

SYS843
Réseaux de neurones et
systèmes flous
Séance 09

Eric Granger
Ismail Ben Ayed

Hiver 2017

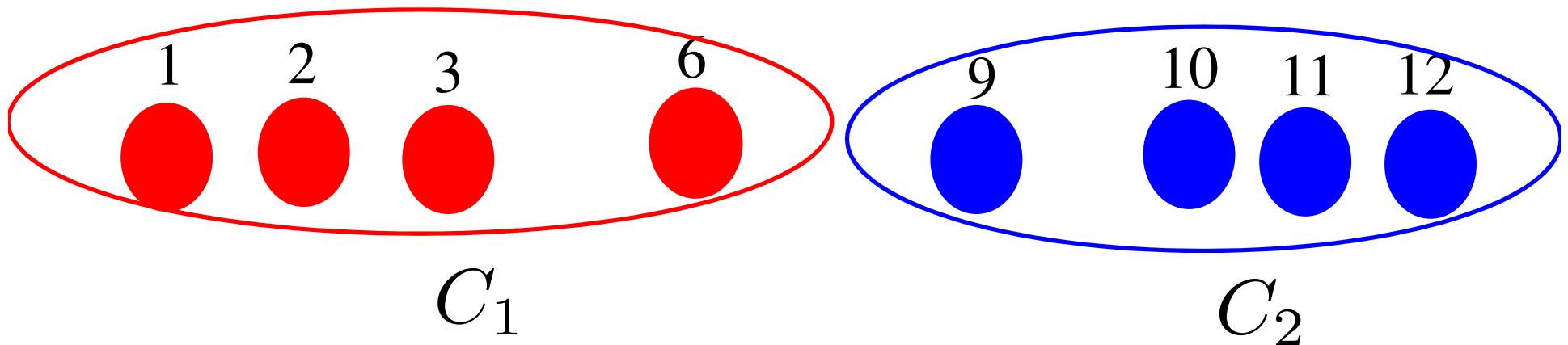
Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

- 1) Algorithme classique k -means
- 2) Modèles probabilistes de clustering
- 3) Mélange de Gaussiennes
- 4) Algorithme fuzzy C-means

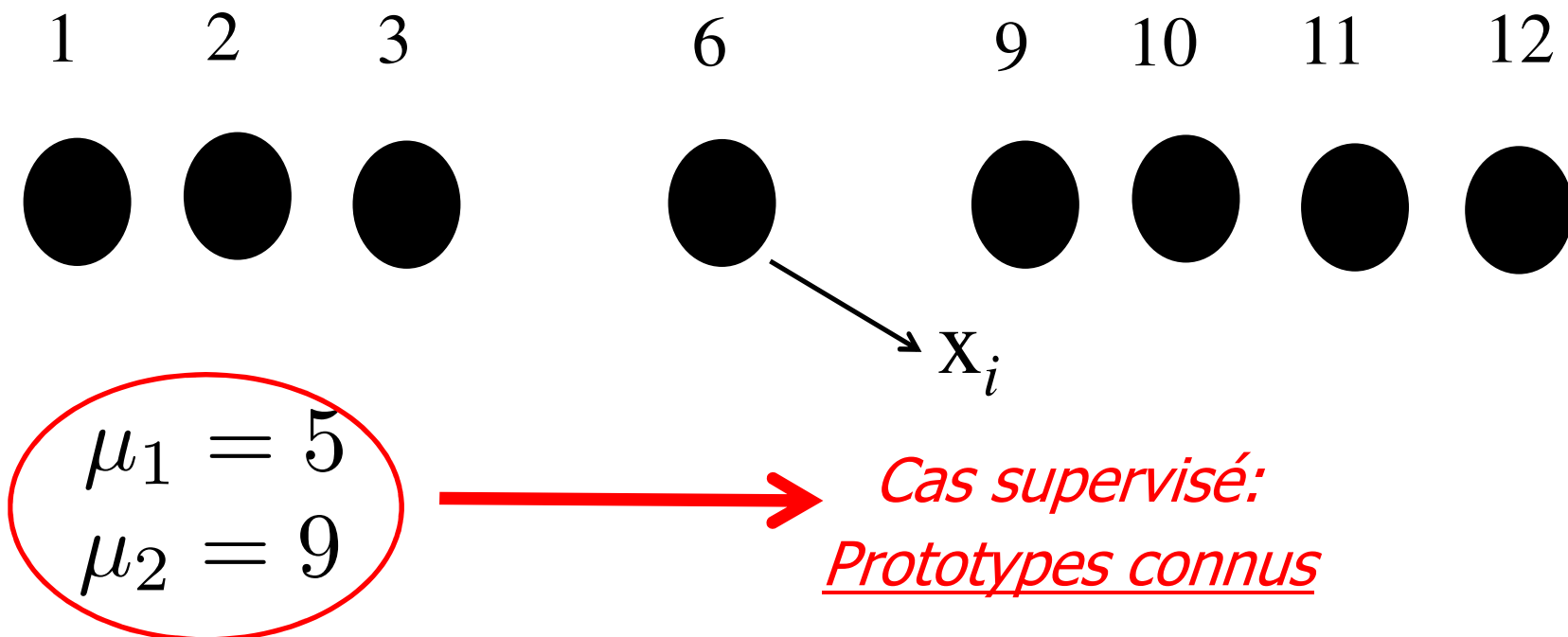
Algorithme *K-means*

- Un algorithme qui identifie des groupes (clusters, catégories) dans un ensemble de points (patrons) de façon non-supervisée
- Produit une partition dure des données.



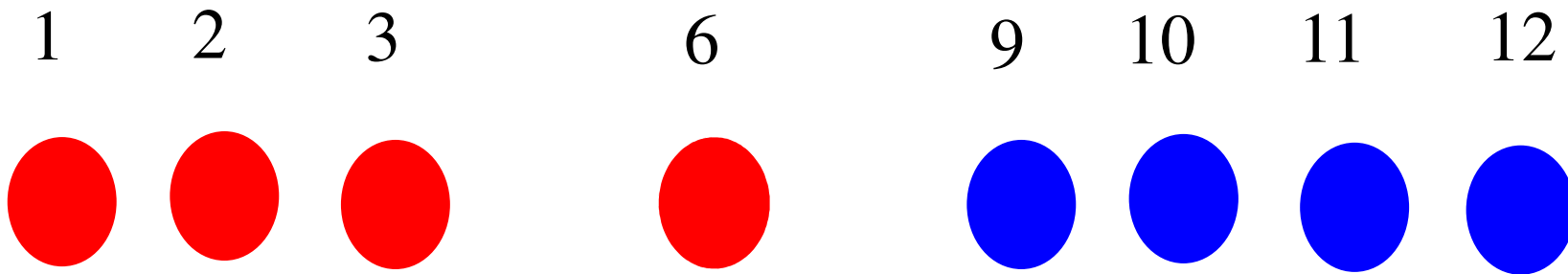
Algorithme *K-means*

• **Objectif:** Regrouper les patrons \mathbf{x}_i d'un ensemble $D_n = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ dans un des K catégories, où chaque catégorie est représenté par un prototype μ_k .



Algorithme *K-means*

• **Objectif:** Regrouper les patrons \mathbf{x}_i d'un ensemble $D_n = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ dans un des K catégories, où chaque catégorie est représenté par un prototype μ_k .



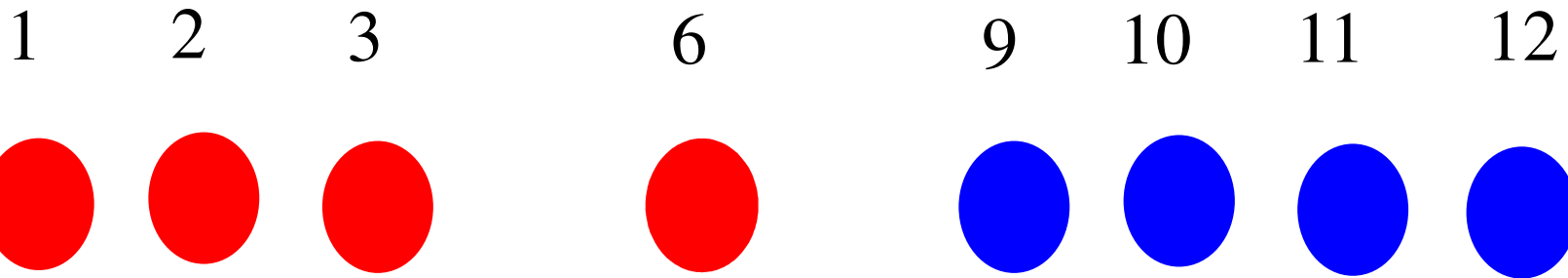
$$\begin{aligned}\mu_1 &= 5 \\ \mu_2 &= 9\end{aligned}$$



Cas supervisé:
Classification à distance minimum

Algorithme *K-means*

• **Objectif:** Regrouper les patrons \mathbf{x}_i d'un ensemble $D_n = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ dans un des K catégories, où chaque catégorie est représenté par un prototype μ_k .



Fonctions caractéristiques du groupe k ($k=1,2$)

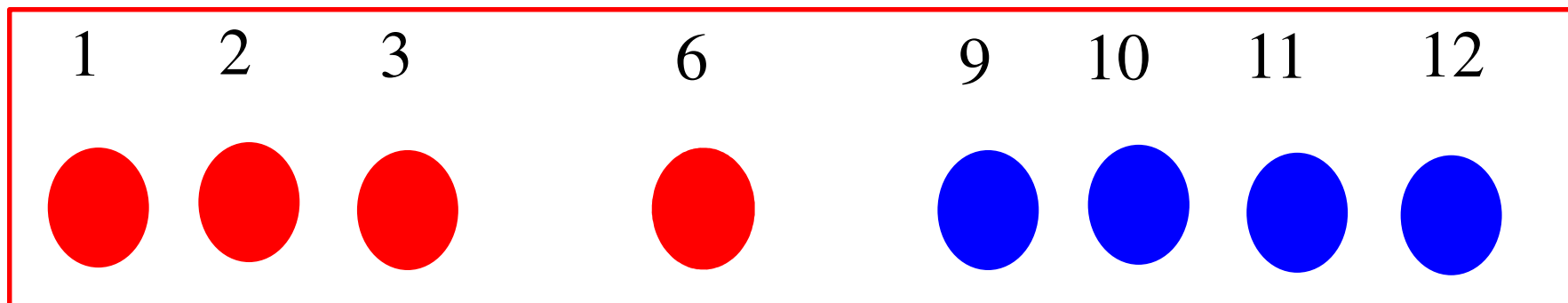
$$r_{ik} = \begin{cases} 1 & \text{si } k = \underset{j}{\operatorname{argmin}} \|\mathbf{x}_i - \mu_j\|^2; \\ 0 & \text{sinon.} \end{cases}$$

Classificateur à distance minimum

Algorithme *K-means*

• **Fonction de coût:** On minimise la somme des erreurs au carré (Cas $K=2$)

$$J = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 + \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2$$



$$\begin{aligned}\mu_1 &= 5 \\ \mu_2 &= 9\end{aligned}$$

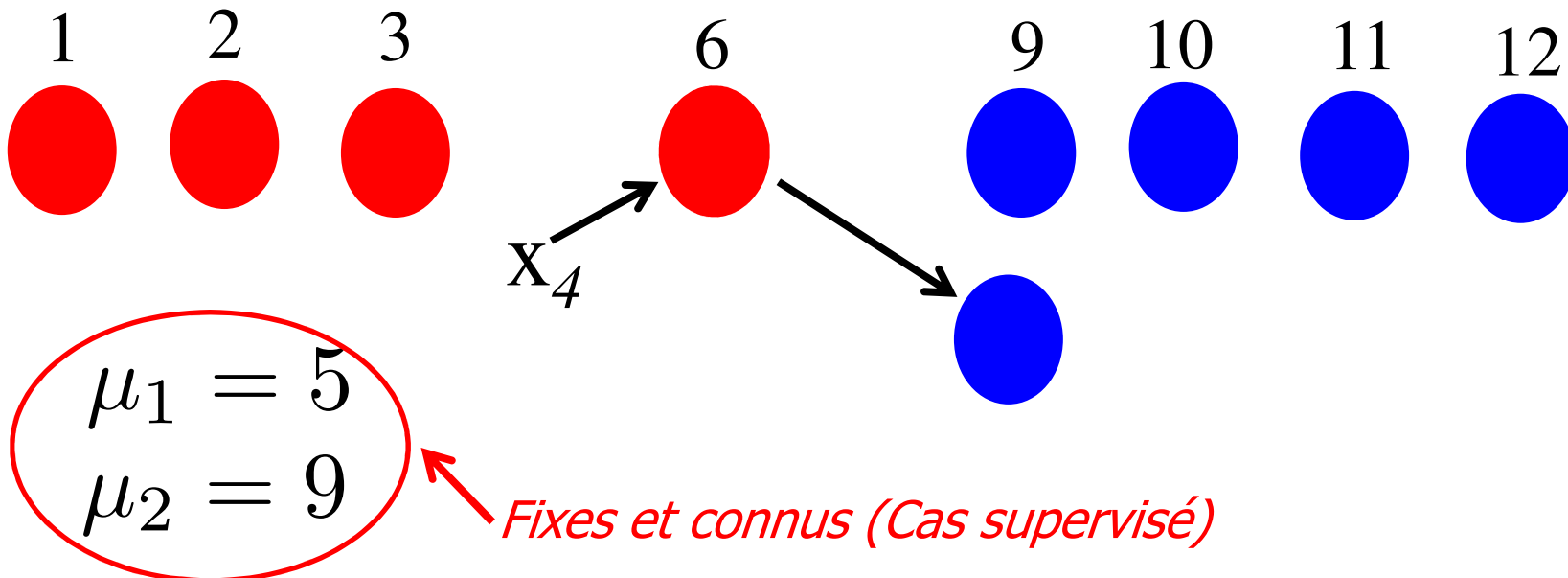
Cette solution est le minimum de la fonction

Fixes et connus (Cas supervisé)

Algorithme *K-means*

- Supposons qu'on change le groupe de \mathbf{x}_4

$$\begin{aligned}\delta J &= -\|\mathbf{x}_i - \mu_1\|^2 + \|\mathbf{x}_i - \mu_2\|^2 \\ &= -\|6 - 5\|^2 + \|6 - 9\|^2 > 0\end{aligned}$$

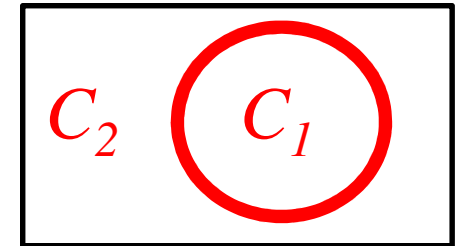


Algorithme *K-means*

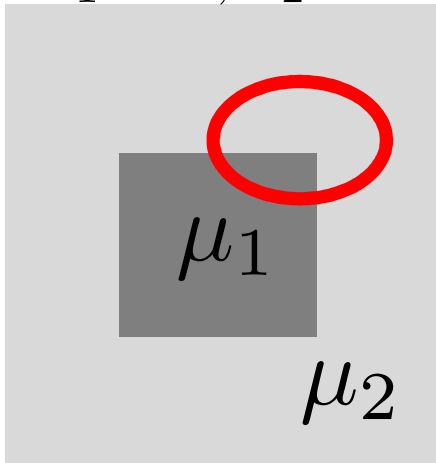
- Un exemple simple qui montre que la fonction de coût est appropriée ($K=2$):

$$F_1 = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 > 0$$

$$F_2 = \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2 > 0$$



$$F_1 > 0, F_2 > 0$$

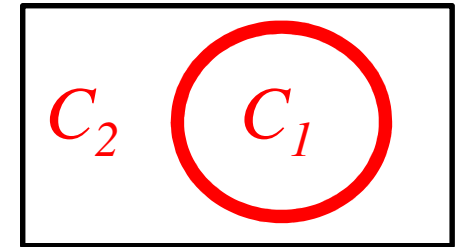


Algorithme *K-means*

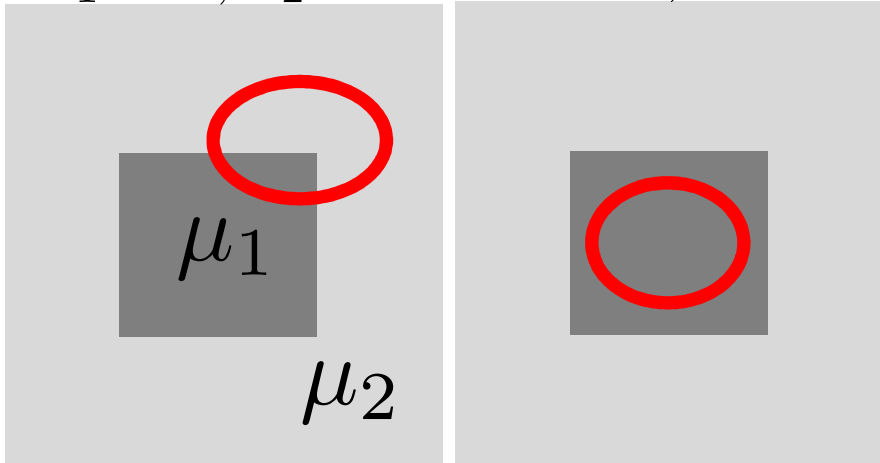
- Un exemple simple qui montre que la fonction de coût est appropriée ($K=2$):

$$F_1 = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 > 0$$

$$F_2 = \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2 > 0$$



$$F_1 > 0, F_2 > 0 \quad F_1 = 0, F_2 > 0$$

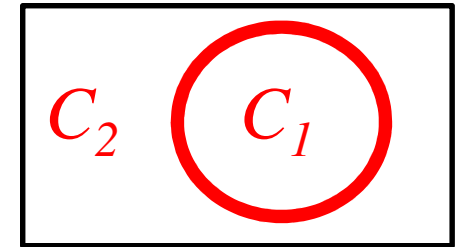


Algorithme *K-means*

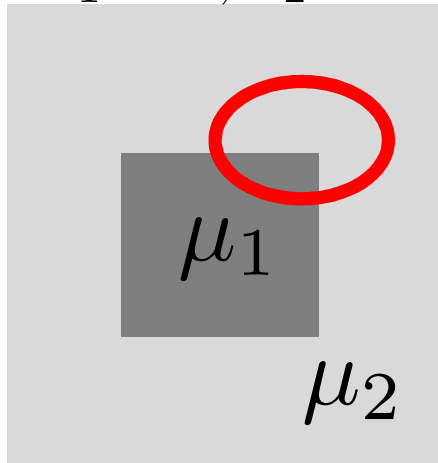
- Un exemple simple qui montre que la fonction de coût est appropriée ($K=2$):

$$F_1 = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 > 0$$

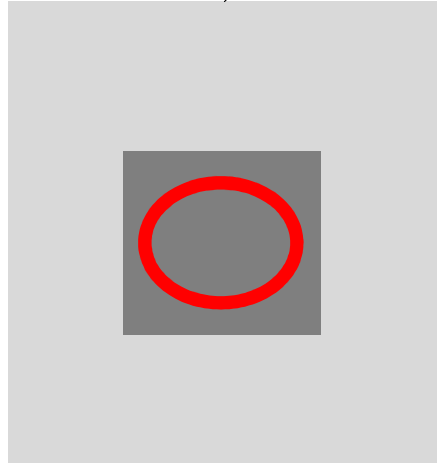
$$F_2 = \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2 > 0$$



