

SYS843

Réseaux de neurones et

systèmes flous

Séance 10

Eric Granger

Ismail Ben Ayed

Hiver 2017

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

1) Catégorisation à noyau et spectrale

- Décalage de moyenne ('*Mean-Shift*')
- K-means basé sur les noyaux (*Kernel K-means*)
- Formulations basées sur les affinités de points
- Analyse spectrale

2) Réseaux auto-organisateurs

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

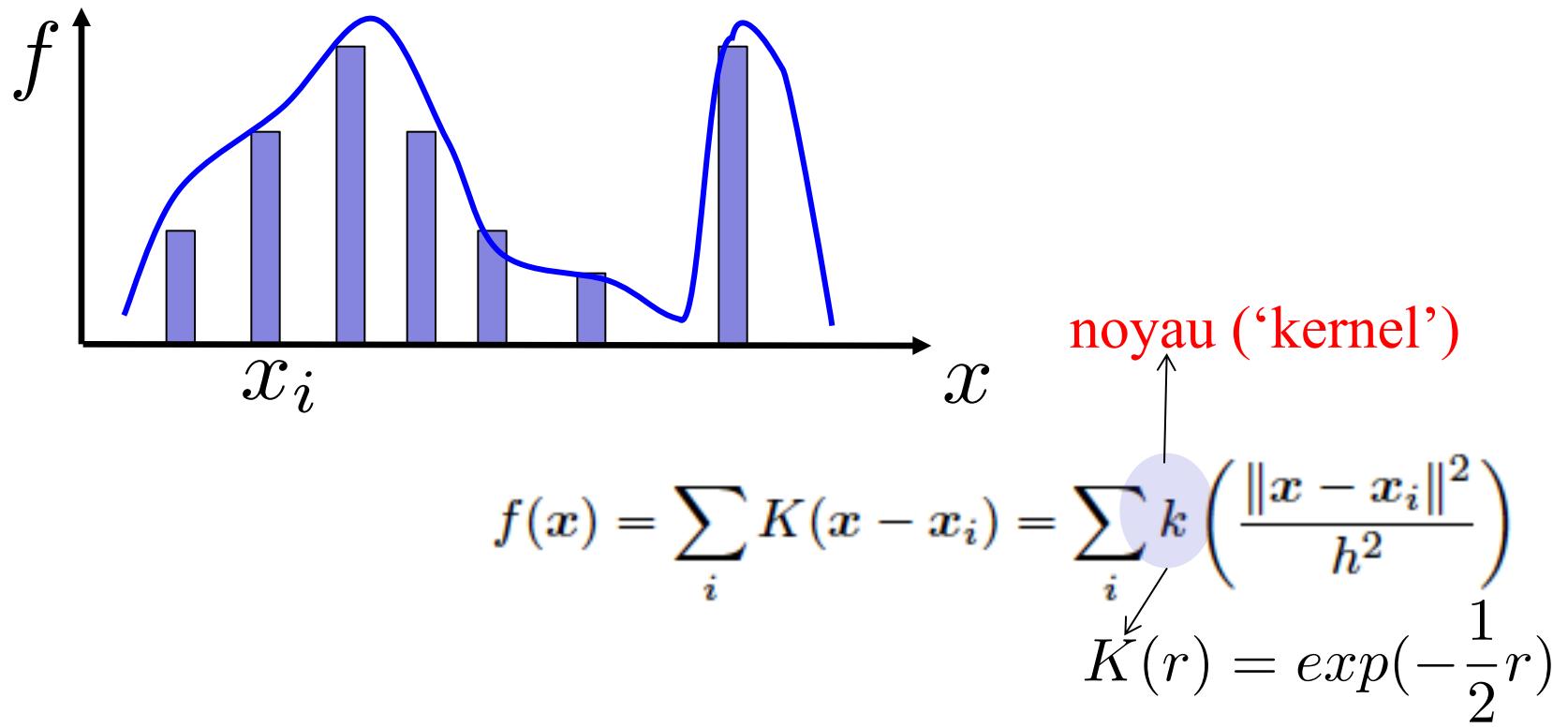
1) Catégorisation à noyau et spectrale

- Décalage de moyenne ('*Mean-Shift*')
- K-means basé sur les noyaux (*Kernel K-means*)
- Formulations basées sur les affinités de points
- Analyse spectrale

2) Réseaux auto-organisateurs

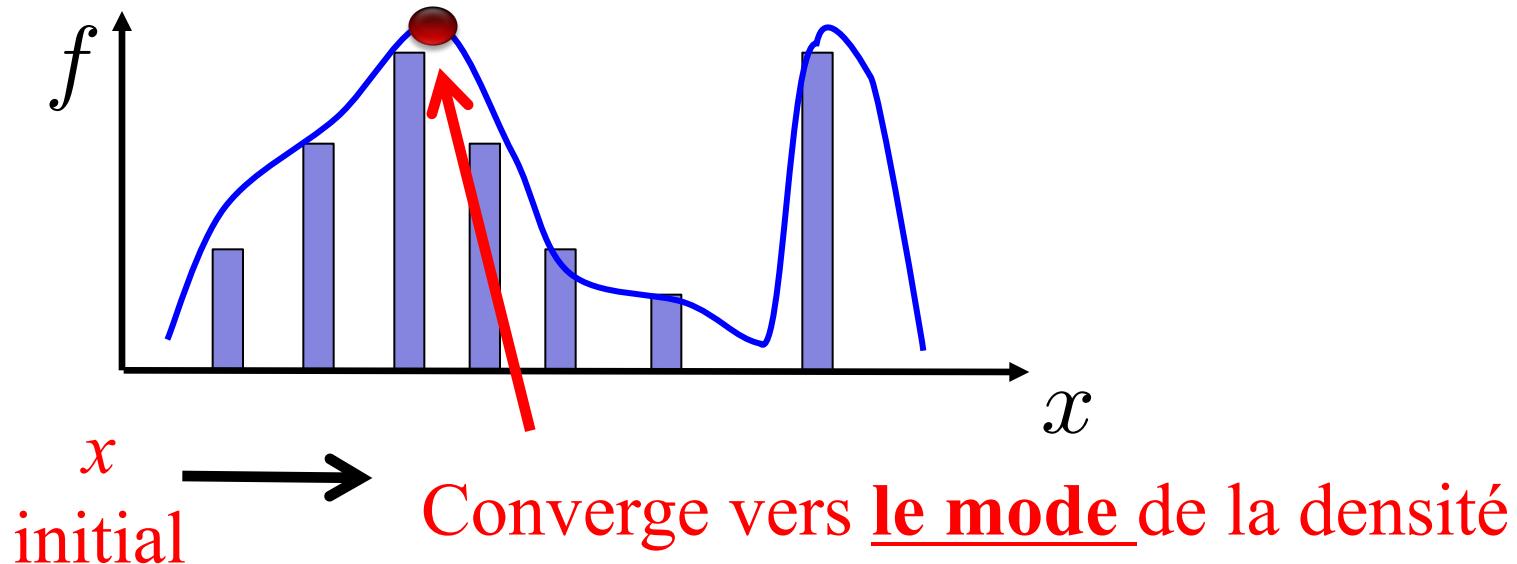
Décalage de moyenne ('Mean-Shift')

- Estimation de densité basée sur les noyaux ('Kernels'):



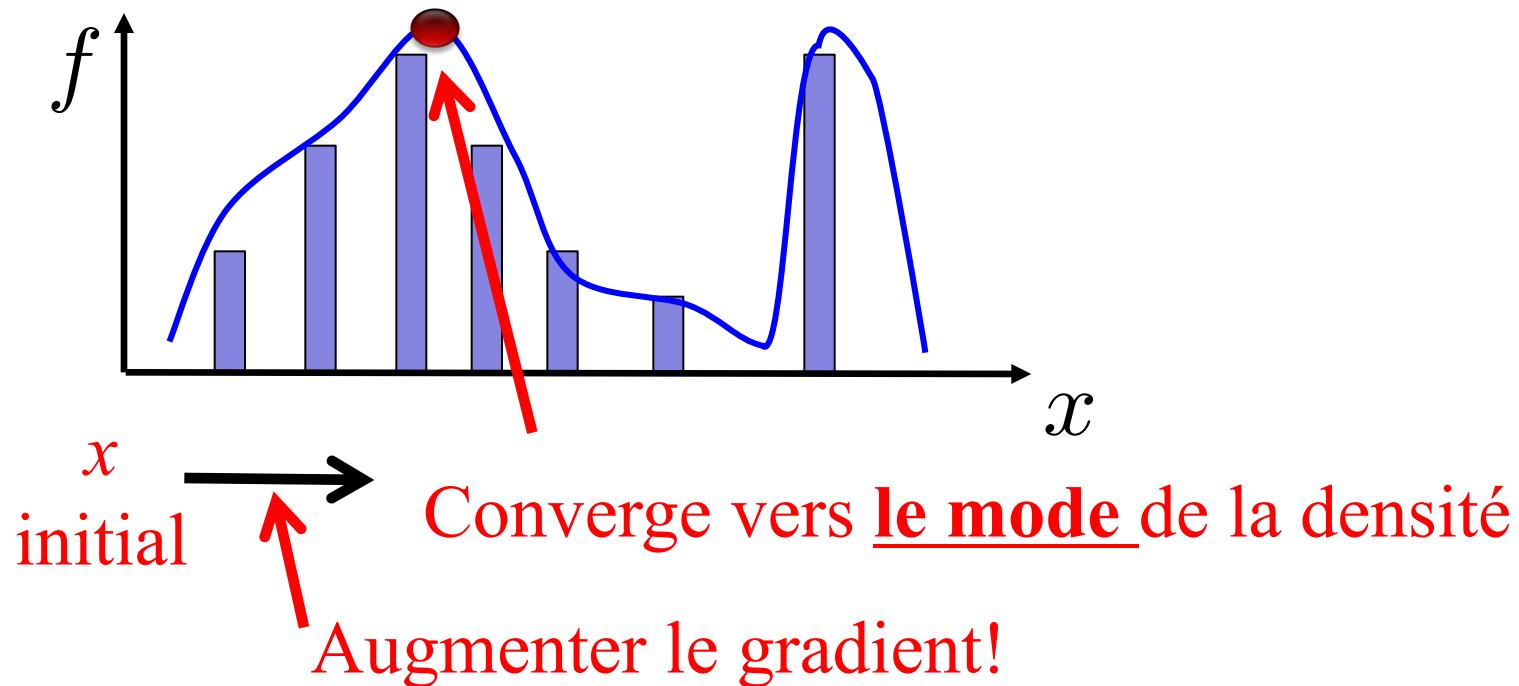
Décalage de moyenne ('Mean-Shift')

- Estimation de densité basée sur les noyaux ('Kernels'):



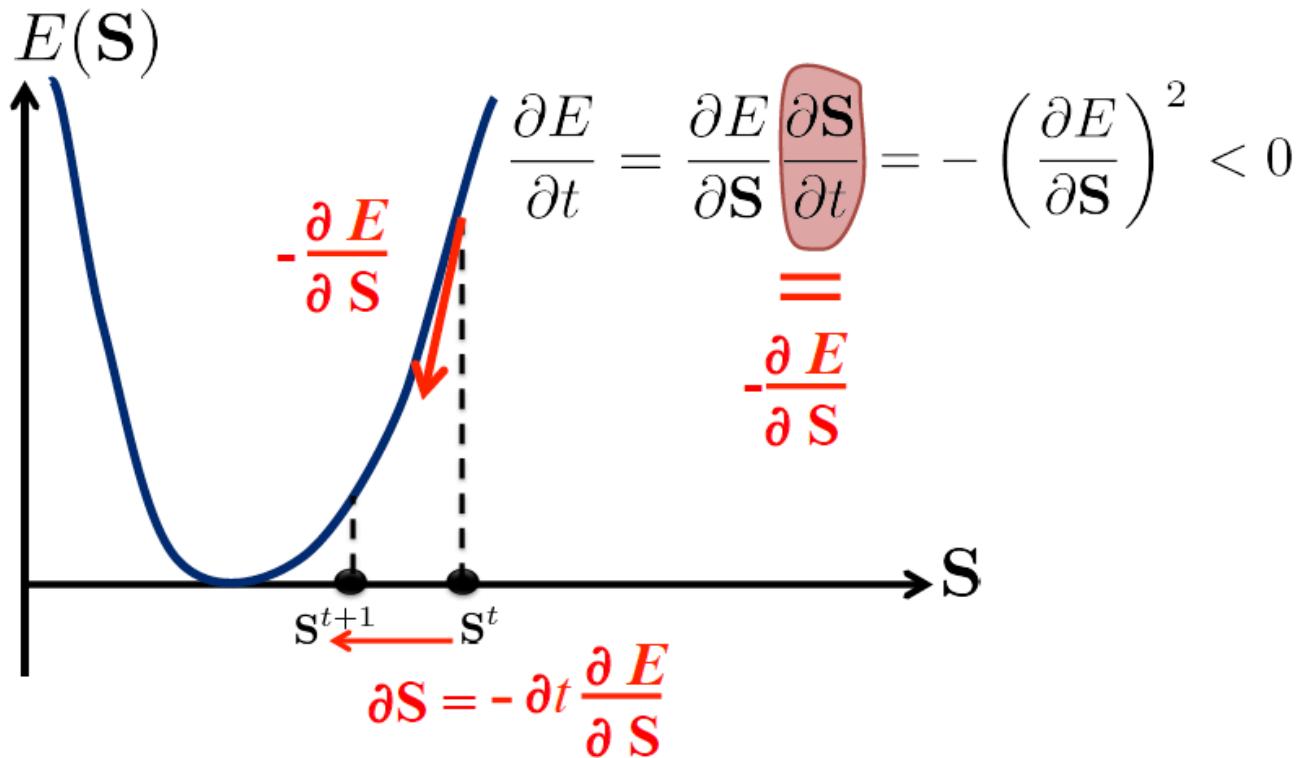
Décalage de moyenne ('Mean-Shift')

