Classification non supervisée par modèles de mélange

Cathy Maugis-Rabusseau

4modIA / INSA Toulouse & ENSEEIHT

2022-2023

Plan

Hypothèses des mélanges finis

- Données : On observe n individus décrits par p variables

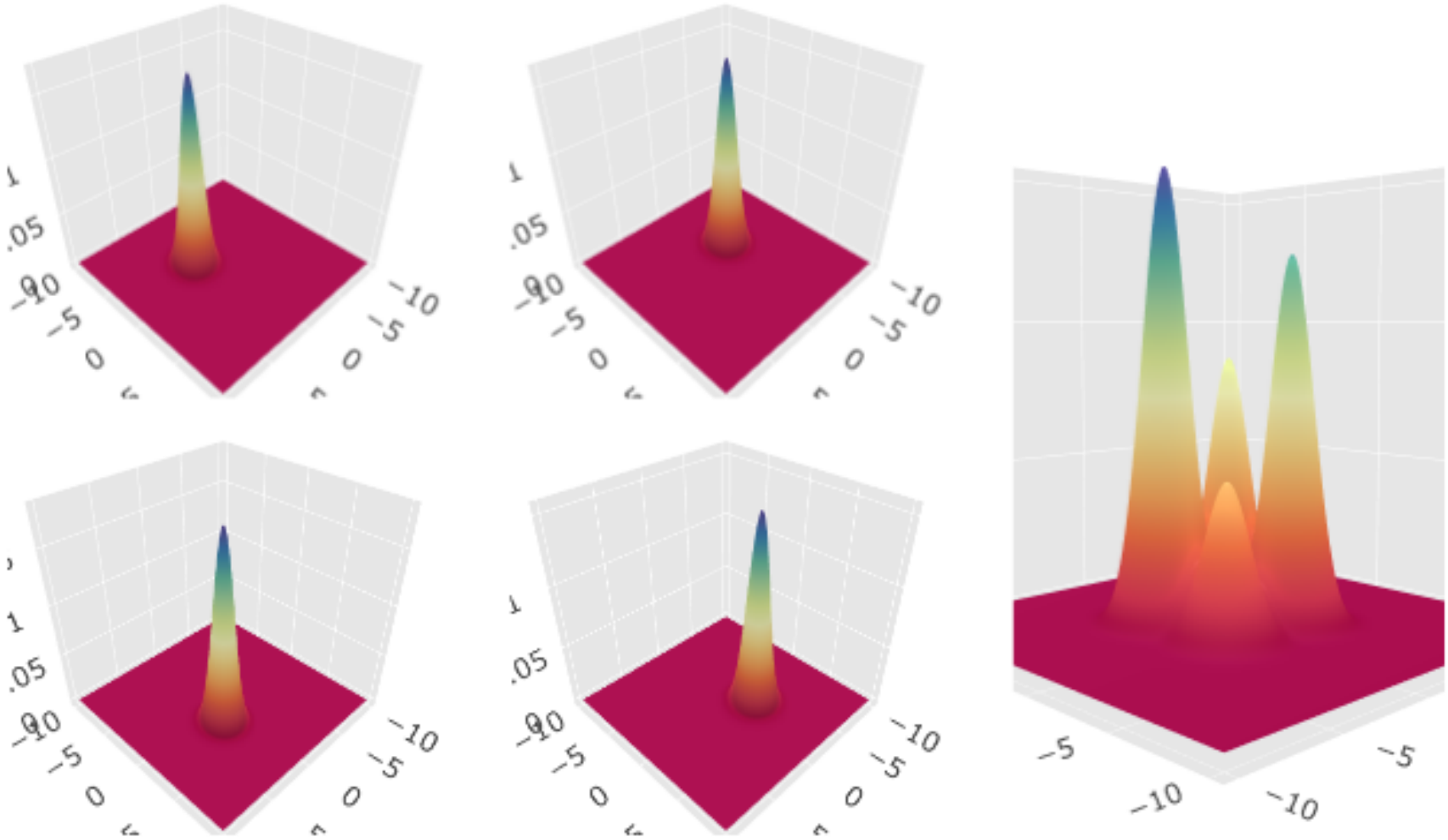
$$\underline{\mathbf{x}} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \quad \text{avec } x_i = (x_{i1}, \dots, x_{ip}) \in \mathcal{X}$$

- On suppose que les données proviennent d'une population contenant plusieurs sous-populations

- Chaque sous-population est modélisée indépendamment des autres (**choix d'une loi de distribution pour chaque sous-population**).

- La population totale est alors vue comme un mélange de ces sous-populations. Le modèle résultant est un **modèle de mélange fini**.

Mélanges finis



Mélanges finis

- Un modèle de mélange à K composantes est de la forme

$$f(.|\theta_K) = \sum_{k=1}^{K} \pi_k f_k(.|\alpha_k)$$

  - $(\pi_1, \ldots, \pi_K)$ sont les proportions du mélange

$$\forall k \in \{1, \ldots, K\},\ \pi_k \in [0,1] \text{ et } \sum_{k=1}^{K} \pi_k = 1$$

  - $f_k(.|\alpha_k)$ est la densité de la $k$ème sous-population
  - $\theta_K = (\pi_1, \ldots, \pi_K, \alpha_1, \ldots, \alpha_K)$

- Le choix des f_k dépend de la nature des données

Structure cachée du modèle

- Données observées : $\underline{\mathbf{x}} = (x_1, \ldots, x_n)$ avec $x_i = (x_{i1}, \ldots, x_{ip})$
- Données manquantes (variables latentes) : $\underline{\mathbf{z}} = (z_1, \ldots, z_n)$ avec $z_i = (z_{i1}, \ldots, z_{iK})$ tel que $z_{ik} = \mathbb{1}_{i \in \mathcal{C}_k}$.
- Données complétées : $(\underline{\mathbf{x}}, \underline{\mathbf{z}}) = \{(x_1, z_1), \ldots, (x_n, z_n)\}$
- Rem : $\underline{\mathbf{z}}$ défini une partition des données observées $\underline{\mathbf{x}}$: $\mathcal{C}_k = \{i \in \{1 \ldots, n\}; z_{ik} = 1\}$ pour $k = 1, \ldots, K$

Modèle génératif

- On suppose les proportions $\pi_1, \ldots, \pi_K$ et les densités $f_1, \ldots, f_k$ fixées
- Les données sont alors générées de la façon suivante :
  - $z_i = (z_{i1}, \ldots, z_{iK}) \sim \mathcal{M}(1; \pi_1, \ldots, \pi_K)$ (loi multinomiale)
    un individu i appartient à $\mathcal{C}_k$ avec probabilité π_k : $\mathbb{P}(z_{ik} = 1 | \theta_K) = \pi_k$
  - x_i est généré selon la densité f_k si i appartient à $\mathcal{C}_k$ $(z_{ik} = 1)$

$$f(x_i | z_{ik} = 1, \theta_K) = f_k(x_i | \alpha_k)$$

- Ainsi

$$f(x_i, z_i | \theta_K) = \prod_{k=1}^{K} \{\pi_k f_k(x_i | \alpha_k)\}^{z_{ik}}$$

et

$$f(x_i | \theta_K) = \sum_{k=1}^{K} f(x_i | z_{ik} = 1, \theta_K) \mathbb{P}(z_{ik} = 1 | \theta_K) = \sum_{k=1}^{K} \pi_k f_k(x_i | \alpha_k)$$

Les grandes étapes

1. Collection de modèles :

$$\forall K \in \mathbb{N}^\star,\ \mathcal{S}_K = \left\{ x \in \mathbb{R}^p \mapsto f(x|\theta_K) = \sum_{k=1}^{K} \pi_k f_k(.|\alpha_k) \right\}$$

Mélanges à K=2 classes | **Mélanges à K=3 classes** | **Mélanges à K=4 classes** | **Mélanges à K=5 classes** | ...

