Apprentissage non supervisé : Partitionnement de données (*Clustering*)

Joseph Salmon

Université de Montpellier

Plan

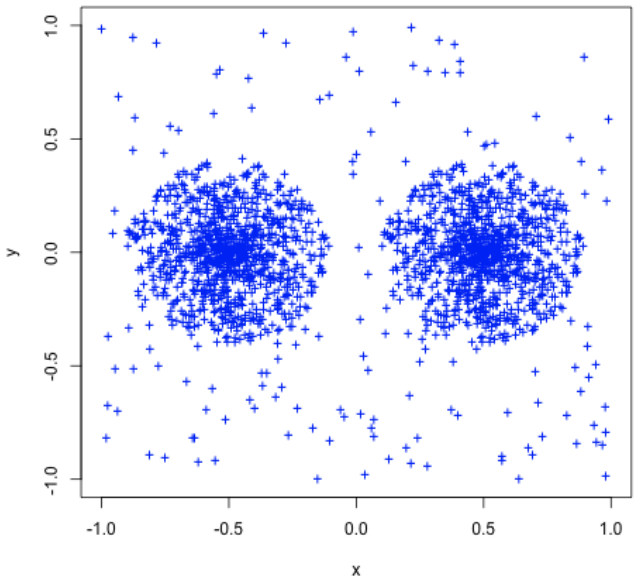
Introduction

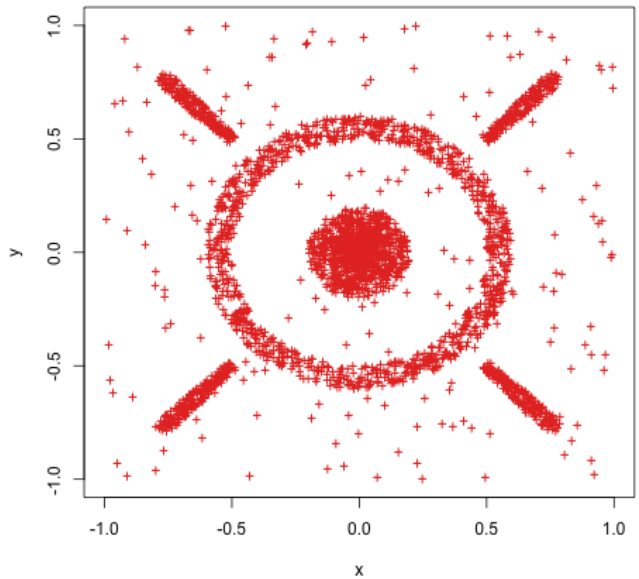
Objectifs :

- Structurer les données
- Regrouper les observations “proches” en “classes”

Vocabulaire :

- partitionner les données $=$ *Clustering*
- une méthode *non-supervisé* (sans étiquettes, *i.e.*, sans y)





Exemples d'applications

Gestion - Marketing :

- données : infos client, produits, ...
- but : segmenter la clientèle, définir des profils

Exemples d'applications

Gestion - Marketing :

- données : infos client, produits, ...
- but : segmenter la clientèle, définir des profils

Traitement Naturel du Langage NLP :

- données : texte, email, ...
- but : grouper automatiquement les textes proches

Exemples d'applications

Gestion - Marketing :

- données : infos client, produits, ...
- but : segmenter la clientèle, définir des profils

Traitement Naturel du Langage NLP :

- données : texte, email, ...
- but : grouper automatiquement les textes proches

Sociologie :

- données : attributs d'un individu, *e.g.*, revenus, sexe, ...
- but : former des catégories de population

Exemples d'applications

Gestion - Marketing :

- données : infos client, produits, ...
- but : segmenter la clientèle, définir des profils

Traitement Naturel du Langage NLP :

- données : texte, email, ...
- but : grouper automatiquement les textes proches

Sociologie :

- données : attributs d'un individu, *e.g.*, revenus, sexe, ...
- but : former des catégories de population

Analyse génomique :

- données : gênes
- but : former des groupes homogènes de gênes

Intérêt divers

- visualisation
- accélération des calculs (appliquer des algorithmes sur des sous parties plus petites)
- stabilisation des résultats (appliquer des algorithmes sur des sous parties plus homogènes)

Exemple analyse numérique : permet de **pré-conditionner** les données pour utiliser des outils d'algèbre linéaire

Cadre mathématique

- Données : $X = [x_1, \ldots, x_n]^\top \in \mathbb{R}^{n \times d}$ (matrice $n \times d$)
  x_i : i-ème observation (i-ème ligne de X)
  n : nombre d'observations / individus / exemples
  d : dimension des variables explicatives (features), covariables

Tailles usuelles pour n :

- Santé / Bio : $10 \leq n \leq 1000$ (patients) , $1000 \leq d \leq 10^5$
- Entreprises PME / Nationales : $1000 \leq n \leq 10^7$ (clients)
- Entreprises GAFA etc. : $10^6 \leq n \leq 10^{10}$ (clients)
- Visions par ordinateurs $10^0 \leq n \leq 10^{10}$ (images) ; $d =$ nombres de pixels

Rem: ici pas d'"étiquettes" collectées (coûteuses) !
Anecdote : Imagenet (2009), base de données ayant "relancé" les réseaux de neurones, recours à la "production participative" (*crowdsourcing*)

Notion de proximité

Enjeu :

- Mesurer la proximité de deux observations x_1, x_2

Ingrédients :

- fonction de similarité : plus la mesure est faible, plus les objets sont similaires ($\approx$ à une distance / divergence)
- fonction de dissimilarité : plus la mesure est grande, plus les objets sont similaires

Distances usuelles

$x_1 \in \mathbb{R}^d, x_2 \in \mathbb{R}^d$

- Distance euclidienne :

$$d^2(x_1, x_2) = \sum_{i=1}^{d} (x_1^i - x_2^i)^2$$

- Distance de Manhattan :

$$d(x_1, x_2) = \sum_{i=1}^{d} |x_1^i - x_2^i|$$

- Distance de Minkowski :

$$d(x_1, x_2) = \left(\sum_{i=1}^{d} |x_1^i - x_2^i|^p \right)^{\frac{1}{p}}$$

- Distance de Mahalanobis (W matrice symétrique, $\succ 0$)

$$d^2(x_1, x_2) = (x_1 - x_2)^\top W (x_1 - x_2) = \sum_{i,j} W_{i,j} (x_1^i - x_2^i)(x_1^j - x_2^j)$$

Cas des variables discrètes

Distance de Hamming : nombre de coefs où les vecteurs diffèrent

$$d(x_1, x_2) = \sum_{i=1}^{d} \mathbb{1}_{\{x_1^i \neq x_2^i\}}$$

Exemple :

$$x_1 = (0, 1, 2, 1, 2, 1)^\top, x_2 = (1, 0, 2, 1, 0, 1)^\top : d(x_1, x_2) = 3$$

Bonus : pour données non “numériques”, mesure indépendante de l’encodage

Par exemple : si menthe=0, vanille=1, chocolat=2 ; la distance euclidienne rendrait menthe plus proche de vanille que de chocolat...