- Estimation de densité basée sur les noyaux ('Kernels'):



Méthode du gradient (Rappel)

- Descente de gradient:



Décalage de moyenne ('Mean-Shift')

$$\nabla f(x) = \sum_i (x_i - x)G(x - x_i) = \sum_i (x_i - x)g\left(\frac{\|x - x_i\|^2}{h^2}\right)$$

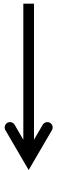
↓

Gradient

Décalage de moyenne ('Mean-Shift')

$$\nabla f(x) = \sum_i (x_i - x) G(x - x_i) = \sum_i (x_i - x) g\left(\frac{\|x - x_i\|^2}{h^2}\right)$$

Gradient



$$\nabla f(x) = \left[\sum_i G(x - x_i) \right] m(x)$$



Mean-Shift → $m(x) = \frac{\sum_i x_i G(x - x_i)}{\sum_i G(x - x_i)} - x$

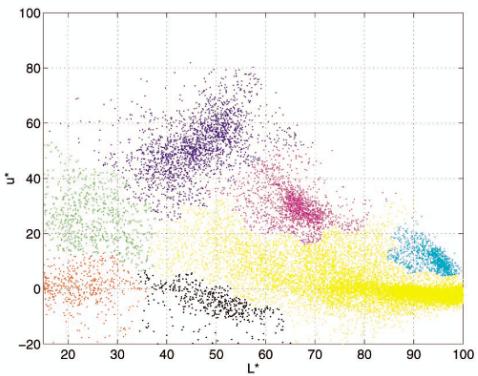
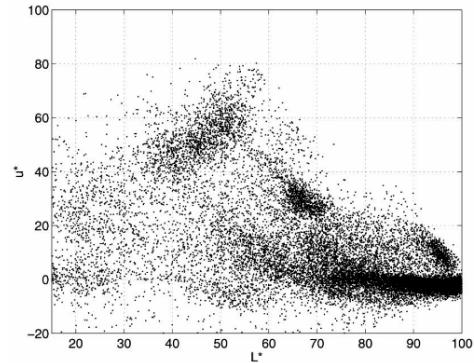
Décalage de moyenne ('Mean-Shift')

- Tout ce qu'on à faire pour chaque point: Commencer au point et faire ces mises à jours itératives (très rapides) pour converger au mode (cluster) le plus proche

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{m}(\mathbf{y}_k) = \frac{\sum_i \mathbf{x}_i G(\mathbf{y}_k - \mathbf{x}_i)}{\sum_i G(\mathbf{y}_k - \mathbf{x}_i)}.$$

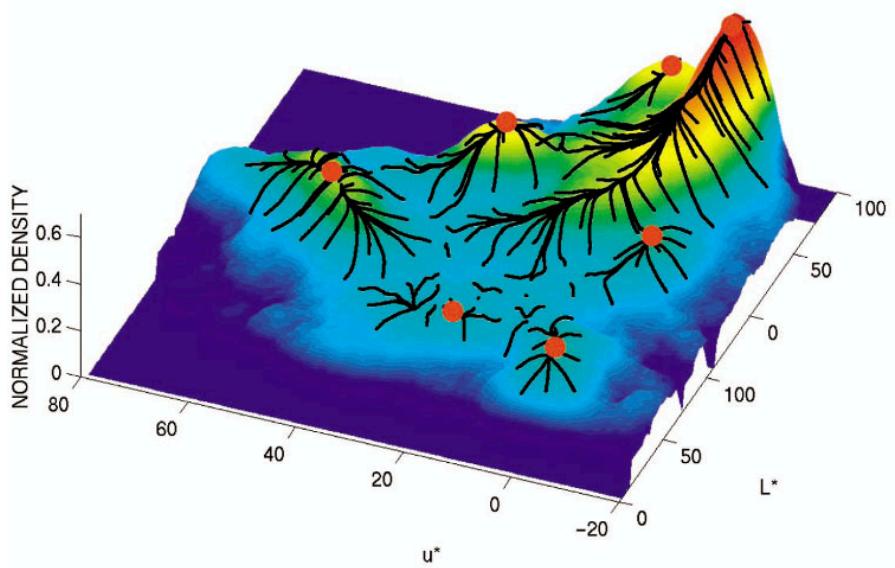
- Pas besoin de calculer la densité de tous les points (efficace pour des dimensions larges)
- Pas besoin de connaître le nombre de catégories!

Décalage de moyenne: Exemples

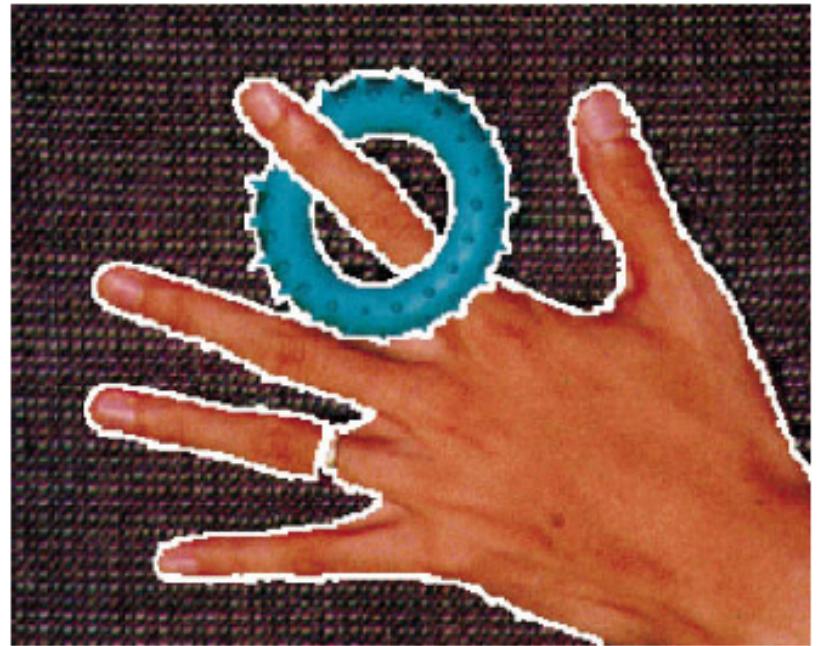


[Comaniciu et Meer, *IEEE TPAMI* 02]

~ 10,000 citations!



Décalage de moyenne: Exemples



[Comaniciu et Meer, *TPAMI* 02]

~ 10,000 citations!

Décalage de moyenne: Exemples

<https://www.youtube.com/watch?v=hJg7ik4x95U>

<https://www.youtube.com/watch?v=pE5KUzHFSnk>

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

1) Catégorisation à noyau et spectrale

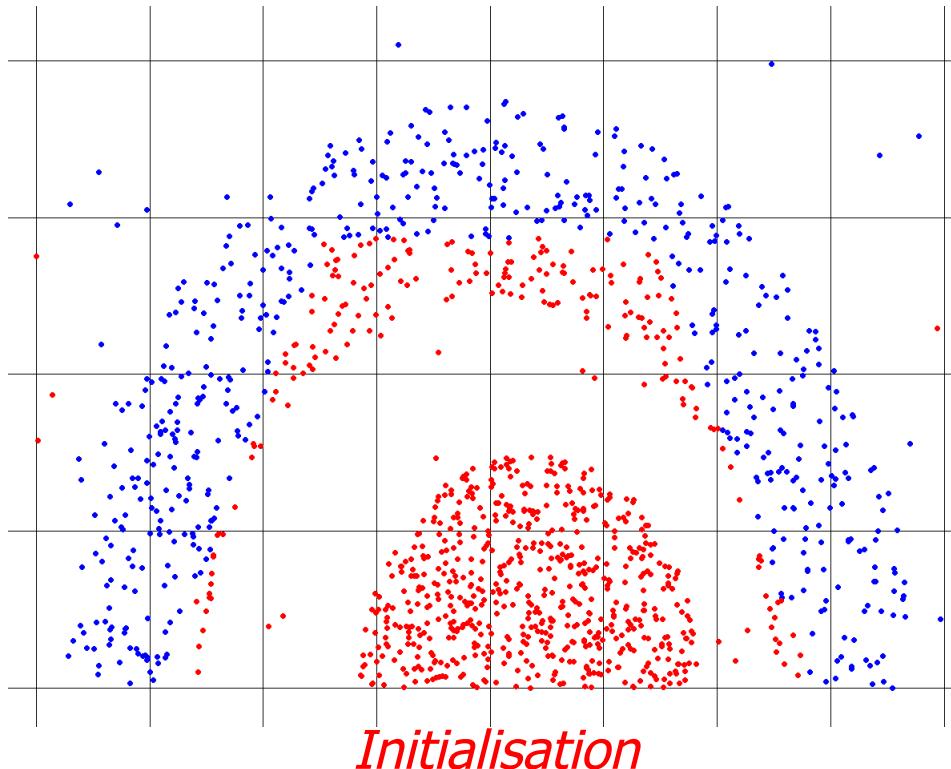
- Décalage de moyenne ('*Mean-Shift*')
- K-means basé sur les noyaux (*Kernel K-means*)
- Formulations basées sur les affinités de points
- Analyse spectrale

2) Réseaux auto-organisateurs

K-means: Séparation non-linéaire?

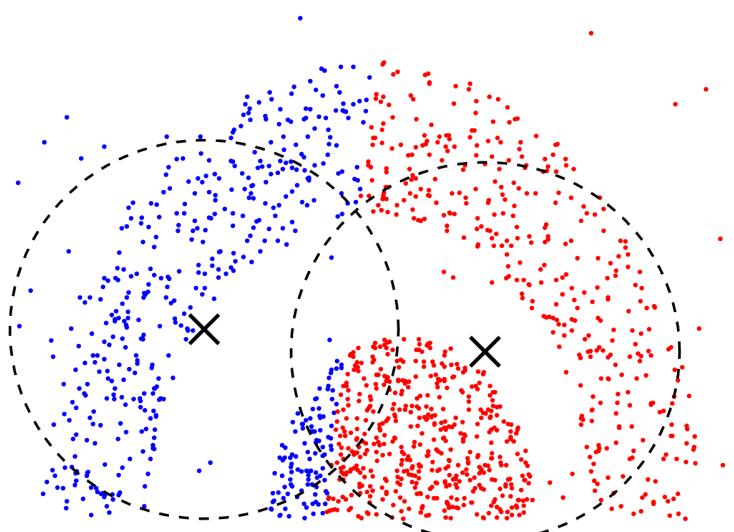
- K-means:

$$J = \sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} \|\mathbf{x}_i - \mu_k\|^2$$

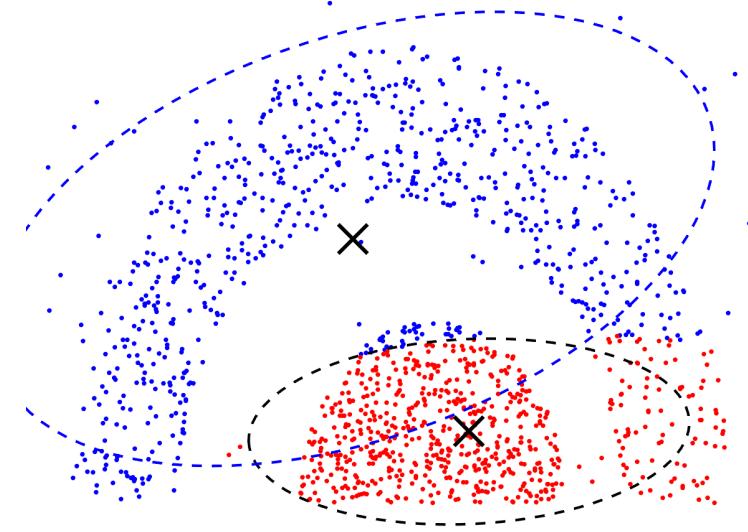


K-means: Séparation non-linéaire?

