BCA) = Tr(CAB) \tag{8}$$

on a, grâce aux formules (8) et (6) :

$$\frac{\partial (Tr((y_k - F_{e_i}(y_{k-p}^{k-1}))^T \Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p}^{k-1}))))}{\partial \theta_j} = \left((y_k - F_{e_i}(y_{k-p}^{k-1}))(y_k - F_{e_i}(y_{k-p}^{k-1}))^T\right)_{kl}$$

et grâce à la formule (7)

$$\frac{\partial \ln(\det(\Sigma_{e_i}^{-1}))}{\partial \theta_j} = (\Sigma_{e_i})_{kl}$$

En résumé, le i-ème élément de u_i vaut (en utilisant la symétrie de Σ_{e_i}) :

$$\begin{array}{l} b_k(i) \times \left((\Sigma_{e_i})_{kl} - \left((y_k - F_{e_i}(y_{k-p}^{k-1}))(y_k - F_{e_i}(y_{k-p}^{k-1}))^T\right)_{kl}\right), \; si \; k \neq l \\ b_k(i) \times \frac{1}{2}\left((\Sigma_{e_i})_{kl} - \left((y_k - F_{e_i}(y_{k-p}^{k-1}))(y_k - F_{e_i}(y_{k-p}^{k-1}))^T\right)_{kl}\right), \; si \; k = l \end{array} \tag{9}$$

Si θ_j est un des paramètres de la fonction de régression F_{e_i} Supposons que θ_j correspond à la composante l du vecteur paramètre ω_{e_i}, on a :

$$\frac{\partial b_k}{\partial \theta_j} = (0, \cdots, \frac{\partial b_k(i)}{\partial \theta_j}, \cdots, 0)^T$$

où $\frac{\partial b_k(i)}{\partial \theta_j}$ vaut :

$$b_k(i) \times \frac{\partial - \frac{1}{2}[(d\ln(2\pi) - \ln(\det(\Sigma_{e_i}^{-1})) + Tr((y_k - F_{e_i}(y_{k-p}^{k-1}))^T \Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p}^{k-1}))]}{\partial \theta_j}$$

$$= -\frac{1}{2}\frac{\partial [Tr((y_k - F_{e_i}(y_{k-p}^{k-1}))^T \Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p}^{k-1}))]}{\partial \theta_j} \times b_k(i)$$

et en utilisant la formule (8)

$$\frac{\partial b_k(i)}{\partial \theta_j} = -\frac{1}{2}\frac{\partial [Tr(\Sigma_{e_i}^{-1}(y_k - F_{e_i}(y_{k-p}^{k-1})(y_k - F_{e_i}(y_{k-p}^{k-1}))^T]}{\partial \theta_j} \times b_k(i)$$

On note $\left(y_k - F_{e_i}(y_{k-p}^{k-1})\right)(l)$(resp. $\left(F_{e_i}(y_{k-p}^{k-1})\right)(l)$) la l-ième composante du vecteur $\left(y_k - F_{e_i}(y_{k-p}^{k-1})\right)$ (resp. $\left(F_{e_i}(y_{k-p}^{k-1})\right)$), en utilisant la formule de dérivées des composées de fonctions (vraie aussi dans le cas vectoriel (Cartan [2]) et la formule (6), on aura :

$$\frac{\partial b_k(i)}{\partial \theta_j} =$$

$$b_k(i) \times \sum_{1 \leq m \leq l \leq d} (\Sigma_{e_i}^{-1})_{lm} \left(\left(F_{e_i}(y_{k-p}^{k-1}) - y_k\right)(l) \frac{\partial F_{e_i}(y_{k-p}^{k-1})(m)}{\partial \theta_j} + \left(F_{e_i}(y_{k-p}^{k-1}) - y_k\right)(m) \frac{\partial F_{e_i}(y_{k-p}^{k-1})(l)}{\partial \theta_j} \right)$$