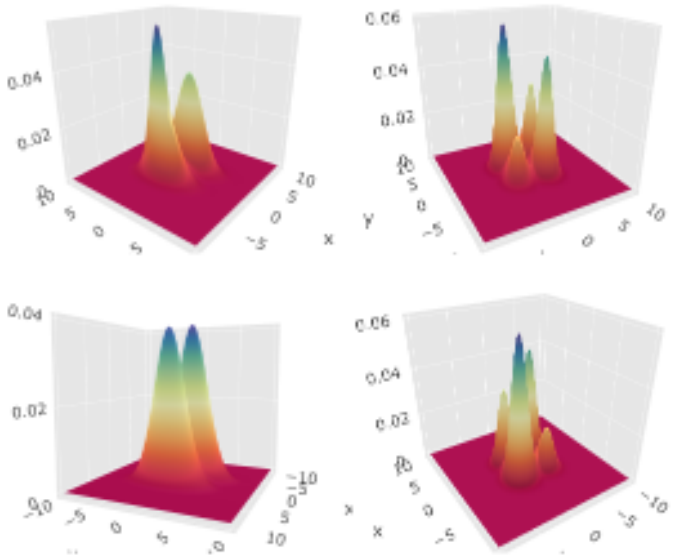
$\Rightarrow$ Choix initial de la modélisation

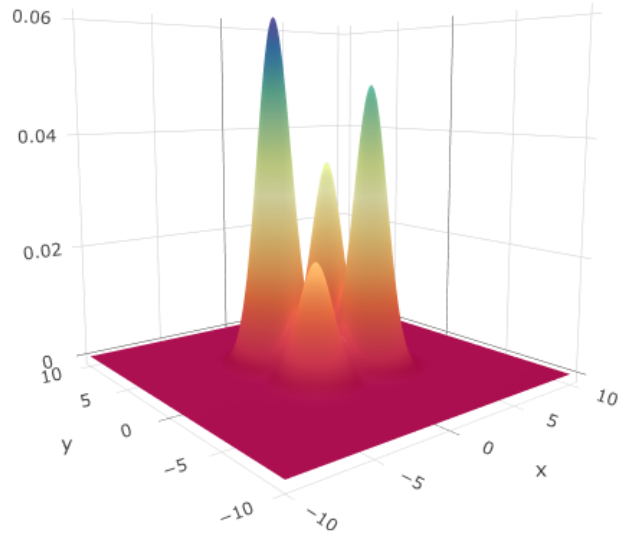
2. Dans chaque modèle $\mathcal{S}_K$: on détermine le mélange qui s'ajuste le mieux aux données: $f(.|\hat{\theta}_K)$

$\Rightarrow$ Besoin d'un algorithme d'estimation des paramètres $(\hat{\theta}_K)$

Les grandes étapes

3. Choisir le "meilleur" mélange parmi $f(.|\hat{\theta}_2), f(.|\hat{\theta}_3), \ldots, f(.|\hat{\theta}_{K_{\max}})$





$\Rightarrow$ Besoin d'un critère de sélection de modèles pour déterminer $\hat{K}$ et donc choisir $f(.|\hat{\theta}_{\hat{K}})$.

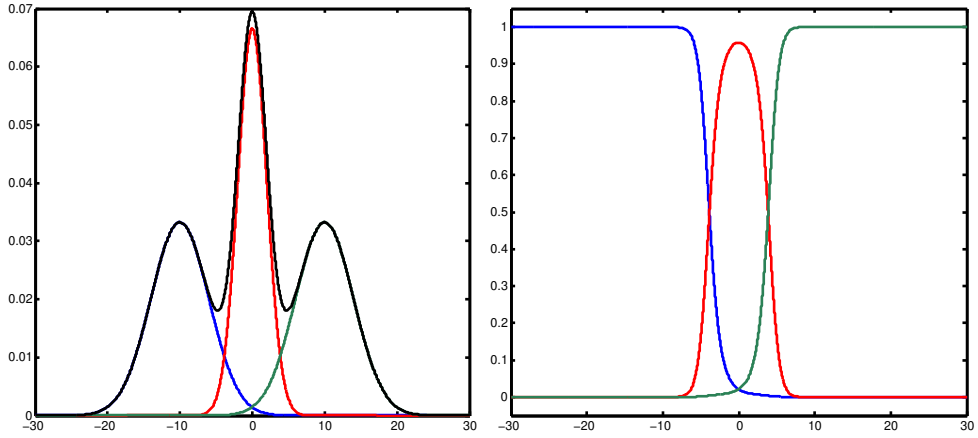
4. Règle du "MAP" pour en déduire une classification des données

Etape 4 : Règle du MAP

- Principe : chaque individu est affecté à la classe pour laquelle il a la plus forte probabilité d'appartenance conditionnellement à l'estimation des paramètres

- Probabilité conditionnelle d'appartenance de i à $\mathcal{C}_k$ avec le vecteur de paramètres θ

$$\begin{aligned} t_{ik}(\theta) &= \mathbb{P}(z_{ik} = 1 | x_i, \theta) \\ &= \frac{\pi_k f_k(x_i|\alpha_k)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x_i|\alpha_\ell)} \end{aligned}$$