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



K-means

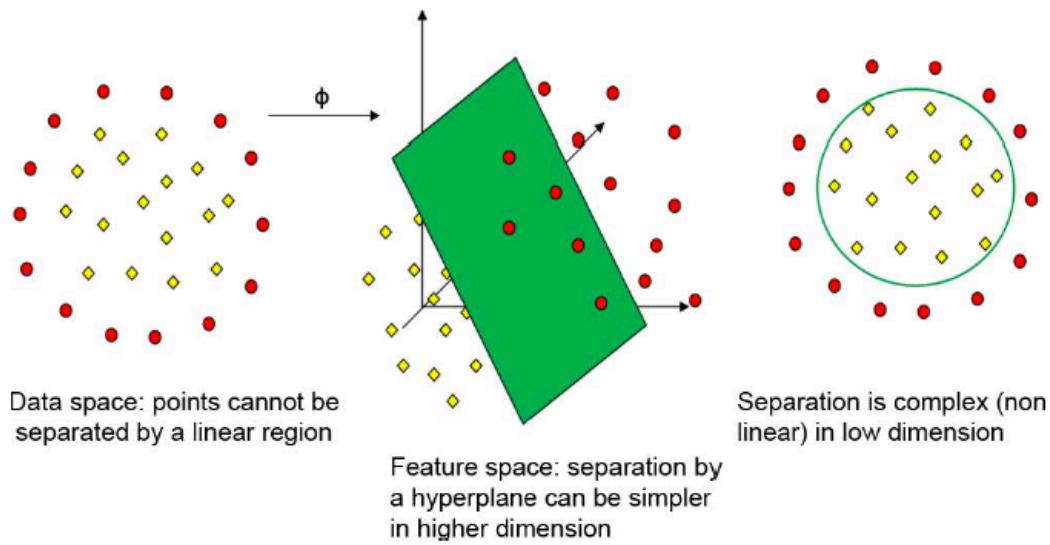


K-means elliptique

Catégorisation à noyaux

Kernel K-means

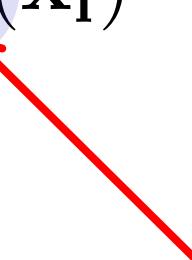
$$J = \sum_{\mathbf{x}_i \in C_1} \|\phi(\mathbf{x}_i) - \mu_1\|^2 + \sum_{\mathbf{x}_i \in C_2} \|\phi(\mathbf{x}_i) - \mu_2\|^2$$



Catégorisation à noyaux

Kernel K-means

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



Fonction qui transforme les données de l'espace original à un espace de plus haute dimension

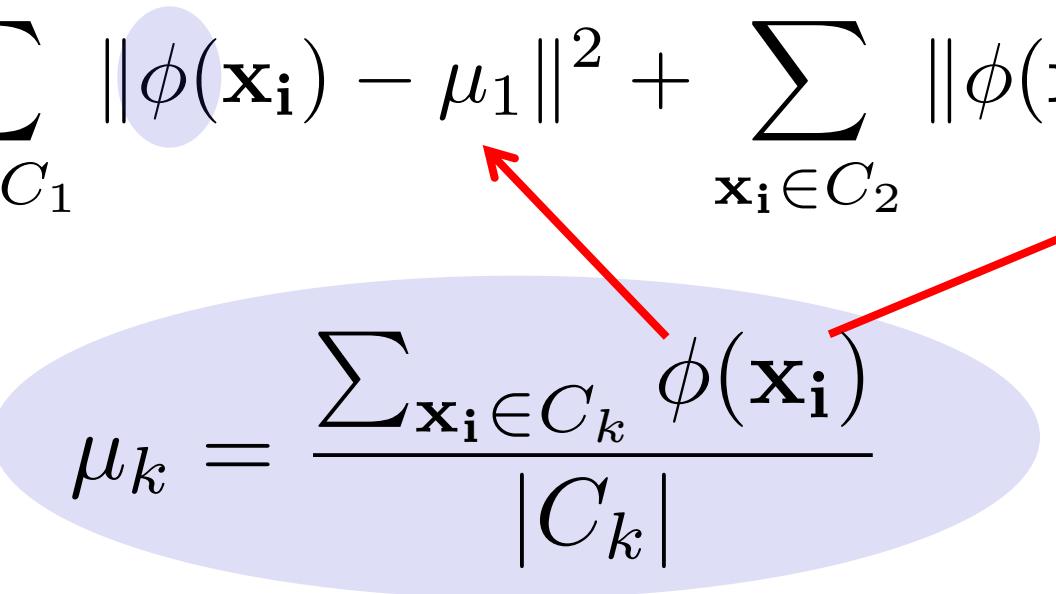
Comment avoir cette fonction?

Catégorisation à noyaux

Kernel K-means

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

$\mu_k = \frac{\sum_{\mathbf{x}_i \in C_k} \phi(\mathbf{x}_i)}{|C_k|}$



*Moyenne optimale comme dans K-means:
 Encore une fois on ne connaît pas la fonction transformatrice*

Catégorisation à noyaux

En mettant les moyenne optimale dans la fonction coût

$$J = \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_1} \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_q)\|^2}{2|C_1|} + \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_2} \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_q)\|^2}{2|C_2|}$$

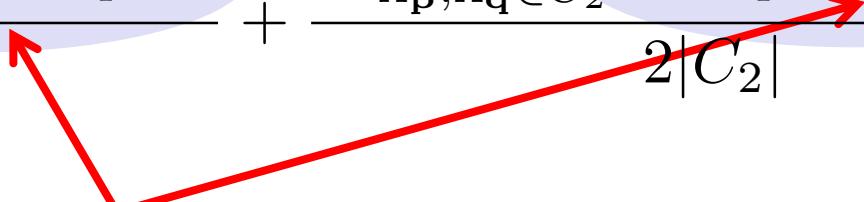

*Il n'y a plus de moyennes dans la fonction:
 Juste les distances entre les paires de points.
 La fonction transformatrice est encore présente*

Catégorisation à noyaux

En mettant les moyenne optimale dans la fonction coût

$$J = \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_1} \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_q)\|^2}{2|C_1|} + \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_2} \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_q)\|^2}{2|C_2|}$$

D_{pq}



On appelle ce critère «Average distortion (AD)» dans la littérature des méthodes spectrales

[Roth et al., TPAMI 03]

Catégorisation à noyaux

- Mr. Mercer nous donne une solution!

Mercer theorem: Une fonction noyau (*kernel function*) positive semi-définie (psd) est un produit scalaire dans un espace de plus haute dimension:

$$k(x, y) = \phi(x)^T \phi(y)$$

fondation noyau (kernel)

Produit scalaire dans le nouveau espace

[Muller et al., IEEE TNN 01]

Catégorisation à noyaux

- Mr. Mercer nous donne une solution!

Exemple de fonction noyau

$$k(x, y) = \exp \frac{-(x - y)^2}{\sigma}$$

[Muller et al., *IEEE TNN 01*]

Catégorisation à noyaux

- Mr. Mercer nous donne une solution!

Mercer theorem: Nous permet d'exprimer la transformation avec les noyaux

$$\|\phi(\mathbf{x}) - \phi(\mathbf{y})\|^2 = k(x, x) + k(y, y) - 2k(x, y)$$



Transformation



Fonction noyau (kernel)

[Muller et al., IEEE TNN 01]

Catégorisation à noyaux

Mercer theorem: On peut aussi exprimer les distances entre les points et les prototypes (dans le nouveau espace) en fonction des noyaux!

Uniquement en fonction des noyaux!

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

Donc: On a tous les ingrédients pour faire K-means dans le nouveau espace
 (Sauf que la complexité est quadratique maintenant)!

[Muller et al., IEEE TNN 01]

Catégorisation à noyaux

- Kernel *K-means* est équivalent à la minimisation de la fonction suivante, qui s'exprime avec **les affinités (associations)** entre les paires de points:

$$J = - \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_1} k(\mathbf{x}_p, \mathbf{x}_q)}{|C_1|} - \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_2} k(\mathbf{x}_p, \mathbf{x}_q)}{|C_2|} + \text{Constante}$$



$$- \sum_{\mathbf{x}_p \in C_1} f_{C_1}(\mathbf{x}_p)$$

Densité des données dans le clusters (basée sur les noyaux)!

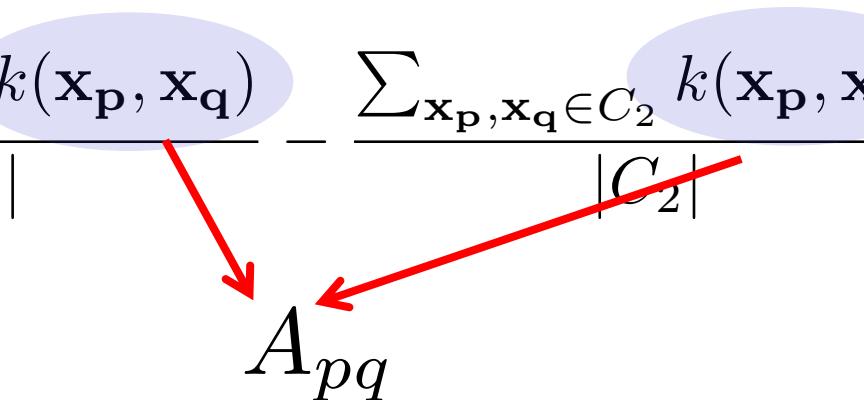
[Dhillon et al., KDD 04]
 Tang et al., ICCV 2015

Catégorisation à noyaux

- Kernel *K-means* est équivalent à la minimisation de la fonction suivante, qui s'exprime avec les **affinités** (**associations**) entre les paires de points:

$$J = -\frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_1} k(\mathbf{x}_p, \mathbf{x}_q)}{|C_1|} - \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_2} k(\mathbf{x}_p, \mathbf{x}_q)}{|C_2|} + \text{Constante}$$

A_{pq}



On appelle ce critère «Average Association (AA)» dans la littérature des méthodes spectrales

[Shi and Malik, IEEE TPAMI]
 ~ 10,000 citations!

Équivalence des critères

- Même si la matrice d'affinités A n'est pas positive semi-définie (psd), on peut exprimer le problème sous forme *Kernel K-means*:

$$k := \frac{A + A^T}{2} + \lambda I$$

$$-\sum_k \sum_{\mathbf{x}_p \in C_k} \|\phi(\mathbf{x}_p) - \mu_k\|$$



$$-\sum_k \frac{\sum_{\mathbf{x}_p, \mathbf{x}_q \in C_k} A(\mathbf{x}_p, \mathbf{x}_q)}{|C_k|}$$

Kernel K-means $A(\mathbf{x}_p, \mathbf{x}_q) = \phi(\mathbf{x}_p)^T \phi(\mathbf{x}_q)$

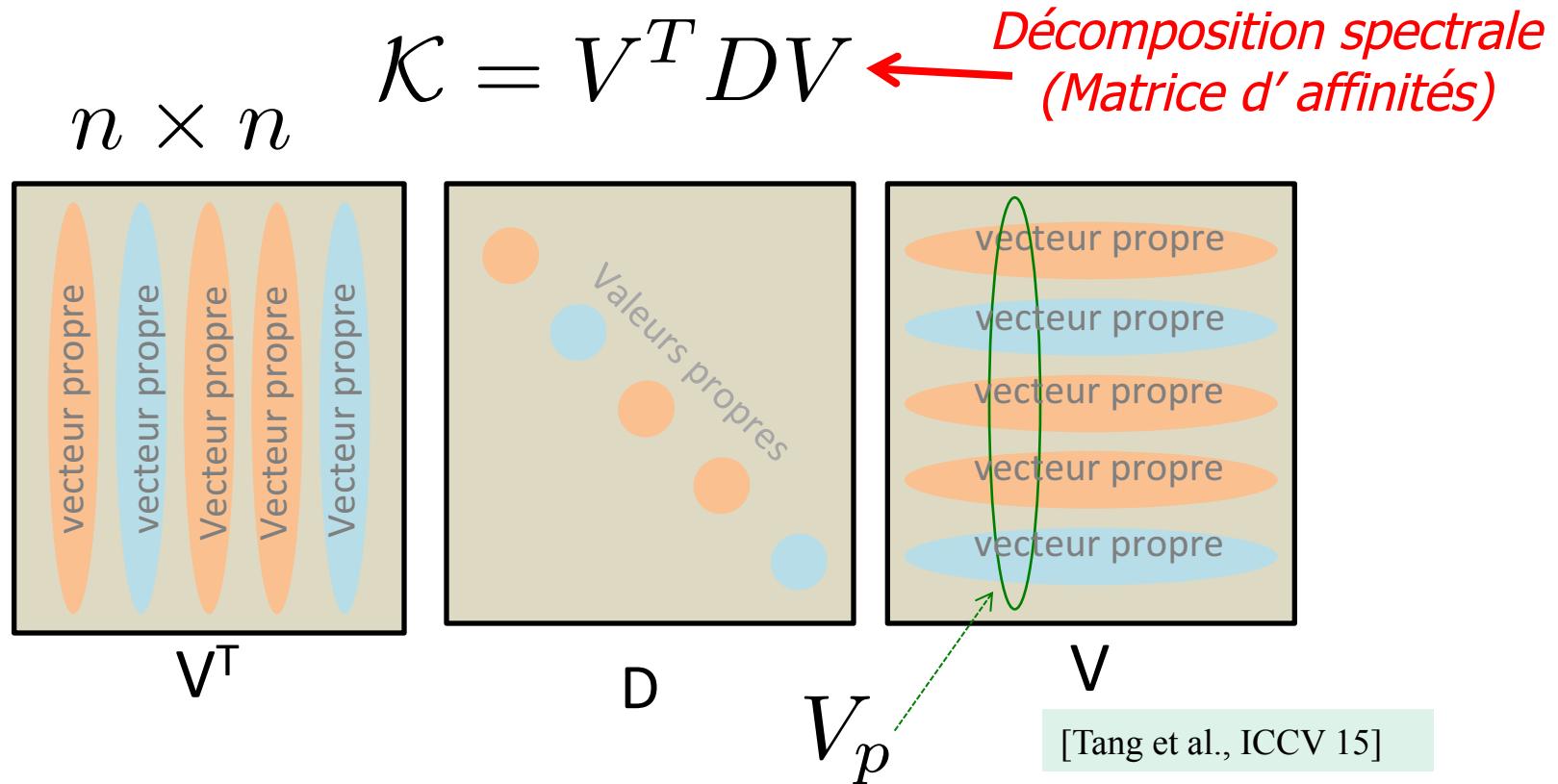
AA

[Dhillon et al., KDD 04]

[Tang et al., ICCV 15]

Analyse spectrale

- Est ce qu'on peut avoir la transformation ('mapping')?
- Oui, si l'ensemble de points est fini (tjs le cas en pratique)!