4.1.2 Calcul de $\frac{\partial p_k}{\partial \theta_j}$ suivant θ_j

On sait que p_k vérifie la récurrence :

$$p_{k+1} = \frac{AB_k p_k}{b_k^T p_k}$$

En dérivant cette expression par rapport au paramètre θ_j on a :

$$\frac{\partial p_{k+1}}{\partial \theta_j} = \frac{\partial AB_k p_k}{\partial \theta_j} \times \frac{1}{b_k^T p_k} + AB_k p_k \times \frac{\partial b_k^T p_k}{\partial \theta_j} \times \left(-\frac{1}{(b_k^T p_k)^2}\right)$$

soit

$$\frac{\partial p_{k+1}}{\partial \theta_j} = \left(\frac{\partial AB_k}{\partial \theta_j} p_k + AB_k \frac{\partial p_k}{\partial \theta_j}\right) \times \frac{1}{b_k^T p_k} + AB_k p_k \times \left(\frac{\partial b_k^T}{\partial \theta_j} p_k + b_k^T \frac{\partial p_k}{\partial \theta_j}\right) \times \left(-\frac{1}{(b_k^T p_k)^2}\right)$$

on a alors :

$$\frac{\partial p_{k+1}}{\partial \theta_j} = \frac{AB_k}{b_k^T p_k}\left[I - \frac{p_k b_k^T}{b_k^T p_k}\right] \frac{\partial p_k}{\partial \theta_j} + \left(\frac{\partial AB_k}{\partial \theta_j}\right) \frac{p_k}{b_k^T p_k} - \frac{AB_k p_k}{(b_k^T p_k)^2} \left(\frac{\partial b_k^T}{\partial \theta_j} p_k\right)$$

d'où :

$$\frac{\partial p_{k+1}}{\partial \theta_j} = \frac{AB_k}{b_k^T p_k}\left[I - \frac{p_k b_k^T}{b_k^T p_k}\right] \frac{\partial p_k}{\partial \theta_j} + \left(\frac{\partial A}{\partial \theta_j} B_k + A \frac{\partial B_k}{\partial \theta_j}\right) \frac{p_k}{b_k^T p_k} - \frac{AB_k p_k}{(b_k^T p_k)^2} \left(\frac{\partial b_k^T}{\partial \theta_j} p_k\right) \tag{10}$$

avec, si p_1 est la distribution initiale : $\frac{\partial p_1}{\partial \theta_j} = 0$ pour tout j .

Calcul de $\frac{\partial A}{\partial \theta_j}$

Si θ_j est un coefficient de la matrice A :$\theta_j = v_{lm}$ On a :

$$\frac{\partial A}{\partial \theta_j} = C(v_{lm})$$

avec $C(v_{lm})$ une matrice dont tous les coefficients sont nuls, sauf la colonne m : C_m qui est telle que :

$$\begin{cases} C_m(i) = -a_{im} a_{lm} \text{ si } i \neq l \\ C_m(i) = a_{lm}(1 - a_{lm}) \text{ si } i = l \end{cases} \tag{11}$$

Calcul de $\frac{\partial B_k}{\partial \theta_j}$

- Si θ_j est un coefficient de la matrice de la matrice $\Sigma_{e_i}^{-1}$ ou de la fonction de régression F_{e_i}, on déduit facilement $\frac{\partial B_k}{\partial \theta_j}$de $\frac{\partial b_k(i)}{\partial \theta_j}$, puisque c'est la matrice :

$$diag(0, ..., \frac{\partial b_k(i)}{\partial \theta_j}, ..., 0)$$

- Sinon $\frac{\partial B_k}{\partial \theta_j}$ est la matrice nulle.

