L'algorithme EM : une courte présentation

Frédéric Santos
CNRS, UMR 5199 PACEA
Avenue des facultés, Bât. B8
33400 Talence Cedex
Courriel : frederic.santos@u-bordeaux1.fr

5 août 2015

Résumé

Après avoir brièvement rappelé le principe de l'estimation par maximum de vraisemblance, ce document détaille comment et dans quel contexte l'algorithme EM peut être utilisé pour sa résolution.

Divers développements théoriques sont fournis — au moins pour le cadre discret — mais les problèmes fins de convergence sont éludés.

Enfin, quelques simulations informatiques sur des cas concrets sont réalisées. Les exemples d'application choisis sont très classiques, et concernent l'utilisation de l'algorithme EM pour l'estimation de paramètres dans le cadre d'un modèle de mélange.

Table des matières

1. Estimation par maximum de vraisemblance

1.1. Principe

Rappelons le principe de l'estimation par maximum de vraisemblance (ci-après désigné « MV »).

On dispose de n observations, considérées comme les réalisations de n variables aléatoires indépendantes et identiquement distribuées $(X_1, \cdots, X_n)$.

On se place dans le cadre d'un modèle statistique paramétrique $(E_X, \mathcal{E}, (\mathbb{P}_\theta)_{\theta\in\Theta})$: chaque X_i, $i = 1, \cdots, n$, suit une loi gouvernée par le vecteur de paramètres $\theta \in \mathbb{R}^d$. Par exemple, les X_i peuvent être des gaussiennes, toutes de même de loi $\mathcal{N}(\mu, \sigma^2)$ inconnue : on a alors $\theta = (\mu, \sigma^2)$, et $\Theta = \mathbb{R}^2$ sauf indication contraire.

On note $L_\theta(x_i)$, $L(x_i; \theta)$ ou plus souvent $L(x_i|\theta)$ dans la littérature anglophone, la densité de X_i. On appelle vraisemblance de l'échantillon, la densité jointe de $X_1, \cdots, X_n$:

$$L_n(x_1, \cdots, x_n; \theta) = \prod_{i=1}^{n} L(x_i; \theta) \tag{1}$$

Dans le cas particulier d'une loi discrète, cela se ramène à :

$$L_n(x_1, \cdots, x_n; \theta) = \mathbb{P}_\theta \{X_1 = x_1, \cdots, X_n = x_n\} = \prod_{i=1}^{n} \mathbb{P}_\theta \{X_i = x_i\} \tag{2}$$

Définition 1 (EMV). — L'estimateur au sens MV de θ est :

$$\widehat{\theta} = \arg\max_{\theta} \; L_n(x_1, \cdots, x_n; \theta) \tag{3}$$

C'est donc, à échantillon fixé, la valeur du vecteur de paramètres θ qui rend aussi vraisemblables que possible les observations obtenues.

Pour des raisons de commodité analytique, on préfère souvent maximiser $\log(L_n)$ plutôt que L_n. La fonction log étant strictement croissante, cela revient au même, tout en facilitant considérablement les calculs puisque :

$$\log\left(L_n(x_1, \cdots, x_n; \theta)\right) = \sum_{i=1}^{n} \log\left(L(x_i; \theta)\right) \tag{4}$$

et qu'il est généralement bien plus simple de travailler sur une somme plutôt que sur un produit.

1.2. Exemple

Estimation du paramètre d'une loi binomiale. — On joue à pile ou face avec une pièce potentiellement truquée. Sur 30 lancers, on obtient 22 fois « pile » et 8 fois « face ».

Formellement, on considère que les 30 lancers $(x_1, \cdots, x_{30})$ sont des réalisations i.i.d. suivant une loi $\mathcal{B}(p)$ de paramètre p inconnu, prenant la valeur 1 si on obtient pile, et 0 si on obtient face. Ainsi, le nombre $X = \sum X_i$ de « pile » obtenus suit une loi binomiale $\mathcal{B}(30, p)$.

Intuitivement, on estimerait p par la fréquence empirique de « pile », c'est-à-dire par $\widehat{p} = 22/30$. Il est aisé de vérifier qu'il s'agit de l'estimateur MV de p.

La vraisemblance de X est :

$$L_{30}(x_1, \cdots, x_{30}; p) = \prod_{i=1}^{30} \mathbb{P}_p \{X_i = x_i\} = p^{\sum X_i}(1-p)^{30-\sum X_i}$$

et donc sa log-vraisemblance est :

$$\log\left(L_{30}(x_1, \cdots, x_{30}; p)\right) = \log(p)\sum_{i=1}^{30} X_i + \log(1-p)(30 - \sum_{i=1}^{30} X_i)$$

On cherche à maximiser la log-vraisemblance : son maximum doit annuler sa dérivée. On résout donc :

$$\frac{\mathrm{d}}{\mathrm{d}p}\log\left(L_{30}(x_1, \cdots, x_{30}; \widehat{p}_{\mathrm{MV}})\right) = 0 \iff \frac{\sum X_i}{\widehat{p}_{\mathrm{MV}}} - \frac{(30 - \sum X_i)}{1 - \widehat{p}_{\mathrm{MV}}} = 0$$

d'où finalement comme attendu :

$$\widehat{p}_{\mathrm{MV}} = \frac{\sum X_i}{30}$$

Le lecteur pourra s'assurer qu'il s'agit bien d'un maximum en vérifiant que la dérivée seconde est négative.

2. « Philosophie » de l'algorithme EM

L'algorithme EM — pour Expectation-Maximisation — est un algorithme itératif du à Dempster, Laird et Rubin (1977). Il s'agit d'une méthode d'estimation paramétrique s'inscrivant dans le cadre général du maximum de vraisemblance.