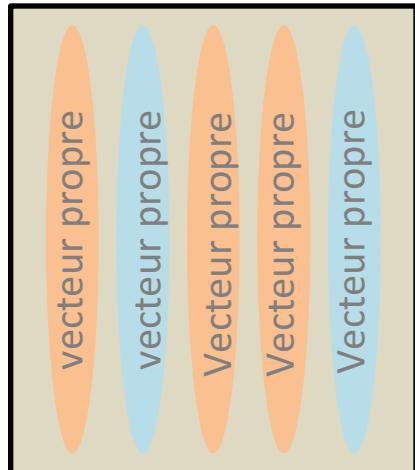
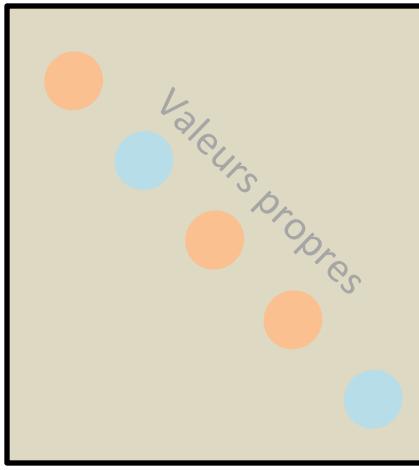
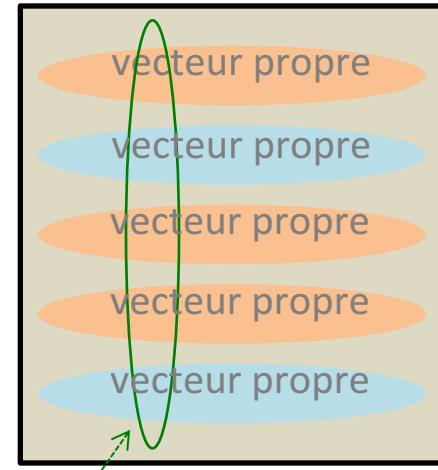
Analyse spectrale

- Si les valeurs propres sont positives (matrice psd):

$$D = \sqrt{D} \sqrt{D} \rightarrow \boxed{\phi(\mathbf{x}_p) = \sqrt{D} V_p}$$

Dimension très large

n × n


 V^T

 D

 V
 V_p

[Tang et al., ICCV 15]

Analyse spectrale

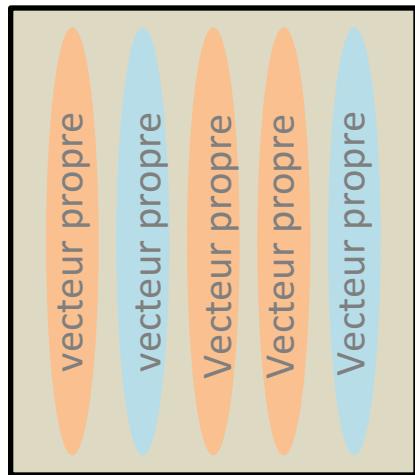
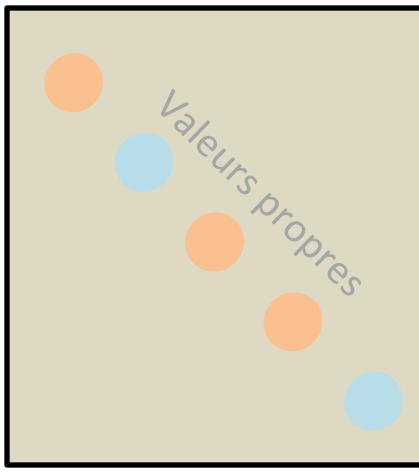
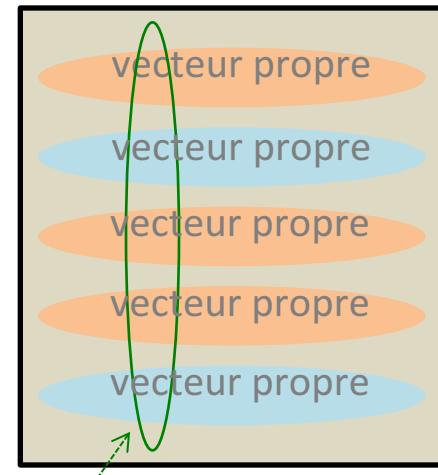
- On peut contrôler la dimension (sans perdre trop de précision):



Large valeurs propres



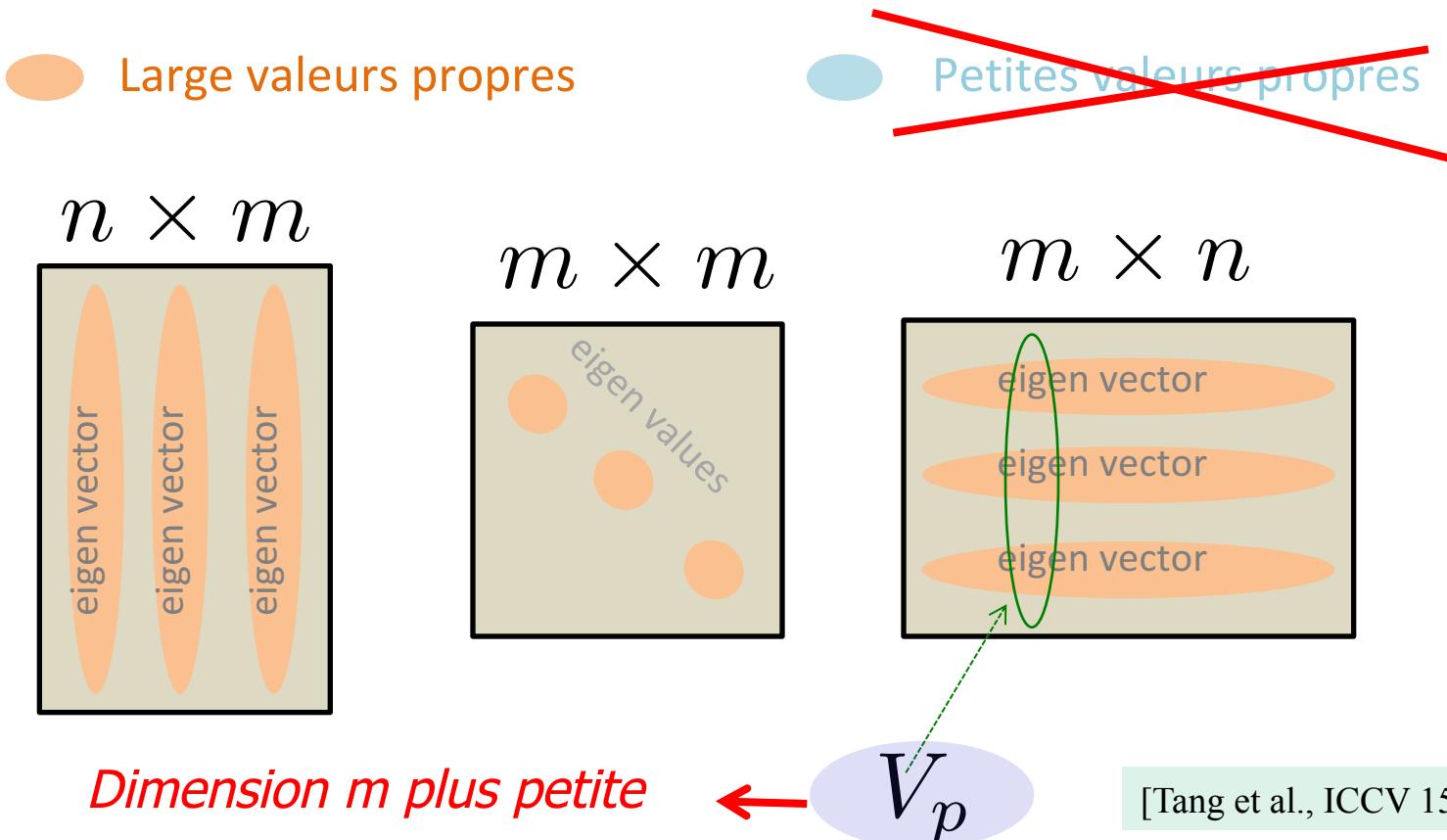
Petites valeurs propres


 V^T

 D

 V
 V_p

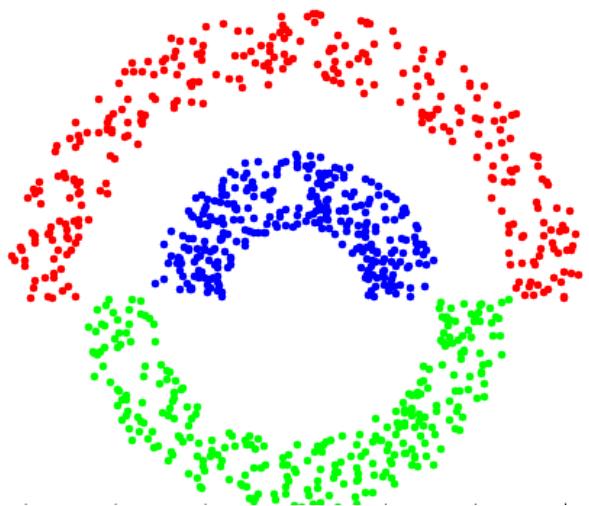
[Tang et al., ICCV 15]

Analyse spectrale

- On peut contrôler la dimension (sans perdre trop de précision):