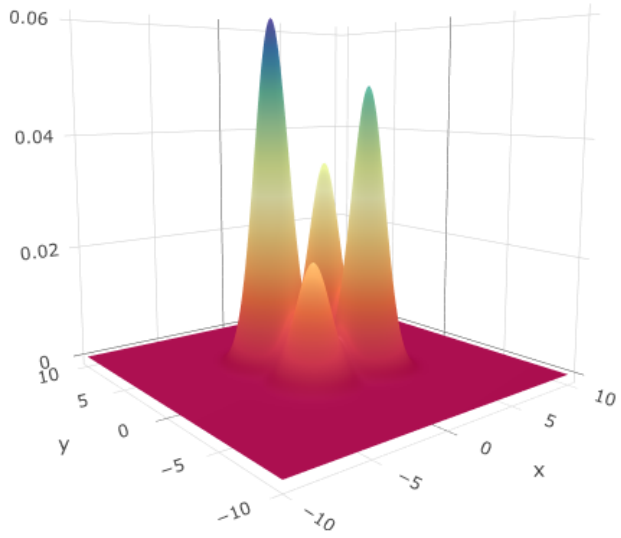


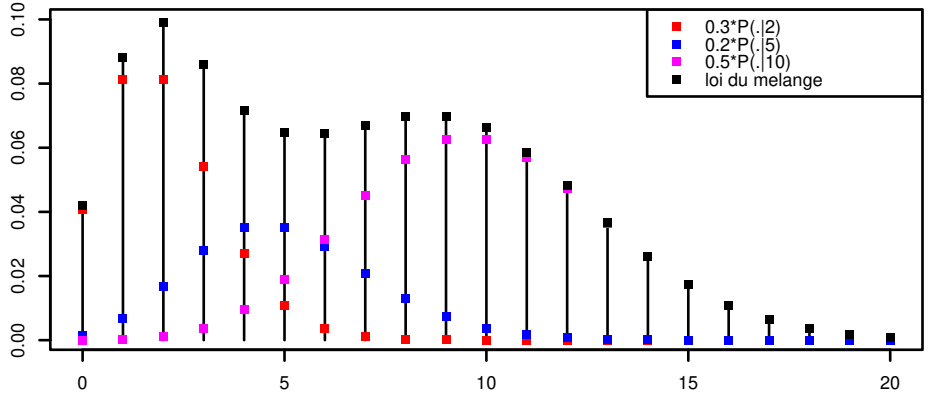
- Règle du maximum a posteriori (MAP) avec $\hat{\theta}_{\hat{K}}$:

$$i \in \mathcal{C}_k \text{ si } t_{ik}\left(\hat{\theta}_{\hat{K}}\right) > t_{i\ell}\left(\hat{\theta}_{\hat{K}}\right) \ \forall \ell \neq k$$

Etape 1 : Choix des distributions

- Données quantitatives : mélanges gaussiens, mélanges de Student, ...
- Données de comptages : mélanges de Poisson, de binomiales négatives, ...
- Données qualitatives : mélanges de multinomiales, ...
- Données compositionnelles : mélanges de Dirichlet, ...





Plan

Estimation du maximum de vraisemblance

- On désire déterminer le vecteur des paramètres $\theta_K = (\pi_1, \ldots, \pi_K, \alpha_1, \ldots, \alpha_K)$ qui **maximise la logvraisemblance** :

$$\mathcal{L}(\underline{\mathbf{x}}|\theta_K) = \mathcal{L}(x_1, \ldots, x_n|\theta_K) = \sum_{i=1}^{n} \ln \left[\sum_{k=1}^{K} \pi_k f_k(x_i|\alpha_k) \right]$$

- Ce problème de maximisation ne possède généralement pas de solution analytique

- $\Rightarrow$ Algorithme d'optimisation pour approcher $\hat{\theta}_K$

Vraisemblance classifiante

- Algorithme s'appuyant sur la notion de données complétées :

$$(\underline{\mathbf{x}}, \underline{\mathbf{z}}) = \{\underbrace{(x_1, \ldots, x_n)}_{\text{données observées}}, \underbrace{(z_1, \ldots, z_n)}_{\text{variables latentes}}\}$$

- Logvraisemblance des données complétées ou logvraisemblance classifiante :

$$\begin{aligned}
\mathcal{L}(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\theta_K) &= \ln\left[f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\theta_K)\right] \\
&= \ln\left[\prod_{i=1}^{n}\prod_{k=1}^{K}\left\{\pi_k f_k(x_i|\alpha_k)\right\}^{z_{ik}}\right] \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\left\{\ln(\pi_k) + \ln(f_k(x_i|\alpha_k))\right\}
\end{aligned}$$

Algorithme EM [7]

- EM = Expectation Maximization
- Maximisation par itérations successives de l'espérance de la logvraisemblance classifiante conditionnellement aux observations et une valeur courante des paramètres $\theta^{(r)}$:

$$\begin{aligned} \mathcal{Q}\left(\theta_K|\theta_K^{(r)}\right) &:= \mathbb{E}\left[\ln(f(\underline{\mathbf{x}},\underline{\mathbf{z}}|\theta_K)|\underline{\mathbf{x}},\theta_K^{(r)}\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K}\mathbb{E}\left[z_{ik}|\underline{\mathbf{x}},\theta_K^{(r)}\right]\{\ln(\pi_k)+\ln(f_k(x_i|\alpha_k))\} \end{aligned}$$

avec $\mathbb{E}\left[z_{ik}|\underline{\mathbf{x}},\theta_K^{(r)}\right] = \mathbb{P}\left(z_{ik}=1|\underline{\mathbf{x}},\theta_K^{(r)}\right) = t_{ik}(\theta_K^{(r)}) := t_{ik}^{(r)}$

Algorithme EM

- Initialisation: $\theta_K^{(0)}$
- Étape E: Calcul de $\mathcal{Q}\left(\theta_K|\theta_K^{(r)}\right) \Leftrightarrow$ calcul des $t_{ik}^{(r)}$

$$t_{ik}^{(r)} = \mathbb{P}\left(z_{ik}=1|x_i,\theta_K^{(r)}\right) = \frac{\pi_k^{(r)} f_k\left(x_i|\alpha_k^{(r)}\right)}{\sum_{\ell=1}^{K} \pi_\ell^{(r)} f_\ell\left(x_i|\alpha_\ell^{(r)}\right)}$$

- Étape M: Déterminer $\theta_K^{(r+1)} = \underset{\theta_K\in\Theta_K}{\text{argmax}}\ \mathcal{Q}\left(\theta_K|\theta_K^{(r)}\right)$

$$\begin{cases} \pi_k^{(r+1)} = \frac{1}{n}\sum_{i=1}^{n} t_{ik}^{(r)} \\ \\ (\alpha_1^{(r+1)},\ldots,\alpha_K^{(r+1)}) = \underset{(\beta_1,\ldots,\beta_K)}{\text{argmax}} \sum_{i=1}^{n}\sum_{k=1}^{K} t_{ik}^{(r)} \ln[f_k(x_i|\beta_k)] \end{cases}$$

Caractéristiques de l'algorithme EM

- Propriétés :

Propriété 1

A chaque itération de l'algorithme EM, la logvraisemblance croit.

Propriété 2

L'algorithme EM converge mais pas nécessairement vers le maximum global de la logvraisemblance.