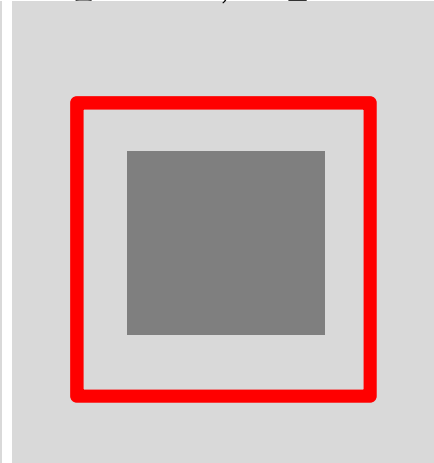
$$F_1 > 0, F_2 > 0$$



$$F_1 = 0, F_2 > 0$$



$$F_1 > 0, F_2 = 0$$

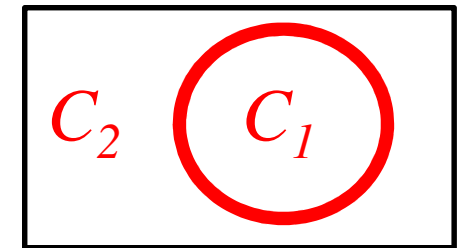


Algorithme *K-means*

- Un exemple simple qui montre que la fonction de coût est appropriée ($K=2$):

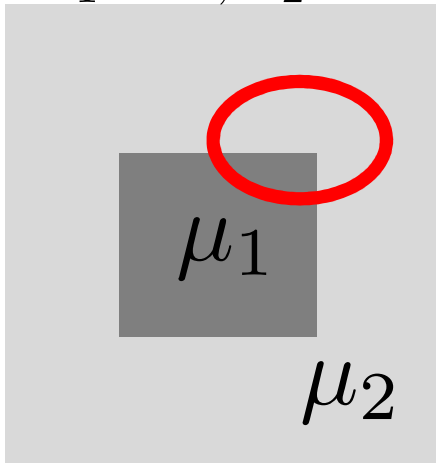
$$F_1 = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 > 0$$

$$F_2 = \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2 > 0$$

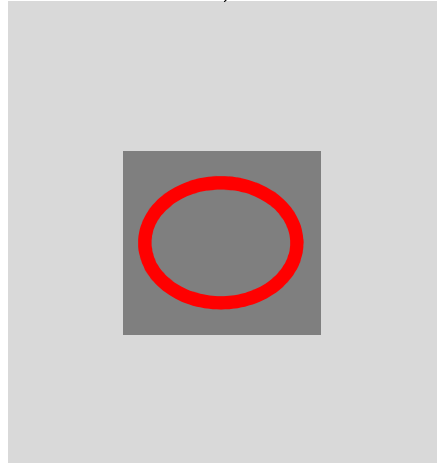


Minimum

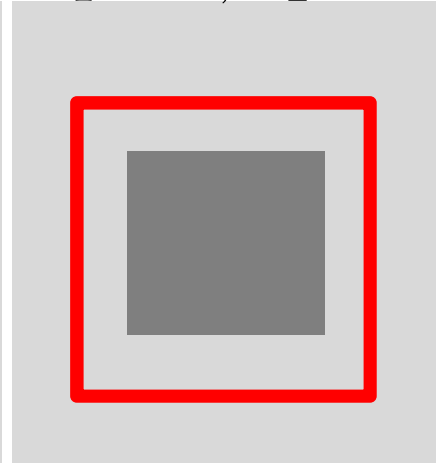
$F_1 > 0, F_2 > 0$



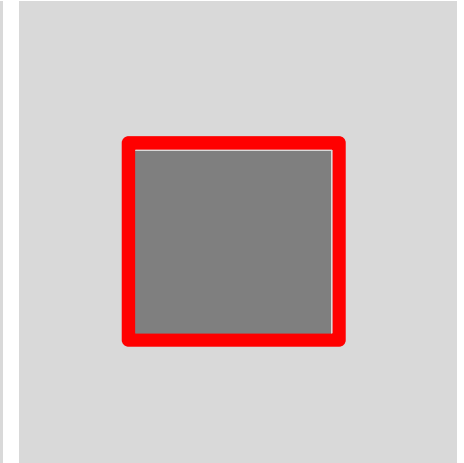
$F_1 = 0, F_2 > 0$



$F_1 > 0, F_2 = 0$



$F_1 = 0, F_2 = 0$



K-means: Fonction de coût

- $K=2$:

$$J(C_1, C_2) = \sum_{\mathbf{x}_i \in C_1} \|\mathbf{x}_i - \mu_1\|^2 + \sum_{\mathbf{x}_i \in C_2} \|\mathbf{x}_i - \mu_2\|^2$$

- $K>2$:

$$J(C_1, C_2, \dots, C_K) = \sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \|\mathbf{x}_i - \mu_k\|^2$$

K-means: Fonction de coût

- K arbitraire:

$$J(C_1, C_2, \dots, C_K) = \sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \|\mathbf{x}_i - \mu_k\|^2$$

$$r_{ik} = \begin{cases} 1 & \text{si } \mathbf{x}_i \in C_k; \\ 0 & \text{sinon.} \end{cases}$$

*Fonction
de partition*

$$J(r) = \sum_{k=1}^K \sum_{i=1}^N r_{ik} \|\mathbf{x}_i - \mu_k\|^2$$


Algorithme *K-means*

Processus itératif à deux étapes

Initialise K prototypes μ_k ou la partition r de façon aléatoire

1. **Partition (r):** Assigne la catégorie la plus *proche* à chaque patron \mathbf{x}_i en optimisant J par rapport à r

$$r_{ik} = \begin{cases} 1 & \text{si } k = \operatorname{argmin}_j \|\mathbf{x}_i - \mu_j\|^2; \\ 0 & \text{sinon.} \end{cases}$$

 *Approche discriminative*

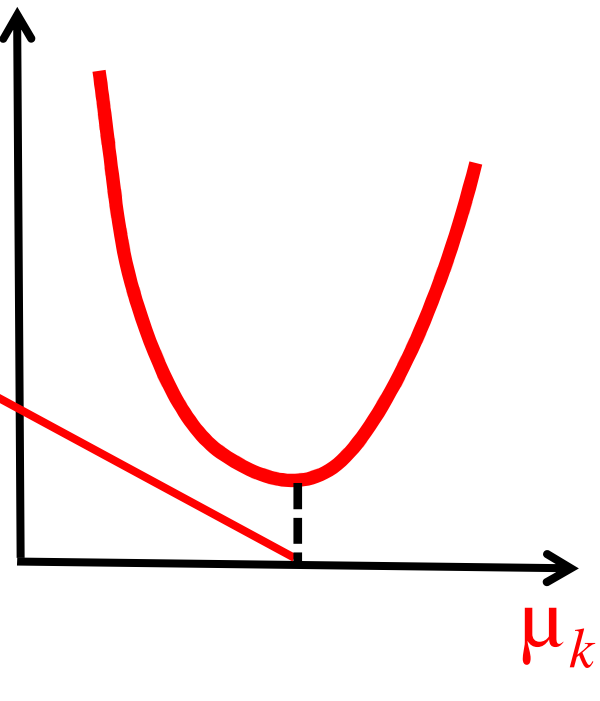
2. **Apprentissage:** Mise-à-jour du prototype de chaque catégorie en optimisant J par rapport aux prototypes μ_k
3. **Critère d'arrêt:** Si la fonction coût change, revenir à l'étape 2

Algorithme *K-means*

Apprentissage: mise à jour des prototypes

Optimisation de J par rapport aux K prototypes μ_k

$$\frac{\partial J}{\partial \mu_k} = 0$$

$$2 \sum_{i=1}^N r_{ik} (\mathbf{x}_i - \mu_k) = 0$$


Algorithme *K-means*

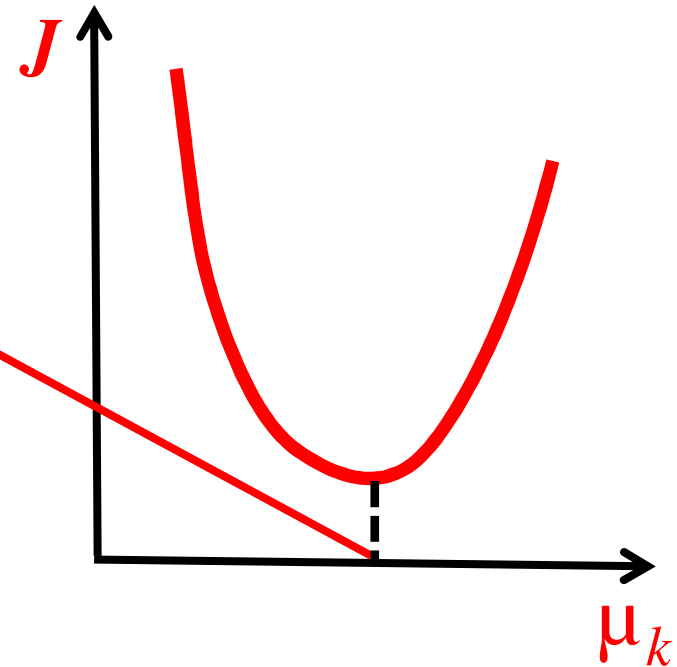
Apprentissage: mise à jour des prototypes

Optimisation de J par rapport aux K prototypes μ_k

$$\frac{\partial J}{\partial \mu_k} = 0$$

$$\frac{\sum_{i=1}^N r_{ik} \mathbf{x}_i}{\sum_{i=1}^N r_{ik}}$$

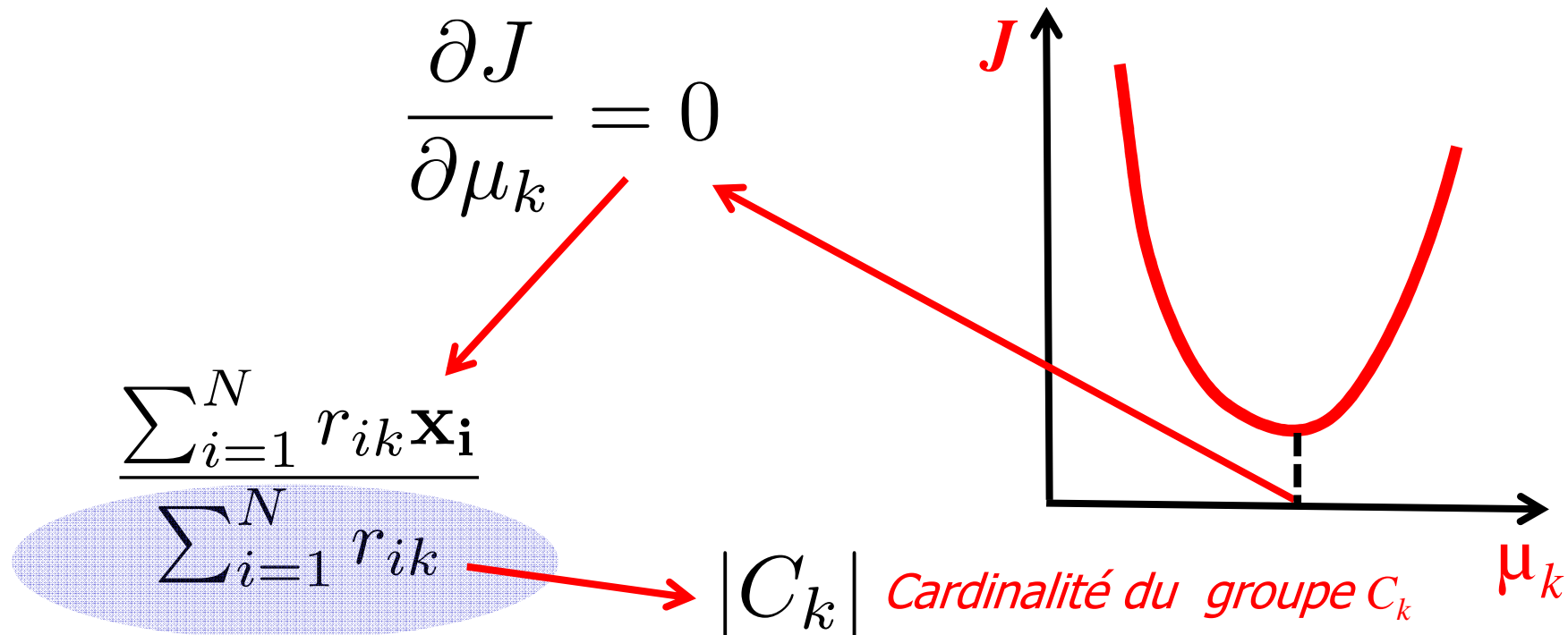
Moyenne dans groupe C_k



Algorithme *K-means*

Apprentissage: mise à jour des prototypes

Optimisation de J par rapport aux K prototypes μ_k



Algorithme *K-means*

Alterner le deux étapes jusqu'à convergence

Initialise K prototypes μ_k ou la partition r de façon aléatoire

1. **Partition (r):** Assigne la catégorie la plus *proche* à chaque patron \mathbf{x}_i en optimisant J par rapport à r

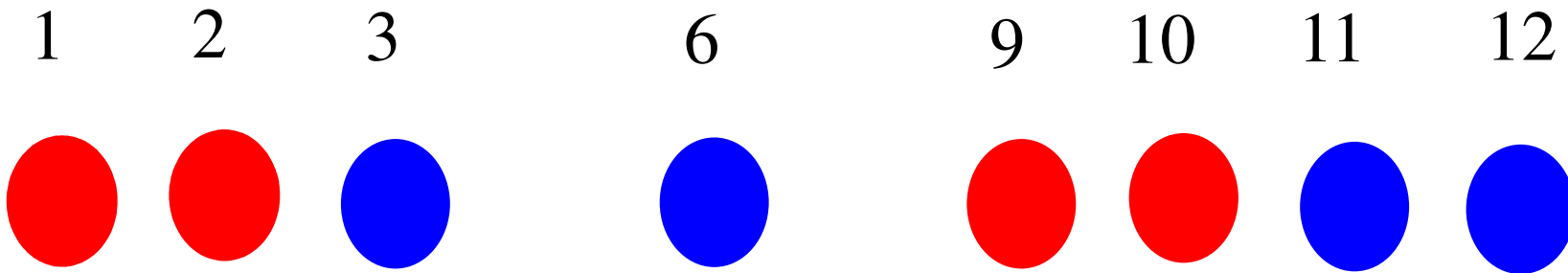
$$r_{ik} = \begin{cases} 1 & \text{si } k = \operatorname{argmin}_j \|\mathbf{x}_i - \mu_j\|^2; \\ 0 & \text{sinon.} \end{cases}$$

→ *Approche discriminative*

2. **Apprentissage:** Mise-à-jour du prototype de chaque catégorie en optimisant J par rapport aux prototypes μ_k

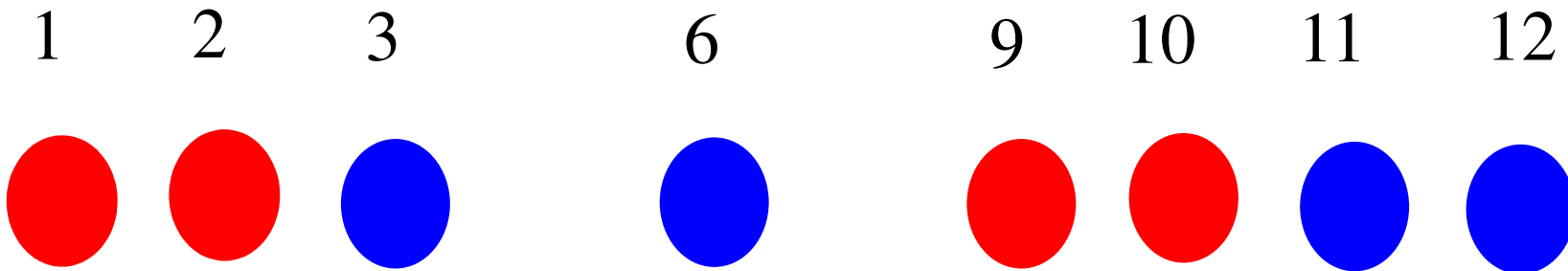
$$\mu_k = \frac{\sum_{i=1}^N r_{ik} \mathbf{x}_i}{\sum_{i=1}^N r_{ik}}$$