5 Résumé

Avec les notations précédentes, pour estimer les paramètres de l'estimateur du maximum de vraisemblance du modèle, on doit maximiser :

$$\ln(L(y_1, \cdots, y_n)) = \sum_{k=1}^{n} \ln(b_k^T p_k)$$

Ce qui s'obtient par une méthode d'optimisation différentielle classique, car le gradient se calcule ainsi :

$$\frac{\partial \ln(L(y_1, \cdots, y_n))}{\partial \theta_j} = \sum_{k=1}^{n} \frac{\frac{\partial b_k^T p_k}{\partial \theta_j}}{b_k^T p_k}$$

et $\frac{\partial b_k^T p_k}{\partial \theta_j}$, se calcule récursivement grâce aux formules :

$$\frac{\partial b_k^T p_k}{\partial \theta_j} = \frac{\partial b_k}{\partial \theta_j}^T p_k + b_k^T \frac{\partial p_k}{\partial \theta_j}$$

avec

$$\begin{cases} \frac{\partial b_k}{\partial \theta_j} = 0 \text{ si } \theta_j \in A \\ \frac{\partial b_k}{\partial \theta_j} = b_k(i) \times \frac{1}{2} \left((\Sigma_{e_i})_{ll} - \left((y_k - F_{e_i}(y_{k-p}^{k-1}))(y_k - F_{e_i}(y_{k-p}^{k-1}))^T \right)_{ll} \right) \text{ si } \theta_j = \left(\Sigma_{e_i}^{-1}\right)_{ll} \\ \frac{\partial b_k}{\partial \theta_j} = b_k(i) \times \left((\Sigma_{e_i})_{lm} - \left((y_k - F_{e_i}(y_{k-p}^{k-1}))(y_k - F_{e_i}(y_{k-p}^{k-1}))^T \right)_{lm} \right) \text{ si } \theta_j = \left(\Sigma_{e_i}^{-1}\right)_{l \neq m} \\ \frac{\partial b_k}{\partial \theta_j} = b_k(i) \times \sum_{m \leq l} (\Sigma_{e_i}^{-1})_{lm} \left(\left(F_{e_i}(y_{k-p}^{k-1}) - y_k \right)(l) \frac{\partial F_{e_i}(y_{k-p}^{k-1})(m)}{\partial \theta_j} + \left(F_{e_i}(y_{k-p}^{k-1}) - y_k \right)(m) \frac{\partial F_{e_i}(y_{k-p}^{k-1})(l)}{\partial \theta_j} \right) \\ \text{si } \theta_j \in \omega_{e_i} \end{cases}$$

et $\frac{\partial p_k}{\partial \theta_j}$ qui vérifie la récurrence :

$$\frac{\partial p_{k+1}}{\partial \theta_j} = \frac{AB_k}{b_k^T p_k} \left[I - \frac{p_k b_k^T}{b_k^T p_k} \right] \frac{\partial p_k}{\partial \theta_j} + \left(\frac{\partial A}{\partial \theta_j} B_k + A \frac{\partial B_k}{\partial \theta_j} \right) \frac{p_k}{b_k^T p_k} - \frac{AB_k p_k}{(b_k^T p_k)^2} \left(\frac{\partial b_k^T}{\partial \theta_j} p_k \right)$$

De plus si p_1 est la distribution initiale : $\frac{\partial p_1}{\partial \theta_j} = 0$ pour tout j .
et

$$\begin{cases} \frac{\partial B_k}{\partial \theta_j} & = & diag(0, \cdots, \frac{\partial b_k(i)}{\partial \theta_j}, \cdots, 0) & \text{si } \theta_j \in F_{e_i} \text{ ou } \Sigma_{e_i}^{-1} \\ \frac{\partial B_k}{\partial \theta_j} & = & 0 & \text{sinon} \end{cases}$$

$$\left\{ \begin{array}{lcll} \frac{\partial A}{\partial \theta_j} & = & C(\upsilon_{lm}) & \text{si } \theta_j \in A,\, \theta_j = \upsilon_{lm} \\ \frac{\partial A}{\partial \theta_j} & = & O_{N\times N} & \text{sinon} \end{array} \right.$$

et $C(\upsilon_{lm})$ définie par (11).

Remarque Pour la mise à jour des paramètres le long de la direction d'optimisation calculée, il est important de vérifier que les matrices de covariance inverses restent définies positives et de diminuer le pas de descente le cas échéant.

