TP R : modèle de markov caché

Fabrice Rossi

1 Rappels

1.1 Modèle de Markov caché (MMC)

On se donne une séquence temporelle de N observations dans $\mathbb{R}^d$, les $(x_i)_{1\leq i\leq N}$. On suppose que chaque observation est issue d'un mélange de K densités paramétriques sur $\mathbb{R}^d$, les $p_k(x|w_k)$, chaque w_k désignant l'ensemble des paramètres de la densité p_k. Pour tenir compte de la structure temporelle des données, on ne fait pas l'hypothèse classique d'indépendance. On suppose au contraire qu'il existe des variables cachées les $(z_i)_{1\leq i\leq N}$, prenant leurs valeurs dans $\{1,\dots,K\}$, avec des dépendances markoviennes (et une stationnarité temporelle).

Plus précisément, la vraisemblance des $(z_i)_{1\leq i\leq N}$ est supposée être de la forme suivante, celle d'une chaîne de Markov homogène :

$$p(z_1,\dots,z_N|\pi,A) = p(z_1|\pi)\prod_{i=2}^{N} p(z_i|z_{i-1},A),$$

où π et A sont les paramètres de la chaîne, c'est-à-dire :

$$\begin{aligned} p(Z_1 = k|\pi) &= \pi_k,\ \forall k,\\ p(Z_i = k|Z_{i-1} = l, A) &= A_{lk},\ \forall k,l. \end{aligned}$$

On suppose ensuite que chaque x_i dépend directement seulement du z_i correspondant, c'est-à-dire :

$$p(x_1,\dots,x_N,z_1,\dots,z_N|w,\pi,A) = \prod_{i=1}^{N} p(x_i|z_i,w)p(z_1|\pi)\prod_{i=2}^{N} p(z_i|z_{i-1},A),$$

avec $w = (w_1,\dots,w_K)$, l'ensemble des paramètres des K densités et avec

$$p(x_i|z_i = k,w) = p_k(x_i|w_k).$$

Dans la suite de l'énoncé, on note indifféremment z_i pour une valeur dans $\{1,\dots,K\}$ et pour un vecteur binaire de $\{0,1\}^K$ ne comportant qu'une unique coordonnée non nulle. On a $z_i = k$ si et seulement si le vecteur z_i comporte $K-1$ 0 et un 1 en position k.

1.2 Algorithme forward backward

L'algorithme EM peut être utilisé pour estimer par maximum de vraisemblance les paramètres du modèle, en s'appuyant sur un algorithme forward backward spécifique. On calcule les quantités suivantes :

$$\begin{aligned} \alpha_{i,k} &= p(x_1,\dots,x_i,Z_i = k|w,\pi,A),\\ \beta_{i,k} &= p(x_{i+1},\dots,x_N|Z_i = k,w,\pi,A). \end{aligned}$$

1.2.1 Phase forward

Le calcul des $\alpha_{i,k}$ se fait en augmentant i. On a ainsi

$$\begin{aligned}\alpha_{1,k} &= p_k(x_1|w_k)\pi_k,\ \forall k,\\ \alpha_{i,k} &= p_k(x_i|w_k)\sum_{l=1}^{K}\alpha_{i-1,l}A_{lk},\ \forall k,\ \forall i>1.\end{aligned}$$

1.2.2 Phase backward

Au contraire, le calcul des $\beta_{i,k}$ se fait en réduisant i. On a ainsi

$$\begin{aligned}\beta_{N,k} &= 1,\ \forall k,\\ \beta_{i,k} &= \sum_{l=1}^{K}\beta_{i+1,l}p_l(x_{i+1}|w_l)A_{kl},\ \forall k,\ \forall i<N.\end{aligned}$$

1.2.3 Vraisemblance

On montre que pour tout i

$$p(x_1,\dots,x_N|w,\pi,A)=\sum_{k=1}^{K}\alpha_{i,k}\beta_{i,k}.$$

En particulier, en prenant $i=N$, on a

$$p(x_1,\dots,x_N|w,\pi,A)=\sum_{k=1}^{K}\alpha_{N,k}.$$

1.2.4 Distribution des variables cachées

En utilisant les résultats de l'algorithme, on peut calculer les lois à posteriori des variables cachées, sachant les variables observées, ce qui donne

$$\begin{aligned}p(Z_i=k|x_1,\dots,x_N,w,\pi,A) &:= \gamma_{i,k},\\ &= \frac{\alpha_{i,k}\beta_{i,k}}{p(x_1,\dots,x_N|w,\pi,A)},\end{aligned}$$

et

$$\begin{aligned}p(Z_{i-1}=l,Z_i=k|x_1,\dots,x_N,w,\pi,A) &:= \xi_{i-1,l,i,k},\\ &= \frac{\alpha_{i-1,l}p_k(x_i|w_k)A_{lk}\beta_{i,k}}{p(x_1,\dots,x_N|w,\pi,A)}.\end{aligned}$$