K-means: Un exemple simple



Initialisation aléatoire de la partition r

K-means: Un exemple simple

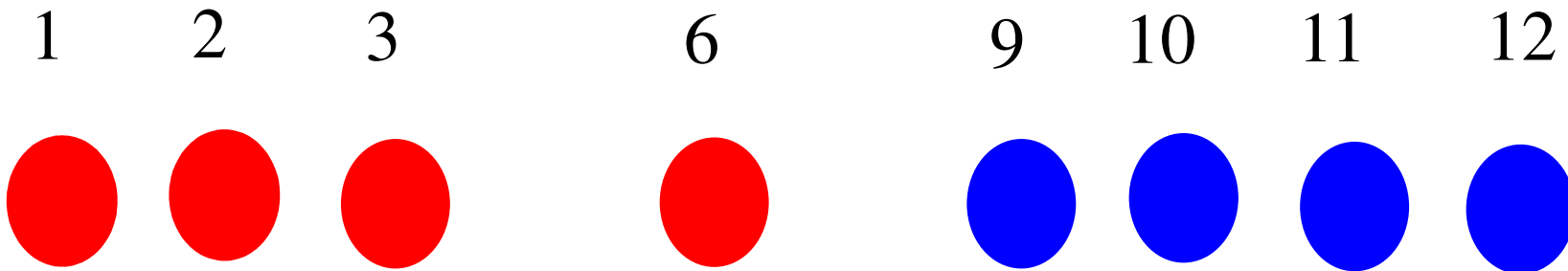


*Mise à jour des prototypes avec r fixé
(itération 1.1)*

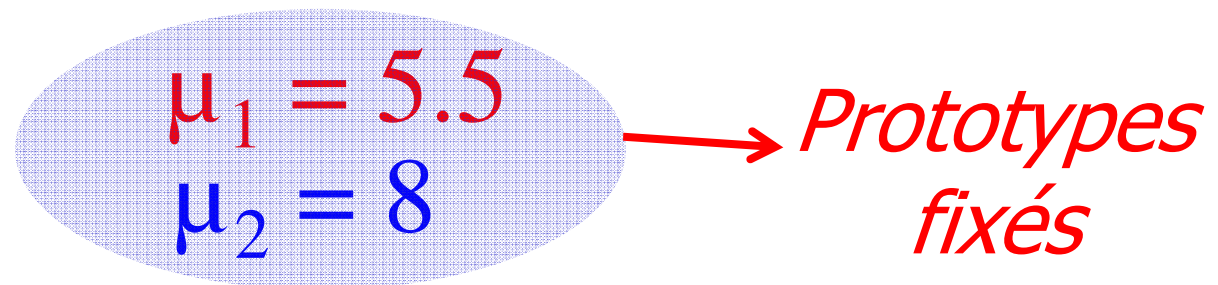
$$\mu_1 = 5.5$$

$$\mu_2 = 8$$

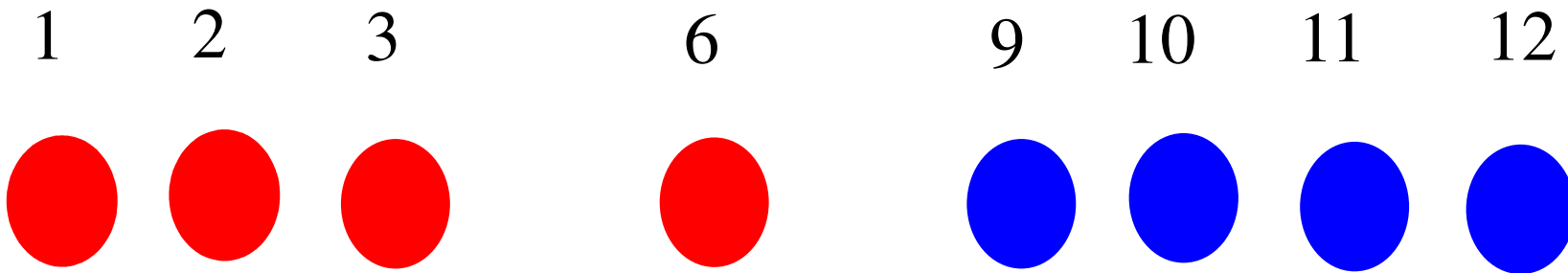
K-means: Un exemple simple



Mise à jour de la partition avec les prototypes fixés (itération 1.2)



K-means: Un exemple simple

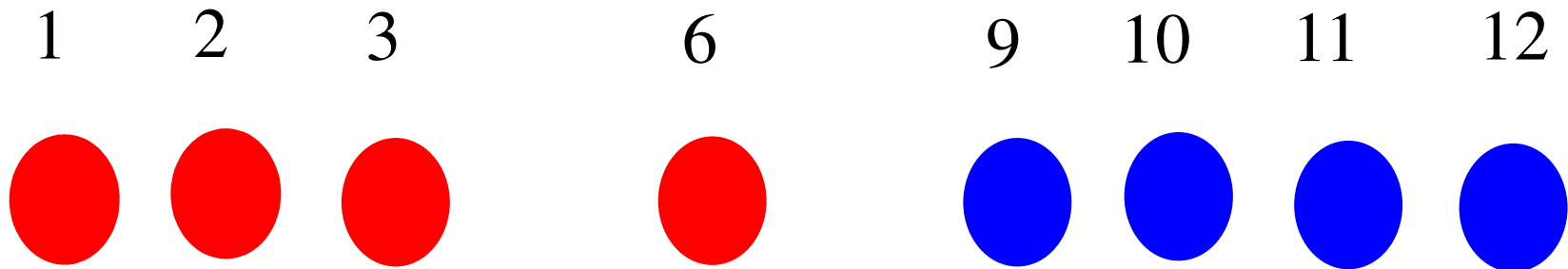


*Mise à jour des prototypes avec r fixé
(itération 2.1)*

$$\mu_1 = 3 \searrow$$

$$\mu_2 = 10.5 \nearrow$$

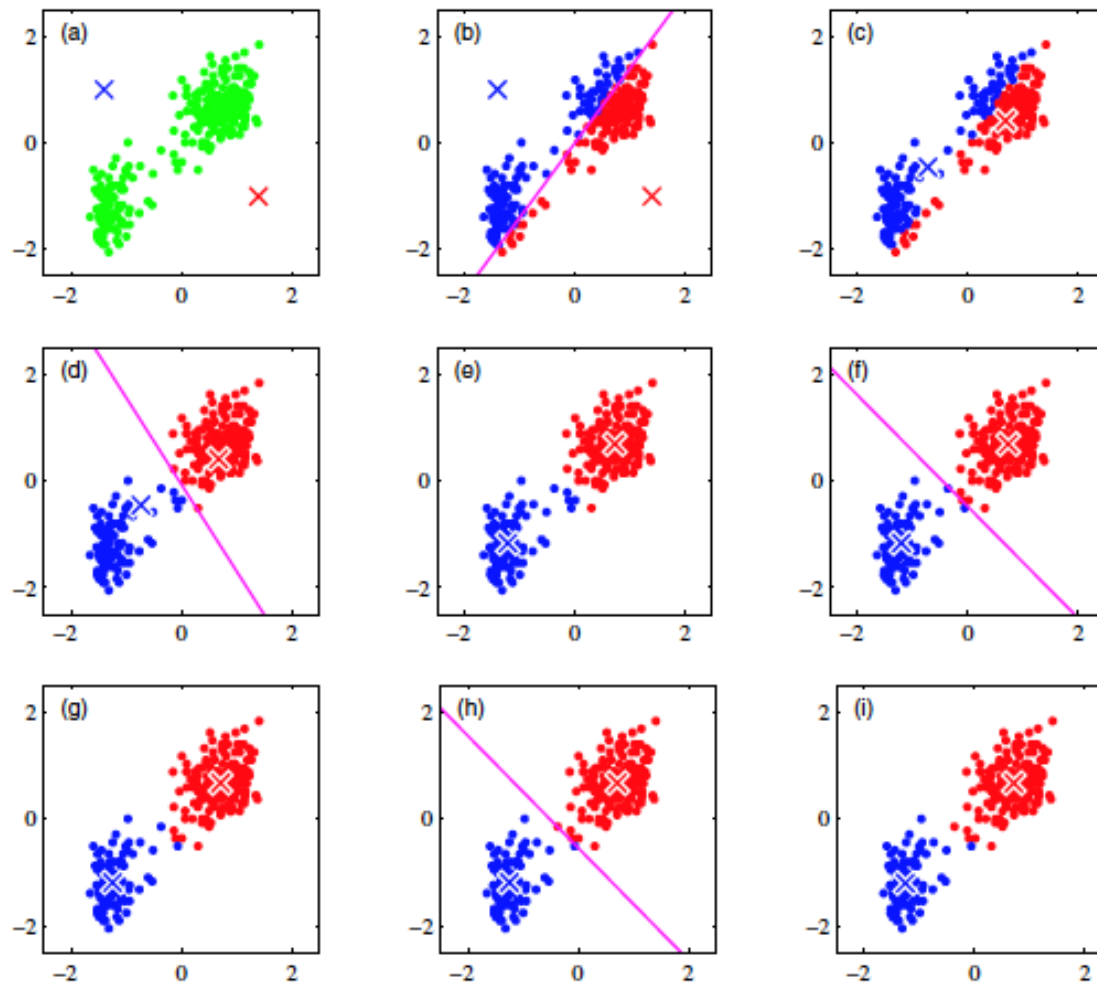
K-means: Un exemple simple



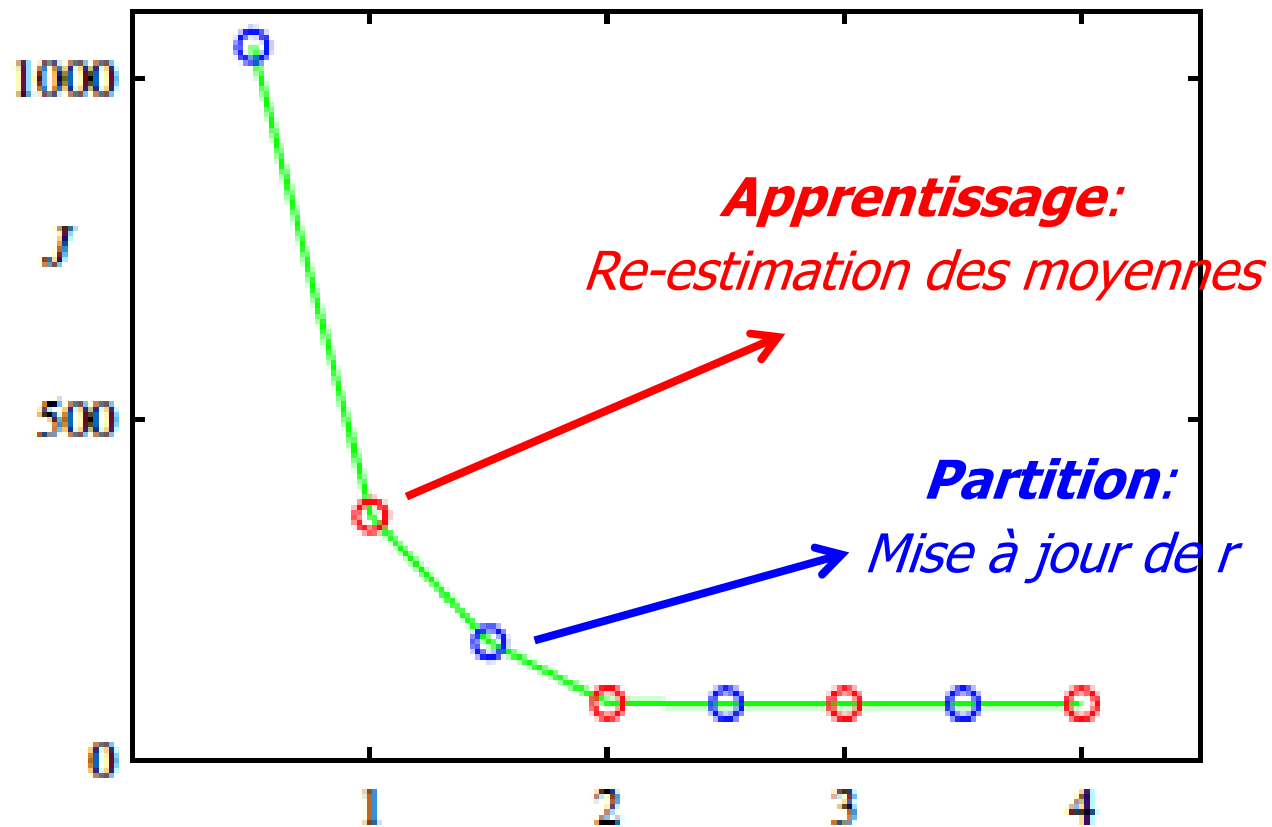
*Mise à jour de la partition (itération 2.2):
Convergence (pas de changement)*

$\mu_1 = 3$
 $\mu_2 = 10.5$ → *Prototypes
fixés*

K-means: Autres exemples



K-means: Autres exemples



K-means: Autres exemples

<https://www.youtube.com/watch?v=BVFG7fd1H30>

K-means: Autres exemples

$K = 2$



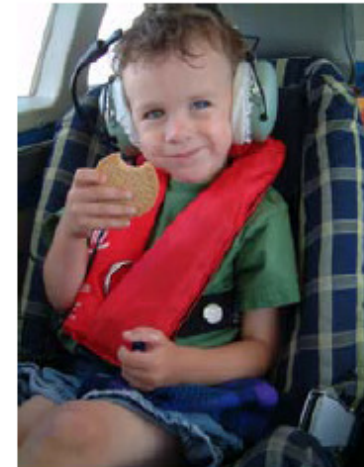
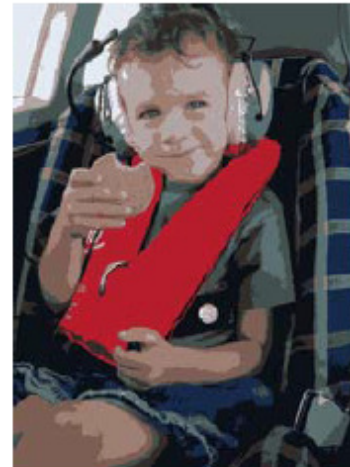
$K = 3$



$K = 10$



Original image

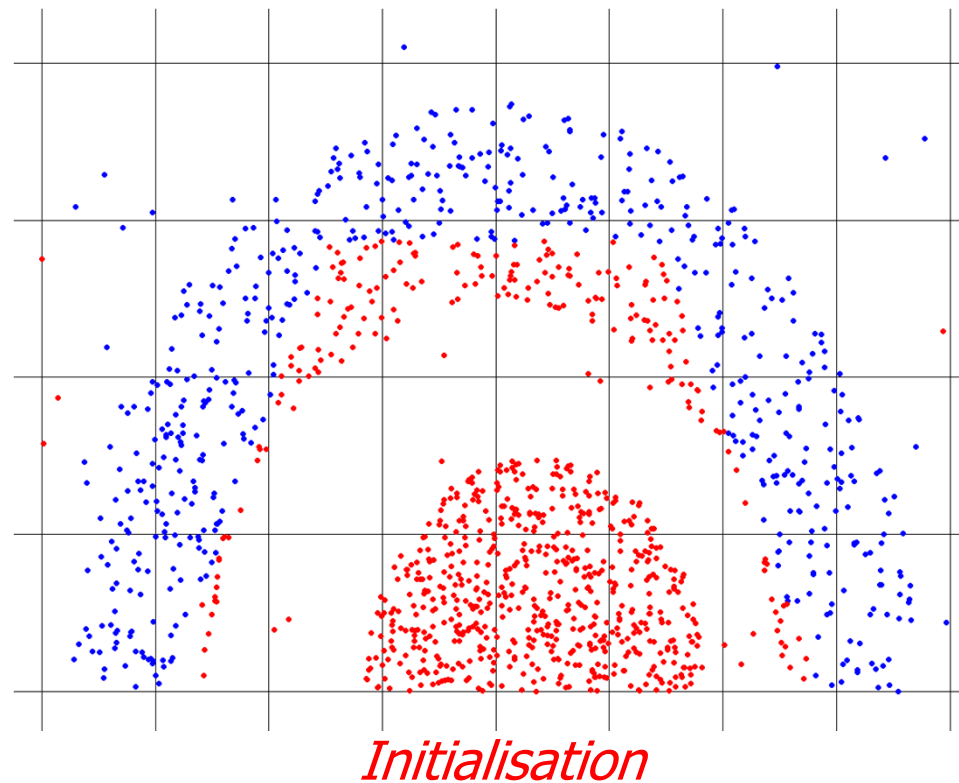


Algorithme *K-means*

Avantages	Inconvénients
<ul style="list-style-type: none">+ très simple et efficace+ généralité – peut fonctionner pour une mesure arbitraire<ul style="list-style-type: none">➤ <i>On va voir une généralisation dans ce qui suit</i>+ stabilité – converge toujours rapidement vers un minimum (soit local ou global)<ul style="list-style-type: none">➤ <i>Une approche par <u>fonction</u> <u>auxiliaire</u></i>	<ul style="list-style-type: none">– mauvaise performance pour des données non-séparable linéairement– très sensible aux conditions initiales (partition, nombre et/ valeur des moyenne)– doit choisir le nombre K de catégories d'avance

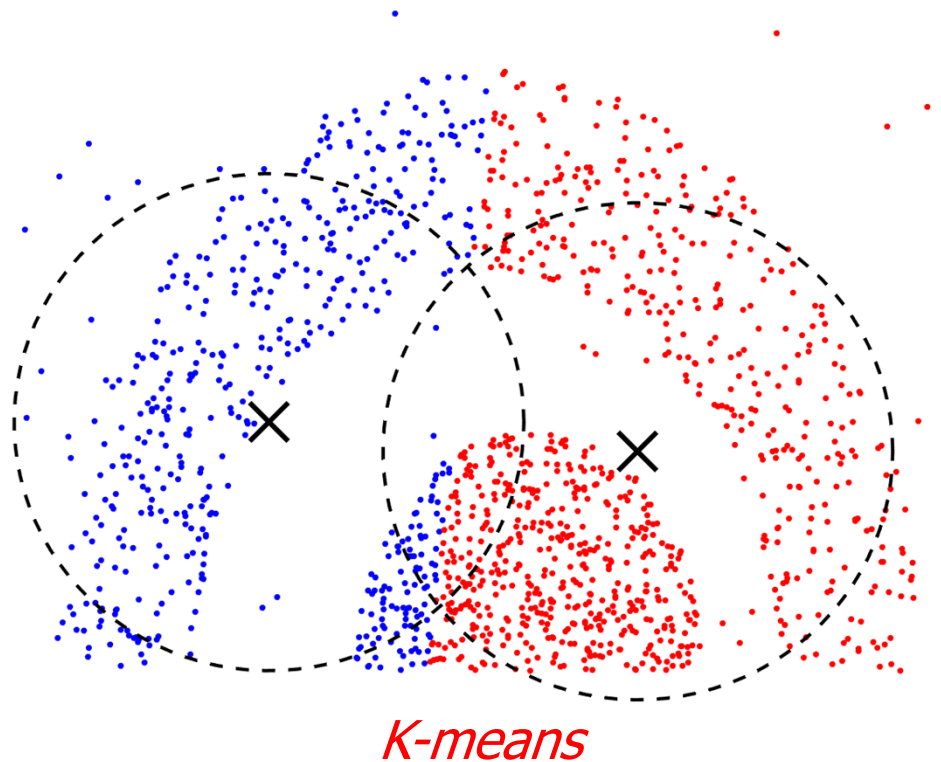
Séparation non-linéaire?