6 Application : Estimation récursive

6.1 Estimation récursive du maximum de vraisemblance

Un estimateur récursif θ_{n+1} du vecteur paramètre θ basé sur les $n+1$ premières observations de $(y_t)_{t\in\mathbb{N}^*}$ est de la forme :

$$\theta_{n+1} = \theta_n + \gamma_n H_n h(y_{n+1}, \theta_n)$$

où $h(y,\theta)$ est la fonction score, H_n une matrice adaptative et γ_n est une suite de gain satisfaisant

$$\gamma_n \leq 0,\ \sum_{n=1}^{\infty} \gamma_n = \infty,\ \sum_{n=1}^{\infty} \gamma_n^2 < \infty \tag{12}$$

Pour des observations indépendantes avec une densité $f(y,\theta)$, la fonction score est

$$h(y,\theta) = \left\{ \frac{\partial \ln\left(f(y,\theta)\right)}{\partial \theta_i},\ 1 \leq i \leq dimension\, of\, \theta \right\}$$

et le choix optimal pour H_n est l'inverse de la matrice d'information , i.e. $H_n^{-1} = I(\theta_n)$, où

$$I\left(\theta\right) = E\left[h(y,\theta)h(y,\theta)^T\right]$$

Le calcul de cette matrice d' information requiert une intégration numérique ce qui est très coûteux en temps de calcul. On utilisera donc à la place une estimation de cette matrice, i. e.

$$H_n^{-1} = \frac{1}{n}\sum_{k=1}^{n} h(y_k, \theta_{k-1})h(y_k, \theta_{k-1})^T$$

La matrice H_n peut être estimé récursivement grâce au lemme d'inversion de matrice de Ricatti : (en notant $h_n = h(y_n, \theta_{n-1})$)

$$H_n = \frac{1}{1-\gamma_n}\left(H_n - \frac{\gamma_n H_{n-1} h_n h_n^T H_{n-1}}{(1-\gamma_n) + \gamma_n h_n^T H_n h_n}\right)$$

6.2 Estimation récursive du maximum de vraisemblance pour un modèle autorégressif à changement de régime markovien

Dans notre cas les observations ne sont pas i.i.d., cependant la log-vraisemblance (**??**) a une forme similaire à la log-vraisemblance pour le cas i.i.d. On en déduit,l'algorithme récursif pour notre cas : Notons

$$\theta_n = ({\omega_{e_i}^n}^T, \cdots, {\omega_{e_N}^n}^T, \upsilon_{11}^n, \cdots, \upsilon_{(N-1)N}^n, (\Sigma_{e_1}^{n^{-1}})_{11}, \cdots, (\Sigma_{e_1}^{n^{-1}})_{dd}, \cdots, (\Sigma_{e_N}^{n^{-1}})_{dd})^T$$

le paramètre au temps n, et A_n la matrice associée avec $\left(\upsilon_{11}^n, \cdots, \upsilon_{(N-1)N}^n\right)$, on a

$$\begin{cases} \theta_{n+1} = \theta_n + \gamma_n H_n h_{n+1} \\ H_n = \frac{1}{1-\gamma_n}\left(H_n - \frac{\gamma_n H_{n-1} h_n h_n^T H_{n-1}}{(1-\gamma_n)+\gamma_n h_n^T H_n h_n}\right) \end{cases}$$

avec h_n le vecteur gradient tel que la j-ème coordonnées soit :

$$h_n(j) = \frac{\partial {b_n}^T}{\partial \theta_n^j} p_n + b_n^T \frac{\partial p_n}{\partial \theta_n^j}$$

où

$$p_{n+1} = \frac{A_n B_n p_n}{b_n^T p_n}$$

et

$$\frac{\partial p_{n+1}}{\partial \theta_n^j} = \frac{A_n B_n}{b_n^T p_n}\left[I - \frac{p_n b_n^T}{b_n^T p_n}\right]\frac{\partial p_n}{\partial \theta_n^j} + \left(\frac{\partial A_n}{\partial \theta_n^j} B_n + A_n \frac{\partial B_n}{\partial \theta_n^j}\right)\frac{p_n}{b_n^T p_n} - \frac{A_n B_n p_n}{(b_n^T p_n)^2}\left(\frac{\partial b_n^T}{\partial \theta_n^j} p_n\right)$$