Panorama de clustering

http://scikit-learn.org/stable/modules/clustering.html

Plan

Qualité d'une partition

- $\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K$: partition de $[\![1, n]\!]$, en K classes (K fixé ici)
- $|\mathcal{C}_1| = N_1, \ldots, |\mathcal{C}_K| = N_K$
- "centres" $\mu_1, \ldots, \mu_K$

Inertie intra-cluster

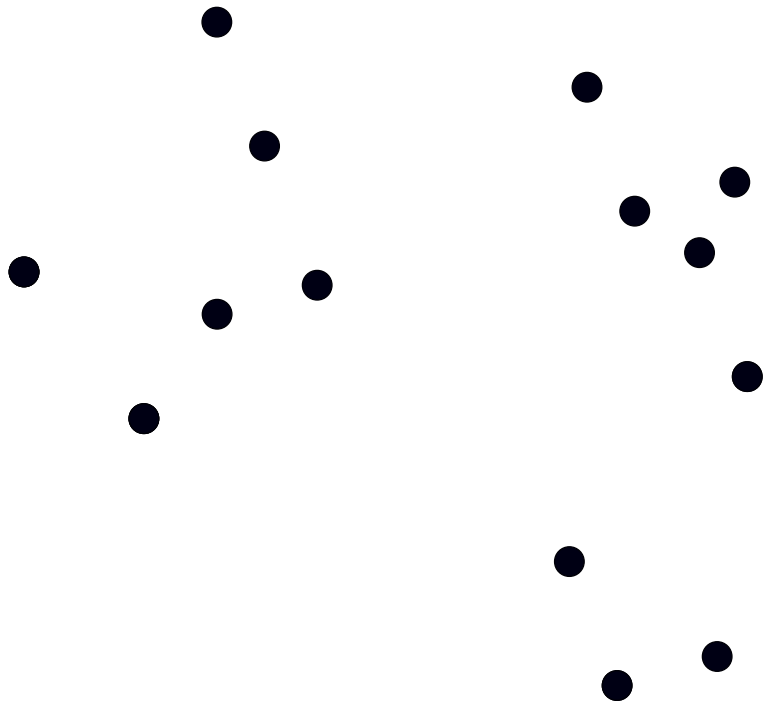
$$I = I(\mu, \mathcal{C}) = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} d^2 (x_i, \mu_k)$$

Rem: vocabulaire similaire à la mécanique

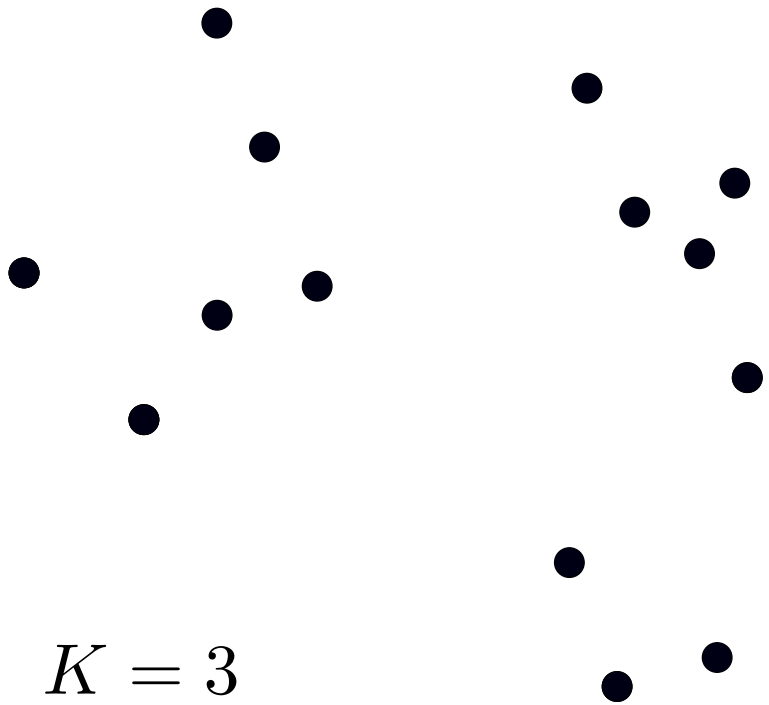
Stratégie :

(i) Minimiser l'inertie intra-cluster

Visualisation

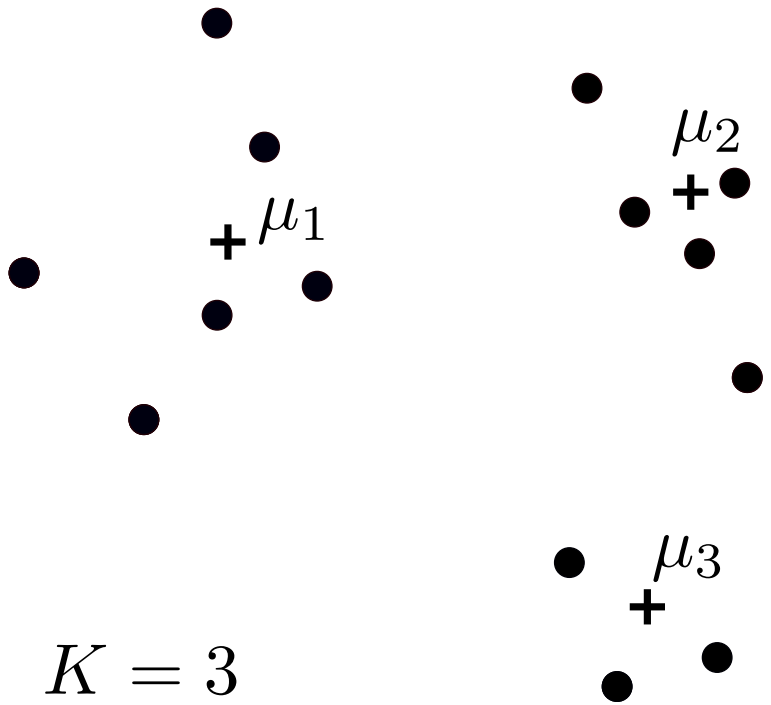


Visualisation

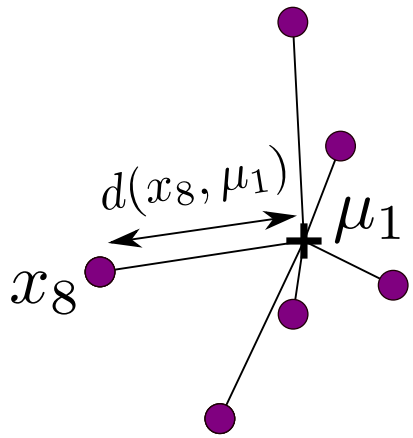


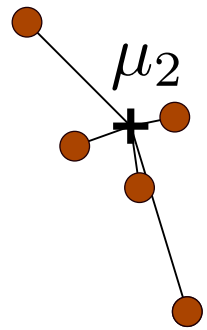
$K = 3$

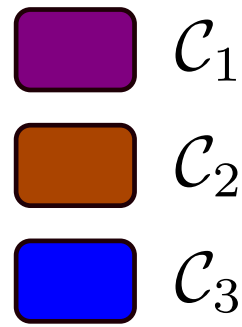
Visualisation

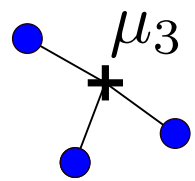


Visualisation









$K = 3$

La méthode du k-means

Contexte : On dispose d'un échantillon de taille n : $x_1, \ldots, x_n$ à valeurs dans $\mathbb{R}^d$

Principe :

- Classe $\mathcal{C}_k$: représentée par un **centroïde** $\mu_k \in \mathbb{R}^d$
- Formation des classes : affecter au centroïde le plus proche

Heuristique :
Déterminer K centroïdes $\mu_1, \ldots, \mu_K$ minimisant :

$$\mathcal{E}_n(\mu_1, \ldots, \mu_K) = \sum_{i=1}^{n} \min_{1 \leq k \leq K} d(x_i, \mu_k)^2$$

Idéalement il faut résoudre le problème (non-convexe / NP-dur) :

$$\min_{\mu_1, \ldots, \mu_K \in (\mathbb{R}^d)^K} \mathcal{E}_n(\mu_1, \ldots, \mu_K)$$

Algorithme de Lloyd[1], [2] : optimisation alternée

Objectif : minimiser le critère de distorsion $\mathcal{E}_n(\mu_1, \ldots, \mu_K)$

Initialisation des K centres $\mu_1, \ldots, \mu_K$

Affectation de chaque observation au centre le plus proche

Mise à jour des centres, en calculant la moyenne empirique des observations dans chaque classe

Itérer jusqu'à convergence

Rem: algorithme non publié par Lloyd, alors aux Bell Labs en 1957

---

(1). S. Lloyd. "Least squares quantization in PCM". In : *IEEE Trans. Inf. Theory* 28.2 (1982), p. 129-137.

(2). H. Steinhaus. "Sur la division des corps matériels en parties". In : *Bull. Acad. Polon. Sci. Cl. III.* 4 (1956), 801-804 (1957).