Lorsque les seules données dont on dispose ne permettent pas l'estimation des paramètres, et/ou que l'expression de la vraisemblance est analytiquement impossible à maximiser, l'algorithme EM peut être une solution. De manière grossière et vague, il vise à fournir un estimateur lorsque cette impossibilité provient de la présence de données cachées ou manquantes [1] — ou plutôt, lorsque la connaissance de ces données rendrait possible l'estimation des paramètres.

L'algorithme EM tire son nom du fait qu'à chaque itération il opère deux étapes distinctes :

- la phase « Expectation », souvent désignée comme « l'étape E », procède comme son nom le laisse supposer à l'estimation des données inconnues, sachant les données observées et la valeur des paramètres déterminée à l'itération précédente ;
- la phase « Maximisation », ou « étape M », procède donc à la maximisation de la vraisemblance, rendue désormais possible en utilisant l'estimation des données inconnues effectuée à l'étape précédente, et met à jour la valeur du ou des paramètre(s) pour la prochaine itération.

En bref, l'algorithme EM procède selon un mécanisme extrêmement naturel : s'il existe un obstacle pour appliquer la méthode MV, on fait simplement sauter cet obstacle puis on applique effectivement cette méthode.

Le côté itératif de l'algorithme pourra peut-être paraître un peu mystérieux pour l'instant, mais comme nous le verrons, l'algorithme garantit que la vraisemblance augmente à chaque itération, ce qui conduit donc à des estimateurs de plus en plus corrects.

1. Il ne s'agit pas de la même chose... Les données manquantes peuvent être vues comme des « trous » dans la base de données du statisticien, par exemple des non-réponses à un questionnaire, alors qu'une donnée cachée lui est par définition inaccessible.

3. Aspects théoriques

Pour le détail théorique de l'algorithme EM, nous nous en tiendrons au cas discret, suffisant pour beaucoup d'applications, dont la génétique — voir à ce sujet [Mor08] pour un bel exposé.

La démarche ci-dessous est inspirée de [Bor04], et est compréhensible avec un bagage mathématique raisonnable. Elle se généralise sans difficulté au cas continu.

3.1. Rappels d'analyse convexe réelle

Définition 2 (Fonction convexe). — Une application $f : [a, b] \to \mathbb{R}$ est dite *convexe* sur $[a, b]$ si pour tous x_1, x_2 de cet intervalle et tout $\lambda \in [0, 1]$,

$$f(\lambda x_1 + (1-\lambda)x_2) \leqslant \lambda f(x_1) + (1-\lambda)f(x_2) \tag{5}$$

f est dite *strictement convexe* si l'inégalité (5) est stricte, et f est dite *concave* si $-f$ est convexe.

Théorème 1. — Si f est deux fois dérivable sur $[a, b]$ et si $f''(x) \geqslant 0$ pour tout $x \in [a, b]$, alors f est convexe sur cet intervalle. ♢

Théorème 2 (Inégalité de Jensen). — Soit f une fonction convexe définie sur un intervalle I. Si $x_1, \cdots, x_n \in I$ et $\lambda_1, \cdots, \lambda_n \geqslant 0$ tels que $\sum \lambda_i = 1$, alors :

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leqslant \sum_{i=1}^{n} \lambda_i f(x_i)$$

La démonstration est immédiate en procédant par récurrence. ♢

Exemple. — La fonction $x \mapsto -\log(x)$ est donc strictement convexe sur $\mathbb{R}_+^\star$ puisque sa dérivée seconde, $x \mapsto 1/x^2$, est strictement positive. L'inégalité de Jensen appliquée à la fonction $-\log$ fournit une relation qui sera d'une grande utilité dans ce qui suivra :

$$\log\left(\sum_{i=1}^{n} \lambda_i x_i\right) \geqslant \sum_{i=1}^{n} \lambda_i \log(x_i) \tag{6}$$

3.2. Formalisation d'une itération de l'algorithme

Pour formaliser un peu ce qui est exposé en section 2 :

- nous disposons d'observations i.i.d. $\mathbf{X} = (X_1, \cdots, X_n)$ de vraisemblance notée $P(\mathbf{X}|\theta)$;
- maximiser $\log P(\mathbf{X}|\theta)$ est impossible ;
- on considère des données cachées $\mathbf{Z} = (Z_1, \cdots, Z_n)$ dont la connaissance rendrait possible la maximisation de la « vraisemblance des données complètes », $\log P(\mathbf{X}, \mathbf{Z}|\theta)$;
- comme on ne connaît pas ces données $\mathbf{Z}$, on estime la vraisemblance des données complètes en prenant en compte toutes les informations connues : l'estimateur est naturellement $\mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m}[\log P(\mathbf{X}, \mathbf{z}|\theta)]$ (« étape E » de l'algorithme) ;
- et on maximise enfin cette vraisemblance estimée pour déterminer la nouvelle valeur du paramètre (« étape M » de l'algorithme).

Ainsi, le passage de l'itération m à l'itération $m+1$ de l'algorithme consiste à déterminer :

$$\theta_{m+1} = \arg\max_{\theta} \left\{ \mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m}[\log P(\mathbf{X}, \mathbf{z}|\theta)] \right\} \tag{7}$$

3.3. Preuve de la croissance de la vraisemblance d'une itération à l'autre

À l'itération m, nous disposons d'une valeur $\theta_m \in \mathbb{R}^d$ du vecteur de paramètres. Le but est de la mettre à jour avec une « meilleure » valeur θ, augmentant la vraisemblance, donc telle que $\Delta(\theta, \theta_m) := \log P(\mathbf{X}|\theta) - \log P(\mathbf{X}|\theta_m) \geqslant 0$. On souhaite bien sûr que cette différence soit la plus grande possible.