1.3 Algorithme EM

On peut montrer que la phase M de l'algorithme EM conduit aux formules de mise à jour suivantes pour les paramètres du MMC :

$$\pi_k = \frac{\gamma_{1,k}}{\sum_{l=1}^{K} \gamma_{1,l}},$$
$$A_{lk} = \frac{\sum_{i=2}^{N} \xi_{i-1,l,i,k}}{\sum_{j=1}^{K} \sum_{i=2}^{N} \xi_{i-1,l,i,j}},$$
$$w_k = \arg\max_{w} \sum_{i=1}^{N} \gamma_{ik} \log p_k(x_i|w).$$

1.4 Algorithme de Viterbi

L'algorithme de Viterbi permet de déterminer la séquence d'états cachés la plus probable en utilisant un principe de programmation dynamique. On définit

$$V_{i,k} = \max_{z_1,\ldots,z_{i-1}} p(x_1, \ldots, x_i, Z_1 = z_1, \ldots, Z_{i-1} = z_{i-1}, Z_i = k).$$

On montre

$$V_{1,k} = p(x_1|Z_1 = k)\pi_k,$$
$$V_{i,k} = p_k(x_i|w_k) \max_l (A_{lk} V_{i-1,l}).$$

On note $z^*_{i,k}$ la valeur de l qui réalise le maximum dans l'expression ci-dessus, avec comme convention $z^*_{1,k} = k$. On montre alors que la séquence d'états cachés la plus vraisemblable est obtenue par

$$z^*_N = \arg\max_l V_{N,l},$$
$$z^*_{i-1} = z^*_{i-1,z^*_i}.$$

2 Mise en œuvre en R

2.1 Simulation

On étudie un cas simple dans lequel $d = 1$, c'est-à-dire que les données observées sont dans $\mathbb{R}$. On prend pour les p_k la loi normale centrée en μ_k et de variance σ_k^2.

Écrire une fonction R `mmcSimulation` qui simule un MMC. Elle prendra pour paramètres :

— un entier `K` donnant le nombre d'état du MMC ;
— un entier `N` donnant le nombre d'observations à engendrer ;
— un vecteur `pi` pour le paramètre π ;
— une matrice `A` pour le paramètre A ;
— un vecteur `mu` contenant les μ_k ;
— et un vecteur `sigma` contenant les σ_k.

La fonction doit renvoyer une liste contenant un vecteur `x` d'observations et un vecteur `z` contenant les valeurs des variables cachées. Le paramètre `K` est redondant mais permet de vérifier la cohérence des autres paramètres.

La figure 1 donne un exemple de séquence temporelle engendrée par un MMC et obtenue par une implémentation de la fonction demandée.

2.2 Algorithme forward backward

Pour programmer l'algorithme forward backward en R de façon efficace, il faut le formuler de façon vectorielle et matricielle.

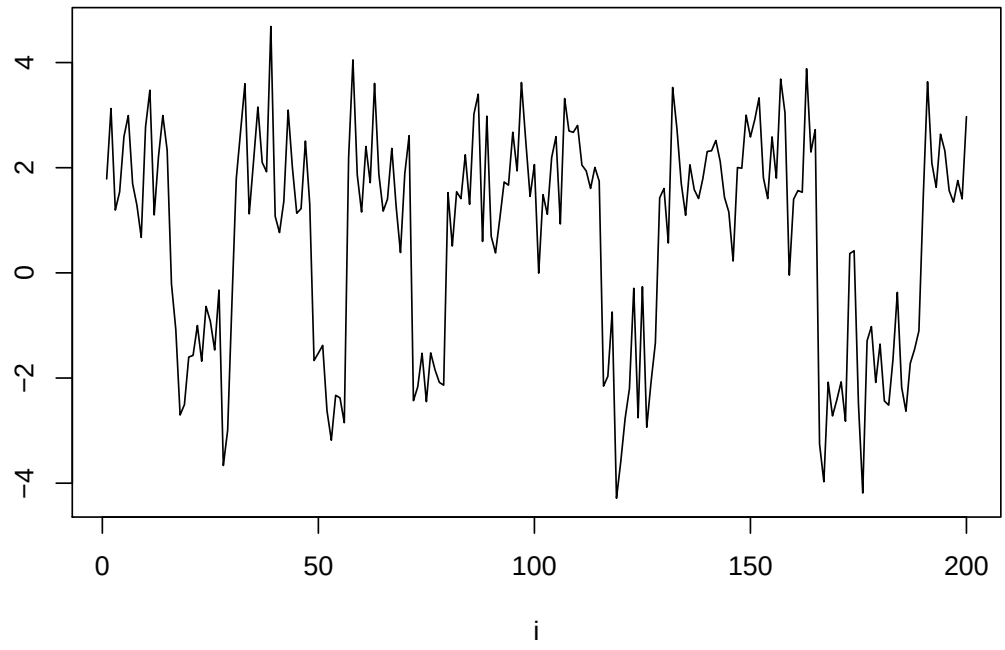

FIGURE 1 – Exemple de séquence temporelle engendrée par un MMC à deux états cachés