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

---

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

tant que *pas convergé* **faire**

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

tant que *pas convergé* **faire**

$t \leftarrow t + 1, \mathcal{C}_1 \leftarrow \emptyset, \ldots, \mathcal{C}_K \leftarrow \emptyset$

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

tant que *pas convergé* **faire**

$t \leftarrow t + 1, \mathcal{C}_1 \leftarrow \emptyset, \ldots, \mathcal{C}_K \leftarrow \emptyset$

pour $i = 1, \ldots, n$ **faire** // Affectation

$h = h^*(x_i) = \underset{k \in [\![1,K]\!]}{\arg\min}\, d^2(x_i, \mu_k)$

$\mathcal{C}_h \leftarrow \mathcal{C}_h \cup \{i\}$

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

tant que *pas convergé* **faire**

$t \leftarrow t + 1, \mathcal{C}_1 \leftarrow \emptyset, \ldots, \mathcal{C}_K \leftarrow \emptyset$

pour $i = 1, \ldots, n$ **faire** // Affectation

$h = h^*(x_i) = \underset{k \in [\![1,K]\!]}{\arg\min}\, d^2(x_i, \mu_k)$

$\mathcal{C}_h \leftarrow \mathcal{C}_h \cup \{i\}$

pour $k = 1, \ldots, K$ **faire** // Estimation

$\mu_k \leftarrow \frac{1}{|\mathcal{C}_k|} \sum_{j \in \mathcal{C}_k} x_j$

k-means

Objectif : trouver une partition optimale

Algorithme : k-means

Entrées : X, K

Initialisation : $t = 0$ et $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$

tant que *pas convergé* **faire**

$t \leftarrow t + 1, \mathcal{C}_1 \leftarrow \emptyset, \ldots, \mathcal{C}_K \leftarrow \emptyset$

pour $i = 1, \ldots, n$ **faire** // Affectation

$h = h^*(x_i) = \underset{k \in [\![1,K]\!]}{\arg\min}\, d^2(x_i, \mu_k)$

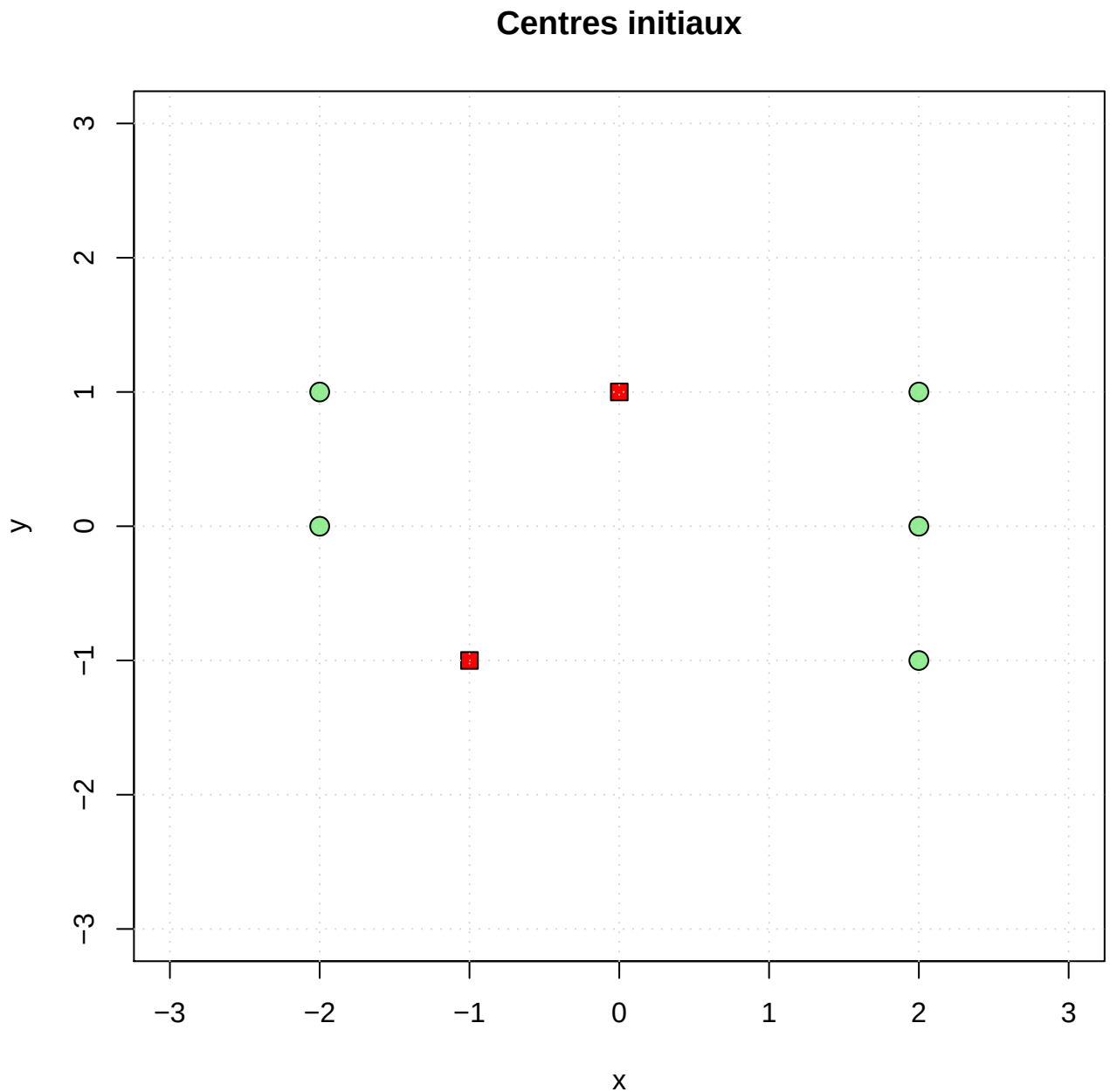
$\mathcal{C}_h \leftarrow \mathcal{C}_h \cup \{i\}$

pour $k = 1, \ldots, K$ **faire** // Estimation

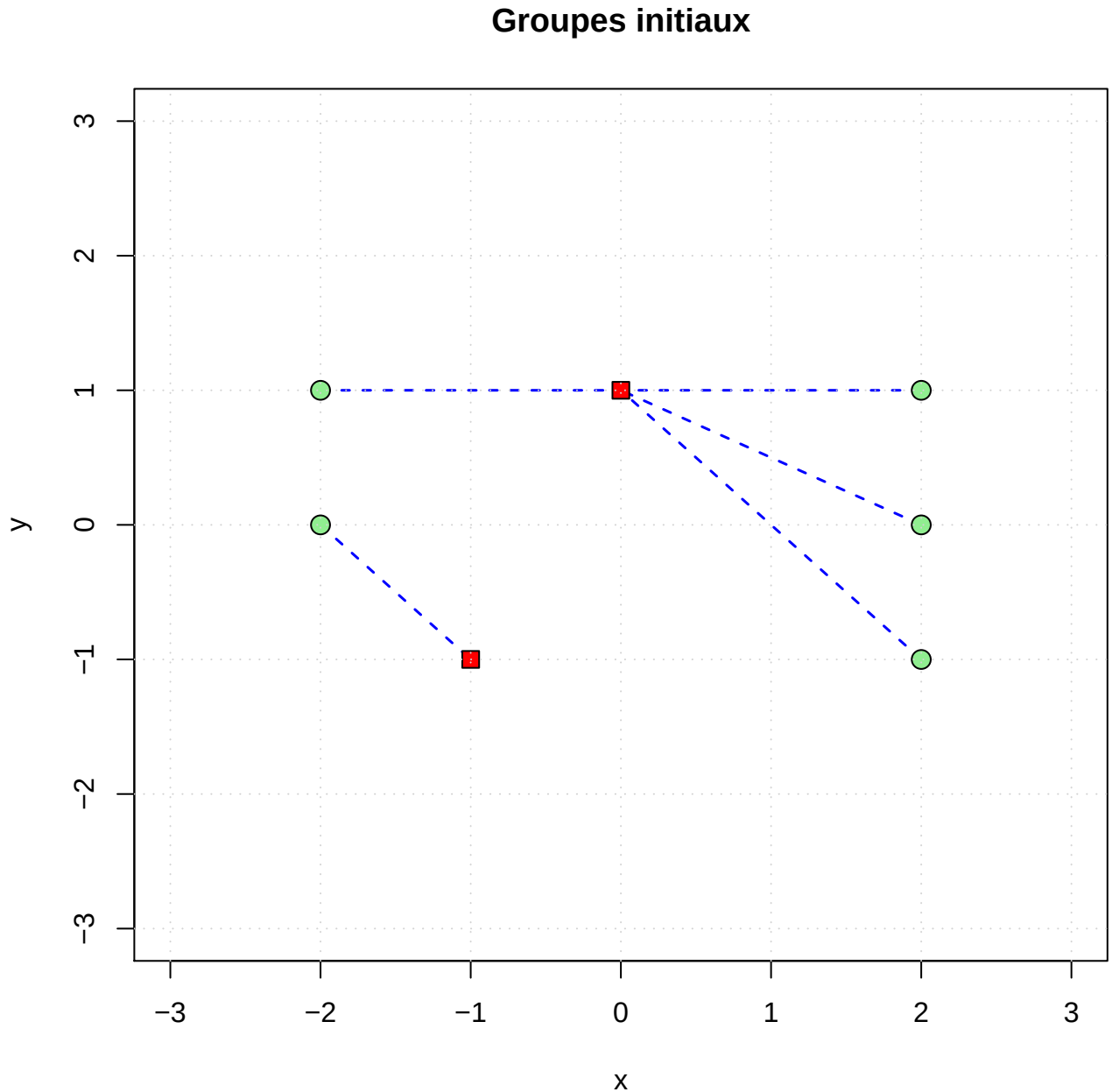
$\mu_k \leftarrow \frac{1}{|\mathcal{C}_k|} \sum_{j \in \mathcal{C}_k} x_j$