- *K-means ne trouve pas des frontières de séparation non-linéaires entre les groupes (clusters)*



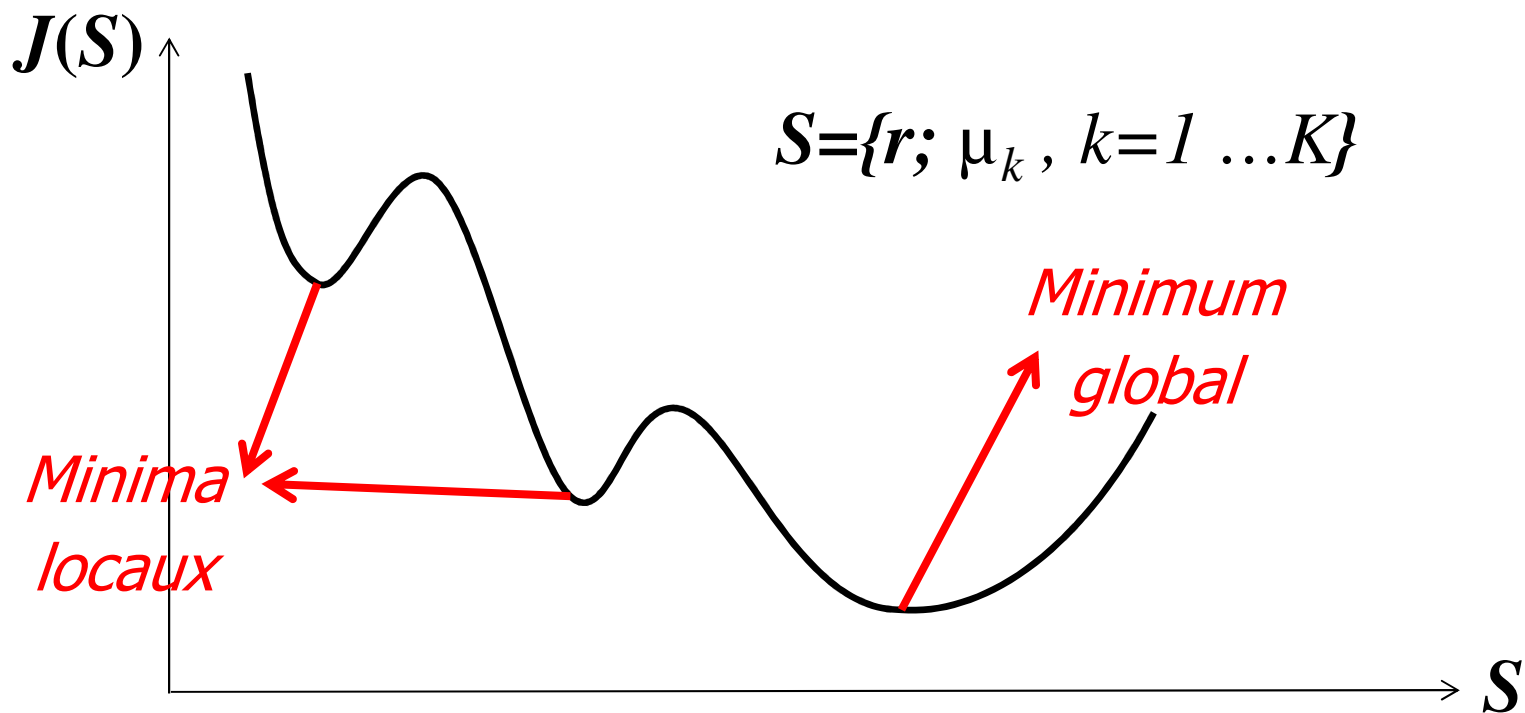
Séparation non-linéaire?

- *K-means ne trouve pas des frontières de séparation non-linéaires entre les groupes (clusters)*



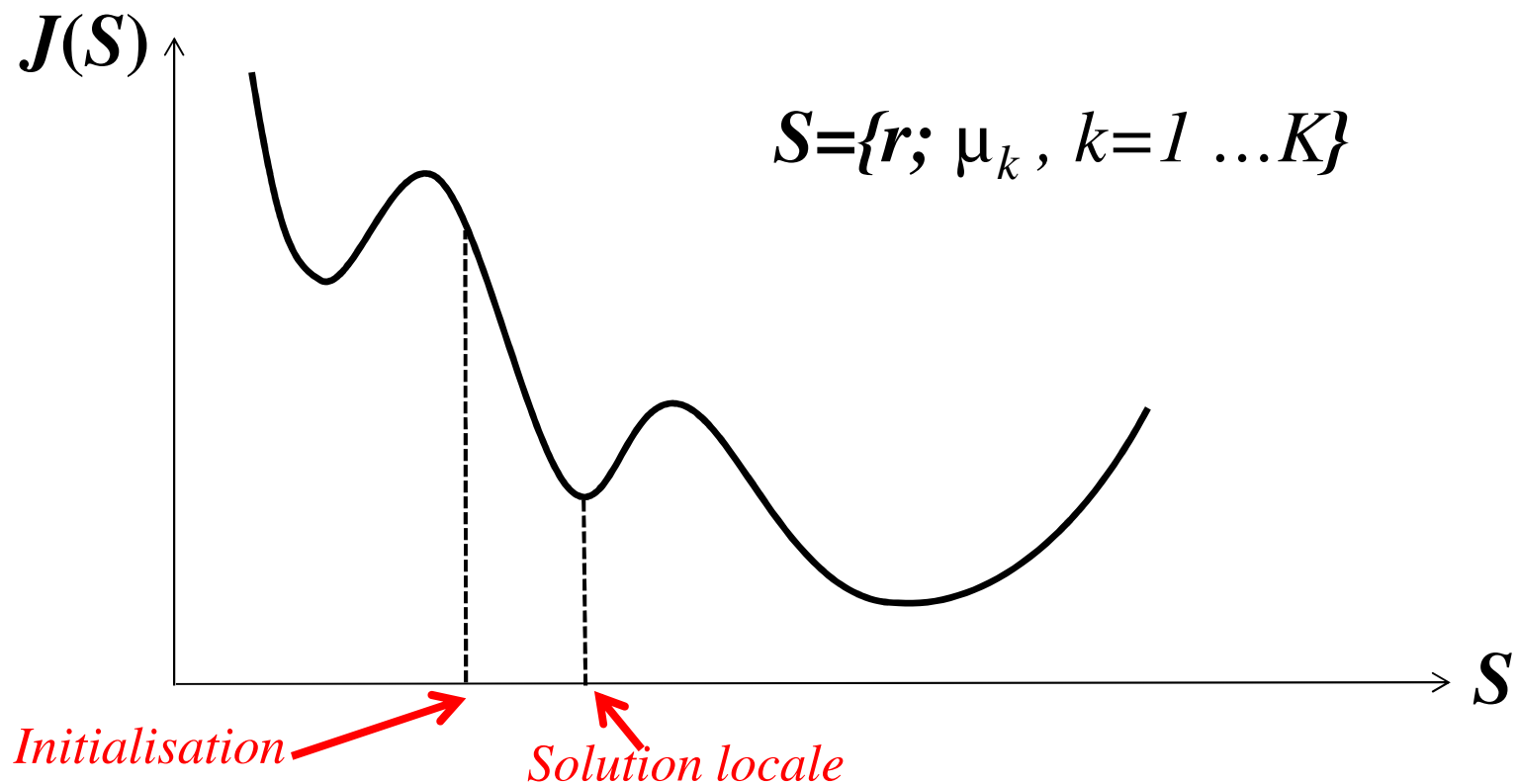
Optimum global?

- *K-means ne garantit pas d'obtenir un minimum global*



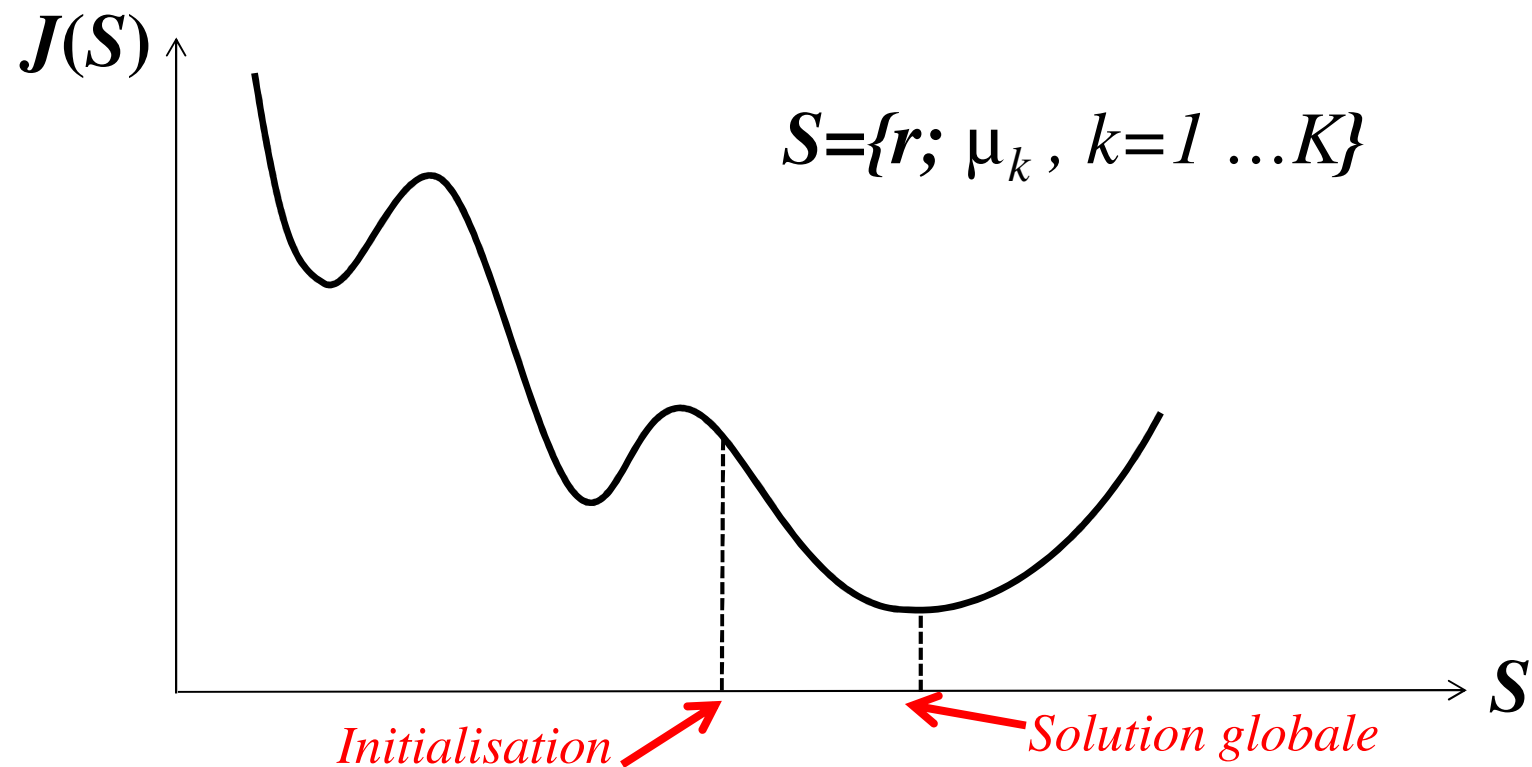
Optimum global?

- *K-means ne garantit pas d'obtenir un minimum global*



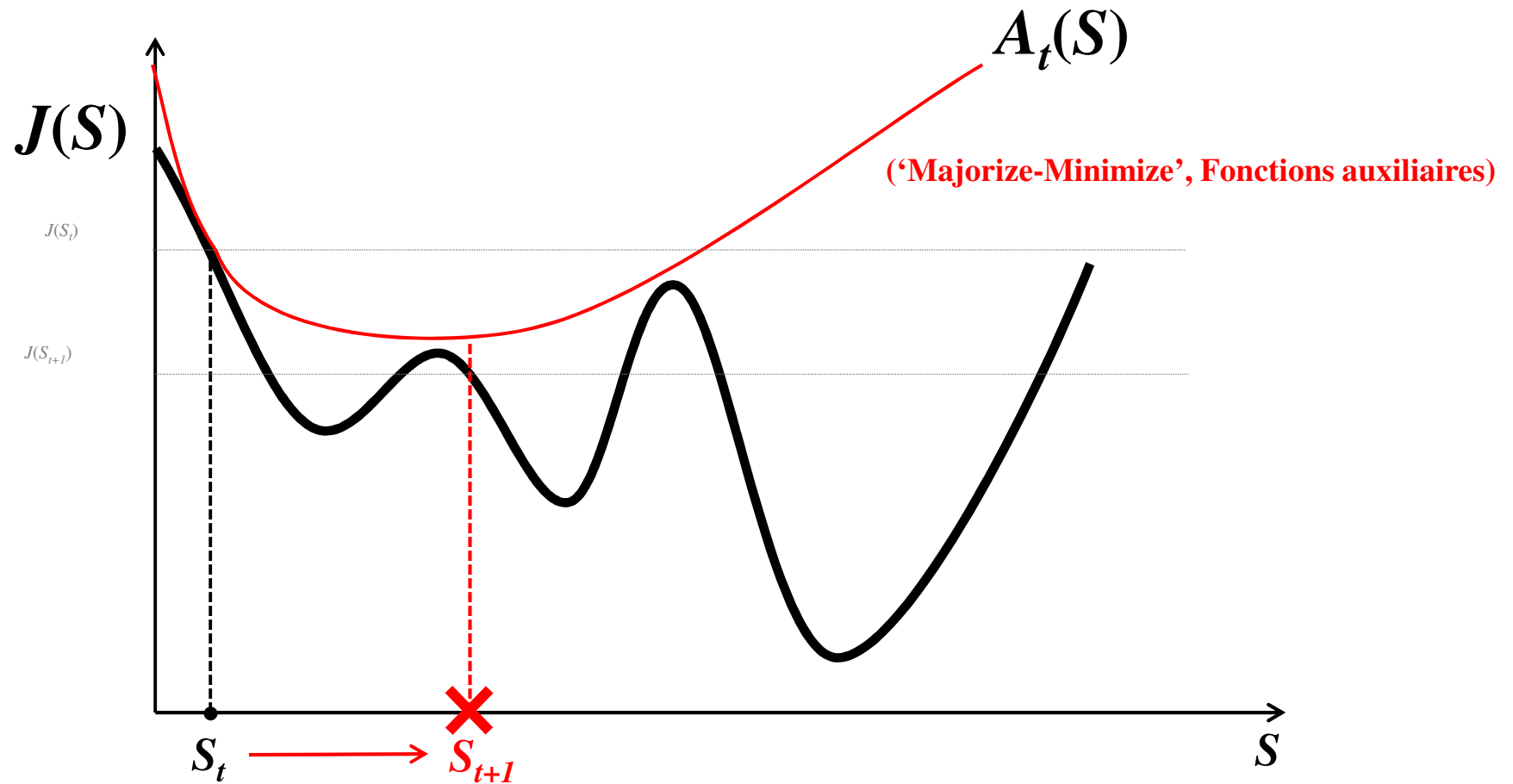
Optimum global?