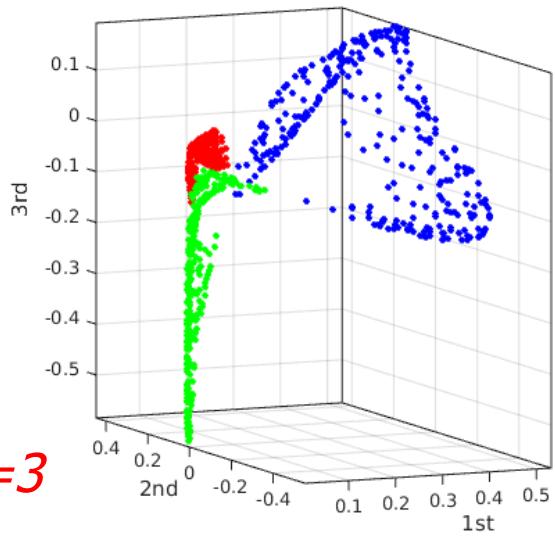


Exemples

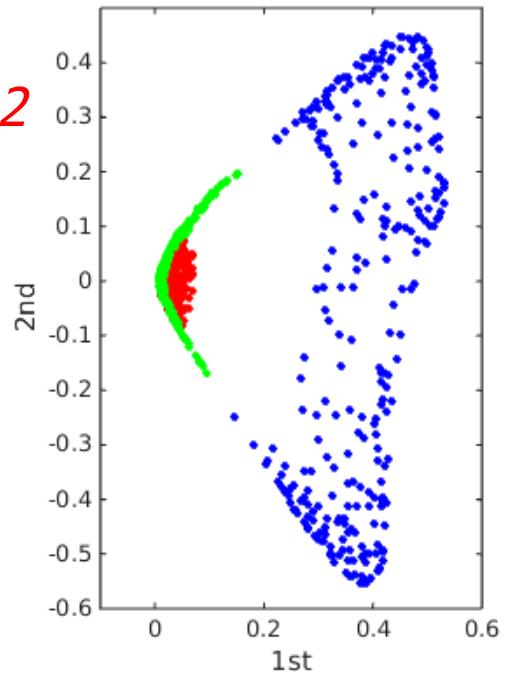


Données 2 D

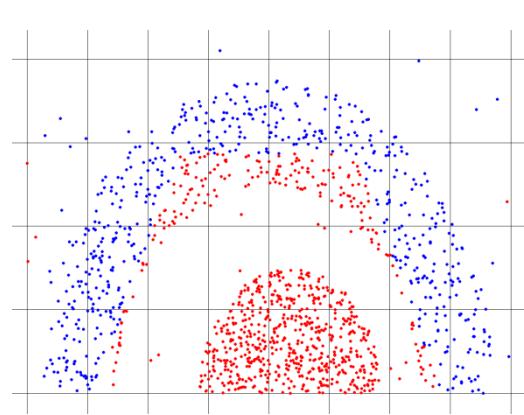
m=3



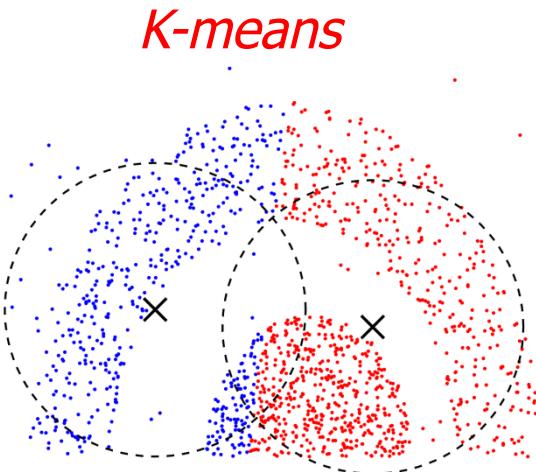
m=2



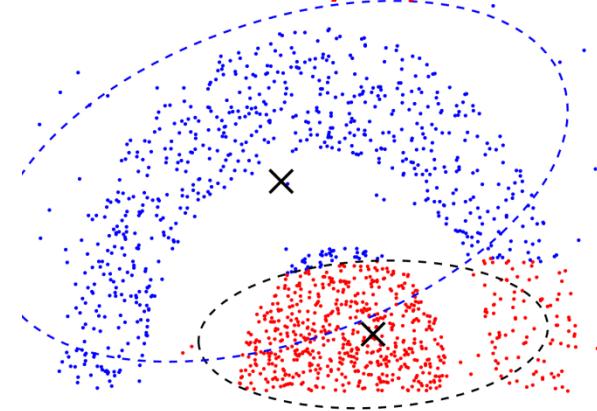
Exemples



K-means

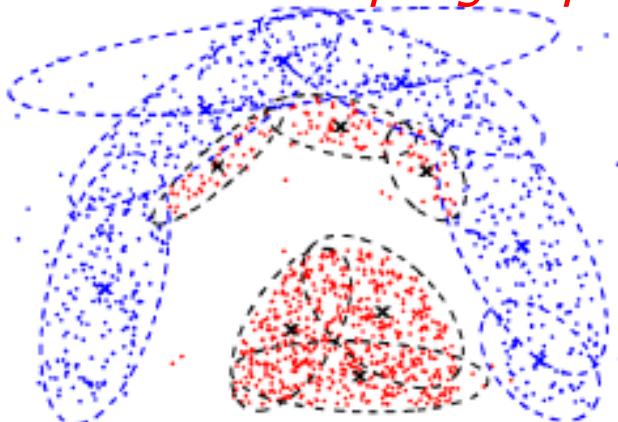


K-means elliptique

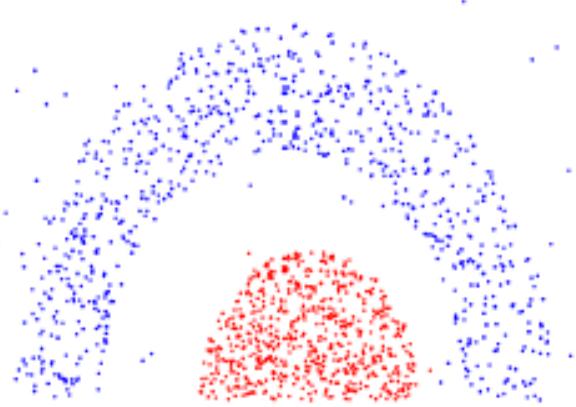


Initialisation

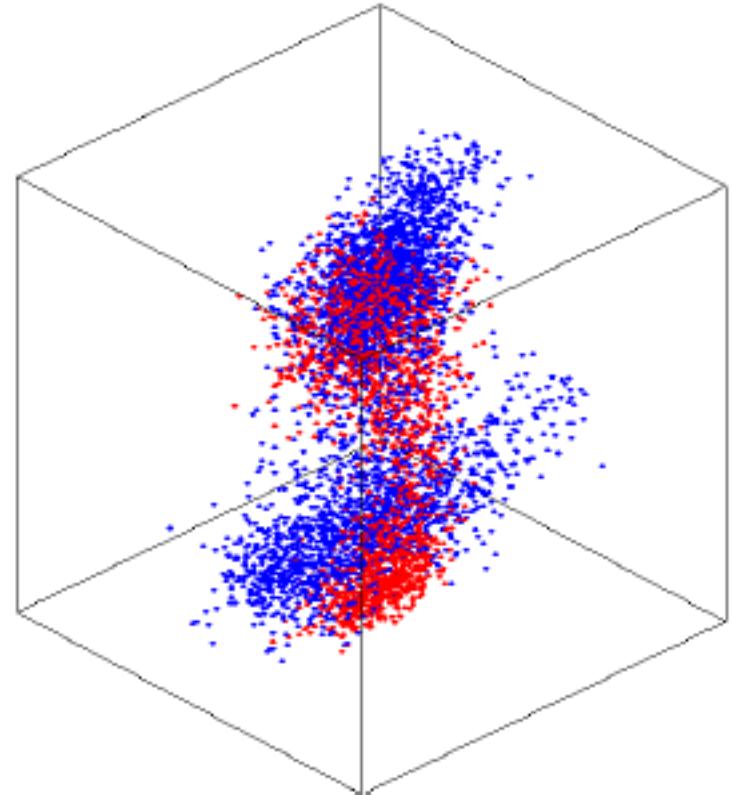
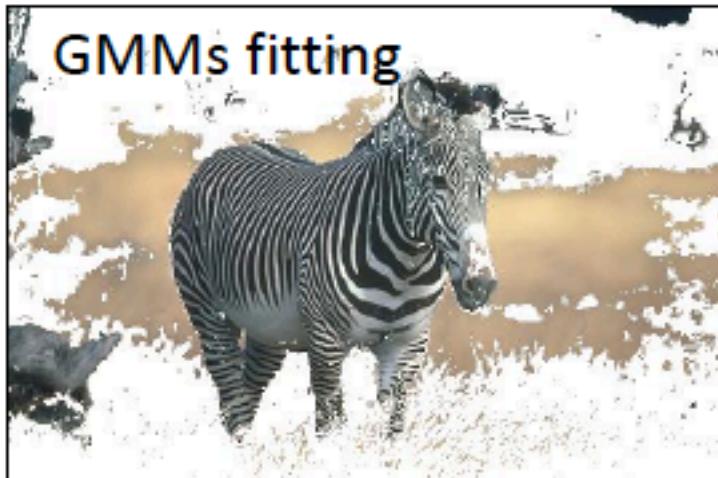
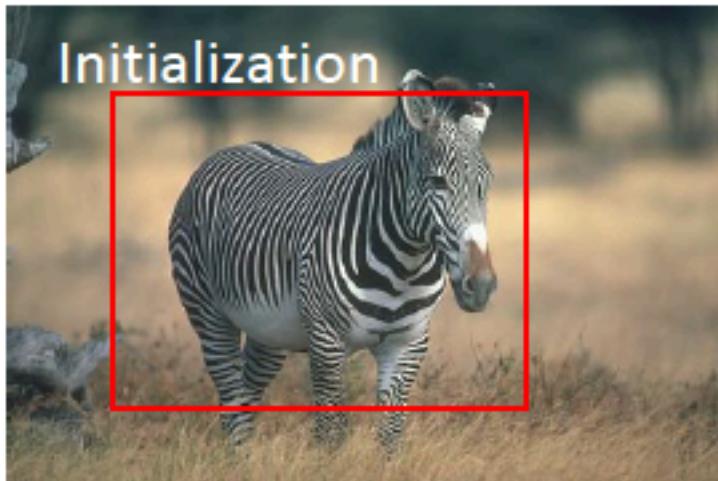
*K-means probabiliste
GMM dans chaque groupe*



Kernel K-means



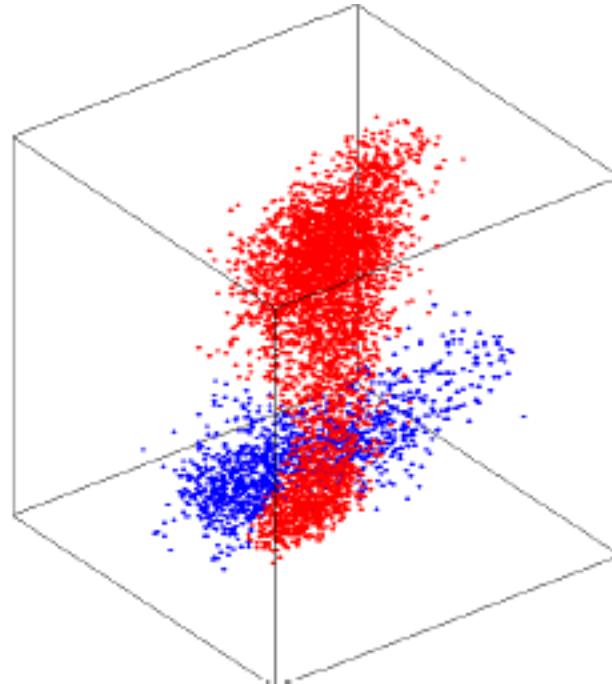
Un cas réel



3D color space

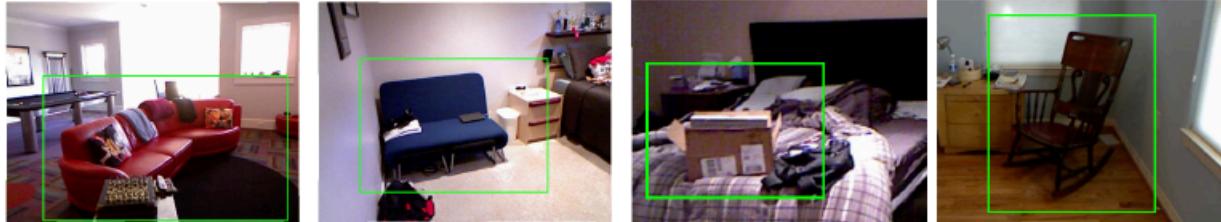
[Tang et al., ICCV 15]

Un cas réel (Kernel K-means)

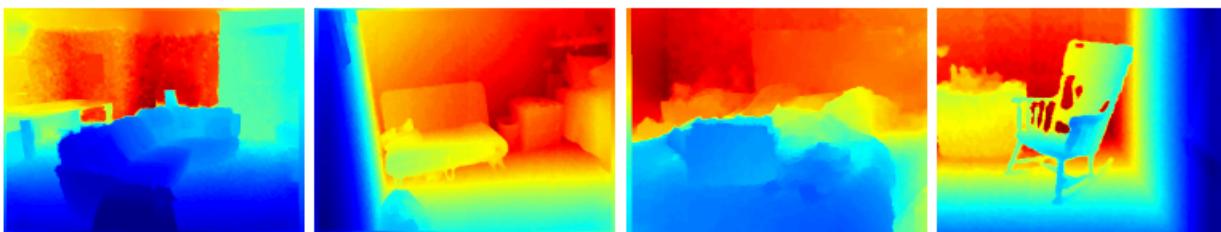


[Tang et al., ICCV 15]