Sorties : $\mu_1, \ldots, \mu_K$ et $\mathcal{C}_1, \ldots, \mathcal{C}_K$

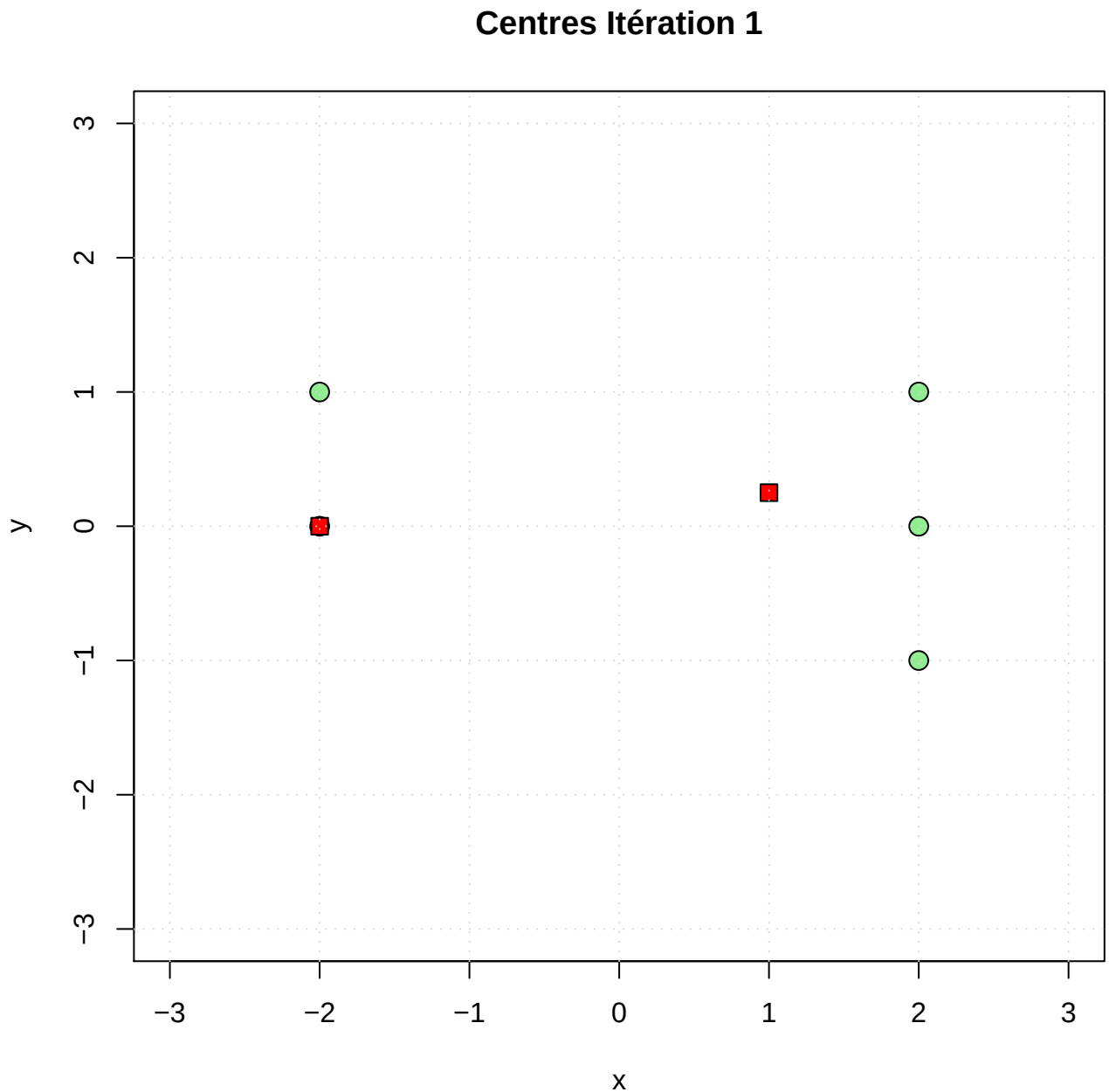
Exemple (1/3)



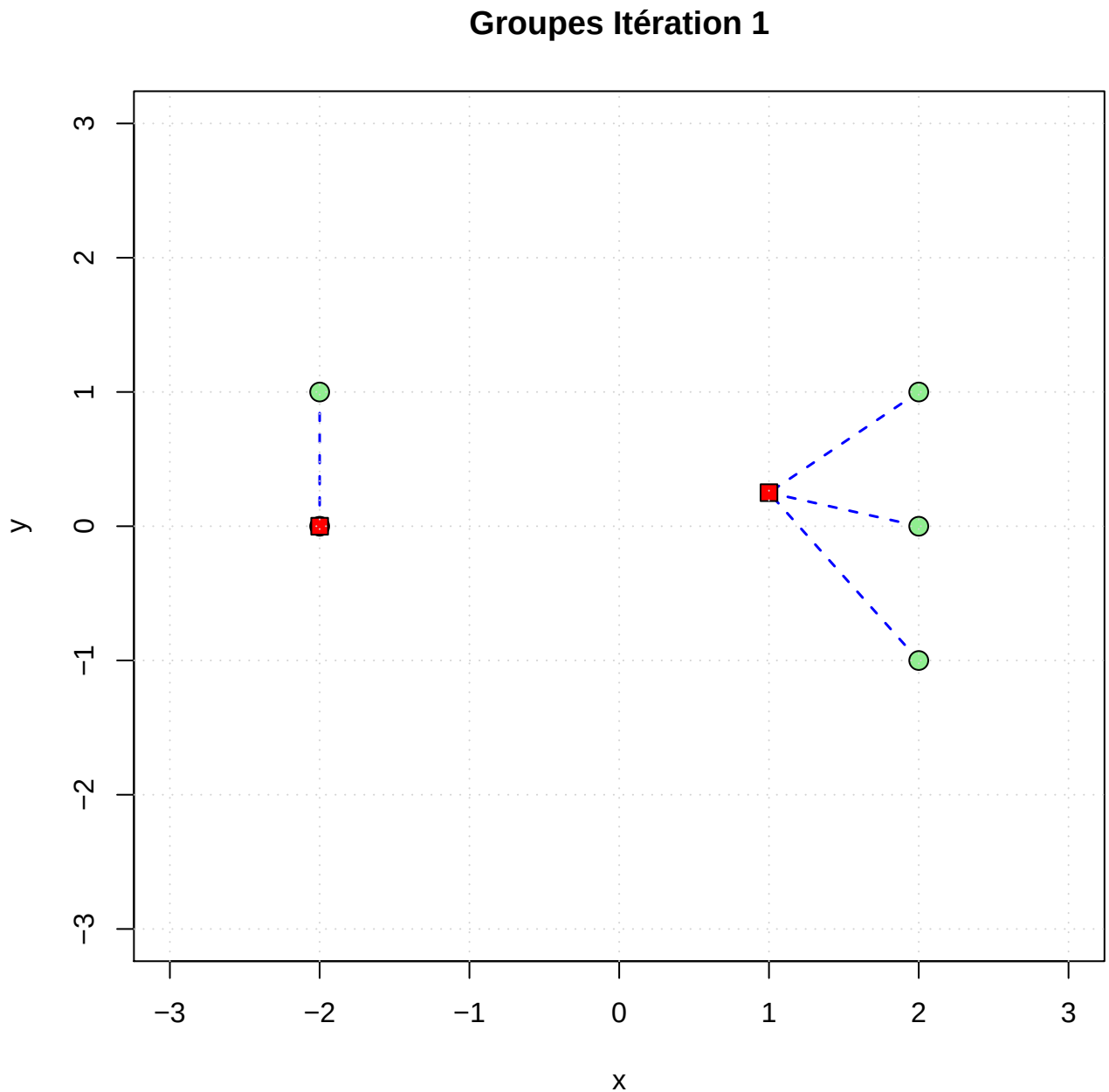
Exemple (1/3)