- En pratique :
  - Facile à mettre en oeuvre
  - Parfois lent à converger (en particulier lorsque les composants sont très mélangés)
  - Sensible à l'initialisation (i.e. au choix de $\theta_K^{(0)}$)

Exemple : mélange gaussien unidim. à 2 composantes

- $x \mapsto \beta\phi(x|\mu_1,\sigma_1^2) + (1-\beta)\phi(x|\mu_2,\sigma_2^2)$ avec

$$\phi(x|\mu_k,\sigma_k^2) = (2\pi\sigma_k^2)^{-\frac{1}{2}} \exp\left[-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right]$$

- Étape E :

$$t_{i1}^{(r)} = \frac{\beta^{(r)}\phi(x_i|\mu_1^{(r)},\sigma_1^{2\,(r)})}{\beta^{(r)}\phi(x_i|\mu_1^{(r)},\sigma_1^{2\,(r)}) + (1-\beta^{(r)})\phi(x_i|\mu_2^{(r)},\sigma_2^{2\,(r)})} = 1 - t_{i2}^{(r)}$$

- Etape M :

$$\beta^{(r+1)} = \frac{1}{n}\sum_{i=1}^{n} t_{i1}^{(r)} \qquad \mu_k^{(r+1)} = \frac{\sum_{i=1}^{n} t_{ik}^{(r)} x_i}{\sum_{i=1}^{n} t_{ik}^{(r)}}$$

$$\sigma_k^{2\,(r+1)} = \frac{\sum_{i=1}^{n} t_{ik}^{(r)} \left(x_i - \mu_k^{(r+1)}\right)^2}{\sum_{i=1}^{n} t_{ik}^{(r)}}$$

Algorithme SEM [4]

- SEM = Stochastique EM
- Ajout d'une étape S de classification aléatoire
  - Étape E: Calcul des $t_{ik}^{(r)}$
  - Étape S: Tirage au hasard d'une classe pour chaque individu : on simule les $\hat{z}_{ik}^{(r)}$ selon $\mathcal{M}\left(1; t_{i1}^{(r)}, \ldots, t_{iK}^{(r)}\right)$
  - Étape M: On actualise les paramètres en remplaçant $t_{ik}^{(r)}$ par $\hat{z}_{ik}^{(r)}$.

Autres variantes de l'algorithme EM

- SAEM (Stochastic Approximation EM) [6]

Compromis entre EM et SEM. L'importance des tirages aléatoires diminue avec le temps.

$$\theta^{(r+1)} = \gamma_r \; \theta_{\text{SEM}}^{(r+1)} + (1 - \gamma_r) \; \theta_{\text{EM}}^{(r+1)}$$

où $\gamma_0 = 1$ et $(\gamma_r)_{r \geq 0}$ est décroissante.

- MCEM (Monte Carlo EM) [9]
  - Etape Monte Carlo E: Simulation de M répétitions de $\underline{\mathbf{z}}$, notées $\underline{\mathbf{z}}'_1, \ldots, \underline{\mathbf{z}}'_M$ selon la distribution conditionnelle de $\underline{\mathbf{z}}|\underline{\mathbf{x}}, \theta^{(r)}$
  - Etape M: Maximisation en θ de

$$\theta \mapsto \frac{1}{M} \sum_{m=1}^{M} \ln[f(\underline{\mathbf{x}}, \underline{\mathbf{z}}'_m|\theta)]$$

qui approche $\mathcal{Q}(\theta_K|\theta_K^{(r)})$.

Approche par MV classifiante

- Approche CMV : $\underline{\mathbf{z}}$ est vu comme un paramètre à estimer
- Estimation simultanée de θ et $\underline{\mathbf{z}}$ en maximisant la logvraisemblance classifiante :

$$\mathcal{L}(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\theta_K) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik} \left\{\ln(\pi_k) + \ln(f_k(x_i|\alpha_k))\right\}$$

- Cette approche revient à chercher une partition telle que chaque classe $\mathcal{C}_k$ soit assimilable à un sous-échantillon issu de la loi $f_k(.|\alpha_k)$.

Algorithme CEM [5]

- CEM = Classification EM
- Etapes de l'algorithme :
  - Étape E: Calcul des $t_{ik}^{(r)}$
  - Étape C: détermination d'une partition des données $\underline{\mathbf{x}}$ par la règle du MAP

$$\hat{z}_{ik}^{(r)} = \begin{cases} 1 & \text{si } t_{ik}^{(r)} > t_{i\ell}^{(r)}, \forall \ell \neq k \\ 0 & \text{sinon} \end{cases}$$

  - Étape M: On actualise les paramètres en remplaçant $t_{ik}^{(r)}$ par $\hat{z}_{ik}^{(r)}$.

Caractéristiques de l'algorithme CEM

- CEM vise à maximiser la vraisemblance complétée pas la vraisemblance.
- CEM converge en un nombre fini d'itérations contrairement à EM
- CEM produit des estimateurs biaisés des paramètres du mélange
- CEM est un algorithme de type Kmeans (voir exercice)

Initialisation

- Les algorithmes de type EM sont sensibles à l'initialisation
- Stratégie en Search/Run/Select:
  - Choix de M positions initiales (RANDOM, CEM, SEM, quelques runs de EM)
  - Quelques itérations de l'algorithme pour chacune des positions
  - Sélection de la position donnant la plus grande vraisemblance (ou vraisemblance classifiante).

Plan

Critères asymptotiques de sélection

$$\hat{K} = \underset{K}{\text{argmin}}\ \text{crit}(K) = \underset{K}{\text{argmin}}\ \ -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \text{pen}(K)$$

- **AIC** (Akaike Information Criterion) [1]
$$\text{crit}(K) = -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \nu_K$$

- **BIC** (Bayesian Information Criterion) [8]
$$\text{crit}(K) = -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n)$$

- **ICL** (Integrated Completed Likelihood) [2]
$$\text{crit}(K) = -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n) + \text{Ent}(K)$$

où ν_K est nombre de paramètres libres des mélanges de $\mathcal{S}_K$

$\text{Ent}(K) = -\sum_{i=1}^{n}\sum_{k=1}^{K} \hat{z}_{ik}\ln[t_{ik}(\hat{\theta}_K)]$ est un terme d'entropie

Critère AIC [1]

- Critère AIC :

$$\text{crit}(K) = -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \nu_K$$

- Critère pour réaliser un compromis en terme de “biais-variance”
- Asymptotiquement, AIC retient le modèle minimisant l’écart de Kullback moyen avec la vraie loi inconnue
- Dans le cadre des modèles de mélanges finis, AIC a tendance à sous-pénaliser

Critère BIC [8]

- Critère fondé sur la maximisation de la vraisemblance intégrée :

$$\hat{K} = \underset{K}{\text{argmax}}\ f(\underline{\mathbf{x}}|K)$$

où $f(\underline{\mathbf{x}}|K) = \int_{\Theta_K} f(\underline{\mathbf{x}}|\theta, K)\Pi(\theta|K)d\theta$ est la vraisemblance intégrée et $\Pi(\theta|K)$ est un prior non informatif.

- Approximation asymptotique : $f(\underline{\mathbf{x}}|K) \approx \ln\left[f(\underline{\mathbf{x}}|\hat{\theta}_K)\right] - \frac{\nu_K}{2}\ln(n)$

- Bayesian Information Criterion (BIC) :

$$\hat{K} = \underset{K}{\text{argmin}}\ \left\{-\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n)\right\}$$

- Asymptotiquement, BIC sélectionne le modèle minimisant l'écart de Kullback avec la vraie loi. En ce sens, BIC est convergeant si le vrai modèle est dans la liste des modèles

Critère ICL [2]

- $\hat{K} = \underset{K}{\text{argmax}}\ f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|K)$ où $f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|K) = \int_{\Theta_K} f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\theta, K)\Pi(\theta|K)d\theta$ est la vraisemblance complète intégrée et $\Pi(\theta|K)$ est un prior non informatif.