Les conditions pour la consistance et la normalité asymptotique de ces procédures sont en général des questions ouvertes même dans le cas i.i.d. Dans cet article le modèle est très général et il n'existe pas de résultats théoriques. Nous verrons cependant que cet estimateur semble très bien se comporter sur des données simulées. Dans la suite, la valeur initiale du pas est $\gamma_0 = 0.08$, il décroît à la vitesse $\frac{1}{n^{1/2+1e-16}}$

7 Simulation avec deux MLP pour fonctions de régression

On simule une série avec deux MLP qui ont 2 entrées,une couche cachée de 3 unités (avec des tangentes hyperboliques pour fonction d'activation) et deux sorties.

Un MLP est une fonction paramétrique non-linéaire.

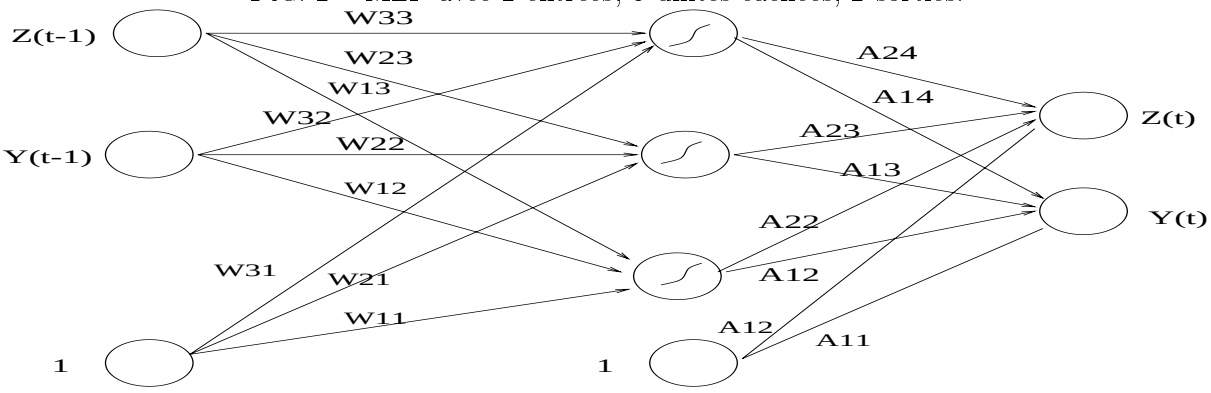
La fonction représentée ici est :

$$Z(t) = A_{24} tanh\left(W_{33} Z(t-1) + W_{32}(Y(t-1) + W_{31}) + A_{23} tanh(W_{23} Z(t-1) + W_{22} Y(t-1) + W_{21}\right)$$
$$+A_{22} tanh(W_{13} Z(t-1) + W_{12} Y(t-1) + W_{11})$$

et

$$Y(t) = A_{14} tanh\left(W_{33} Z(t-1) + W_{32}(Y(t-1) + W_{31}) + A_{13} tanh(W_{23} Z(t-1) + W_{22} Y(t-1) + W_{21}\right)$$
$$+A_{12} tanh(W_{13} Z(t-1) + W_{12} Y(t-1) + W_{11})$$

FIG. 2 – MLP avec 2 entrées, 3 unités cachées, 2 sorties.



– Les vecteurs poids $(W_{11}, \cdots, W_{13}, \cdots, W_{33}, A_{11}, \cdots, A_{14}, \cdots, A_{24})$des deux experts sont les suivants

$$MLP_1 : (0.38, 0.86, 0.88, 0.86, 0.08, -0.64, 0.54, 0.21, 0.23, 0.69, -0.77, -0.42, -0.05, 0.42, -0.52, -0.92, 0.26)$$

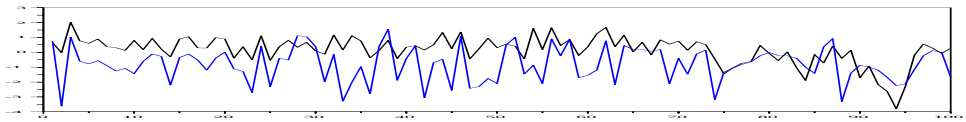
$$MLP_2 : (-0.36, 0.76, 0.70, -0.25, 0.94, 0.18, -0.24, 0.84, 0.16, -0.55, -0.94, -0.73, -0.48, 0.98, 0.40, -0.61, 0.74)$$