Autres exemples



*Données: RGBDXY
(dim 6)*



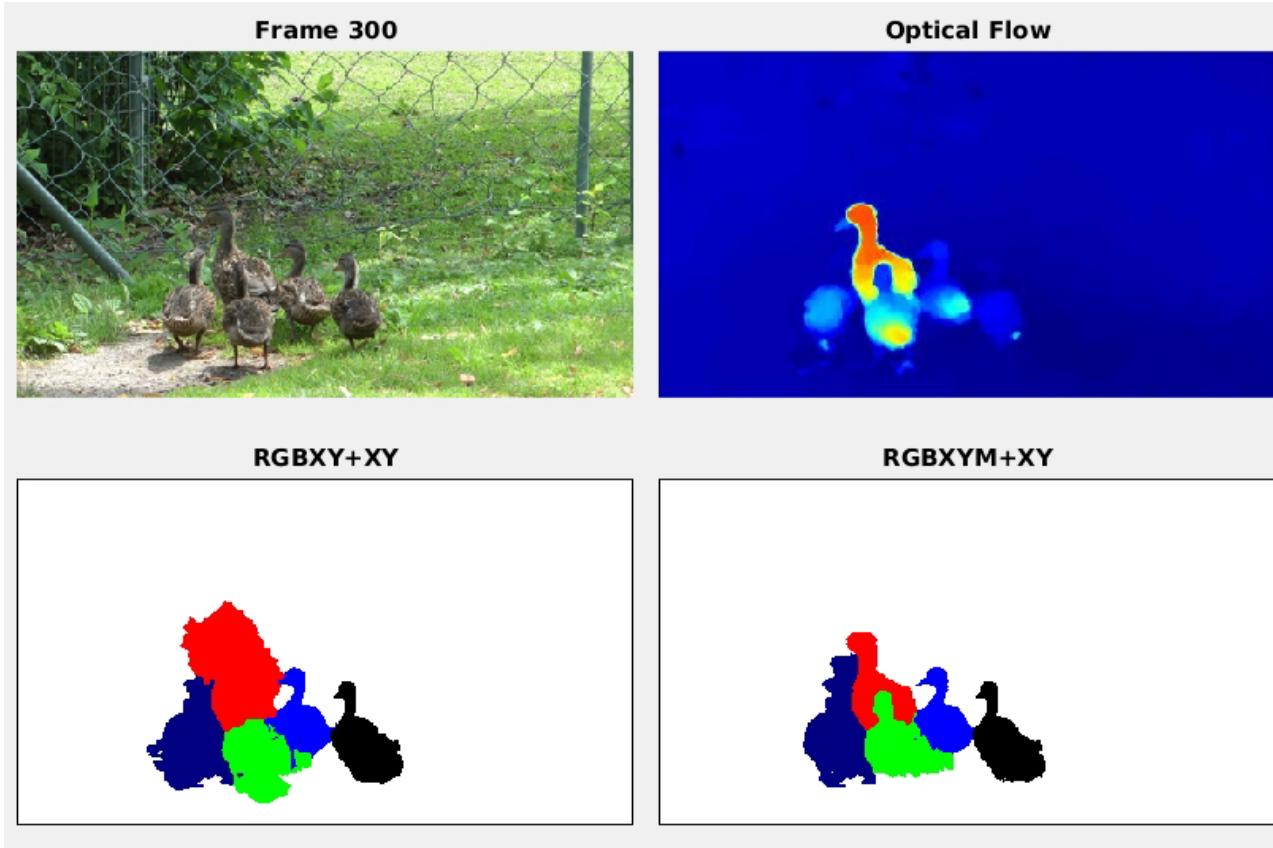
*K-means
probabiliste*

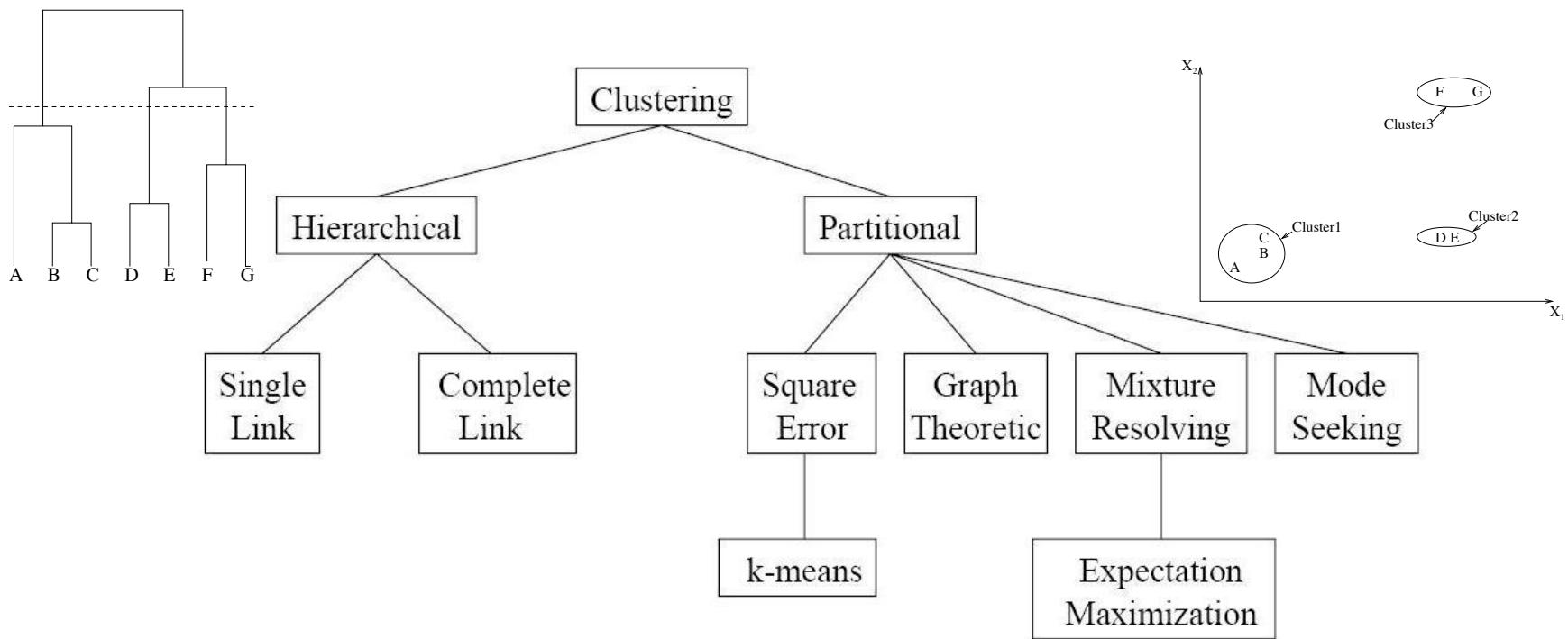


Kernel K-means

[Tang et al., ICCV 15]

Exemples (vidéos)





- R. Xu and D. Wunsch, 'Survey of clustering algorithms,' *IEEE Trans. on Neural Networks*, 16(3):645-678, May 2005.
- A.K. Jain et al., 'Data Clustering: A Review,' *ACM Computing Surveys*, 31(3):264-323, September 1999

Taxonomie des approches

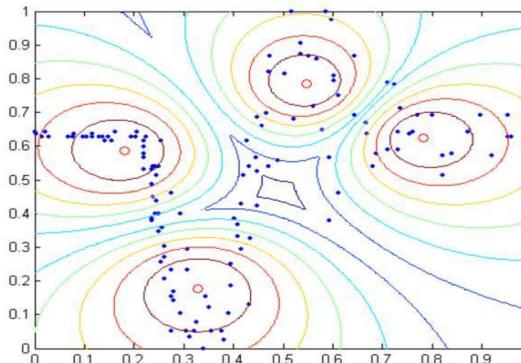
Étapes générales - approches partitionnelles

On veut regrouper les données selon un critère de similarité

1. Initialisation de k prototypes ou catégories du modèle
2. Partitionnement:

- calcul la similarité entre chaque patron \mathbf{x}_i et chaque catégorie
- assigne une ou plusieurs catégories à chaque patron \mathbf{x}_i

3. Apprentissage: adapter les catégories du modèle



Taxonomie des approches

Défis pour la performance

► Choix du nombre de catégories

- 1) *modèles constructifs*
- 2) *critères de validité du modèle*
- 3) *validation empirique avec données indépendantes*

► Gestion des données pour l'apprentissage

- apprentissage *par lot* versus *séquentielle (en-ligne)*

► Modélisation du chevauchement et de la dispersion des données

- partitionnement *dur* versus *mou*

Sommaire

Apprentissage non-supervisé pour la catégorisation de vecteurs

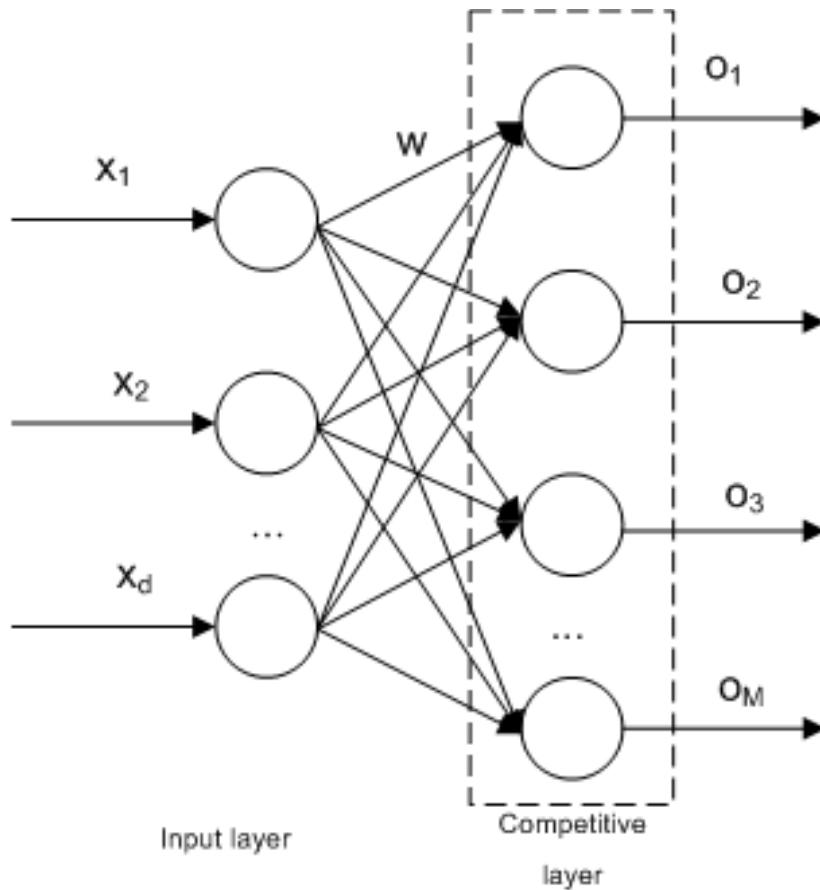
1) Catégorisation à noyau et spectrale

- Décalage de moyenne ('*Mean-Shift*')
- K-means basé sur les noyaux (*Kernel K-means*)
- Formulations basées sur les affinités de points
- Analyse spectrale

2) Réseaux auto-organisateurs

Réseaux auto-organisateurs

Réseaux de neurones compétitifs



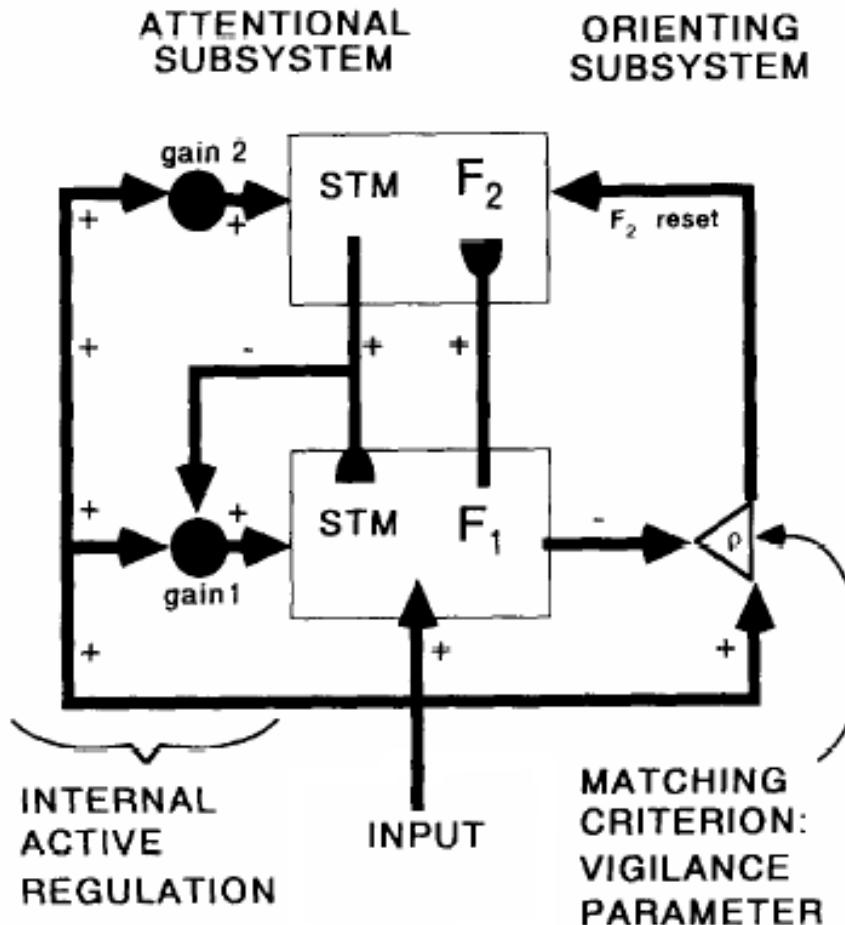
Réseaux auto-organisateurs

Réseaux ART

- ▶ ***Adaptive Resonance Theory:*** une théorie du traitement d'information cognitif
(Grossberg, 1976)
- ▶ **Origine:** une solution au dilemme de *plasticité-stabilité* lors de l'auto-organisation de catégories
 - *Q: comment concevoir un système stable face au bruit, mais adaptatif face aux nouvelles entrées?*
 - *Q: comment apprendre des nouvelles informations sans corrompre les connaissances acquises?*

Réseaux auto-organisateurs

Réseaux ART



Réseaux auto-organisateurs

Réseaux ART

- ▶ Architectures neuroniques capables d'apprendre des catégories stables en réponse à des séquences d'entrées *arbitraires*
 - **sous-système *d'attention*:**
 - 1 couche de neurones (F2) entièrement interconnectées aux noeuds d'entrée F1 et **rôle:** proposer une catégorie (neurone sur la couche F2) gagnante
 - **sous-système *d'orientation*:**
 - mesure la similarité entre entrée et prototype du gagnant
 - **rôle:** accepte le candidat proposé si il est assez similaire, ou bien orienter la recherche parmi les autres catégories

Réseaux auto-organisateurs

Réseaux ART

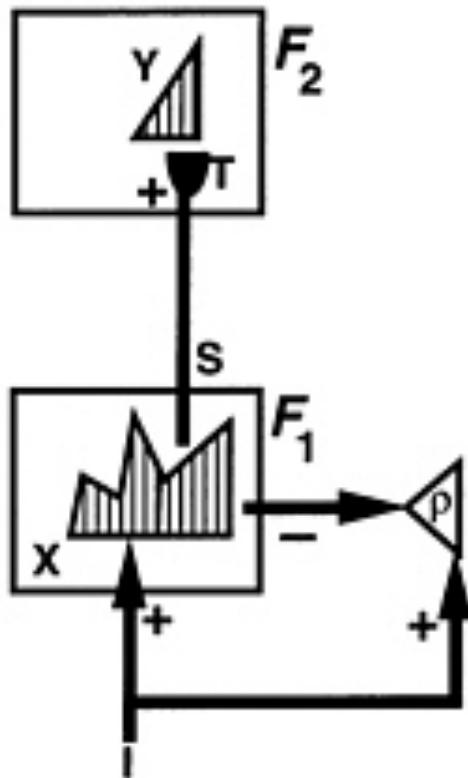
- ▶ **Test de vigilance** – mécanisme pour résoudre le problème d'instabilité face aux nouvelles informations:
 - rétroaction F2 → F1 pour comparer l'entrée avec le prototype du neurone F2 gagnant:
 1. *si entrée et prototype sont assez semblables:* état de résonance qui active l'apprentissage pour ce prototype
 2. *si entrée et prototype ne sont pas assez semblables:* recherche parallèle en mémoire pour choisir un autre neurone de F2
- ▶ **on peut alors apprendre de nouvelles entrées sans corrompre les connaissances acquises...**