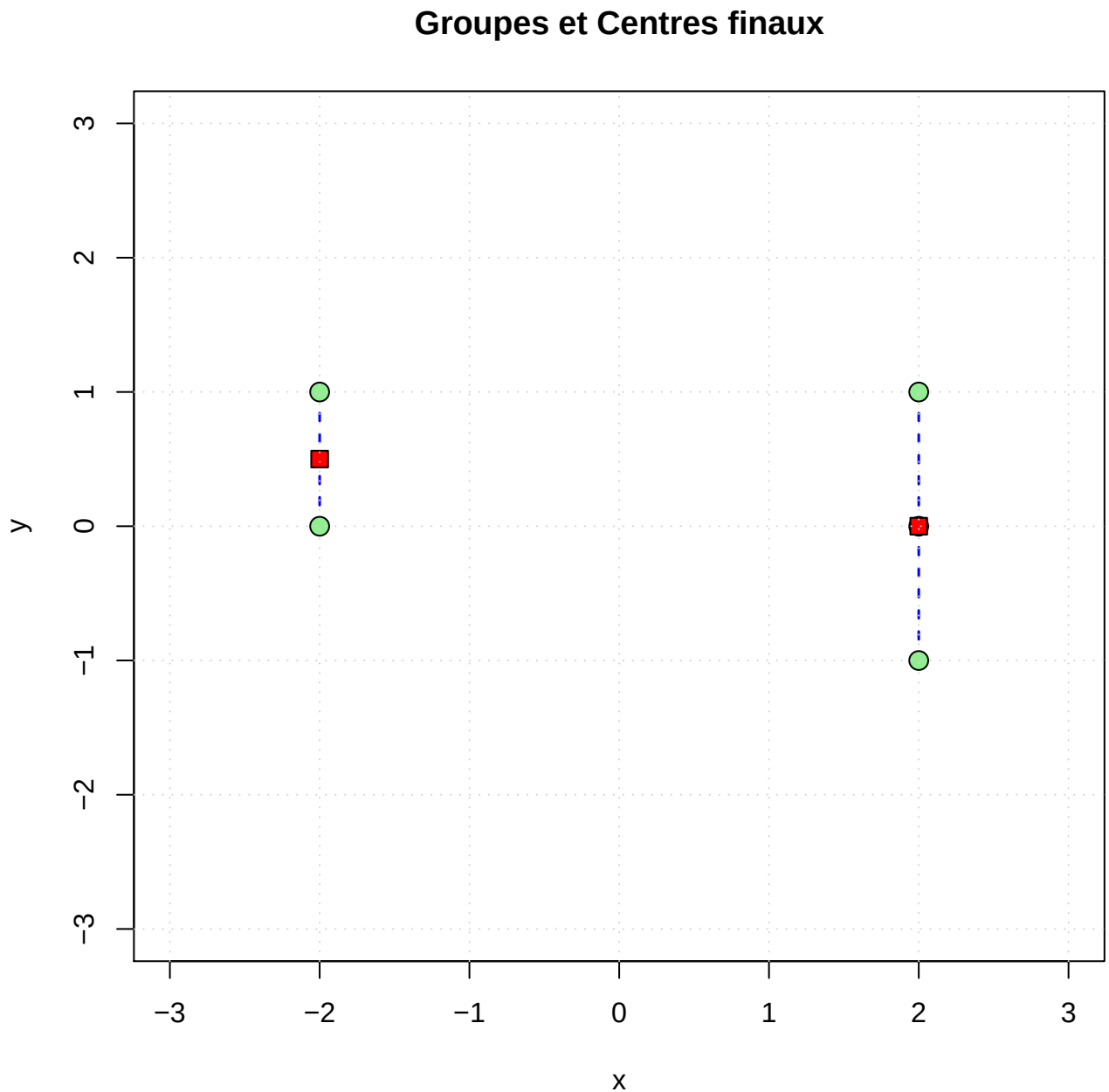
Exemple (1/3)



Exemple (1/3)



Exemple (1/3)



En pratique

- Initialisation : au hasard ou kmeans++ [3]
- Faire tourner plusieurs fois, avec différentes initialisations (choisir la meilleure solution au vue de l'inertie)

(3). D. Arthur et S. Vassilvitskii. "k-means++ : The advantages of careful seeding". In : *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial et Applied Mathematics. 2007, p. 1027-1035.

Théorème (4)

L'algorithme k-means fait décroître l'inertie, et s'arrête donc en un nombre fini d'étapes.

$$I(\mu^t, \mathcal{C}^t) = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k^t} d^2(x_i, \mu_k^t) \geq \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k^t} d^2(x_i, \mu_{h^*(x_i)}^t) \quad \text{(affectation)}$$

$$= \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k^{t+1}} d^2(x_i, \mu_{h^*(x_i)}^t) \geq \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k^{t+1}} d^2(x_i, \mu_k^{t+1}) \quad \text{(estimation)}$$

en remarquant que $h^*(x_i)$ est constant sur $\mathcal{C}_k^{t+1}$ et que

$$\forall \mu \in \mathbb{R}^d, \sum_{i \in \mathcal{C}} d^2(x_i, \mu) \geq \sum_{i \in \mathcal{C}} d^2(x_i, \bar{x}), \text{ avec } \bar{x} = \frac{1}{|\mathcal{C}|} \sum_{i \in \mathcal{C}} x_i$$

Pour la convergence : il n'y a qu'un nombre fini de partitions

(4). A. Banerjee et al. "Clustering with Bregman divergences". In : *J. Mach. Learn. Res.* 6 (2005), p. 1705-1749.

Le k-means en quantification

Quantification vectorielle :

- L'objet de la quantification est de remplacer un ensemble de données par une représentation compacte, sous la forme de centroïdes $\mu_1, \ldots, \mu_K$.
- Une mesure de perte, ou de distorsion, est l'erreur quadratique moyenne.
- L'algorithme des k-means permet de sélectionner les centroïdes minimisant le critère quadratique de distorsion.