- Approximation asymptotique de type BIC:

$$f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|K) \approx \ln\left[f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\hat{\theta}_K^\star)\right] - \frac{\nu_K}{2}\ln(n)$$

où $\hat{\theta}_K^\star = \underset{\theta_K \in \Theta_K}{\text{argmax}}\ f(\underline{\mathbf{x}}, \underline{\mathbf{z}}|\theta_K)$.

- Problème : $\underline{\mathbf{z}}$ est inconnu et donc $\hat{\theta}_K^\star$ est inaccessible $\rightsquigarrow \hat{\underline{\mathbf{z}}} = \text{MAP}(\hat{\theta}_K)$

- Integrated Completed Likelihood :

$$\hat{K} = \underset{K}{\text{argmin}}\ \left\{-\mathcal{L}(\underline{\mathbf{x}}, \hat{\underline{\mathbf{z}}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n)\right\}$$

Différence entre ICL et BIC

- $\text{BIC}(K) = -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n)$

- Relation entre ICL et BIC

$$\begin{aligned}
\text{ICL}(K) &= -\mathcal{L}(\underline{\mathbf{x}}, \underline{\hat{\mathbf{z}}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n) \\
&= -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + \frac{\nu_K}{2}\ln(n) + \text{Ent}(K) \\
&= \text{BIC}(K) + \text{Ent}(K)
\end{aligned}$$

où l'entropie est définie par

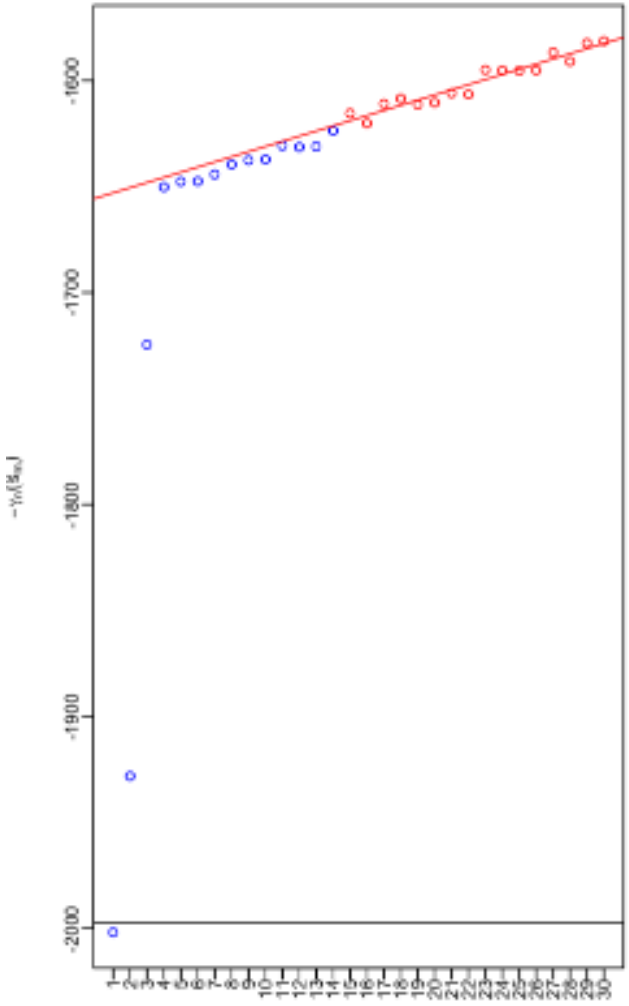
$$\text{Ent}(K) = -\sum_{i=1}^{n}\sum_{k=1}^{K} \hat{z}_{ik} \ln[t_{ik}(\hat{\theta}_K)]$$

Alternative avec l'heuristique de pente

- Critère non-asymptotique

$$\hat{K} = \underset{K}{\mathit{argmin}}\ \mathrm{crit}(K) = \underset{K}{\mathrm{argmin}}\ -\mathcal{L}(\underline{\mathbf{x}}|\hat{\theta}_K) + 2\kappa D_K$$

- Nécessite de déterminer la quantité D_K qui représente la dimension du modèle $\mathcal{S}_K$
- Calibration de la pente inconnue κ



Plan

Mélanges gaussiens multivariés

- Données quantitatives : $x_i \in \mathbb{R}^p$
- Les observations sont supposées issues d'un échantillon de loi

$$f(.|\theta_{K,m}) = \sum_{k=1}^{K} \pi_{k,m}\ \phi(.|\mu_k, \Sigma_{k,m})$$

  - $\phi(.|\mu_k, \Sigma_k)$ la densité d'une loi gaussienne multivariée $\mathcal{N}_p(\mu_k, \Sigma_k)$
  - $\theta_{K,m} = (\pi_{1,m}, \ldots, \pi_{K,m}, \mu_1, \ldots, \mu_K, \Sigma_{1,m}, \ldots, \Sigma_{K,m})$
- Collection de modèles: $\left(\mathcal{S}_{(K,m)}\right)_{(K,m)\in\mathbb{N}^\star\times\mathcal{M}}$ avec
  - $\mathcal{S}_{(K,m)} = \{x \in \mathbb{R}^p \mapsto f(x|\theta_{K,m}); \theta_{K,m} \in \Theta_{K,m}\}$
  - $\mathcal{M}$ = ensemble de formes de mélanges gaussiens

Formes m des mélanges gaussiens

- La décomposition en valeurs propres des matrices variance Σ_k :

$$\Sigma_k = L_k D_k A_k D_k'$$

  - Σ_k est une matrice définie positive de taille $p \times p$
  - $L_k = |\Sigma_k|^{1/p}$ (le volume)
  - D_k la matrice des vecteurs propres de Σ_k (l'orientation)
  - A_k la matrice diagonale des v.p. normalisées de Σ_k (la forme)

- $\Rightarrow$ 3 familles (sphérique, diagonale, générale) $\Rightarrow$ 14 formes

- Proportions supposées égales ou libres

$\Rightarrow$ 28 formes de mélanges gaussiens possibles

Exemple dans $\mathbb{R}^2$

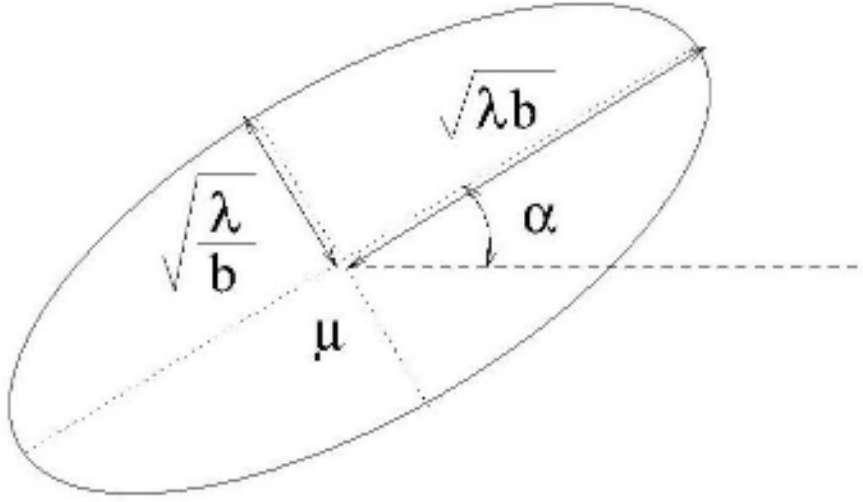
- D est une matrice de rotation définie par un angle α