- *K-means ne garantit pas d'obtenir un minimum global*



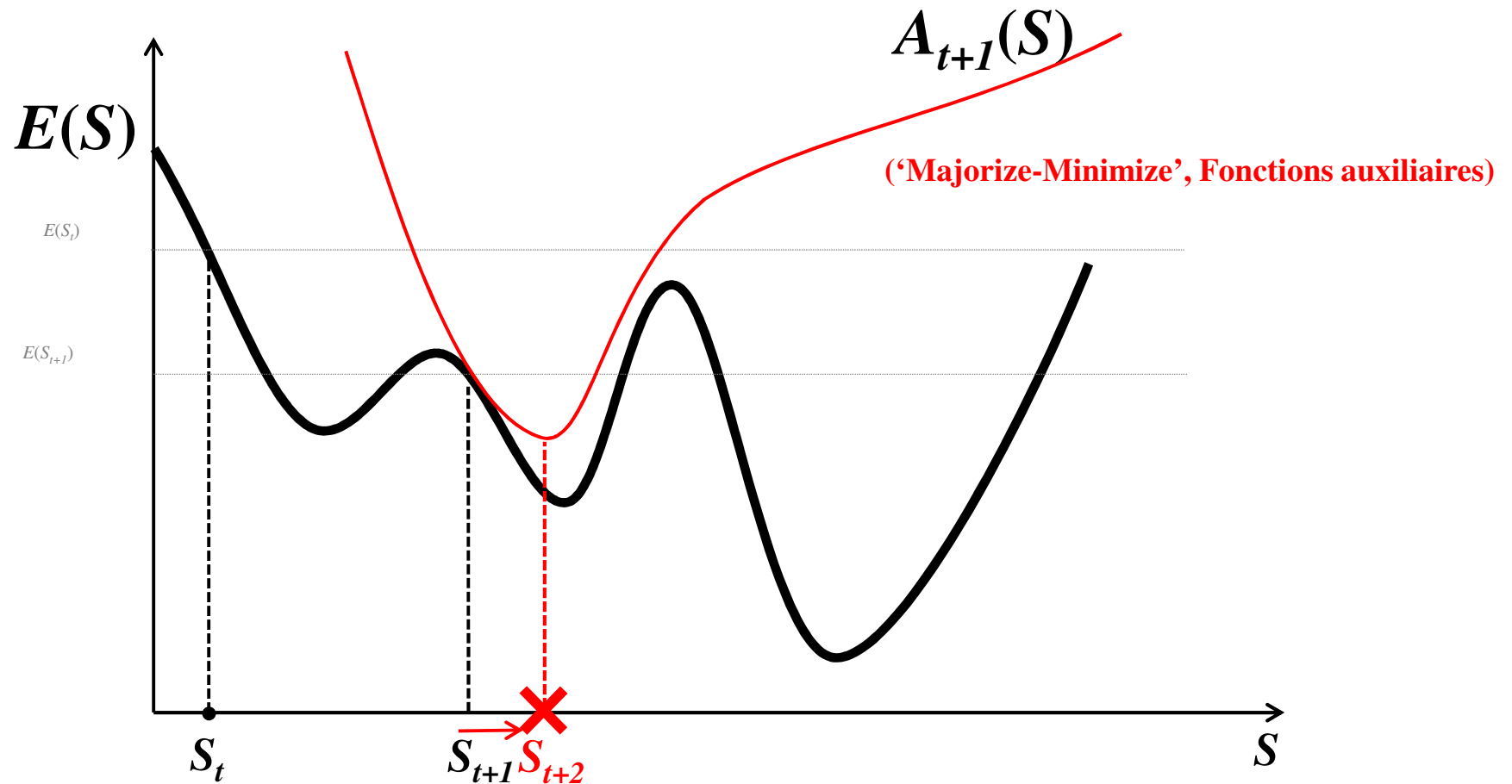
Optimum global?

Pourquoi: Optimisation de fonction auxiliaire



Optimum global?

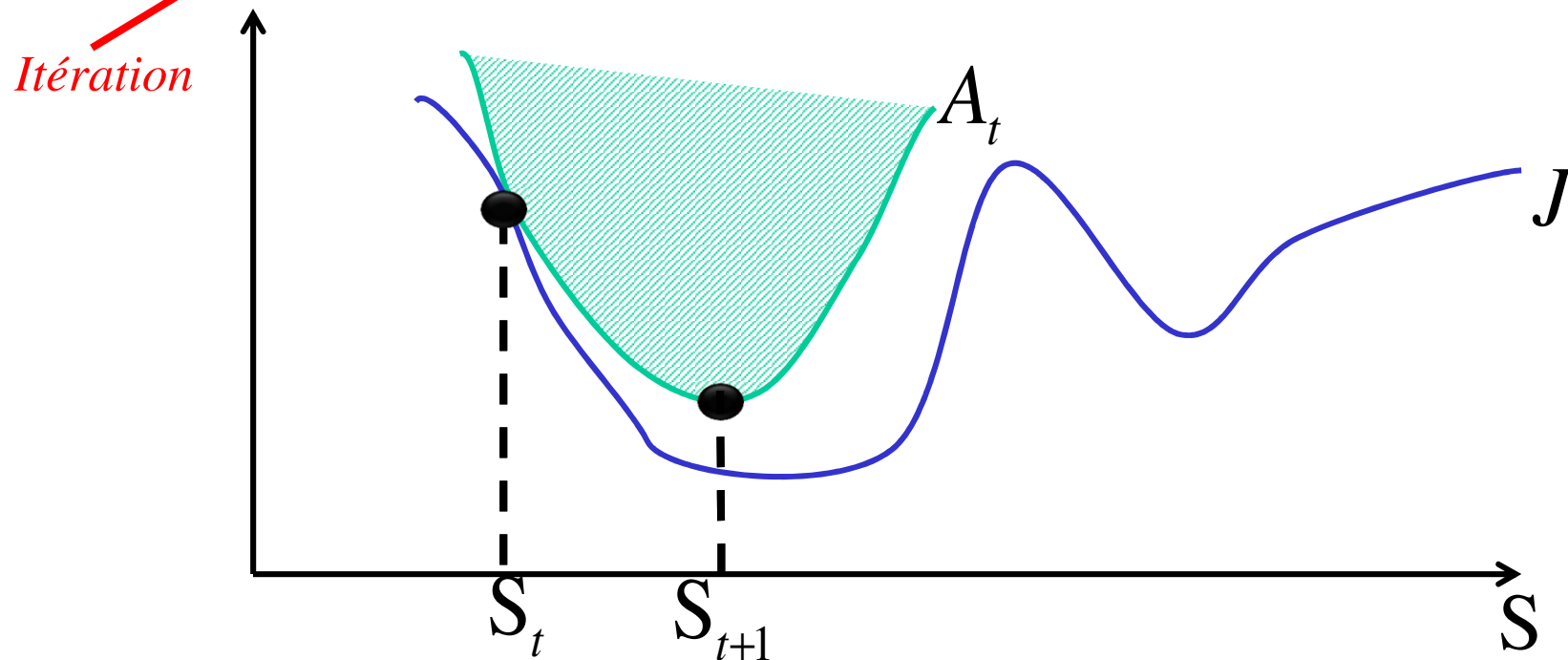
Pourquoi: Optimisation de fonction auxiliaire



Optimum global?

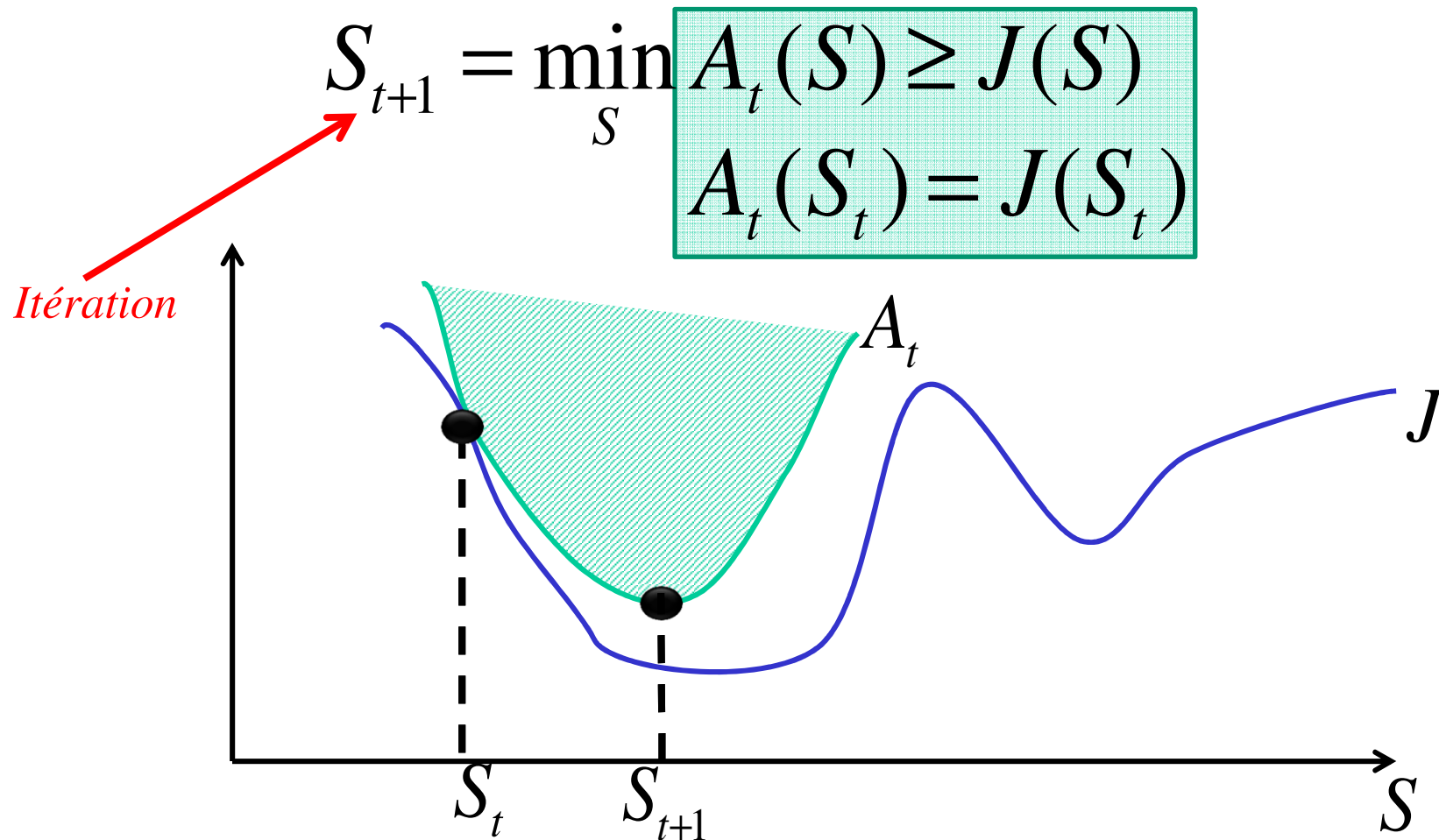
Pourquoi: Optimisation de fonction auxiliaire

$$S_{t+1} = \min_S A_t(S) \geq J(S)$$



Optimum global?

Pourquoi: Optimisation de fonction auxiliaire



Optimum global?

Pourquoi: Optimisation de fonction auxiliaire

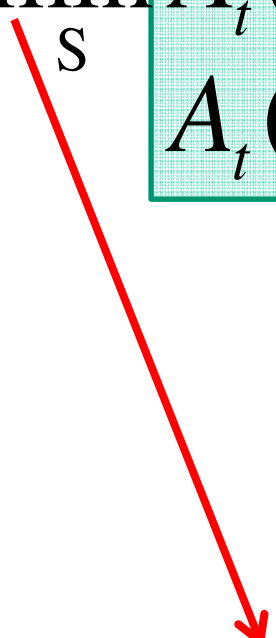
$$S_{t+1} = \min_S A_t(S) \geq J(S)$$
$$A_t(S_t) = J(S_t)$$


$$J(S_{t+1}) \leq A_t(S_{t+1})$$

Optimum global?

Pourquoi: Optimisation de fonction auxiliaire

$$S_{t+1} = \min_S \begin{cases} A_t(S) \geq J(S) \\ A_t(S_t) = J(S_t) \end{cases}$$


$$J(S_{t+1}) \leq A_t(S_{t+1}) \leq A_t(S_t)$$

Optimum global?

Pourquoi: Optimisation de fonction auxiliaire

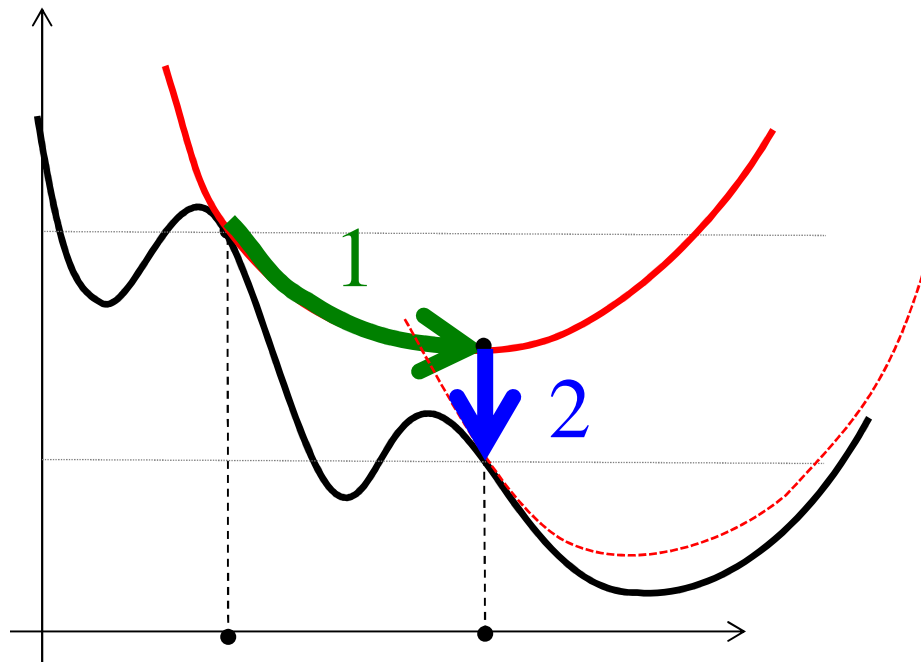
$$S_{t+1} = \min_S \begin{cases} A_t(S) \geq J(S) \\ A_t(S_t) = J(S_t) \end{cases}$$

$$J(S_{t+1}) \leq A_t(S_{t+1}) \leq A_t(S_t) = J(S_t)$$

Fonction coût est diminuée à chaque itération

K-means optimise une fonction aux.?

1. **Partition:** Assigne la catégorie la plus *proche* à chaque
2. **Apprentissage:** Mise-à-jour du prototype de chaque catégorie



[Tang *et al.*, ICCV 2015]

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

- 1) Algorithme classique k -means
- 2) Modèles probabilistes de clustering
- 3) Mélange de Gaussiennes
- 4) Algorithme fuzzy C -means

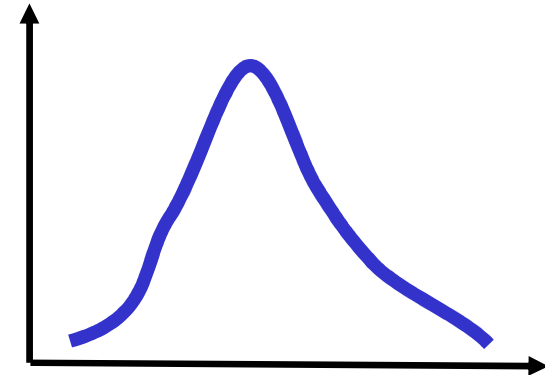
K-means probabiliste

$$J(r, \theta_1, \theta_2, \dots, \theta_k) = - \sum_{k=1}^K \sum_{i=1}^N r_{ik} \log Pr(\mathbf{x}_i | \theta_k)$$

$$= - \sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \log Pr(\mathbf{x}_i | \theta_k)$$

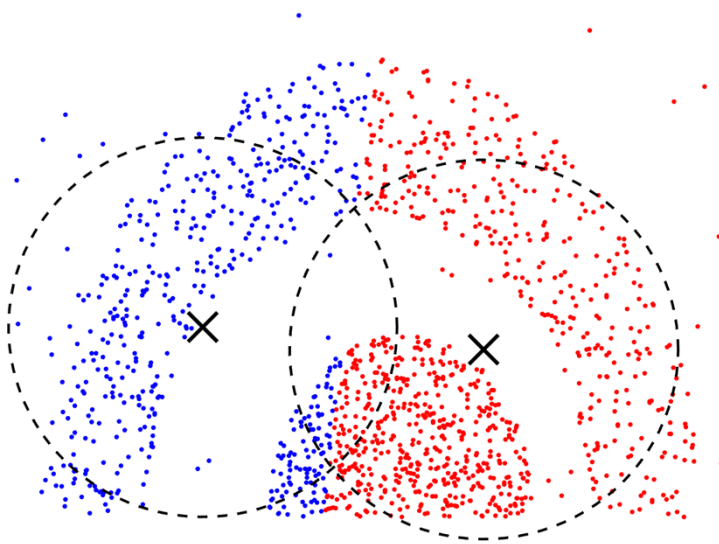
Loi Gaussienne: *K-means* elliptique

$$Pr(\mathbf{x}_i | \mu_k, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{(\mathbf{x}_i - \mu_k)^2}{2\sigma_k^2}}$$