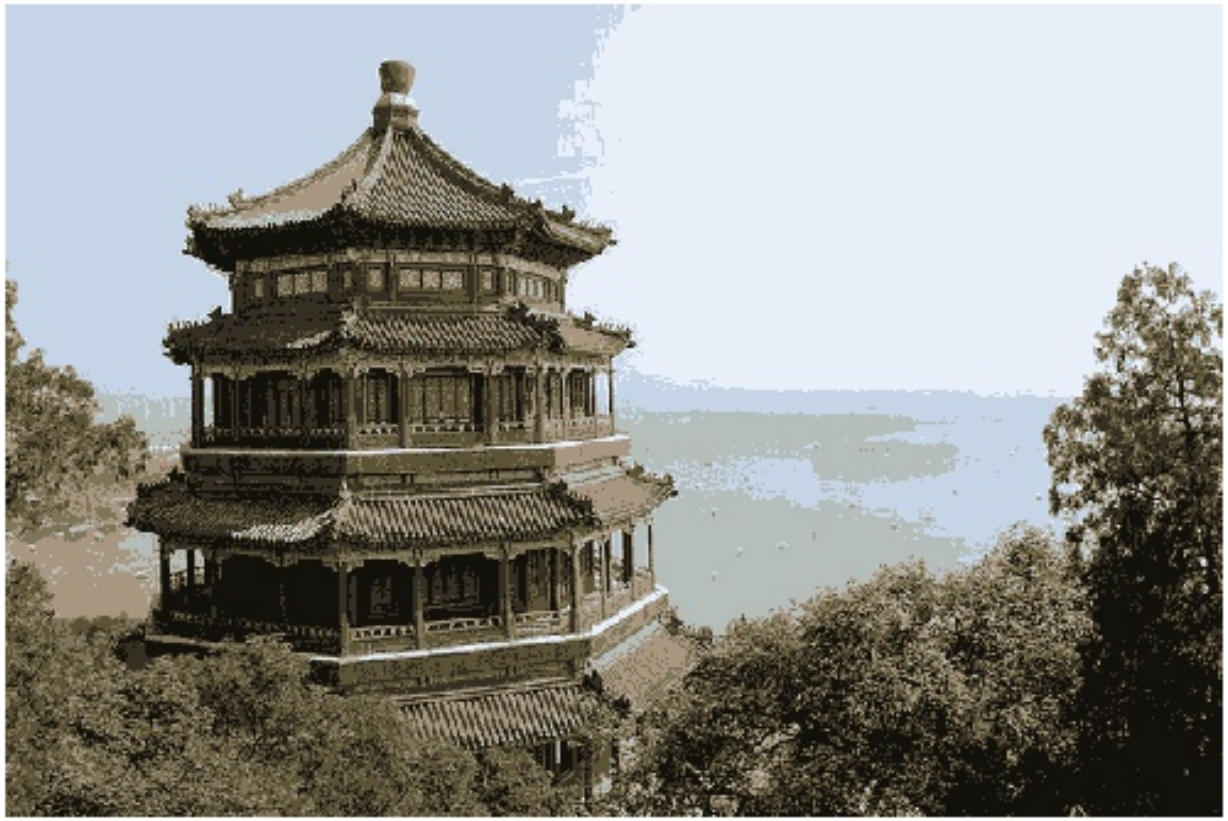
Application imagerie : compression d'images ou de signaux

$n = 427 \times 640$, $d = 3$ (RGB)

Application imagerie : compression d'images ou de signaux

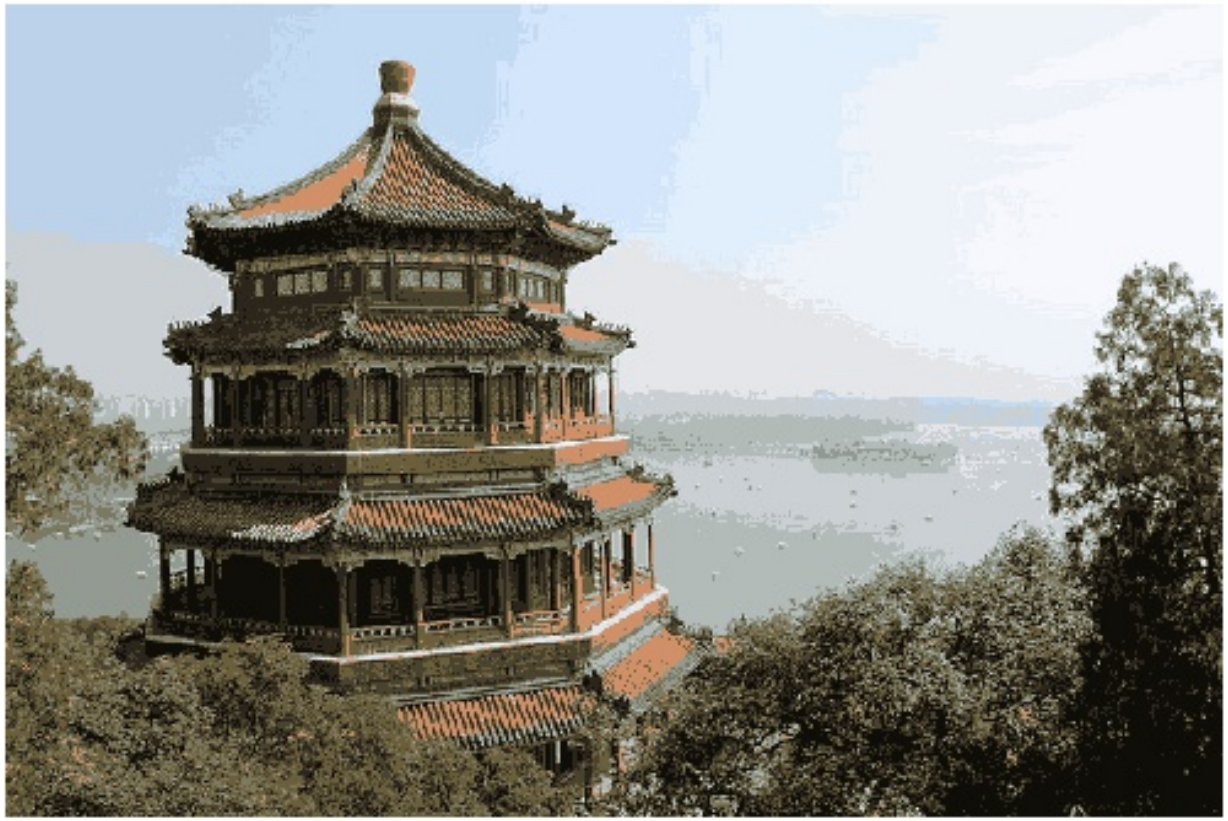
$$K = 2$$

Application imagerie : compression d'images ou de signaux



$$K = 8$$

Application imagerie : compression d'images ou de signaux



$$K = 16$$

Application imagerie : compression d'images ou de signaux

$$K = 32$$

Application imagerie : compression d'images ou de signaux

$$K = 64$$

Géométrie des classes

Définition : Partition de Voronoi

Les K centres $\mu_1, \ldots, \mu_K$ induisent une partition de $\mathbb{R}^d$ appelé la **partition de Voronoi** $V_1, \ldots, V_K$, où :

- $V_k = \{x \in \mathbb{R}^d \ : \ \|x - \mu_k\| \leq \min_{\ell \neq k} \|x - \mu_\ell\|\}$
- $V_1 \cup \cdots \cup V_K = \mathbb{R}^d$
- $V_k \cap V_\ell = \emptyset$, pour $k \neq \ell$ (aux bords près...).