1. En considérant les vecteurs $\alpha_i = (\alpha_{i,1}, \ldots, \alpha_{i,K})^T$ et $\beta_i = (\beta_{i,1}, \ldots, \beta_{i,K})^T$, réécrire les équations de l'algorithme forward backward sous une forme matricielle.
2. Écrire une fonction R `mmcFB` qui applique cet algorithme. Elle prendre pour paramètres :
   — un entier `K` donnant le nombre d'état du MMC ;
   — un vecteur `x` contenant les observations ;
   — un vecteur `pi` pour le paramètre π ;
   — une matrice `A` pour le paramètre A ;
   — un vecteur `mu` contenant les μ_k ;
   — et un vecteur `sigma` contenant les σ_k.
   La fonction doit renvoyer une liste contenant une matrice `alpha` et une matrice `beta`. Chaque matrice contient en colonnes les vecteurs correspondants (α_i et β_i).

2.3 Vraisemblance et lois à posteriori

1. Écrire une fonction R `vraisemblance` qui calcule la vraisemblance des données à partir du résultat de la fonction `mmcFB`.
2. Écrire une fonction R `postProb` qui calcule les $\gamma_{i,k}$ et les $\xi_{i-1,l,i,k}$ à partir du résultat de la fonction `mmcFB` et des autres éléments utiles pour le calcul.

2.4 Algorithme EM

1. Établir des estimateurs pour les μ_k et σ_k à partir des formules générales de l'EM.
2. Écrire une fonction `param` qui prend en entrée les résultas de `postProb` (et toute autre information utile) et renvoie en résultats les nouvelles valeurs de paramètres du modèle.

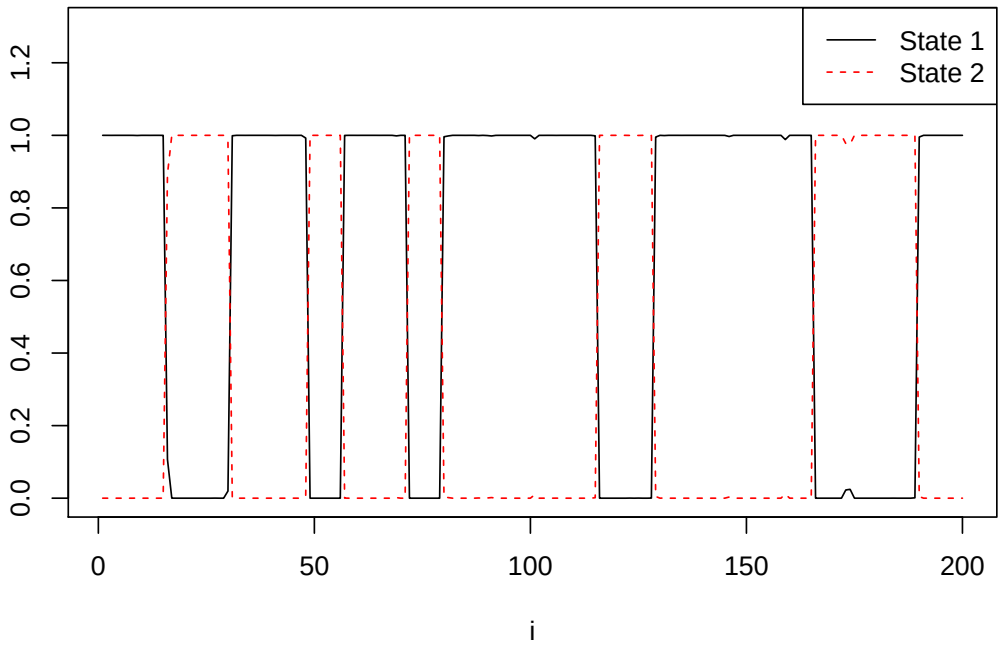

FIGURE 2 – Probabilités à posteriori des états cachés sachant les données

3. Écrire une fonction `mmcEM` qui applique l'algorithme EM à une séquence temporelle et un paramètre K fixant le nombre d'états cachés du MMC. Attention, l'initialisation de l'alrogithme joue un rôle important.
4. Tracer l'estimation finale des probabilités à posteriori.

2.5 Algorithme de Viterbi

Implémenter en R l'agorithme de Viterbi.