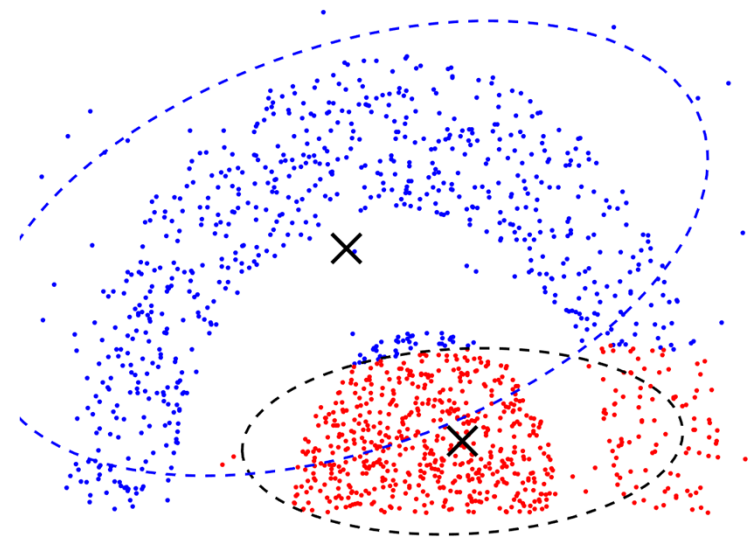


Loi Gaussienne: *K-means* elliptique

K-means probabiliste

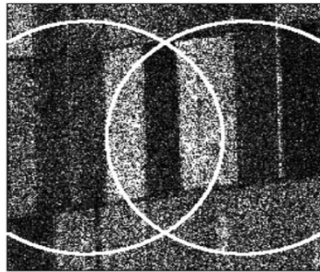


K-means

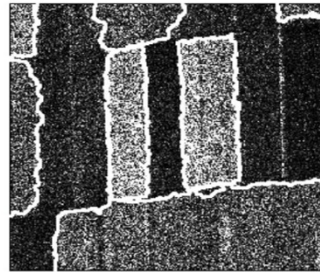


K-means elliptique

K-means probabiliste



(a)



(b)



(c)



(d)



(e)

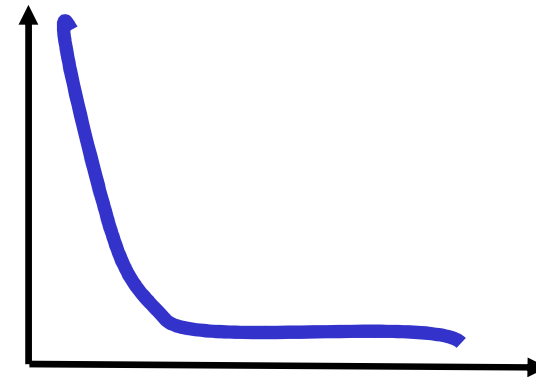


(f)

$$-\sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \log Pr(\mathbf{x}_i | \theta_k)$$

$$Pr(\mathbf{x}_i | \mu_k) = \frac{1}{\mu_k} e^{-\frac{\mathbf{x}_i}{\mu_k}}$$

Loi exponentielle



[Ben Ayed *et al.*, TPAMI 05]

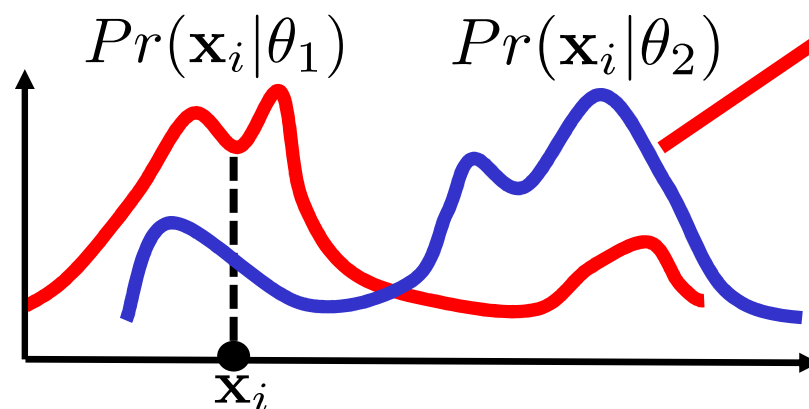
K-means probabiliste

- On peut utiliser des distributions plus complexes pour chaque cluster



$$\log Pr(\mathbf{x}_i | \theta_1) > \log Pr(\mathbf{x}_i | \theta_2) ?$$

*Modèles plus descriptifs
pour les distributions
(Mélange de Gaussiennes ou
histogrammes)*



[Rother et al., SIGGRAPH 04]

Interprétation entropique

$$-\sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \log \text{Pr}(\mathbf{x}_i | \theta_k)$$

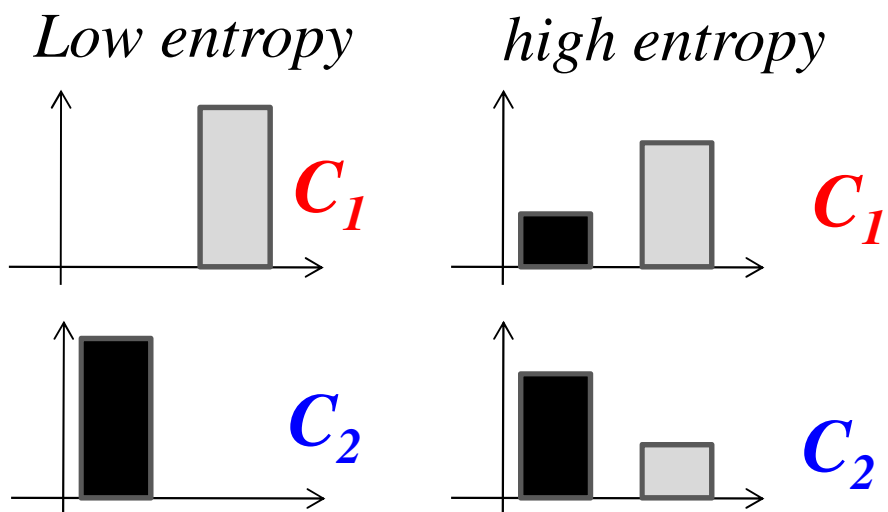
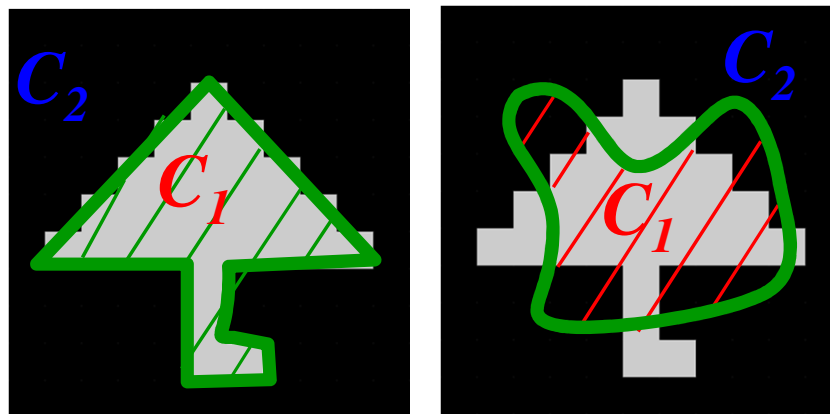
\approx

$$\sum_{k=1}^K |C_k| H(\text{Pr}(\mathbf{x}_i | \theta_k))$$

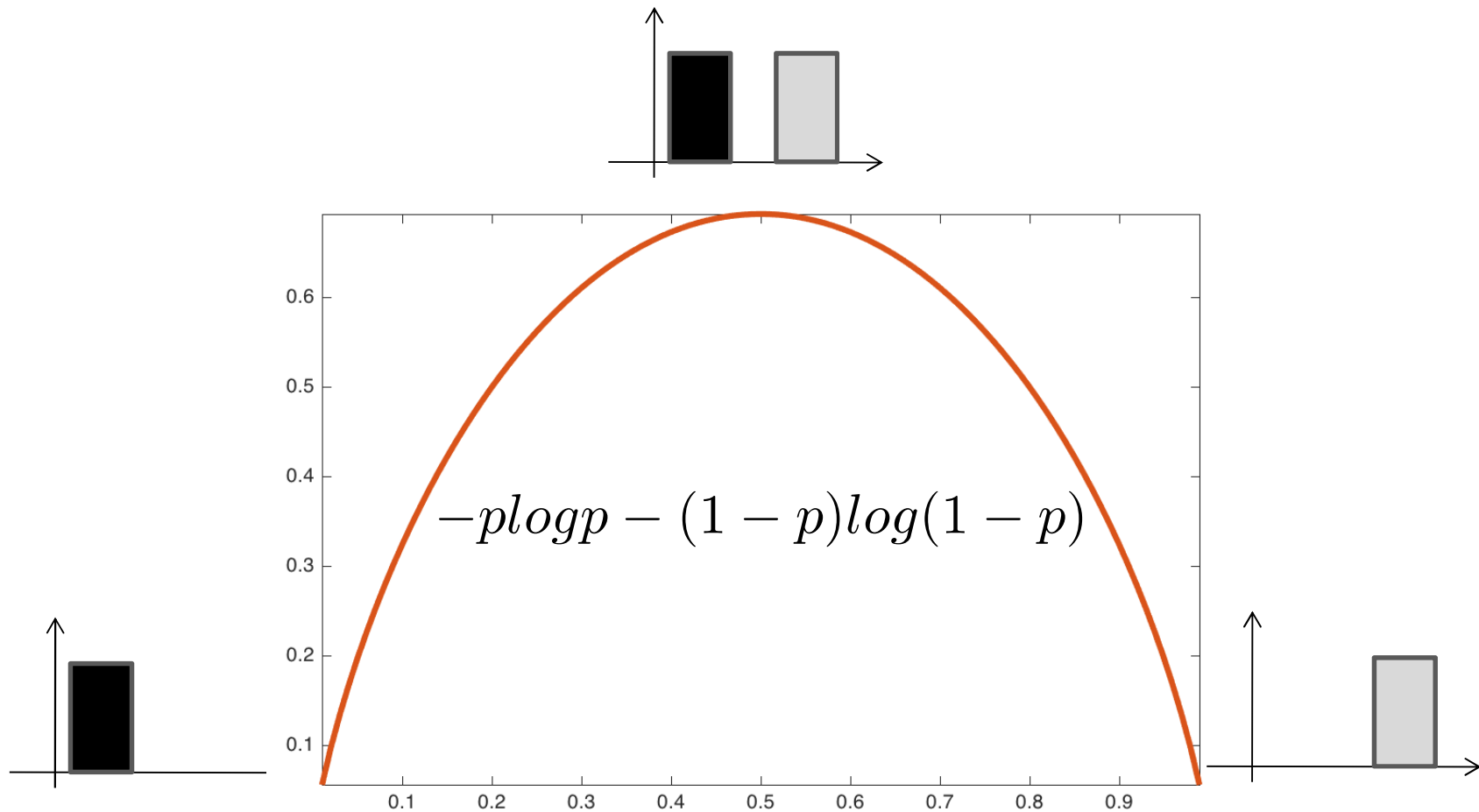
Entropie

[Kearns *et al.*, UAI 97]

Interprétation entropique



Interprétation entropique



K-means probabiliste

Préfère des solutions équilibrées en terme de cardinalité des clusters

$$-\sum_{k=1}^K \sum_{i \in C_k} \log \text{Pr}^k(X_i)$$

[Kearns *et al.*, UAI 97]

$$\sum_{k=1}^K |C^k| \cdot KL(X^k | P^k) + |\Omega| [H(C | X) - H(C)]$$

Entropie de la partition

Attention: différente de l'entropie des données

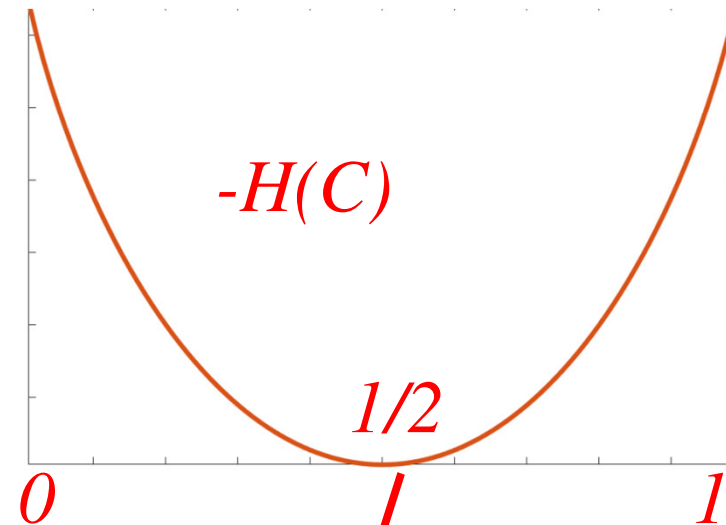
K-means probabiliste

Préfère des solutions équilibrées en terme de cardinalité des clusters

$$H(C) = -\sum_{k=1}^K \frac{|C^k|}{|\Omega|} \log \frac{|C^k|}{|\Omega|}$$



$$-H(C) = KL(C \mid U) = \sum_{k=1}^K \frac{|C^k|}{|\Omega|} \log \frac{|C^k|/|\Omega|}{1/K}$$



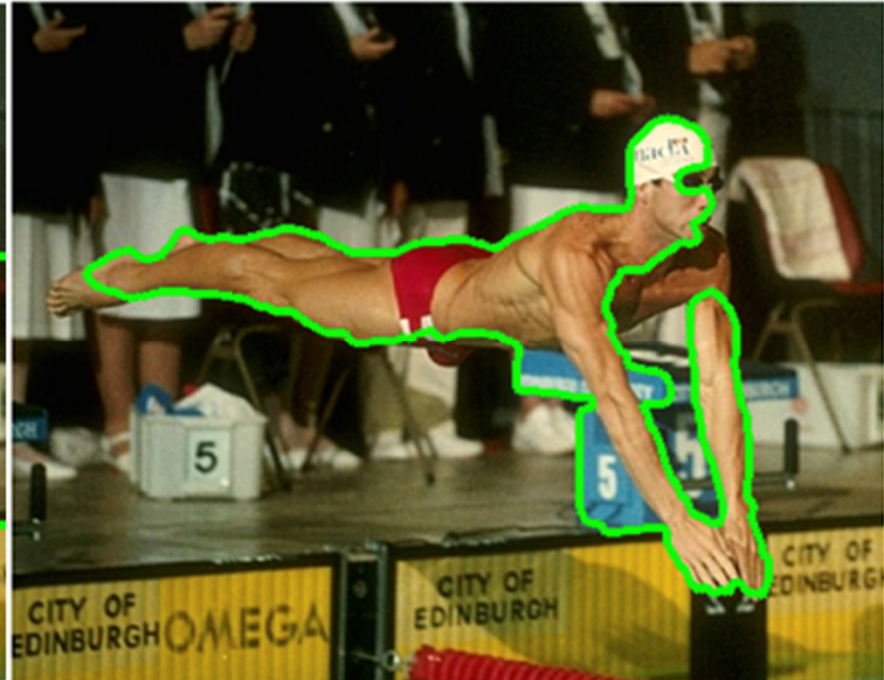
Les deux clusters ont la même cardinalité (proportion)

K-means: Des partitions équilibrée



Biaisé: *K-means*

[Rother *et al.*, SIGGRAPH 04]



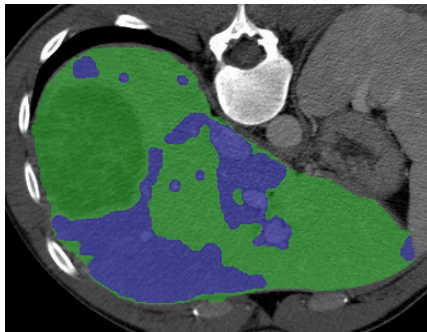
Non Biaisé: *K-means* + $H(C)$

[Boykov *et al.*, ICCV 2015]

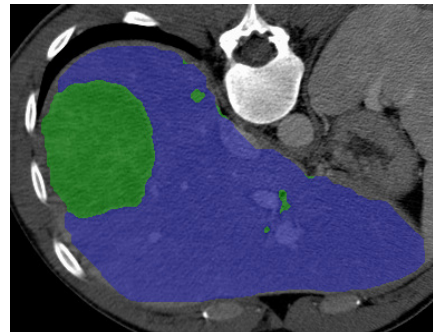
A priori de proportion

$$-\sum_{k=1}^K \sum_{\mathbf{x}_i \in C^k} \log w_k Pr^k(\mathbf{x}_i)$$

[Boykov *et al.*, ICCV 2015]

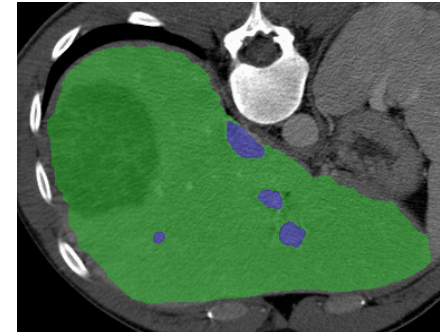


K-means



$$w_1=0.25$$

$$w_2=0.75$$

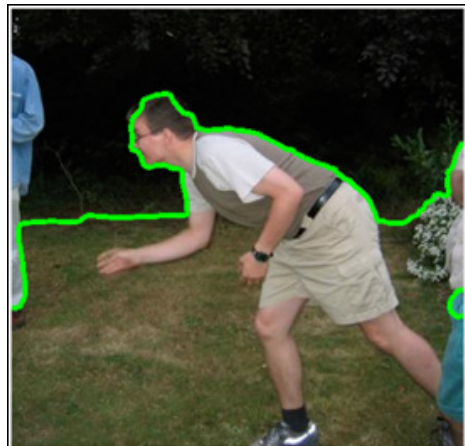
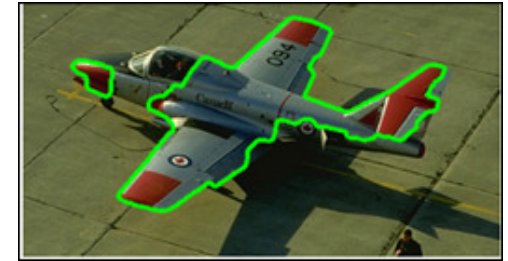
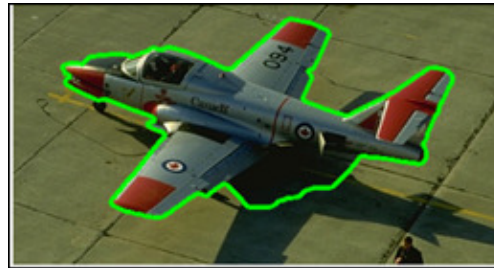


$$w_1=0.96$$

$$w_2=0.04$$

$-H(S) = KL(S|U)$ Remplacée par $\rightarrow KL(C|W) = \sum_{k=1}^K \frac{|C^k|}{|\Omega|} \log \frac{|C^k|/|\Omega|}{w_k}$