Réseaux auto-organisateurs

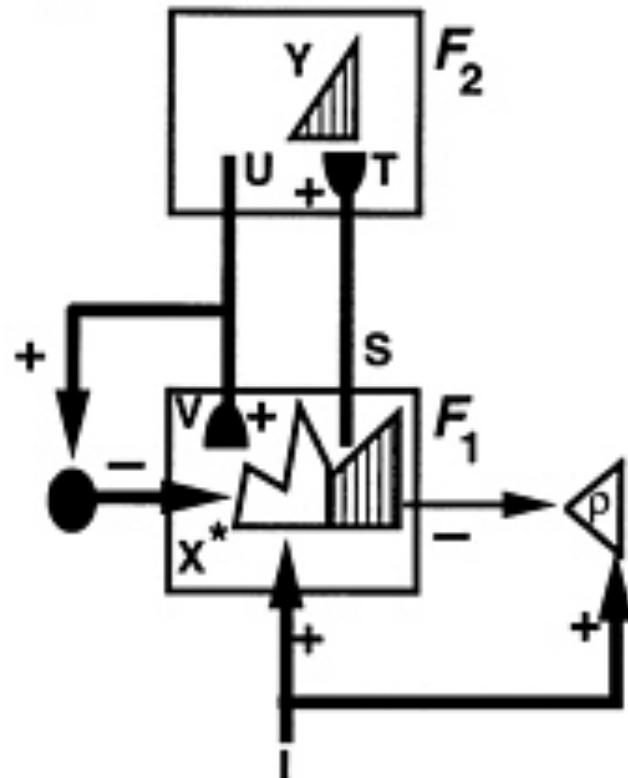
Réseaux ART

- ▶ Illustration du processus de recherche:

A



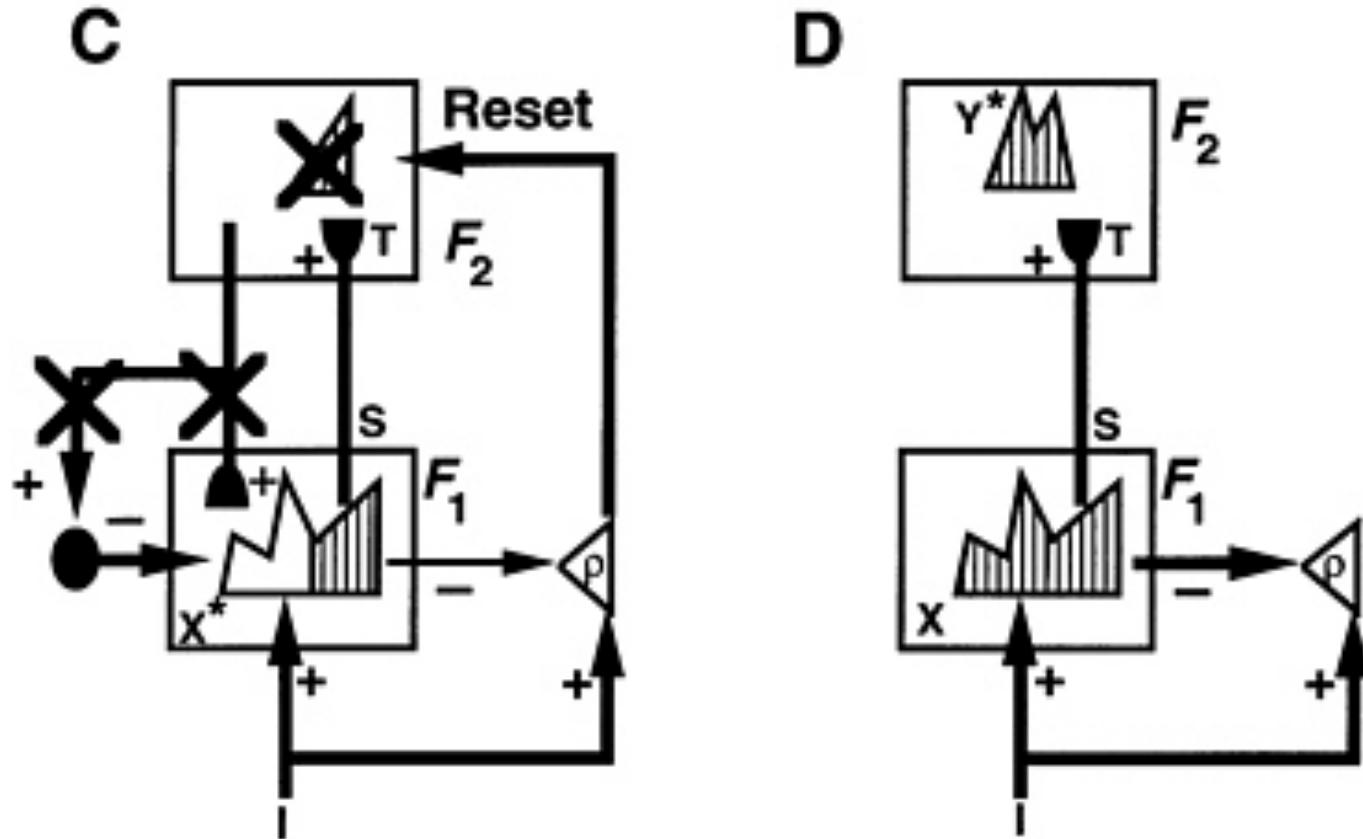
B



Réseaux auto-organisateurs

Réseaux ART

► Illustration du processus de recherche:



Réseaux auto-organisateurs

Réseaux ART

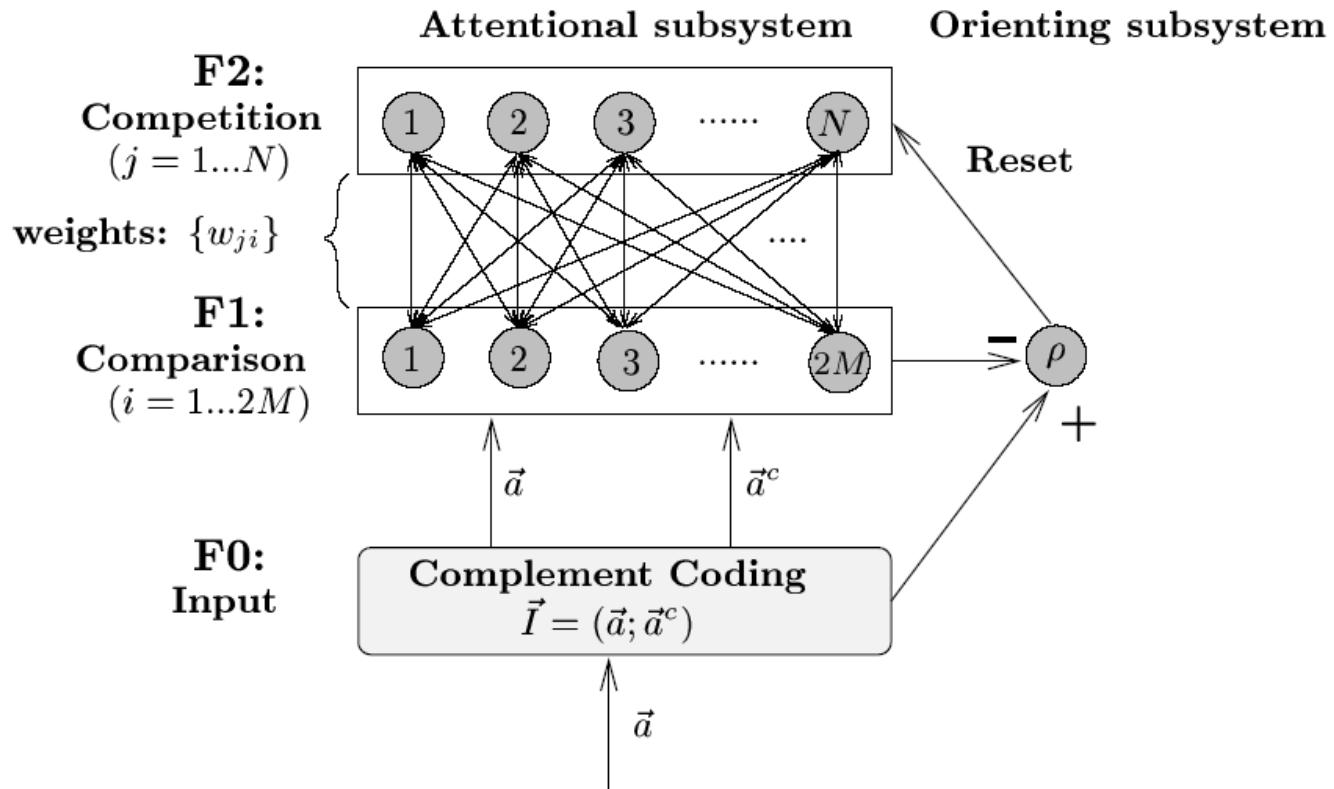
► Variantes populaires de la famille ART:

Architecture	Apprentissage	Entrées
ART1	non-supervisé	binaire
ART2, ART2-A et ART2-AE	non-supervisé	binaire+analogique
ART3	non-supervisé	binaire+analogique
fuzzy ART	non-supervisé	binaire+analogique
ARTMAP	supervisé	binaire
fuzzy ARTMAP	supervisé	binaire+analogique
ART-EMAP, ARTMAP-IC et distributed ARTMAP	supervisé	binaire+analogique
Bayesian ARTMAP, Gaussian ARTMAP, PROBART et PFAM	supervisé	binaire+analogique

Réseaux auto-organisateurs

Réseau fuzzy ART

► Architecture simplifiée:



Réseaux auto-organisateurs

Réseau fuzzy ART

► Description algorithmique:

1. Initialisation des poids:

- tous les neurones de F2 sont *non commis*
- tous les poids dans \mathbf{W} sont fixés à 1

2. Encodage d'un patron d'entrée:

- un nouveau patron entrée normalisé \mathbf{a} est présenté à la couche F0: $\mathbf{a} = \{a_i \in [0,1] : i = 1, 2, \dots, M\}$
- F0 réalise le *codage en complément*, qui donne un vecteur $\mathbf{A} = (\mathbf{a}; \mathbf{a}^c)$ avec $a_i^c = 1 - a_i$
- le vecteur \mathbf{A} active la couche F1

Réseaux auto-organisateurs

Réseau fuzzy ART

► Description algorithmique: (suite)

3. Choix de catégorie:

- la fonction de choix est calculé pour chaque neurone de la couche F2:

$$T_j(\mathbf{A}) = \frac{|\mathbf{A} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|}$$

- F2 est une couche de compétition WTA, où le gagnant est le neurone J tel que:

$$J = \arg \max \{ T_j : j = 1, 2, \dots, N \}$$

Réseaux auto-organisateurs

Réseau fuzzy ART

► Description algorithmique: (suite)

4. Critère de vigilance:

- compare la similarité entre \mathbf{A} et \mathbf{w}_J sur la couche F1 selon:

$$\frac{|\mathbf{A} \wedge \mathbf{w}_J|}{|\mathbf{A}|} = \frac{|\mathbf{A} \wedge \mathbf{w}_J|}{M} \geq \rho$$

- passe le test:

- neurone J est sélectionné, et \mathbf{w}_J peut apprendre \mathbf{A} (passe à l'étape 5)

- échoue le test:

- neurone J est désactivé pour \mathbf{A} ($T_J = -1$), et on retourne à l'étape 3 pour rechercher un autre J qui peut passer le test de vigilance

Réseaux auto-organisateurs

Réseau fuzzy ART

► Description algorithmique: (suite)

5. Mise à jour du prototype de J :

- le vecteur prototype \mathbf{w}_J du neurone J est adapté selon:

$$\mathbf{w}'_J = \beta(\mathbf{A} \wedge \mathbf{w}_J) + (1 - \beta)\mathbf{w}_J$$