Cependant, comme précédemment exposé en section 2, on ne sait pas maximiser $P(\mathbf{X}|\theta)$, donc on ne sait pas non plus maximiser $\Delta(\theta, \theta_m)$... Un moyen d'optimiser malgré tout, dans une certaine mesure, cette différence, peut consister à chercher une fonction $\theta \mapsto \delta(\theta|\theta_m)$ que l'on sait maximiser, et qui est telle que :

$$\left\{ \begin{array}{rcl} \Delta(\theta, \theta_m)) & \geqslant & \delta(\theta|\theta_m) \quad \forall \theta \in \mathbb{R}^d \\ \delta(\theta_m|\theta_m) & = & 0 \end{array} \right. \tag{8}$$

Ainsi, $\delta(\theta|\theta_m)$ borne inférieurement $\Delta(\theta, \theta_m)$, et son maximum est au moins égal à 0. Trouver un θ' qui maximise $\theta \mapsto \delta(\theta|\theta_m)$ conduit donc mécaniquement à obtenir $\Delta(\theta', \theta_m) \geqslant 0$, c'est à dire une nouvelle valeur θ' plus vraisemblable des paramètres.

Afin de trouver une telle fonction δ, nous utilisons une représentation marginale de la vraisemblance selon les « données cachées » $\mathbf{Z} = (Z_1, \cdots, Z_n)$:

$$P(\mathbf{X}|\theta) = \sum_{\mathbf{z}} P(\mathbf{X}, \mathbf{z}|\theta) = \sum_{\mathbf{z}} P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta) \tag{9}$$

Il vient alors :

$$\begin{array}{rcll} \Delta(\theta, \theta_m) & = & \log P(\mathbf{X}|\theta) - \log P(\mathbf{X}|\theta_m) & \\ & = & \log \left(\sum_{\mathbf{z}} P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta) \right) \; - \; \underbrace{\sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m)}_{=1} \log P(\mathbf{X}|\theta_m) & (10) \end{array}$$

Cette expression utilise le logarithme d'une somme : en se souvenant de l'inégalité de Jensen (6), on commence à voir apparaître clairement une façon de minorer $\Delta(\theta, \theta_m)$...

Nous réécrivons (10) en introduisant dans la somme de gauche les $P(\mathbf{z}|\mathbf{X}, \theta_m)$ présents dans la somme de droite, afin de se rapprocher de l'expression (7) :

$$\Delta(\theta, \theta_m) = \log \left(\sum_{\mathbf{z}} \frac{P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_m)} \cdot P(\mathbf{z}|\mathbf{X}, \theta_m) \right) \; - \; \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log P(\mathbf{X}|\theta_m) \tag{11}$$

Et enfin, en remarquant que $\sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) = 1$, nous appliquons l'inégalité de Jensen :

$$\begin{array}{rcll} \Delta(\theta, \theta_m) & \geqslant & \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log \left(\frac{P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_m)} \right) \; - \; \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log P(\mathbf{X}|\theta_m) & \\ & = & \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log \left(\frac{P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_m)} \right) \; - \; \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log P(\mathbf{X}|\theta_m) & \\ & = & \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log \left(\frac{P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_m) P(\mathbf{X}|\theta_m)} \right) & (12) \\ & = & \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X}, \theta_m) \log \left(\frac{P(\mathbf{X}, \mathbf{z}|\theta)}{P(\mathbf{X}, \mathbf{z}|\theta_m)} \right) & (13) \\ & =: & \delta(\theta|\theta_m) & \end{array}$$

Nous avons donc obtenu une fonction $\theta \mapsto \delta(\theta|\theta_m)$ vérifiant les conditions (8) — il est évident avec (13) que $\delta(\theta_m|\theta_m) = 0$.

Finalement, nous posons :

$$\begin{aligned}
\theta_{m+1} &= \arg\max_{\theta} \ \delta(\theta|\theta_m) \qquad (14)\\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X},\theta_m) \log\left(\frac{P(\mathbf{X},\mathbf{z}|\theta)}{P(\mathbf{X},\mathbf{z}|\theta_m)}\right)\right\}\\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{X},\theta_m) \log P(\mathbf{X},\mathbf{z}|\theta)\right\}\\
&= \arg\max_{\theta} \left\{ \mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m}\left[\log P(\mathbf{X},\mathbf{z}|\theta)\right]\right\}
\end{aligned}$$

On détermine bien ainsi une valeur $\theta m + 1$ plus vraisemblable que θ_m, puisque :

$$\log P(\mathbf{X}|\theta_{m+1}) - \log P(\mathbf{X}|\theta_m) = \Delta(\theta_{m+1},\theta_m) \ \geqslant \ \delta(\theta_{m+1}|\theta_m) \ \geqslant \ \delta(\theta_m|\theta_m) \ \geqslant \ 0$$

Commentaires. — La mécanique itérative de l'algorithme est très astucieuse, et débouche sur une amélioration progressive et réciproque des données cachées $\mathbf{Z}$ et de la valeur du vecteur de paramètres θ.

En effet, on démarre l'algorithme avec une ignorance absolue des données cachées $\mathbf{Z}$ et en initialisant θ à une valeur θ_0 totalement arbitraire, potentiellement très loin de la réalité. L'algorithme se sert de θ_0 pour estimer $\mathbf{Z}$, puis se sert de $\widehat{\mathbf{Z}}$ pour réestimer les paramètres en une valeur θ_1 plus pertinente.

À l'itération suivante, on améliore donc l'estimation des données cachées $\mathbf{Z}$ puisque cette nouvelle estimation se base cette fois sur θ_1. Et cette meilleure précision sur $\widehat{\mathbf{Z}}$ conduit à son tour à une meilleure précision sur θ_2, etc.