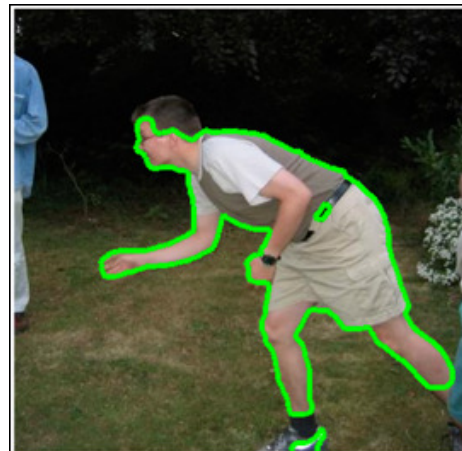
Exemples: A priori/non biaisé



K-means

Erreur: 14.41%

[Rother *et al.*, SIGGRAPH 04]



A priori

Erreur: 4.75%

[Boykov *et al.*, ICCV 2015]

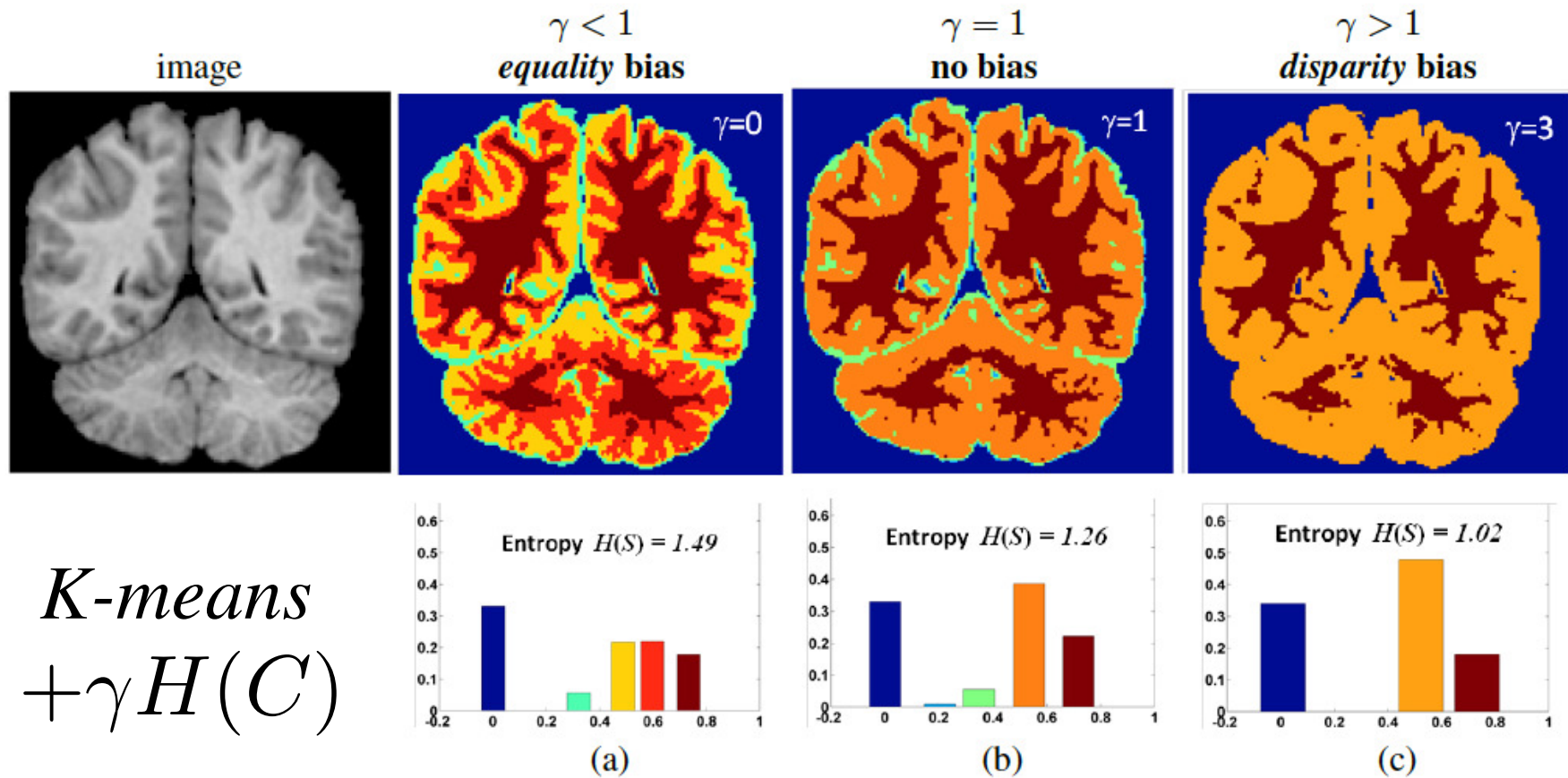


Non-biaisé

Erreur: 7.87%

[Boykov *et al.*, ICCV 2015]

Exemples



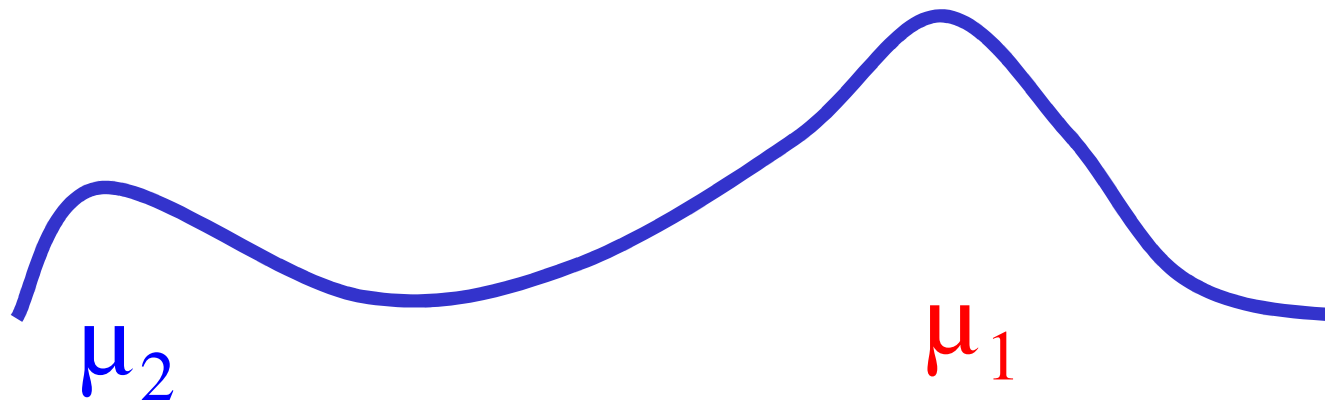
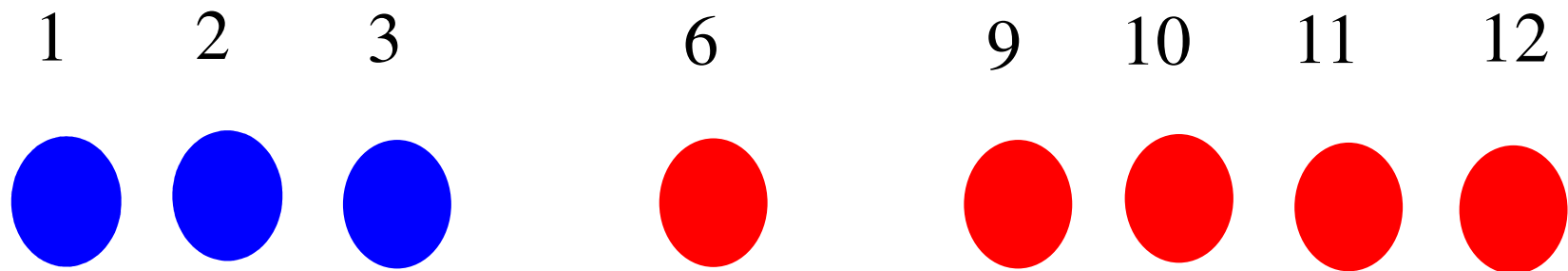
[Boykov *et al.*, ICCV 2015]

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

- 1) Algorithme classique k -means
- 2) Modèles probabilistes de clustering
- 3) Mélange de Gaussiennes
- 4) Algorithme fuzzy C -means

Mélange de Gaussiennes



Approche générative

Mélange de Gaussiennes

► Hypothèses pour l'apprentissage:

- soit $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n\}$ une séquence d'échantillons aléatoires (v.a.) pigé à partir d'une distribution de probabilité inconnue $p(\mathbf{X})$
- *supposons* un modèle probabiliste de la distribution sur les variables dans \mathbf{X} (paramétrisé par θ)
 - les données sont pigées indépendamment à partir d'un mélange de densités:
$$p(\mathbf{x} | \theta) = \sum_{k=1}^K p(k) \cdot p(\mathbf{x} | k, \theta_k)$$
- *nous disposons* d'une instance particulière de \mathbf{X} , soit la base de données $\mathbf{D}_n = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, et $\mathbf{x}_i \in \mathcal{R}^d$ pour $i = 1, 2, \dots, n$

Mélange de Gaussiennes

► Mélange de Gaussiennes (GMM):

- un GMM est un modèle probabiliste de $p(\mathbf{X})$
- la vraisemblance de \mathbf{x} étant donné *une* distribution Gaussienne:

$$p(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\boldsymbol{\Sigma}|}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

- la vraisemblance de \mathbf{x} étant donné un mélange de Gaussiennes:

$$p(\mathbf{x}) = \sum_{k=1}^K w_k \cdot p(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \Leftrightarrow p(\mathbf{x}) = \sum_{k=1}^K P(k) \cdot p(\mathbf{x} | k)$$

où K est le nombre de Gaussiennes et w_k est le poids associé à la Gaussienne k , avec:

$$\sum_k w_k = 1, \quad w_k \geq 0$$

Mélange de Gaussiennes

► Propriétés du GMM:

- les GMM sont connus comme des *approximateurs universels de densités*
- on utilise souvent une matrice de covariance Σ diagonale car le nombre de paramètres à optimiser est plus gérable
 - Gaussiennes sphériques: $\Sigma = \mathbf{I}\sigma^2$
- on entraîne les GMM avec **Expectation-Maximisation (E-M)**
 - *un algorithme efficace de type maximum de vraisemblance (ML) pour optimiser les paramètres d'un modèle probabiliste*

Mélange de Gaussiennes

Algorithme E-M

► Objectif général de E-M:

- étant donnée le modèle paramétrisé par θ , on cherche à maximiser la vraisemblance $p(\mathbf{X}|\theta)$ pour des données \mathbf{D}_n :

$$\theta^* = \arg \max_{\theta} \{ p(\mathbf{X} | \theta) \} = \arg \max_{\theta} \left\{ \prod_{i=1}^n p(\mathbf{x}_i | \theta) \right\}$$

où $\theta = (\theta_1, \theta_2, \dots, \theta_K)$

Mélange de Gaussiennes

Algorithme E-M

► Stratégie de E-M:

1. introduire une *variable cachée* Q telle que sa connaissance permet de simplifier la maximisation de $p(\mathbf{X}|\theta) \rightarrow p(\mathbf{X}, Q|\theta)$
2. appliquer le processus *itératif* suivant, qui s'arrête quand aucune amélioration est possible:
 - **phase E**: étant donnée la valeur actuelle des paramètres et \mathbf{D}_n , estimer la distribution de Q
 - **phase M**: modifier les paramètres afin de maximiser la distribution jointe de \mathbf{D}_n et Q

Mélange de Gaussiennes

E-M pour les GMMs

- ▶ **Variable cachée Q** – pour un GMM, Q décrit l'appartenance de chaque patron aux Gaussiennes

- **remarque:** si Q serait observée pour chaque patron, maximiser la vraisemblance des données serait trivial
- soit la vraisemblance d'un GMM pour un patron \mathbf{x}_i :

$$p(\mathbf{x}_i | \theta) = \sum_{k=1}^K P(k | \theta_k) \cdot p(\mathbf{x}_i | k, \theta_k) \quad \text{pour } i = 1, 2, \dots, n$$

- et la variable indicatrice $q_{i,k} = \begin{cases} 1 & \text{si Gaussienne } k \text{ émet } \mathbf{x}_i \\ 0 & \text{autrement} \end{cases}$

Mélange de Gaussiennes

E-M pour les GMMs

- Puisqu'on introduit une v.a. cachée Q , on a la probabilité jointe de \mathbf{X} et Q :

$$p(\mathbf{X}, \mathbf{Q} | \theta) = \prod_{i=1}^n \prod_{k=1}^K P(k | \theta)^{q_{i,k}} \cdot p(\mathbf{x}_i | k, \theta)^{q_{i,k}}$$

$$\log p(\mathbf{X}, \mathbf{Q} | \theta) = \sum_{i=1}^n \sum_{k=1}^K q_{i,k} \cdot \log P(k | \theta) + q_{i,k} \cdot \log p(\mathbf{x}_i | k, \theta)$$

- **Fonction de coût:** avec E-M, on veut maximiser la vraisemblance totale sur \mathbf{D}_n

$$E_Q[\log p(\mathbf{X}, \mathbf{Q} | \theta) | \mathbf{D}_n, \theta_{t-1}]$$

Mélange de Gaussiennes

E-M pour les GMMs

- **Phase E** – estimation de la probabilité a posteriori de la catégorie k (responsabilités) pour chaque \mathbf{x}_i :

$$\begin{aligned} P(k | \mathbf{x}_i, \theta_{t-1}) &= \frac{P(k | \theta_{t-1}) p(\mathbf{x}_i | k, \theta_{t-1})}{p(\mathbf{x}_i | \theta_{t-1})} \\ &= \frac{P(k | \theta_{t-1}) p(\mathbf{x}_i | k, \theta_{t-1})}{\sum_{h=1}^K P(h | \theta_{t-1}) p(\mathbf{x}_i | h, \theta_{t-1})} \end{aligned}$$

- **Phase M** – trouver les paramètres qui maximisent:

$$E_Q[\log p(\mathbf{X}, \mathbf{Q} | \theta) | \mathbf{D}_n, \theta_{t-1}]$$

Mélange de Gaussiennes

E-M pour les GMMs

► Phase M:

1. moyenne

$$\hat{\mu}_k = \frac{\sum_{i=1}^n P(k | \mathbf{x}_i, \theta_{t-1}) \cdot \mathbf{x}_i}{\sum_{i=1}^n P(k | \mathbf{x}_i, \theta_{t-1})}$$

*Pondérée par
L'a posteriori*

2. variance:

$$\hat{\sigma}_k^2 = \frac{\sum_{i=1}^n P(k | \mathbf{x}_i, \theta_{t-1}) (\mathbf{x}_i - \hat{\mu}_k)^2}{\sum_{i=1}^n P(k | \mathbf{x}_i, \theta_{t-1})}$$

*Multiplicateur de
Lagrange*

3. poids:

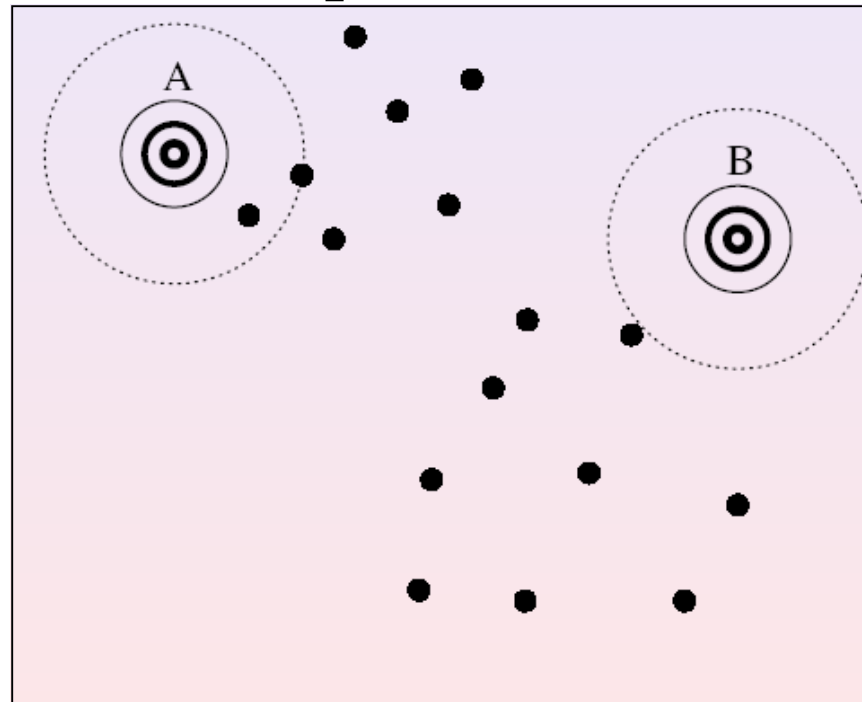
$$\hat{w}_k = \frac{1}{n} \sum_{i=1}^n P(k | \mathbf{x}_i, \theta_{t-1})$$