– La matrice de transition de la chaîne de Markov cachée est

$$A = \begin{pmatrix} 0.95 & 0.1 \\ 0.05 & 0.9 \end{pmatrix}$$

Les matrices de covariance des bruits sont $\Sigma_1 = \begin{pmatrix} 0.29 & 0.34 \\ 0.34 & 1.48 \end{pmatrix}$ et $\Sigma_2 = \begin{pmatrix} 1.09 & 0.33 \\ 0.33 & 0.1 \end{pmatrix}$ On simule une série qui contient 30000 données. Les 100 premières données sont

FIG. 3 – La série simulée



7.1 Estimation à l'aide des 1000 premières observations

Dans toute la suite les temps CPU donnés à titre indicatif sont obtenus sur un PC (400 Mhz).

On estime les paramètres sur 10 initialisations aléatoires du vecteur paramètre. Ici BFGS (cf. Press [7]) désigne un algorithme de gradient du second ordre (méthode quasi-newtonienne).

– Algorithme E.M : On fait 50 itérations de l'algorithme E.M., avec 10 itérations de BFGS pour chaque expert afin de calculer le M-Step.
– Optimisation Différentielle : On fait 100 itérations de BFGS.

– Estimation récursive : Comme l'algorithme n'a pas le temps de converger sur les 1000 premières valeurs de la série, on fait 30 passages.

log-vraisemblance pour les vrais paramètres : -1.20

algorithme E.M.	optimisation différentielle	optimisation récursive
-1.36805	-2.13761	-1.30815
-1.40708	-1.69924	-1.25308
-2.4903	-1.40469	-1.33211
-1.72062	-1.53105	-1.34769
-1.35515	-1.69533	-1.33211
-1.76341	-1.67276	-1.35613
-1.48806	-1.56657	-1.2944
-1.56468	-1.75466	-1.24691
-1.5589	-1.53692	-1.3545
-1.53337	-1.56613	-1.37256
CPU :1365 s.	CPU :664.91 s.	CPU :222 s.

Les matrices de covariances estimées pour le modèle avec la meilleure log-vraisemblance (-1.24691) sont

$$\widehat{\Sigma}_1 = \begin{pmatrix} 0.27 & 0.32 \\ 0.32 & 1.49 \end{pmatrix} \text{ et } \widehat{\Sigma}_2 = \begin{pmatrix} 1.10 & 0.33 \\ 0.33 & 0.10 \end{pmatrix}$$

et la matrice de transition estimée est

$$\widehat{A} = \begin{pmatrix} 0.96 & 0.1 \\ 0.04 & 0.9 \end{pmatrix}$$

7.2 Estimation récursive des paramètres avec toutes les données (30000).

Ici, on fait un seul passage sur toutes les données

log-vraisemblance pour les vrais paramètres : -1.18

Log-vraisemblance finale
-1.20364
-1.19179
-1.20097
-1.22429
-1.20561
-1.18873
-1.22802
-1.29867
-1.2385
-1.18573
CPU : 248.59 s.

Les matrices de covariances estimées pour le modèle avec la meilleure log-vraisemblance (-1.8573) sont

$$\widehat{\Sigma}_1 = \begin{pmatrix} 0.29 & 0.35 \\ 0.35 & 1.51 \end{pmatrix} \text{ et } \widehat{\Sigma}_2 = \begin{pmatrix} 1.10 & 0.33 \\ 0.33 & 0.10 \end{pmatrix}$$

et la matrice de transition estimée est

$$\widehat{A} = \begin{pmatrix} 0.95 & 0.1 \\ 0.05 & 0.9 \end{pmatrix}$$

8 Autre simulation : Fonctions de régressions linéaires, 4 états.

Les paramètres des fonctions autorégressives sont choisis de telle sorte que le modèle est stable.