Cette mécanique apparaîtra beaucoup plus clairement dans l'exemple développé en section 4.2.

Au final, l'algorithme EM fournit donc non seulement une estimation de plus en plus pertinente de θ, mais aussi une estimation de plus en plus pertinente de $\mathbf{Z}$! Si l'algorithme est classiquement utilisé pour l'estimation paramétrique, rien n'empêche de le considérer dualement comme une façon d'estimer les données cachées, si tel est notre but. Une autre utilisation de l'algorithme EM peut donc être la complétion de données manquantes...

3.4. Convergence

Les (non-)propriétés fines de convergence de l'algorithme relèvent d'un tout autre niveau et ne seront pas discutées ici. Il faut simplement noter que, dans certains cas, l'algorithme peut ne converger que vers un point-selle ou un maximum local de la vraisemblance.... si elle en possède un, naturellement. La dépendance en la condition initiale θ_0 choisie arbitrairement est forte : pour certaines mauvaises valeurs, l'algorithme peut rester gelé en un point selle, alors qu'il convergera vers le maximum global pour d'autres valeurs initiales plus pertinentes. L'algorithme EM peut donc parfois nécessiter plusieurs initialisations différentes.

3.5. Généralisation au cas continu

Tout ce qui précède se réécrit assez aisément dans le cas de lois continues. L'inégalité de Jensen s'applique tout aussi bien : on rappelle que si φ est une fonction convexe, alors pour toute variable aléatoire intégrable X, on a $\mathbb{E}[\varphi(X)] \geqslant \varphi(\mathbb{E}[X])$. Cela s'applique d'ailleurs aussi aux lois conditionnelles — on parle d'inégalité de Jensen conditionnelle.

On trouvera notamment dans [Mor08] comme dans [Col97] une réécriture de la démonstration de ce présent document dans le cas continu.

4. Exemples d'application aux modèles de mélange

4.1. Notion de loi de mélange

Définition 3 (Densité de mélange). — On appelle densité mélange, ou loi mélange, une fonction de densité qui est une combinaison linéaire convexe de plusieurs fonctions de densité. Autrement dit, f est une densité mélange s'il existe $K \in \mathbb{N}$, des densités $f_1, \cdots, f_K$ et des réels $p_1, \cdots, p_K$ sommant à 1, tels que :

$$f(x) = \sum_{i=1}^{K} p_i f_i(x) \tag{15}$$

En écrivant les p_i comme des $\mathbb{P}\{Z = i\}$ avec Z variable aléatoire discrète à valeurs dans $[\![1, K]\!]$, on a alors :

$$f(x) = \sum_{i=1}^{K} \mathbb{1}_{\{Z=i\}} \mathbb{P}\{Z = i\} f_i(x) \tag{16}$$

Cela revient à dire que, pour générer une réalisation d'une variable aléatoire suivant la loi f, on génère tout d'abord une réalisation de Z pour obtenir un nombre $j \in [\![1, K]\!]$, puis on génère une réalisation d'une v.a. suivant la loi f_j.

Pour la loi mélange, les densités f_j sont donc les densités conditionnelles sachant $\{Z = j\}$.

Mélange de lois gaussiennes. — Nous simulons ci-après le mélange de deux lois gaussiennes bidimensionnelles, respectivement $\mathcal{N}_2\left(\binom{2}{1}, 2/3\mathrm{Id}_{\mathbb{R}^2}\right)$ et $\mathcal{N}_2\left(\binom{7}{5}, \mathrm{Id}_{\mathbb{R}^2}\right)$. Le choix entre les deux lois s'opère *via* une loi de Bernoulli $\mathcal{B}(2/5)$.

La simulation est réalisée en R, de manière assez inélégante mais de façon à ce que les lecteurs ne connaissant pas ce langage puissent comprendre le code sans problème. Les commandes `rbinom(n,size,prob)` et `rnorm(n,mean,sd)` sont des générateurs (pseudo-)aléatoires de lois binomiale et gaussienne, où `n` est le nombre de tirages que l'on souhaite effectuer. Ainsi, par exemple, la commande `rbinom(5,10,0.4)` produirait 5 réalisations d'une loi $\mathcal{B}(10, 0.4)$.

Le code suivant est à l'origine de la figure 1 :

```
### On initialise le (futur) nuage de points :

X = matrix(nrow=100,ncol=2) # une ligne de la matrice = un point (x,y)
```

```
### Boucle permettant de tirer 100 valeurs issues
### d'un melange gaussien :

for (i in 1:100) {

 Z = rbinom(1,1,2/5) # choix de la loi par tirage de Benoulli

 if (Z == 1) {
  X[i,1] = rnorm(1,2,2/3)
  X[i,2] = rnorm(1,1,2/3)
 } else {
  X[i,1] = rnorm(1,7,1)
  X[i,2] = rnorm(1,5,1)
 }

}
plot(X,xlab="x",ylab="y",main="Melange de gaussiennes bidimensionnelles")
```