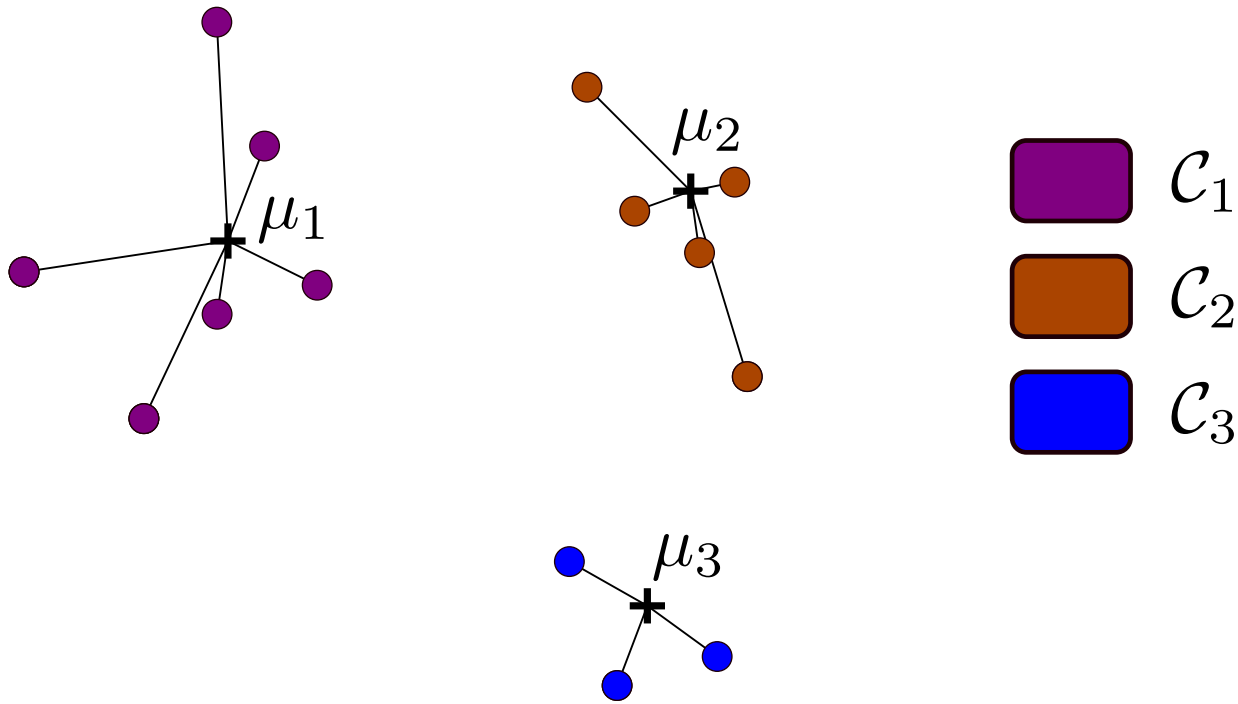
Les V_k sont appelées **cellules** (de Voronoi)

Affectation des classes :

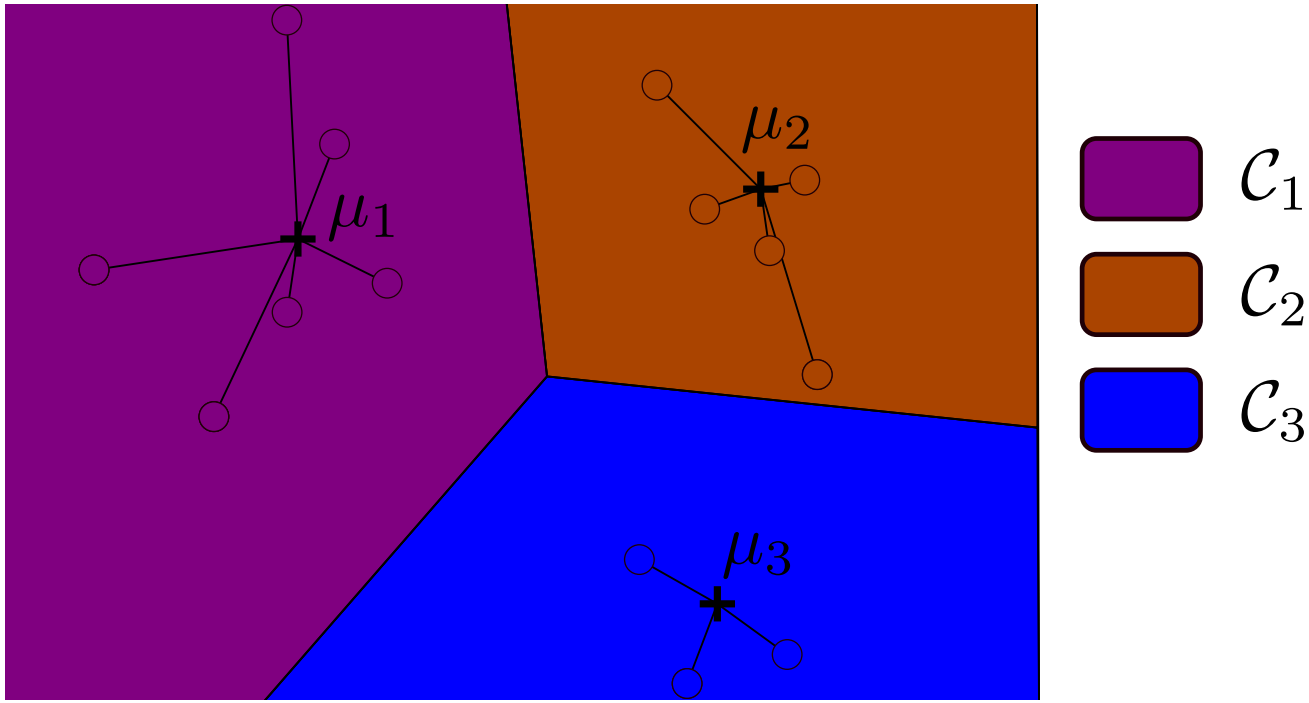
- l'observation x_i est affectée à la k-ième classe si $\|x_i - \mu_k\| \leq \min_{\ell \neq k} \|x_i - \mu_\ell\|$
- dans ce cas, x_i appartient à la **cellule** V_k

Rem: les cellules de Voronoi sont **convexes**

Visualisation



Visualisation



Plan

Définition

Un mélange de lois gaussiennes est une loi dont la densité s'écrit :

$$f(x) = \sum_{m=1}^{M} \alpha_m \phi(x; \mu_m, \Sigma_m), \quad x \in \mathbb{R}^d,$$

où :

(i) α_m sont les coefficients du mélange : $\alpha_m \geq 0$ et $\sum_{m=1}^{M} \alpha_m = 1$.

(ii) $\phi(\cdot; \mu_m, \Sigma_m)$ est la densité de la loi gaussienne, de moyenne μ_m, et de matrice de covariance Σ_m.

Rem: autres familles de lois possibles (Cauchy, Laplace, t-student,)

Estimation des paramètres du modèle

Paramètres à estimer :

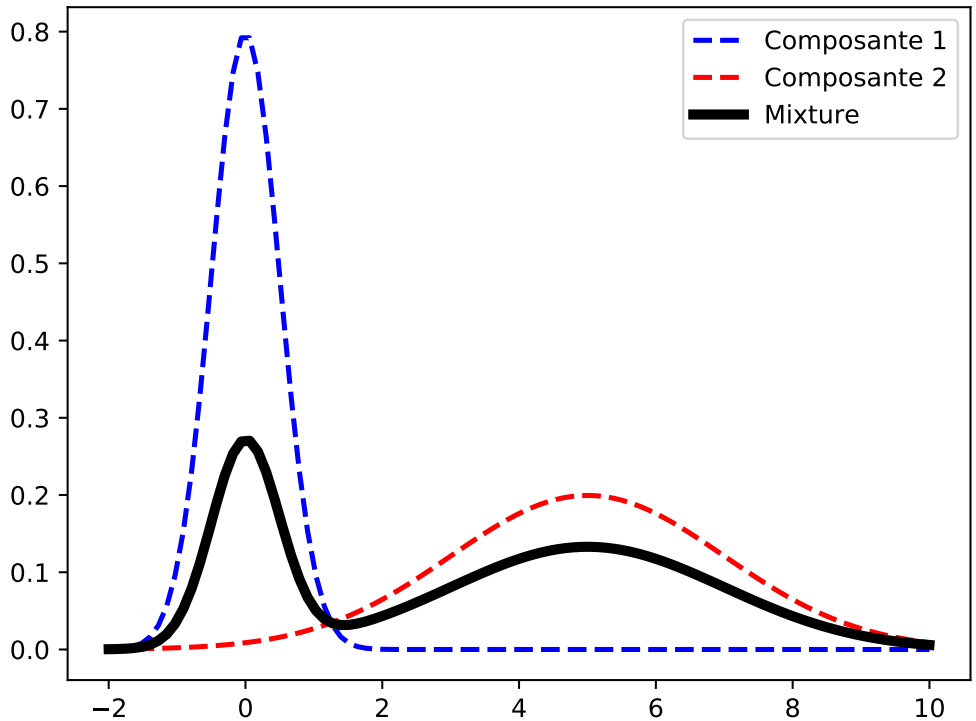
- les coefficients α_m
- les moyennes μ_m
- les matrices de covariance Σ_m,
- (souvent) le nombre de composantes du mélange, M