- Les modèles AR ont pour équations :

$$AR_1 : \begin{array}{c} Y^1(t) \\ Y^2(t) \end{array} = \begin{array}{l} 0.42Y^1(t-1) - 0.37Y^2(t-1) + 0.19 \\ -0.39Y^1(t-1) - 0.40Y^2(t-1) - 0.16 \end{array}$$

$$AR_2 : \begin{array}{c} Y^1(t) \\ Y^2(t) \end{array} = \begin{array}{l} -0.57Y^1(t-1) - 0.19Y^2(t-1) + 0.28 \\ -0.30Y^1(t-1) - 0.37Y^2(t-1) - 0.01 \end{array}$$

$$AR_3 : \begin{array}{c} Y^1(t) \\ Y^2(t) \end{array} = \begin{array}{l} 0.13Y^1(t-1) - 0.17Y^2(t-1) + -0.8 \\ -0.40Y^1(t-1) + 0.47Y^2(t-1) - 0.43 \end{array}$$

$$AR_4 : \begin{array}{c} Y^1(t) \\ Y^2(t) \end{array} = \begin{array}{l} -0.46Y^1(t-1) - 0.50Y^2(t-1) - 0.04 \\ -0.44Y^1(t-1) - 0.48Y^2(t-1) - 0.65 \end{array}$$

- La matrice de transition est

$$A = \begin{pmatrix} 0.9 & 0.1 & 0.02 & 0.02 \\ 0.03 & 0.8 & 0.03 & 0.02 \\ 0.04 & 0.05 & 0.92 & 0.01 \\ 0.03 & 0.05 & 0.03 & 0.95 \end{pmatrix}$$

- Les matrices de covariance des bruits sont

$$\Sigma_1 = \begin{pmatrix} 0.29 & 0.34 \\ 0.34 & 1.48 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 1.09 & 0.33 \\ 0.33 & 0.1 \end{pmatrix}$$

$$\Sigma_3 = \begin{pmatrix} 0.05 & 0.04 \\ 0.04 & 0.05 \end{pmatrix}, \Sigma_4 = \begin{pmatrix} 0.73 & 0.39 \\ 0.39 & 0.34 \end{pmatrix}$$

8.1 Estimation à l'aide des 1000 premières observations

On estime les paramètres sur 10 initialisations aléatoires du vecteur paramètres.

- Algorithme E.M. : On fait 100 itérations de l'algorithme E.M., le calcul du M-Step est direct car le modèle est linéaire.
- Optimisation Différentielle : On fait 200 itérations de BFGS.
- Estimation récursive : Comme l'algorithme n'a pas le temps de converger sur les 1000 premières valeurs de la série, on fait 50 passages.

log-vraisemblance pour les vrais paramètres : -1.44 (1000 données)

algorithme E.M.	optimisation diff.	optimisation récursive
-1.55407	-1.82305	-1.43204
-1.46167	-1.59602	-1.86068
-1.91019	-1.61028	-1.43376
-1.8313	-1.82791	-1.42987
-1.55495	-1.8152	-1.42753
-1.46917	-1.6255	-1.43096
-1.49617	-1.60715	-1.42711
-1.55667	-1.63081	-1.44114
-1.5559	-1.71152	-1.44114
-1.94858	-1.81164	-1.42774
CPU :828 s.(100 it.)	CPU :1129 s.(200 it.)	CPU :550 s. (50 p.)