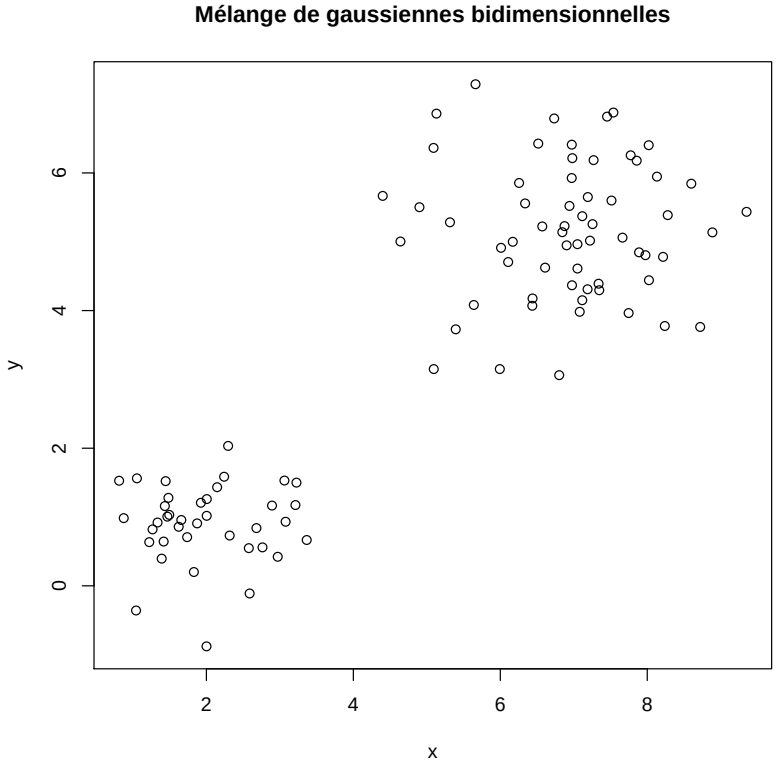

FIGURE 1 – Une simulation de mélange gaussien avec R

4.2. Jeu de pile ou face

Note. — L'exemple suivant est tiré de [Col97], mais légèrement adapté et réécrit avec des notations convenant mieux à nos vieilles habitudes françaises.

On s'amuse à un jeu de lancer de pièce obéissant aux règles suivantes : le joueur dispose de deux pièces, l'une a une probabilité p_1 d'obtenir pile, et l'autre une probabilité p_2. Avant chaque tour, le joueur choisit l'une ou l'autre des deux pièces suivant une loi de Bernoulli de paramètre λ — disons par exemple qu'il a une probabilité λ de choisir la pièce 1, et donc $1-\lambda$ de choisir la pièce 2. Enfin, un tour de jeu consiste en une série de trois lancers consécutifs avec la même pièce, et à noter le résultat obtenu — mais pas la pièce avec laquelle les lancers ont été réalisés.

Par exemple, une série de trois tours de jeu pourra donner lieu au relevé suivant :

$$\{P, P, F\}\,;\{F, F, F\}\,;\{P, F, P\}$$

À partir de ce relevé, le but est naturellement d'estimer les probabilités p_1 et p_2 de chaque pièce. Mais cela s'avère bien difficile par maximum de vraisemblance usuel : on ne sait pas quelle pièce a généré chacun des lancers... Cependant, si cette donnée était connue, l'estimation par MV s'avèrerait triviale — il s'agirait simplement de la fréquence empirique de « pile » obtenue pour chaque pièce, comme nous l'avons vu précédemment.

Nous sommes typiquement dans le cadre d'application de l'algorithme EM : une estimation de paramètres impossible à réaliser avec les seules données dont on dipose, mais la connaissance d'une donnée « cachée » rendrait immédiate la détermination des paramètres.

Formalisation du problème. — Nous dressons le décor suivant :

- On suppose que la partie comportera n tours de jeu.
- Soit $(Z_i)_{1\leqslant i\leqslant n}$ une suite de variables i.i.d. selon la loi $\mathcal{B}(\lambda)$. Si $Z_i = 1$, le joueur utilisera la pièce 1 au i-ème lancer, sinon, si $Z_i = 2$, il utilisera la pièce 2.
- La pièce 1 a une probabilité p_1 d'obtenir pile ; cette probabilité est p_2 pour la pièce 2.
- Le vecteur des paramètres du problème est donc $\theta = (\lambda, p_1, p_2)$.
- Un tour de jeu avec la pièce k (avec $k = 1$ ou 2) est une série de 3 réalisations i.i.d. de loi $\mathcal{B}(p_k)$, de telle sorte que la donnée connue à l'issue du tour de jeu i est par exemple de la forme $X_i = \{P, F, P\}$.
- Soit H_i le nombre de pile obtenus au tour i. Alors, conditionnellement à $Z_i = k$, la vraisemblance de X_i est
$$L(X_i|p_k) = p_k^{H_i}(1-p_k)^{3-H_i} \tag{17}$$
- On note $Y_i = \{X_i, Z_i\}$ les données augmentées pour le tour de jeu i, constituées du résultat obtenu et de la pièce avec laquelle le lancer a été réalisé.
- Enfin, on note $\mathbf{X} = (X_1, \cdots, X_n)$ et $\mathbf{Z} = (Z_1, \cdots, Z_n)$, les données observées et cachées pour l'ensemble des tours de jeu.

Application de l'algorithme EM. — On rappelle que, connaissant une valeur θ_m du vecteur de paramètres θ, l'algorithme EM vise à en trouver une plus vraisemblable, notée θ_{m+1}, telle que :

$$\begin{aligned}
\theta_{m+1} &= \arg\max_{\theta} \left\{ \mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m} \left[\log \left(L_n(\mathbf{X}, \mathbf{Z}) | \theta \right) \right] \right\} && (18) \\
&= \arg\max_{\theta} \left\{ \mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m} \left[\sum_{i=1}^{n} \log \left(L(X_i, Z_i) | \theta \right) \right] \right\} && (19) \\
&= \arg\max_{\theta} \left\{ \sum_{i=1}^{n} \mathbb{E}_{Z_i|X_i,\theta_m} \left[\log \left(L(X_i, Z_i) | \theta \right) \right] \right\} && (20)
\end{aligned}$$

Or ici, les données cachées Z_i — le numéro des pièces aux différents lancers — suivent une loi de Bernoulli.