$$D = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

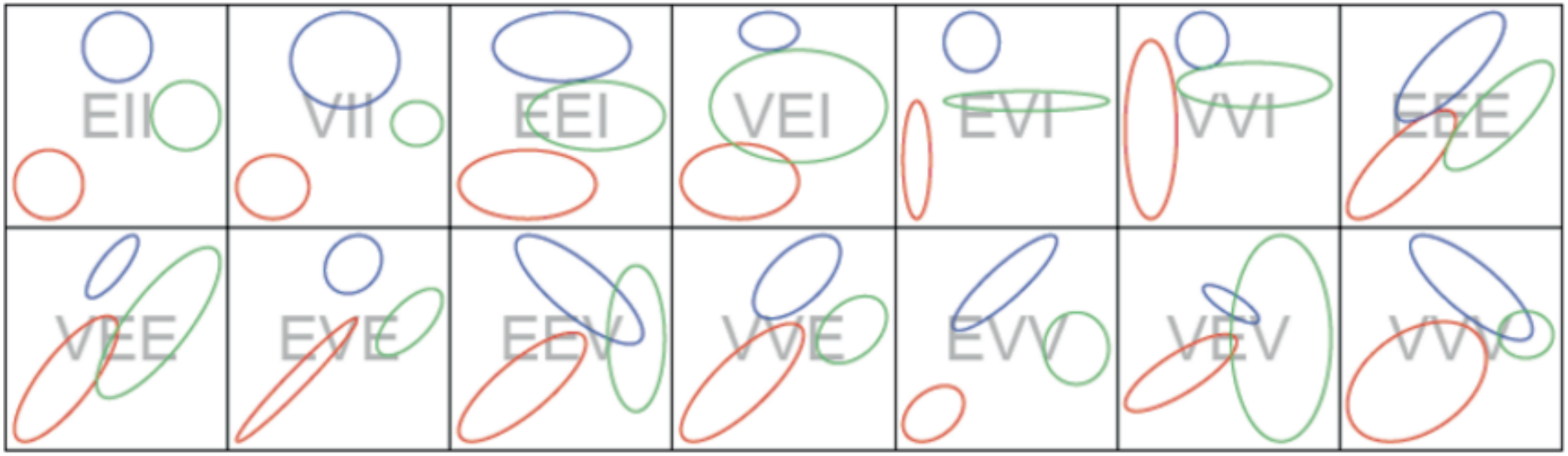
- A est une matrice diagonale de termes diagonaux b et $\frac{1}{b}$

$$A = \begin{pmatrix} b & 0 \\ 0 & \frac{1}{b} \end{pmatrix}$$

- Ellipse d'équidensité ($L = \lambda$ =volume)



Formes m des mélanges gaussiens



Bouveyron et al. [3]

Lien entre GMM et Kmeans

La méthode des Kmeans est un cas particulier des GMM :

- Collection de modèles : mélanges gaussiens sphériques

+

- Estimer les paramètres avec l'algorithme CEM

On peut alors utiliser les critères de sélection de modèles (Etape 3) pour choisir le nombre de classes.

Plan

Quelques librairies avec 

- Librairie `mclust` [Scrucca et al.]

mixtures of multivariate Gaussian

- Librairie `Rmixmod` [Biernacki et al.]

mixtures of multivariate Gaussian or multinomial components

- Librairie `mixture` [McNicholas P.D. et al.]

Gaussian, Student's t, generalized hyperbolic, variance-gamma or skew-t mixtures

- Librairie `movMF` [Horbik and Grün]

mixtures of von Mises-Fisher distributions

- et bien d'autres !

Et avec 

- `sklearn.mixture.GaussianMixture(.)`
- Package `PyMix` [Georgi et al., http://www.pymix.org/]
- Package `Pymixmod` (la version python de mixmod)
- `Multinomial_Mixture_Model` [https://github.com/diningphil/Multinomial_Mixture_Model]
- `studenttmixture` [https://github.com/jlparki/mix_T]
- ...

Exemple des iris avec 

- Utilisation du package `mclust`
- Sélection de modèles avec le critère BIC

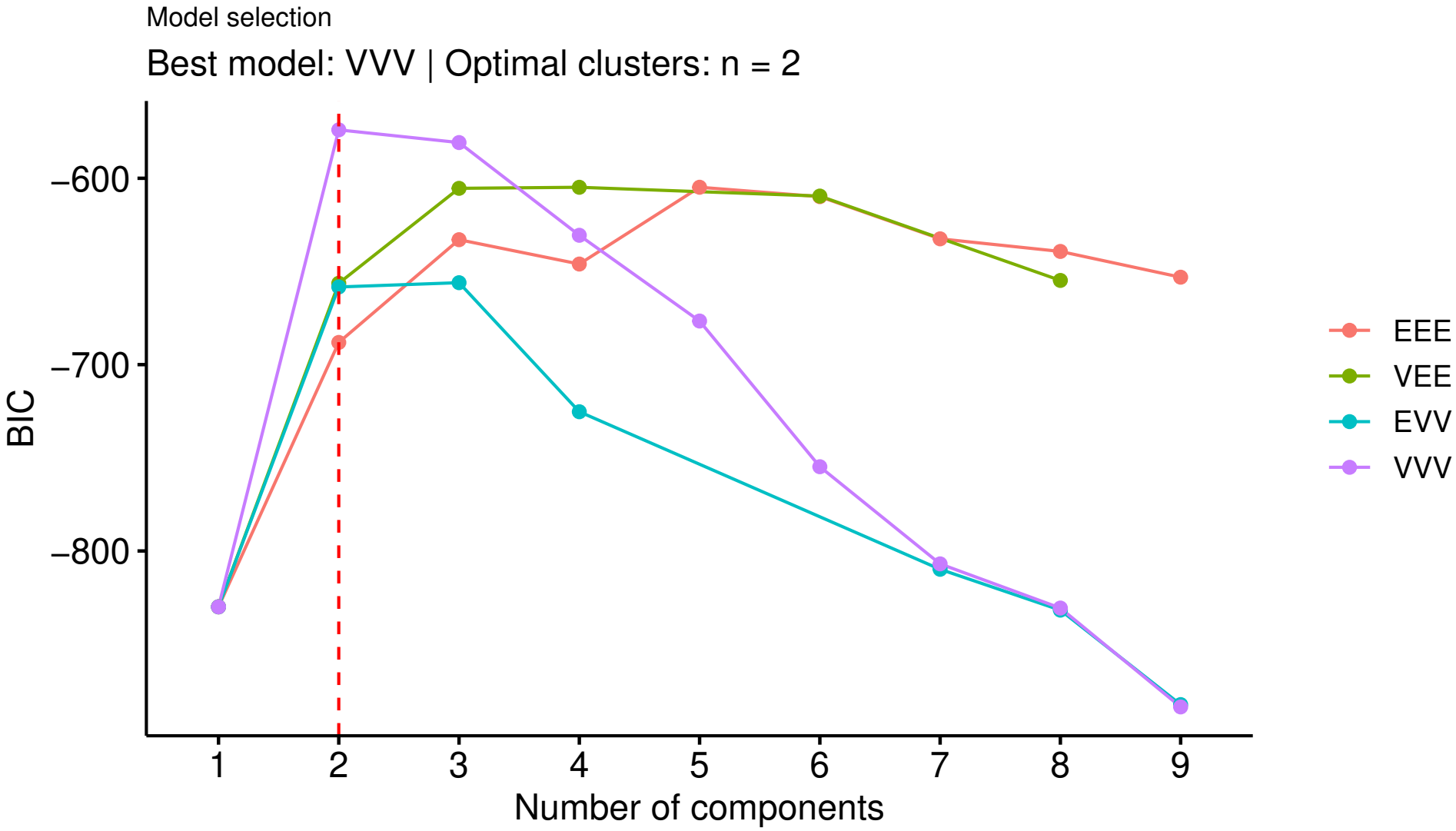
```
data(iris)
library(mclust)
resmclust<-Mclust(iris[,-5],G=1:9,modelNames = c("EEE","VEE","EVV","VVV"))
summary(resmclust)
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 2
components:

 log-likelihood   n df       BIC       ICL
      -214.3547 150 29 -574.0178 -574.0191

Clustering table:
  1   2
 50 100
```

```
# pour accéder aux paramètres
# resmclust$parameters$pro; resmclust$parameters$mean; resmclust$parameters$variance$sigma
```