*(contrainte somme des
poids égale à 1)*

Mélange de Gaussiennes

E-M pour les GMMs

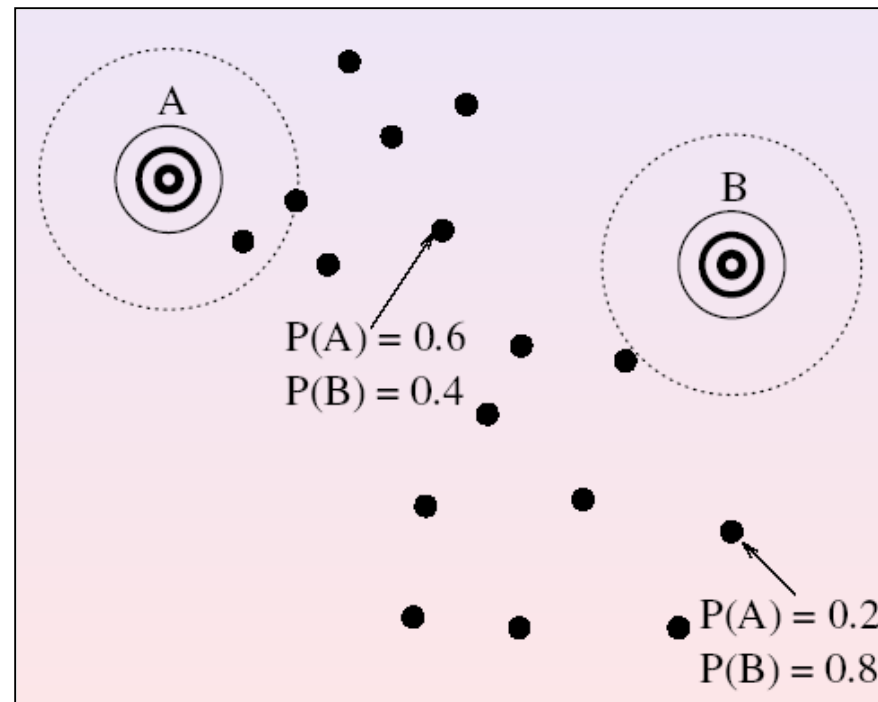
- **Variable cachée** \equiv étiquette de la Gaussienne qui a générée chacun des patrons



Mélange de Gaussiennes

E-M pour les GMMs

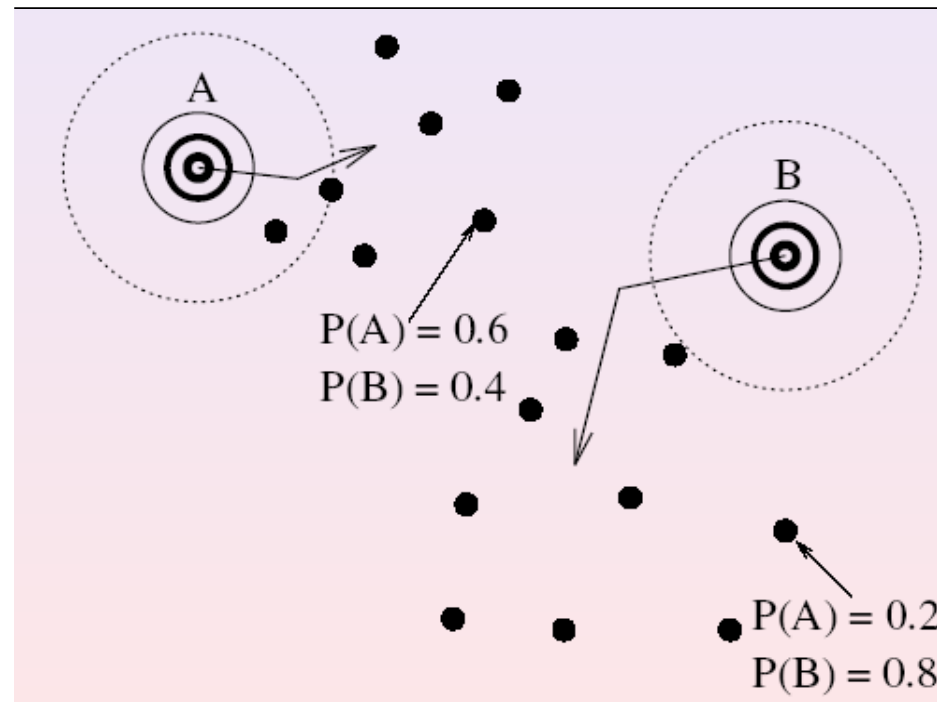
- **étape E:** pour chaque patron, estimez la probabilité qu'il a été généré par chacune des Gaussiennes



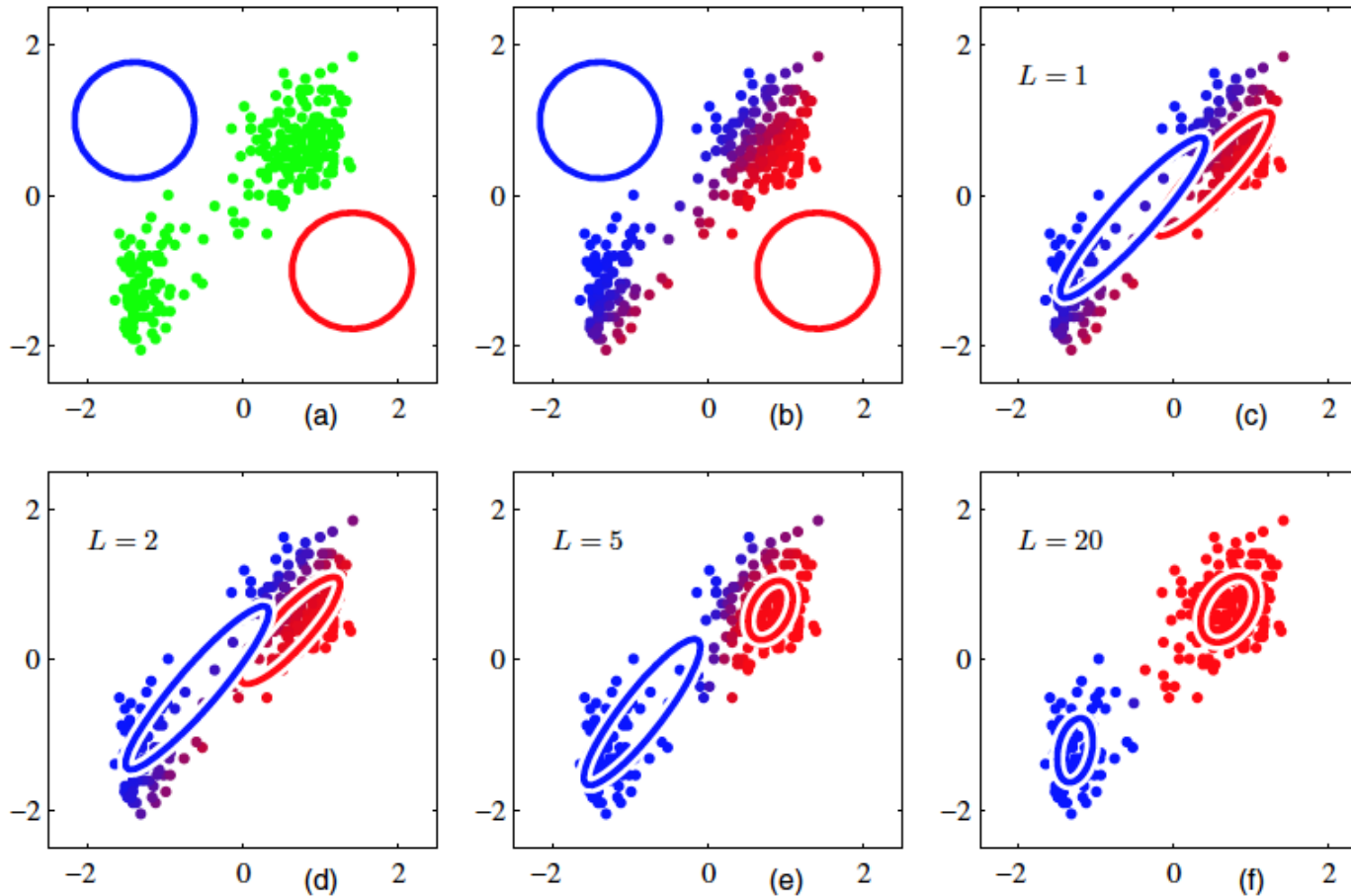
Mélange de Gaussiennes

E-M pour les GMMs

- **étape M:** modifier les paramètres selon la variable cachée, afin de maximiser la vraisemblance des données



Mélange de Gaussiennes



Mélange de Gaussiennes

E-M pour les GMMs

Avantages	Inconvénients
<ul style="list-style-type: none">+ meilleure performance pour des données avec 2+ classes qui se chevauchement+ représentation plus explicite et forte des catégories+ converge toujours vers un minimum (local ou global)	<ul style="list-style-type: none">+ coût computationnelle: l'effort de calcul par itération, et le nombre d'itérations pour converger est énorme+ processus itératif qui est très sensible aux conditions initiales+ converge plus lentement que k-means+ génératif: demande beaucoup de données de référence

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

- 1) Algorithme classique k -means
- 2) Modèles probabilistes de clustering
- 3) Mélange de Gaussiennes
- 4) Algorithme fuzzy C-means

Fuzzy C-Means

Forme générale de la fonction de coût (variables)

$$J_m(\mathbf{U}, \mathbf{v}) = \sum_{k=1}^n \sum_{i=1}^c (\mu_{ik})^m (d_{ik})^2$$

*Vecteur des prototypes
(paramètres des clusters)*

Matrice de partition floue

*Degré d'appartenance du
point k à la catégorie i*

*Mesure de distance entre
point k et la catégorie i*

Fuzzy C-Means

Forme générale de la fonction de coût (Paramètres fixes du modèle)

(Nombre de points à classer) *(Nombre de clusters)*

$$J_m(\mathbf{U}, \mathbf{v}) = \sum_{k=1}^n \sum_{i=1}^c (\mu_{ik})^m (d_{ik})^2$$

Facteur de flou

Fuzzy C-Means

Forme générale de la fonction de coût (Optimisation sous contrainte)

$$\sum_{i=1}^c \mu_{ik} = 1$$

*Avec cette contrainte
qu'il ne faut pas oublier!*

Fuzzy C-Means

La solution optimale est obtenue en minimisant la fonction de coût

- Considérons les ensembles suivants

$$I_k = \{i, \quad 1 \leq i \leq c \quad | \quad d_{ik} = \|\mathbf{x}_k - \mathbf{v}_i\| = 0\} \ ;$$

$$J_k = \{1, 2, \dots, c\} - I_k \ .$$

Fuzzy C-Means

La solution optimale est obtenue en minimisant la fonction de coût

1. Mise à jour des variables d'appartenance flou (similaire à la mise à jour de la partition dans *K-means*)

Premier cas

► $I_k = \emptyset$, alors :

$$\mu_{ik} = \frac{1}{\sum_{j=1}^c \left(\frac{d_{ik}}{d_{jk}} \right)^{\frac{2}{m-1}}} \equiv \frac{\left(\frac{1}{d_{ik}} \right)^{\frac{2}{m-1}}}{\sum_{j=1}^c \left(\frac{1}{d_{jk}} \right)^{\frac{2}{m-1}}}$$

Fuzzy C-Means

La solution optimale est obtenue en minimisant la fonction de coût

1. Mise à jour des variables d'appartenance flou (similaire à la mise à jour de la partition dans *K-means*)

Deuxième cas

$I_k \neq \emptyset$, alors :

$$\mu_{ik} = 0 \quad \forall i \in J_k,$$

$$\text{et} \quad \sum_{i \in I_k} \mu_{ik} = 1,$$

Fuzzy C-Means

La solution optimale est obtenue en minimisant la fonction de coût

2. Mise à jour des prototypes (ou paramètres) des clusters (similaire à la mise à jour de la partition dans *K-means*)

Moyennes pondérées

Cardinalité floue

$$\mathbf{v}_i = \frac{\sum_{k=1}^n (\mu_{ik})^m \mathbf{x}_k}{\sum_{k=1}^n (\mu_{ik})^m}, \quad \forall i.$$

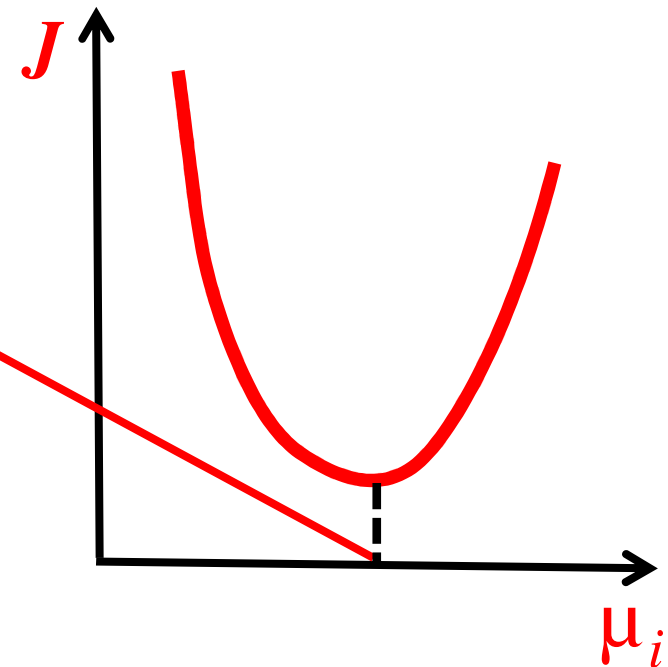
Rappel: Paramètres dans *K-means*

Pour voir le lien: voici la mise à jour
des prototypes qu'on a vue dans *K-means*

$$\frac{\partial J}{\partial \mu_i} = 0$$

$$\frac{\sum_{k=1}^N r_{ik} \mathbf{X}_k}{\sum_{k=1}^N r_{ik}}$$

Moyenne dans groupe C_k



Fuzzy *C-means*

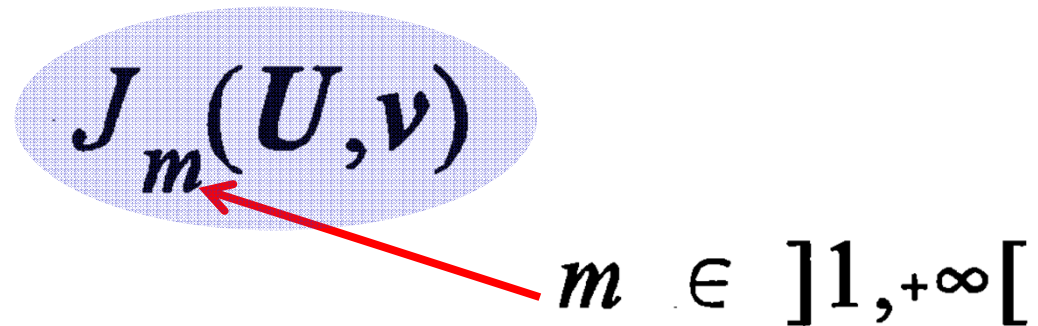
Processus itératif à deux étapes

Initialise K prototypes ou la fonction d'appartenance

1. **Partition flou:** Mise à jour des fonctions d'appartenance flous
2. **Apprentissage:** Mise-à-jour du prototype (moyennes floues) de chaque catégorie
3. **Critère d'arrêt:** Si la fonction coût change, revenir à l'étape 2

Hard *C-means*

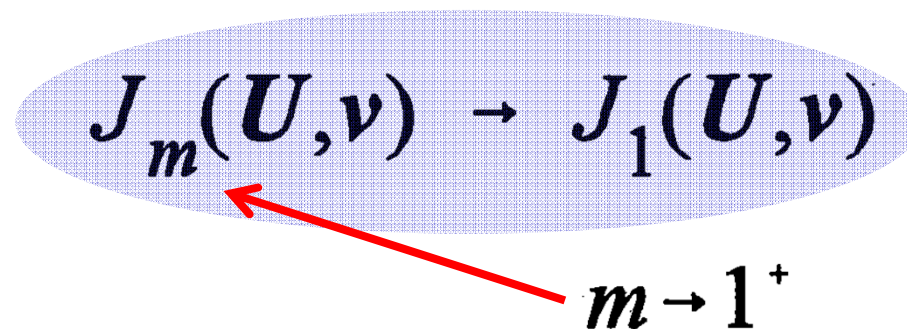
- *Fuzzy C-means*:



A blue oval containing the expression $J_m(U, v)$. A red arrow points from the subscript m to the text $m \in]1, +\infty[$.

$$J_m(U, v)$$
$$m \in]1, +\infty[$$

- *Hard C-means*:



A blue oval containing the expression $J_m(U, v) \rightarrow J_1(U, v)$. A red arrow points from the subscript m to the text $m \rightarrow 1^+$.

$$J_m(U, v) \rightarrow J_1(U, v)$$
$$m \rightarrow 1^+$$

Hard *C-means*

Processus itératif à deux étapes

Initialise K prototypes ou la fonction d'appartenance

1. Mise à jour de la partition

$$\mu_{ik}^{(t)} = \begin{cases} 1; & d_{ik}^{(t)} = \min_{1 \leq j \leq c} \{d_{jk}^{(t)}\} \\ 0; & \text{Autrement} \end{cases}$$

2. Apprentissage: Mise-à-jour du prototype de chaque catégorie

$$v_i = \frac{\sum_{k=1}^n \mu_{ik} x_k}{\sum_{k=1}^n \mu_{ik}}, \quad \forall i.$$

← Moyenne

3. Critère d'arrêt: Si la fonction coût change, revenir à l'étape 2

Fuzzy *C-means*

Rôle du paramètre m

- Contrôle le degré de flou (“amount of fuzziness”)

$$\begin{aligned} m &\rightarrow \infty, \mu_{ik}^{(t)} \rightarrow \frac{1}{c}; \\ m &\rightarrow 1^+, \mu_{ik}^{(t)} \rightarrow 1 \text{ ou } 0 \end{aligned}$$

- La performance du *fuzzy C-means* est évidemment tributaire du choix du paramètre m .