On note, pour $i = 1, \cdots, n$, $\widetilde{p_i} = \mathbb{P}\left\{ Z_i = 1 | X_i, \theta_m \right\}$ la probabilité pour que ce soit la pièce 1 qui ait généré le i-ème lancer sachant le résultat X_i de ce lancer et une certaine valeur θ_m des paramètres. Il s'agit donc d'une probabilité *a posteriori*, dans la terminologie bayésienne.

Remarquons alors que bien évidemment, $1 - \widetilde{p_i} = \mathbb{P}\left\{ Z_i = 2 | X_i, \theta_m \right\}$ est la probabilité *a posteriori* d'avoir choisi la pièce 2 pour le i-ème lancer.

L'expression de (20) se résume alors à :

$$\theta_{m+1} = \arg\max_{\theta} \left\{ \sum_{i=1}^{n} \widetilde{p_i} \log \left(\mathbb{P}\left\{ X_i, Z_i = 1 | \theta \right\} \right) + (1 - \widetilde{p_i}) \log \left(\mathbb{P}\left\{ X_i, Z_i = 2 | \theta \right\} \right) \right\} \qquad (21)$$

La détermination des $\widetilde{p_i}$ permettra le calcul de l'espérance — et donc l'accomplissement de « l'étape E » de l'algorithme EM —, tous les autres termes étant connus.

On a :

$$\begin{aligned}
\widetilde{p_i} &= \mathbb{P}\left\{ Z_i = 1 | X_i, \theta_m \right\} = \frac{\mathbb{P}\left\{ X_i, Z_i = 1 | \theta_m \right\}}{\mathbb{P}\left\{ X | \theta_m \right\}} \\
&= \frac{\mathbb{P}\left\{ X_i | Z_i = 1, \theta_m \right\} \mathbb{P}\left\{ Z_i = 1 \right\}}{\mathbb{P}\left\{ X_i, Z_i = 1 | \theta_m \right\} + \mathbb{P}\left\{ X_i, Z_i = 2 | \theta_m \right\}}
\end{aligned}$$

d'où finalement :

$$\widetilde{p_i} = \frac{\lambda \mathbb{P}\left\{ X_i | Z_i = 1, \theta_m \right\}}{\lambda \mathbb{P}\left\{ X_i | Z_i = 1, \theta_m \right\} + (1 - \lambda) \mathbb{P}\left\{ X_i | Z_i = 2, \theta_m \right\}} \qquad (22)$$

Connaissant θ_m, ceci fournit une expression explicite de $\widetilde{p_i}$, notamment *via* la formule (17).

On a donc :

$$\begin{aligned}
& \mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m} \left[\log \left(L_n(\mathbf{X}, \mathbf{Z}) | \theta \right) \right] \\
= & \sum_{i=1}^{n} \widetilde{p_i} \log \left(\mathbb{P}\left\{ X_i, Z_i = 1 | \theta \right\} \right) + (1 - \widetilde{p_i}) \log \left(\mathbb{P}\left\{ X_i, Z_i = 2 | \theta \right\} \right) \\
= & \sum_{i=1}^{n} \widetilde{p_i} \log \left(\lambda \mathbb{P}\left\{ X_i | Z_i = 1, \theta \right\} \right) + (1 - \widetilde{p_i}) \log \left((1 - \lambda) \mathbb{P}\left\{ X_i | Z_i = 2, \theta \right\} \right) \\
= & \sum_{i=1}^{n} \widetilde{p_i} \log \left(\lambda p_1^{H_i} (1 - p_1)^{3 - H_i} \right) + (1 - \widetilde{p_i}) \log \left((1 - \lambda) p_2^{H_i} (1 - p_2)^{3 - H_i} \right) \\
= & \sum_{i=1}^{n} \widetilde{p_i} \log(\lambda) + (1 - \widetilde{p_i}) \log(1 - \lambda) + \widetilde{p_i} \log \left(p_1^{H_i} (1 - p_1)^{3 - H_i} \right) + (1 - \widetilde{p_i}) \log \left(p_2^{H_i} (1 - p_2)^{3 - H_i} \right)
\end{aligned}$$

Il ne reste plus qu'à effectuer « l'étape M » : maximiser en θ l'expression ci-dessus. On recherche un triplet $\widehat{\theta} = (\widehat{\lambda}, \widehat{p_1}, \widehat{p_2})$ annulant son gradient, donc tel que :

$$\begin{cases} \dfrac{\partial}{\partial\lambda}\mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m}\left[\log\left(L_n(\mathbf{X},\mathbf{Z})|\widehat{\theta}\right)\right] = \dfrac{\sum\widetilde{p_i}}{\widehat{\lambda}} - \dfrac{\sum(1-\widetilde{p_i})}{1-\widehat{\lambda}} & = 0 \\ \dfrac{\partial}{\partial p_1}(\cdot) = \displaystyle\sum_{i=1}^{n}\left(\widetilde{p_i}\,\dfrac{H_i p_1^{H_i-1}(1-p_1)^{3-H_i} - p_1^{H_i}(3-H_i)(1-p_1)^{2-H_i}}{p_1^{H_i}(1-p_1)^{3-H_i}}\right) & = 0 \\ \dfrac{\partial}{\partial p_2}(\cdot) = \displaystyle\sum_{i=1}^{n}\left((1-\widetilde{p_i})\,\dfrac{H_i p_2^{H_i-1}(1-p_2)^{3-H_i} - p_2^{H_i}(3-H_i)(1-p_2)^{2-H_i}}{p_2^{H_i}(1-p_2)^{3-H_i}}\right) & = 0 \end{cases}$$

Un rapide calcul mène à la mise à jour $\widehat{\theta}$ suivante des estimateurs connus θ_m :

$$\widehat{\lambda} = \frac{\sum\widetilde{p_i}}{n} \tag{23}$$

$$\widehat{p_1} = \frac{\sum\frac{H_i}{3}\widetilde{p_i}}{\sum\widetilde{p_i}} \tag{24}$$

$$\widehat{p_2} = \frac{\sum\frac{H_i}{3}(1-\widetilde{p_i})}{\sum(1-\widetilde{p_i})} \tag{25}$$

On posera donc, à l'étape suivante $m+1$ de l'algorithme, $\theta_{m+1} = (\widehat{\lambda}, \widehat{p_1}, \widehat{p_2})$.