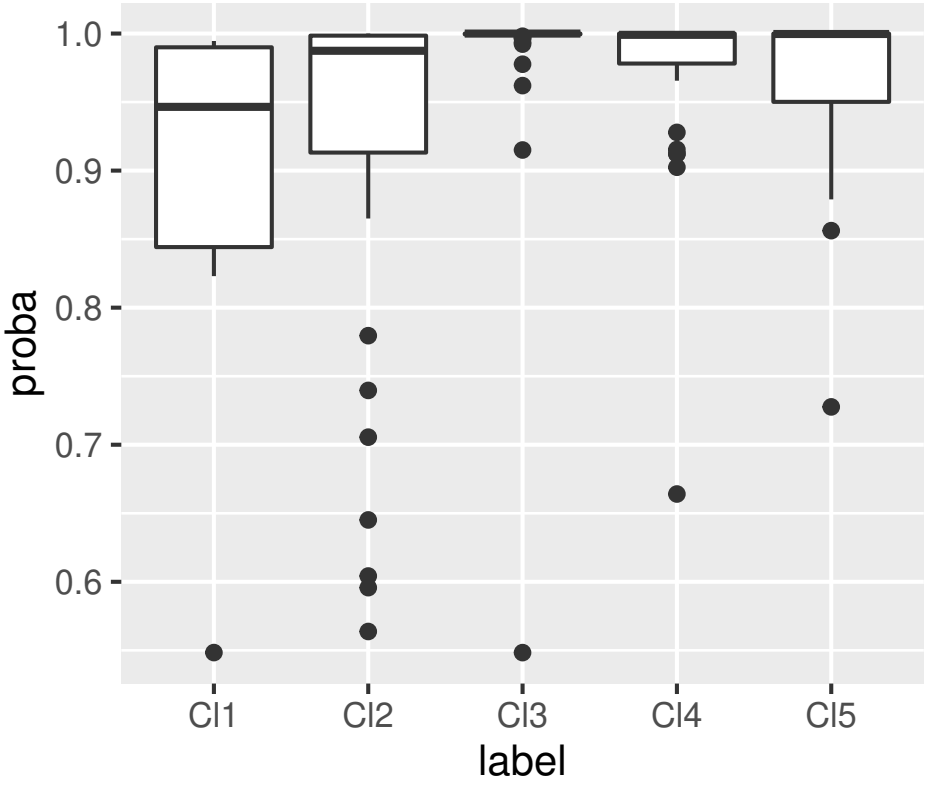
Exemple des iris avec 

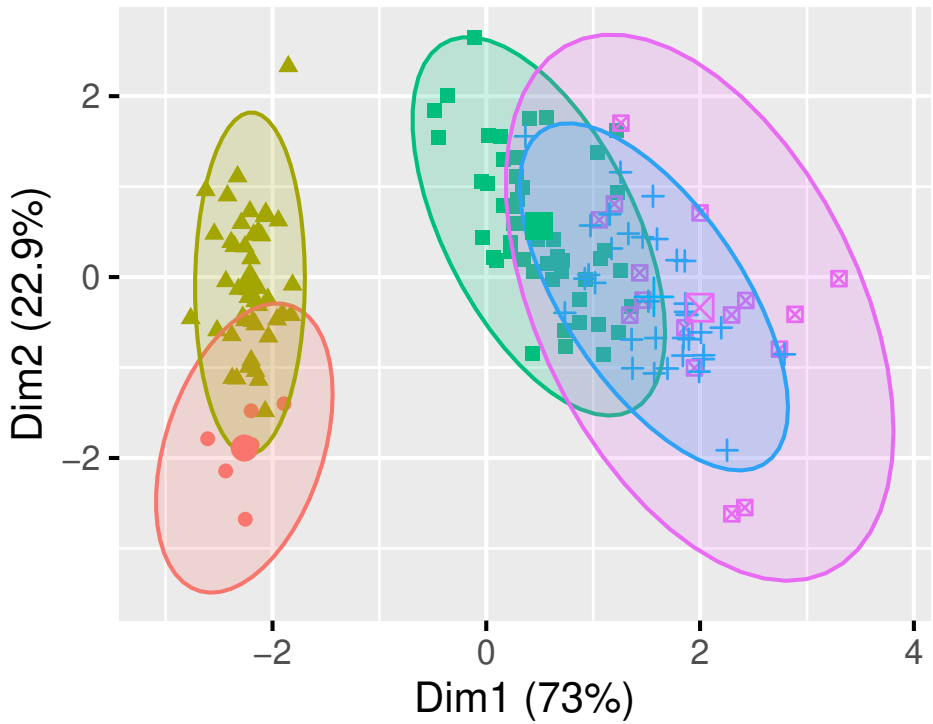
```
fviz_mclust_bic(resmclust)+theme(plot.title = element_text(size =9))+theme(legend.position = "right")
```



Exemple des iris avec 

- Classification à $K = 5$ classes
- Boxplot des probabilités conditionnelles d'appartenance