Problème. L'estimation par maximum de vraisemblance est ardue.
Sur l'échantillon $x_1, \ldots, x_n$, la vraisemblance s'écrit :

$$L(\theta) = \prod_{i=1}^{n} f(x_i; \theta) = \prod_{i=1}^{n} \left(\sum_{m=1}^{M} \alpha_m \phi(x_i; \mu_m, \Sigma_m) \right).$$

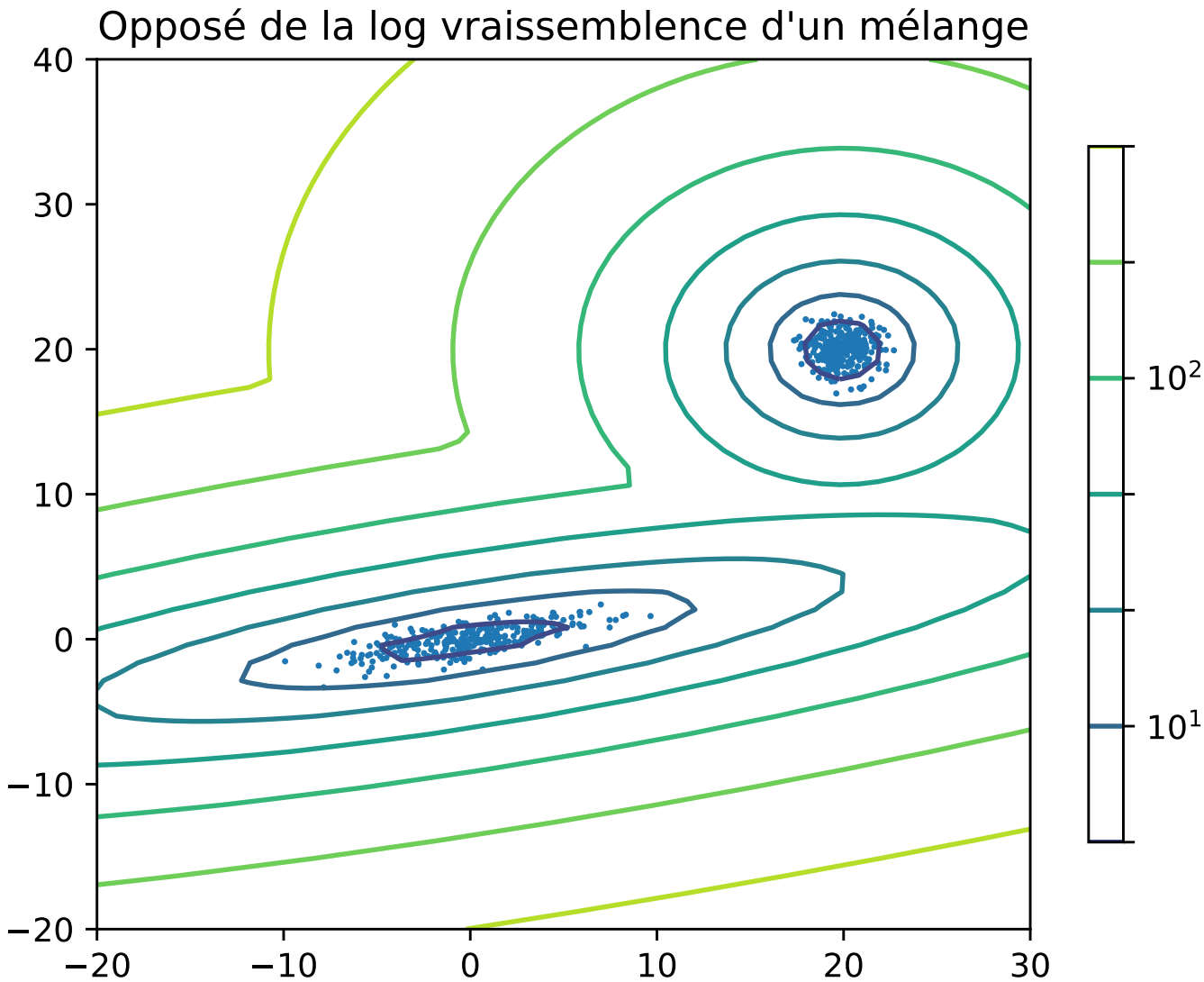
$\longrightarrow$ Pas de formule analytique pour $\hat{\mu}_m$ et $\hat{\Sigma}_m$ si $M > 1$.

Exemple 2D



$$f(x) = \frac{1}{3}\phi(x; \mu_1, \sigma_1^2) + \frac{2}{3}\phi(x; \mu_2, \sigma_2^2), \text{ tq. } \begin{cases} \mu_1 = 0, \sigma_1 = 0.5 \\ \mu_2 = 5, \sigma_2 = 2 \end{cases}$$

Exemple 2D


Opposé de la log vraissemblence d'un mélange
40
30
20
10
0
−10
−20
−20
−10
0
10
20
30
10^2
10^1

Maximum a posteriori

Une fois le modèle ajusté (les coefficients $\hat{\alpha}, \hat{\mu}_m, \hat{\Sigma}_m$ estimés), on affecte x_i à la classe $\mathcal{C}_{\hat{m}_i}$ défini par

$$\hat{m}_i = \arg\max_m \hat{p}_{im} := \frac{\hat{\alpha}_m \phi(x_i; \hat{\mu}_m, \hat{\Sigma}_m)}{\sum_{r=1}^{M} \hat{\alpha}_r \phi(x_i; \hat{\mu}_r, \hat{\Sigma}_r)}$$

Interprétation "Variable cachée" /"variable latente"

G_i : variable aléatoire donnant le groupe auquel x_i appartient, c'est une **variable cachée**, *i.e.*, non observée

La probabilité que l'observation x_i soit dans le groupe $\mathcal{C}_m$ s'écrit :

$$\mathbb{P}\left(G_i = \mathcal{C}_m | X_i = x_i\right) = \frac{\hat{\alpha}_m \phi(x_i; \hat{\mu}_m, \hat{\Sigma}_m)}{\sum_{r=1}^{M} \hat{\alpha}_r \phi(x_i; \hat{\mu}_r, \hat{\Sigma}_r)}.$$

Ainsi

$$\hat{m}_i = \arg\max_m \hat{p}_{im}.$$

Lien avec les k-means

Méthode du k-means

(i) Estimation de M centroïdes.

(ii) Chaque donnée est affectée au centroïde le plus proche

Modèles de mélange :

(i) Estimation M moyennes et matrices de covariance.

(ii) Chaque donnée est affecté au groupe dont la composante du mélange est la plus probable

$\longrightarrow$ La partition obtenue dépend des centroïdes, mais également des matrices de covariances, qui déterminent la forme des groupes

Principe de l'algorithme EM

Maximisation directe de la vraisemblance difficile $\Longrightarrow$ approche alternée (comme pour k-means / algorithme de Lloyd)

Algorithme EM Expectation - Maximisation :

Initialisation : choix d'un mélange de départ.

Expectation Pour chaque donné x_i, calculer la probabilité que x_i soit dans le groupe m

Maximization Étant données les affectations des données en groupes, estimer les paramètres μ_m et Σ_m par maximum de vraisemblance

Itérer 2 et 3 jusqu'à convergence

Complexité du modèle

Pour M composantes, avec $x \in \mathbb{R}^d$, les paramètres sont :

- M moyennes, soit $M \times d$ réels
- M matrice de covariances, soit $M \times d(d+1)/2$ réel
- $(M-1)$ coefficients α_m

Hypothèses sur la variance

Pour simplifier, rajouter des hypothèses sur les Σ_m, *e.g.*, :

Famille sphérique :

- $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_M = \sigma^2 \operatorname{Id}_d$
- Pour chaque m, $\Sigma_m = \sigma_m^2 \operatorname{Id}_d$ (spherical)

Famille diagonale :

- $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_M = \operatorname{diag}(\sigma_1^2, \ldots, \sigma_d^2)$
- $\forall m$, $\Sigma_m = \operatorname{diag}(\sigma_{1,m}^2, \ldots, \sigma_{d,m}^2)$ (diag)

Rem: (full) : sans hypothèse, (tied) : covariance partagée $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_M$

Exemple