Comme le souligne très justement M. Collins dans [Col97], ces formules sont parfaitement intuitives :

- Le paramètre λ est la probabilité de choisir la pièce 1 à chaque lancer, et son estimateur à chaque itération de l'algorithme est la moyenne des probabilités *a posteriori* d'avoir choisi la pièce 1 à chaque tour de jeu.
- Le paramètre p_1 est la probabilité d'obtenir un pile avec la pièce 1. Dans $\widehat{p_1}$, nous reconnaissons la fréquence empirique de « pile » obtenus au i-ème lancer : il s'agit de $H_i/3$. Pour estimer p_1, on moyenne en fait ces différentes fréquences empiriques en les pondérant par leur probabilité *a posteriori* d'avoir été issues de la pièce 1. Ainsi, la fréquence empirique de « pile » obtenus au tour i entrera d'autant plus en compte dans l'estimation de p_1, qu'il est plausible que ce lancer ait été effectué avec la pièce 1. Là encore, cela paraît intuitivement la solution qu'on aurait choisie *a bisto de nas*.
- Même interprétation pour le paramètre p_2...

4.3. Mélanges gaussiens

On reprend l'exemple du modèle de mélange de deux lois gaussiennes, présenté en section 4.1, p. 7.

Soit $\mathbf{X} = (X_1, \cdots, X_n)$ un échantillon i.i.d. d'observations issues d'un mélange de deux gaussiennes bidimensionnelles, et soit $\mathbf{Z} = (Z_1, \cdots, Z_n)$ la donnée cachée où Z_i détermine la distribution dont est issue X_i :

$$\mathcal{L}\left(X_i|\left\{Z_i = 1\right\}\right) = \mathcal{N}_2(\mu_1, \Sigma_1) \quad ; \quad \mathcal{L}\left(X_i|\left\{Z_i = 2\right\}\right) = \mathcal{N}_2(\mu_2, \Sigma_2)$$

avec $\mathbb{P}\left\{Z_i = 1\right\} = \lambda_1$ et $\mathbb{P}\left\{Z_i = 2\right\} = \lambda_2 = 1 - \lambda_1$.

Ne connaissant que le nuage de points $\mathbf{X}$ — tel celui en figure 1 p. 8 —, on cherche à estimer les 5 paramètres inconnus $\theta = (\lambda, \mu_1, \Sigma_1, \mu_2, \Sigma_2)$.

La vraisemblance des données complètes est :

$$L_n(\mathbf{X}, \mathbf{Z}|\theta) = \prod_{i=1}^{n} \left[\sum_{j=1}^{2} \mathbb{1}_{\{Z_i=j\}} \lambda_j f_j(X_i) \right] \tag{26}$$

où $f_j : \mathbb{R}^2 \to \mathbb{R}$ est une densité gaussienne bidimensionnelle de paramètres μ_j et Σ_j :

$$f_j(x) = \frac{1}{2\pi \det(\Sigma_j)^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_j)^\top \Sigma_j^{-1}(x-\mu_j) \right)$$

Il vient donc, pour log-vraisemblance des données complètes :

$$\log(L_n(\mathbf{X}, \mathbf{Z}|\theta)) = \sum_{i=1}^{n} \left[\sum_{j=1}^{2} \mathbb{1}_{\{Z_i=j\}} \left(\log(\lambda_j) - \log(2\pi) - \frac{1}{2}\log(\det(\Sigma_j)) - \frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j) \right) \right] \tag{27}$$

La suite de la démarche est parfaitement calquée sur celle du lancer de pièce : à chaque itération, l'étape E nécessite de définir la distribution *a posteriori* de Z_j connaissant X_j et θ_m. On définit :

$$\widetilde{p_{i,j}} := \mathbb{P}\left\{ Z_i = j | X_i = x_i, \theta_m \right\} = \frac{\lambda_j f_j(x_i)}{\lambda_1 f_1(x_i) + \lambda_2 f_2(x_i)} \tag{28}$$

la probabilité *a posteriori* pour que le point X_i soit issu de la distribution $f_j \equiv \mathcal{N}(\mu_j, \Sigma_j)$, connaissant θ_m.

Alors, on a :

$$\mathbb{E}_{\mathbf{Z}|\mathbf{X},\theta_m}\left[\log L_n(\mathbf{X}, \mathbf{Z}|\theta)\right] = \sum_{i=1}^{n}\sum_{j=1}^{2} \widetilde{p_{i,j}} \left(\log(\lambda_j) - \log(2\pi) - \frac{1}{2}\log(\det(\Sigma_j)) - \frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j) \right)$$

La maximisation en θ de cette expression, bien qu'un peu lourde, ne présente aucune difficulté majeure, et conduit aux estimateurs suivants (pour $j = 1$ ou 2) :

$$\begin{cases} \lambda_j^{(m+1)} &= \frac{1}{n}\sum_{i=1}^{n} \widetilde{p_{i,j}} \\ \mu_j^{(m+1)} &= \frac{\sum_{i=1}^{n} \widetilde{p_{i,j}} x_i}{\sum_{i=1}^{n} \widetilde{p_{i,j}}} \\ \Sigma_j^{(m+1)} &= \frac{\sum_{i=1}^{n} \widetilde{p_{i,j}} (x_i - \mu_j^{(m+1)})(x_i - \mu_j^{(m+1)})^\top}{\sum_{i=1}^{n} \widetilde{p_{i,j}}} \end{cases} \tag{29}$$

Même remarque que pour le lancer de pièces : ces estimateurs ont une interprétation intuitive très claire, et exactement identique à ceux du précédent exemple.