Exemple des iris avec 

- Utilisation du package `mclust`
- Sélection de modèles avec le critère ICL

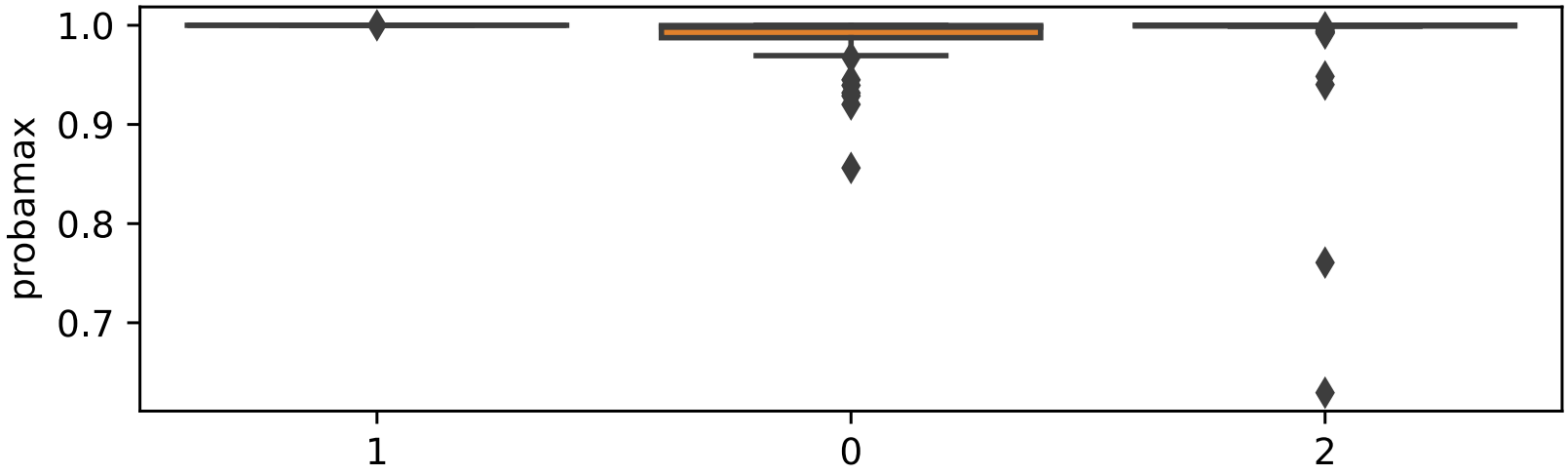
```
# On peut faire la même chose avec mclustICL pour le critere
modICL=mclustICL(iris[,-5],G=1:9,modelNames = c("EEE","VEE","
summary(modICL)
```

```
Best ICL values:
             VVV,2      VVV,3      VEE,3
ICL      -574.0191 -584.05221 -612.28970
ICL diff    0.0000  -10.03311  -38.27061
```

Exemple des iris avec 

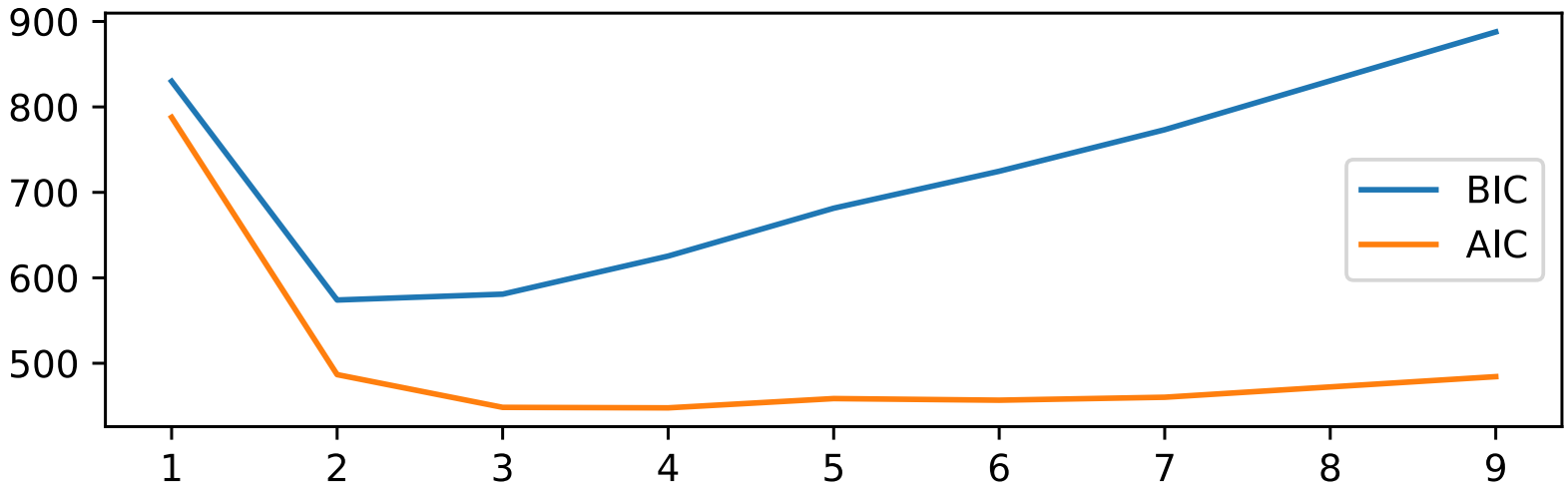
Mélange à $K = 3$ composantes

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.mixture import GaussianMixture as GMM
pyiris=r.iris
X=pyiris.iloc[:, [0, 1, 2, 3]].values

gmm=GMM(n_components=3,covariance_type="full").fit(X)

labels=gmm.predict(X)
probas=gmm.predict_proba(X)

df=pd.DataFrame({'label':list(map(str,labels)), 'probamax':np.amax(probas, axis=1)})
sns.boxplot(data=df,x='label',y='probamax')
```



Exemple des iris avec 

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture as GMM
X=r.iris.iloc[:, [0, 1, 2, 3]].values
n_components = np.arange(1, 10)
models = [GMM(n, covariance_type='full', random_state=0).fit(X) for n in n_components]
plt.plot(n_components, [m.bic(X) for m in models], label='BIC')
plt.plot(n_components, [m.aic(X) for m in models], label='AIC')
plt.legend(loc='best')
plt.xlabel('n_components');
```



References I

[1] H. Akaike. "Information theory and an extension of the maximum likelihood principle". In: *Second International Symposium on Information Theory (Tsahkadsor, 1971)*. Budapest: Akadémiai Kiadó, 1973, pp. 267–281.

[2] Christophe Biernacki, Gilles Celeux, and Gérard Govaert. "Assessing a Mixture Model for Clustering with the Integrated Completed Likelihood". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.7 (2000), pp. 719–725.

[3] Charles Bouveyron et al. *Model-based clustering and classification for data science: with applications in R*. Vol. 50. Cambridge University Press, 2019.

[4] G. Celeux and J. Diebolt. "The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem". In: *Computational Statistics Quarterly* 2 (1985), pp. 73–82.

References II

[5] G. Celeux and G. Govaert. "A classification EM algorithm for clustering and two stochastic versions". In: *Computational Statistics and Data Analysis* 14.3 (1992), pp. 315–332.

[6] B. Delyon, M. Lavielle, and E. Moulines. "Convergence of a stochastic approximation version of the EM algorithm". In: *Annals of Statistics* 27 (1999), pp. 94–128.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm (with discussion)". In: *Journal of the Royal Statistical Society. Series B.* 39.1 (1977), pp. 1–38.

[8] Gideon Schwarz. "Estimating the Dimension of a Model". In: *The Annals of Statistics* 6.2 (1978), pp. 461–464.

[9] G. C. G. Wei and M. A. Tanner. "A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms". In: *Journal of the American Statistical Association* 85 (1990), pp. 699–704.