Les matrices de covariances estimées pour le modèle avec la meilleure log-vraisemblance (-1.42711) sont

$$\widehat{\Sigma}_1 = \begin{pmatrix} 0.28 & 0.29 \\ 0.29 & 1.46 \end{pmatrix} \text{ et } \widehat{\Sigma}_2 = \begin{pmatrix} 0.90 & 0.27 \\ 0.27 & 0.08 \end{pmatrix}$$

$$\widehat{\Sigma}_3 = \begin{pmatrix} 0.05 & 0.04 \\ 0.04 & 0.05 \end{pmatrix} \text{ et } \widehat{\Sigma}_4 = \begin{pmatrix} 0.73 & 0.38 \\ 0.38 & 0.35 \end{pmatrix}$$

et la matrice de transition estimée est

$$A = \begin{pmatrix} 0.91 & 0.18 & 0.02 & 0.03 \\ 0.04 & 0.74 & 0.03 & 0.01 \\ 0.04 & 0.04 & 0.92 & 0.01 \\ 0.01 & 0.04 & 0.03 & 0.95 \end{pmatrix}$$

8.2 Estimation récursive des paramètres avec toutes les données (30000).

On fait 2 passages sur toutes les données car le nombre de paramètres est plus grand que dans l'expérience précédente.

Log-vraisemblance finale ($\mathrm{l}(\theta_0)$=-1.21)
-1.21196
-1.21053
-1.2113
-1.21094
-1.24045
-1.23448
-1.21135
-1.21122
-1.21096
-1.21145
CPU : 776 s. (2 passages)

Les matrices de covariances estimées pour le modèle avec la meilleure log-vraisemblance (-1.21053) sont

$$\widehat{\Sigma}_1 = \begin{pmatrix} 0.30 & 0.38 \\ 0.38 & 1.58 \end{pmatrix} \text{ et } \widehat{\Sigma}_2 = \begin{pmatrix} 1.15 & 0.27 \\ 0.35 & 0.10 \end{pmatrix}$$

$$\widehat{\Sigma}_3 = \begin{pmatrix} 0.05 & 0.04 \\ 0.04 & 0.05 \end{pmatrix} \text{ et } \widehat{\Sigma}_4 = \begin{pmatrix} 0.73 & 0.38 \\ 0.38 & 0.33 \end{pmatrix}$$

et la matrice de transition estimée est

$$A = \begin{pmatrix} 0.90 & 0.10 & 0.02 & 0.02 \\ 0.03 & 0.79 & 0.03 & 0.02 \\ 0.04 & 0.06 & 0.92 & 0.01 \\ 0.03 & 0.05 & 0.03 & 0.95 \end{pmatrix}$$

9 Conclusion

La méthode décrite permet de calculer la log-vraisemblance et sa dérivée de manière relativement simple. Cela fournit une alternative à l'algorithme d'estimation classique (algorithme E.M.), pour ce genre de modèle . La forme additive de la log-vraisemblance, permet en plus d'estimer récursivement les paramètres, ce qui est réputé robuste vis-à-vis des minima locaux (les simulations le confirment) et permet aussi d'estimer les paramètres sur des séries extrêmement longues. Enfin, le calcul exact du gradient de la log-vraisemblance, pourra dans l'avenir fournir une méthode pour identifier le modèle grâce à une procédure du type Step-Wise descendant (cf [3]).

Références

[1] H. Boulard and N. Morgan. *Connectionist speech recognition : a hybrid approach.* Kluwer academic publ., 1994.

[2] H. Cartan. *Calcul différentiel.* Herman, 1970.

[3] M. Cottrell, et al. Neural modeling for time series : a statistical stepwise method for weight elimination. *IEEE Transaction on Neural Networks*, 6 :1355–1364, 1995.

[4] L. R. Holst. Recursive estimation in Switching autoregressions with a Markov regime . *Journal of time series analysis*, 77 :257–287, 1994.

[5] L. Mevel. Statistique asymptotique pour les modèles de Markov cachées. Thèse, Université de Rennes 1, 1997.

[6] A. B. Poritz. Linear predictive hidden Markov models and the speech signals. *IEEE transaction on signal processing*, 41(8) :2557–2573, 1982.

[7] William H. Press, et al. *Numerical recipes in C : The art of scientific computing.* Cambridge University Press, 1992.

[8] L. R. Rabiner. A tutorial on hidden Markov models and selected application in speech application. *proceedings of the IEEE*, 77 :257–287, 1989.

[9] J. Rynkiewicz. Hybrid HMM/MLP models for time series prediction. In *ESANN'99*, 1999.