Remarques. — Ces formules se généralisent de manière triviale à des lois gaussiennes p-dimensionnelles, avec $p \neq 2$ quelconque.

4.4. Paramètres d'un mélange gaussien : implémentation en R

On touche ici à l'une des « limites » de l'algorithme EM : bien que son emploi effectif fasse appel à un ordinateur, chaque cas nécessite au prélable une étude sur papier pour déterminer la formule des estimateurs calculés à chaque itération. Il n'existe pas « une » implémentation de l'algorithme qui soit applicable à toutes les situations : bien au contraire, chaque implémentation est applicable à une et une seule situation.

L'algorithme EM est en effet une méthode très générale, et les détails de l'implémentation pour une situation donnée dépendent évidemment du problème, du modèle sous-jacent, etc. Il est également à noter que dans ce document nous n'avons exposé que des exemples très simples, mais dans beaucoup de situations pratiques, l'étape E de l'algorithme nécessite de faire appel à des méthodes MCMC pour avoir une valeur approchée de l'espérance conditionnelle...

Voici par exemple l'algorithme EM appliqué à la recherche des paramètres d'un mélange de deux gaussiennes unidimensionnelles $\mathcal{N}(\mu_1, \sigma_1^2)$ et $\mathcal{N}(\mu_2, \sigma_2^2)$, avec choix entre les deux gaussiennes suivant une loi $\mathcal{B}(\lambda)$. Nous choisirons ici $\lambda = 0,4$; $\mu_1 = 124$; $\mu_2 = 157$; $\sigma_1 = 8$; $\sigma_2 = 7$.

Les formules (29) et (28) du cas bidimensionnel s'adaptent très simplement au cas de gaussiennes unidimensionnelles, et ce sont elles qui sont implémentées ci-dessous :

```
#####################################################################
# G\'en\'eration de donn\'ees issues d'un m\'elange gaussien : #
#####################################################################

X = NULL
X = matrix(nrow=100,ncol=1)

### Boucle permettant de tirer 100 valeurs issues
### d'un m\'elange gaussien :

for (i in 1:100) {

 Z = rbinom(1,1,2/5) # choix de la loi par tirage de Benoulli

 if (Z == 1) {
  X[i] = rnorm(1,124,8)
 } else {
  X[i] = rnorm(1,157,7)
 }
}


#############
# Algo EM : #
#############
```

```
## D\'efinition arbitraire des valeurs initiales des param\‘etres :

lambda1 = 0.2
lambda2 = 0.8
mu1 = 110
mu2 = 170
sigma1 = 5
sigma2 = 5


K = 30 # on fera K it\'erations de l'algorithme

for (i in 1:K) {

 ## Application de la formule (28) :
 vrais1 = lambda1*dnorm(X,mean=mu1,sd=sigma1)
 vrais2 = lambda2*dnorm(X,mean=mu2,sd=sigma2)
 vrais12 = vrais1 / (vrais1 + vrais2) # probas a posteriori p_{i,1}
 vrais22 = vrais2 / (vrais1 + vrais2) # probas a posteriori p_{i,2}

 ## Mise \‘a jour de lambda1 = P(Z=1 | X,Thteta) :
 lambda1 = mean(vrais12)
 lambda2 = 1-lambda1

 ## Mise \‘a jour de mu1 et mu2 :
 mu1 = sum(vrais12*X)/sum(vrais12)
 mu2 = sum(vrais22*X)/sum(vrais22)

 ## Mise \‘a jour de sigma1 et sigma2 :
 sigma1 = sqrt(sum(vrais12*(X-mu1)^2)/(sum(vrais12)))
 sigma2 = sqrt(sum(vrais22*(X-mu2)^2)/(sum(vrais22)))
}
```

À l'issue d'une simulation, l'algorithme EM renvoie les valeurs suivantes des paramètres :

$$\lambda = 0,34$$

$$\mu_1 = 126,36 \ ; \ \sigma_1 = 7,08$$

$$\mu_2 = 157,14 \ ; \ \sigma_2 = 6,37$$

On pourra noter la précision tout à fait convenable sur les valeurs obtenues.

4.5. Autres perspectives

La reconnaissance de mélanges gaussiens est une des applications fondamentales de l'algorithme EM, et a de nombreuses applications : traitement du signal, traitement de l'image, notamment en imagerie médicale.

L'algorithme EM est également très utilisé pour déterminer les paramètres d'un modèle de Markov caché (*Hidden Makov Model* en Anglais, abrégé HMM).

Références

[Bor04] Sean Borman : The Expectation Maximization algorithm : A short tutorial. `www.isi.edu/natural-language/teaching/cs562/.../B06.pdf`, Juillet 2004.

[Col97] Michael Collins : The EM algorithm. `www.cse.unr.edu/~bebis/CS679/Readings/EM_Algorithm_Review.pdf`, Septembre 1997.

[Mor08] Stephan Morgenthaler : *Génétique statistique*. Collection « Statistique et probabilités appliquées ». Springer, 2008.