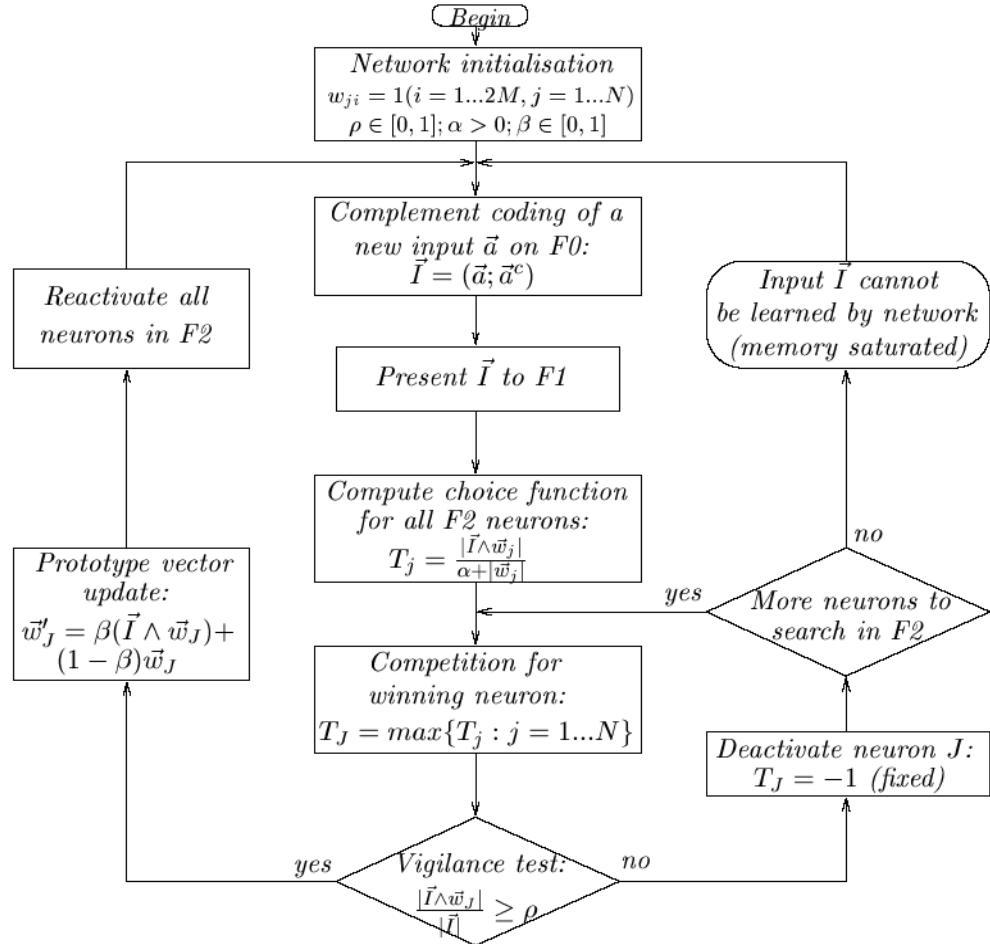
- pour une nouvelle catégorie $\mathbf{w}_J = \mathbf{A}$
- apprentissage rapide: $\beta = 1$
- apprentissage lente: $0 < \beta < 1$

Retour à l'étape 2 pour prendre une autre entrée

Réseaux auto-organisateurs

Réseau fuzzy ART

► Diagramme de flot de données

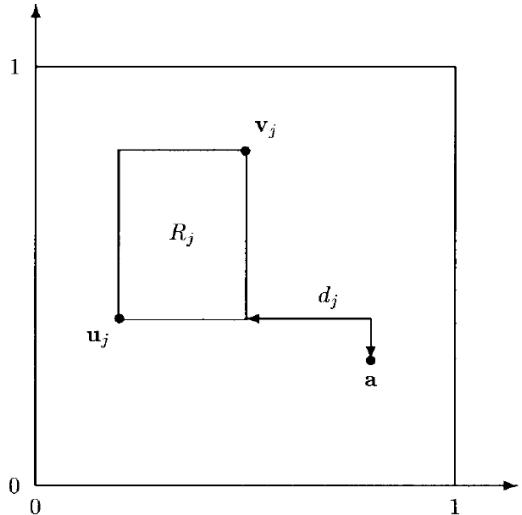


Réseaux auto-organisateurs

Réseau fuzzy ART

► Interprétation géométrique:

- chaque catégorie j peut être représentée comme un hyper-rectangle R_j dans un espace à M -dimensions
- le prototype a la forme: $\mathbf{w}_j = (\mathbf{u}_j; \mathbf{v}_j^c)$
- les dimensions de l'hyper-rectangle: $|R_j| = |\mathbf{v}_j - \mathbf{u}_j|$

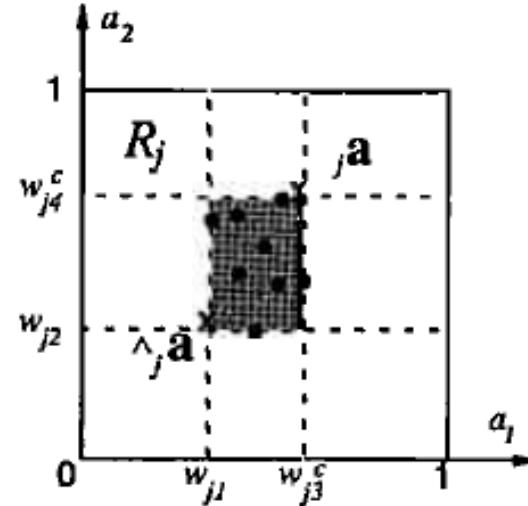
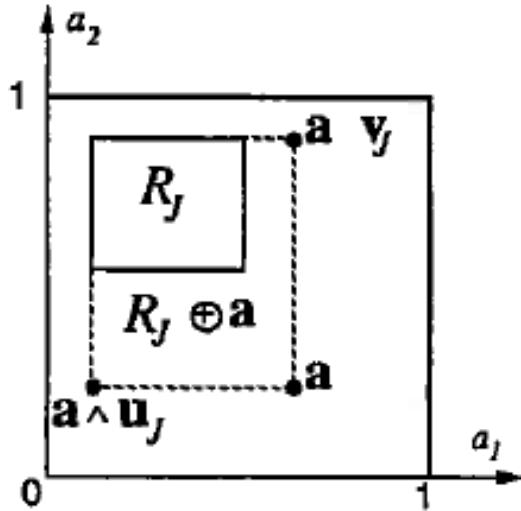


$$\begin{aligned}
 T_j &= \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|} \\
 &= \frac{M - |R_j| - (|\mathbf{w}_j| - |\mathbf{I} \wedge \mathbf{w}_j|)}{\alpha + M - |R_j|} \\
 &= \frac{M - |R_j| - |\mathbf{w}_j - (\mathbf{I} \wedge \mathbf{w}_j)|}{\alpha + M - |R_j|} \\
 &= \frac{M - |R_j| - d_j}{\alpha + M - |R_j|}
 \end{aligned}$$

Réseaux auto-organisateurs

Réseau fuzzy ART

- ▶ Interprétation géométrique: (suite)
 - mise à jour des prototypes:
 - neurone F2 non-commis: R_j représente un patron
 - neurone F2 commis: R_j englobe tous les patrons qui lui sont assignés, avec $|R_j| \leq M(1 - \rho)$



Réseaux auto-organisateurs

Réseaux SOM

Kohonen Self-Organizing Maps (SOM):

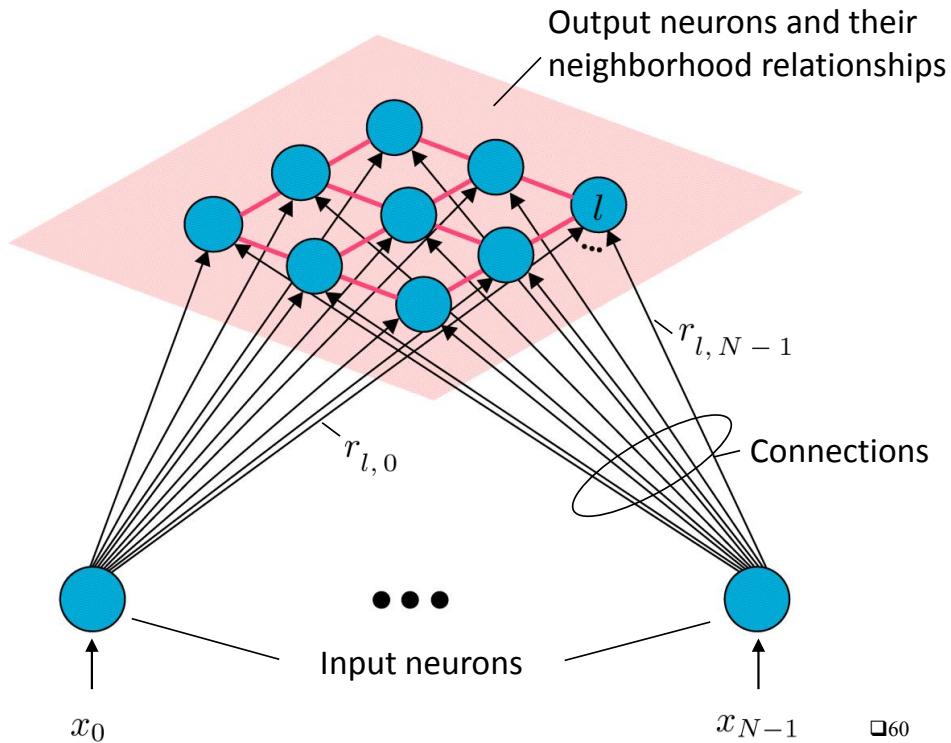
- Un réseau de neurones non-supervisé à 1 couches – chaque noeud d'entrée est connecté à tous les neurones de sortie
- Principe « *winner-takes-all* »: le SOM active toujours le neurone avec la plus grande similarité (plus petite distance) au patron d'entrée
- La distance entre patrons d'entrée et prototypes sert comme fonction d'entrée
- La fonction d'activation est radiale, i.e., un fonction qui décroît de façon monotone:

$$f_{\text{act}}^{(l)} : \mathbb{R}^+ \rightarrow [0, 1] \quad f_{\text{act}}^{(l)}(0) = 1$$

Réseaux auto-organisateurs

SOM – Fondements

► Architecture simplifiée:



Réseaux auto-organisateurs

SOM - Fondements

La performance d'un SOM est influencée par sa topologie:

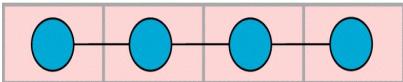
- Les neurones de sortie sont interconnectés par des relations de voisinage
- Ces relations, appelées topologies, sont décrites par une fonction de distance
- SOM apprend des prototypes de catégorie et une topologie (relation de voisinage entre le neurones de sortie)
- Sans relations de voisinage, le SOM se comporte comme un réseau compétitif régulier dans l'espace des patrons d'entrée

Réseaux auto-organisateurs

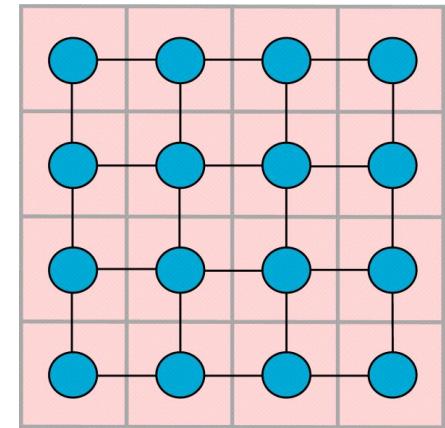
SOM – Fondements

► Voisinage des neurones de sorties: exemples de topologies communes

topologie 1-D



topologie 2-D



- lignes noires: voisins d'un neurone
- lignes grises: contour de régions associé à un neurone

Il est possible d' avoir des topologies avec plus de dimensions mais ceci augmentent la complexité de visualisation

Réseaux auto-organisateurs

SOM - Règle d'apprentissage (séquentiel)

1. **Initialisation:** choix aléatoire de prototypes pour L_{out} catégories, $\mathbf{r}_l = (r_{l,0}, r_{l,1}, \dots, r_{l,N-1})$ avec $l = 1, 2, \dots, L_{\text{out}}$
2. **Attribution:** Pour un vecteur d'entrée $\mathbf{x} = (x_1, x_2, \dots, x_{N-1})$ de D_n , calculer la distance à chaque prototype l de catégorie. Choisir le neurone gagnant avec la distance minimum:

$$l_w = \operatorname{argmin}\{\|\mathbf{x}(t) - \mathbf{r}_l(t)\|^2 : l = 1, 2, \dots, L_{\text{out}}\}$$

3. **Apprentissage:** Tous les prototypes sont adaptées selon la fonction de voisinage $f_{\text{nb}}(\mathbf{x})$ et la règle:

$$\mathbf{r}_l(t+1) = \mathbf{r}_l(t) + \mu(t) f_{\text{nb}} \left(d_{\text{nb}}^{(l, l_w)}(t), \rho(t) \right) (\mathbf{x}(t) - \mathbf{r}_l(t))$$

4. **Critère d'arrêt:** revenir à l'étape 2 si le nombre d'itérations n'est pas atteint, ou si la valeur des prototypes \mathbf{r}_l change par rapport à la dernière itération

Réseaux auto-organisateurs

SOM - Règle d'apprentissage (séquentiel)

Fonction de voisinage ou de topologie: $f_{\text{nb}}(x)$

- Représente les relations de voisinage entre les neurones de sortie (catégories)
- Cette fonction est définie sur une grille (pas dans l'espace des entrées)
- Elle doit être une fonction radiale uni modale qui atteint une valeur maximum pour le neurone $l = l_w$
- Le taux d'apprentissage $\mu(t)$ et le rayon du voisinage $\rho(t)$ doivent diminuer à chaque itération t selon, e.g.:

$$\mu(t) = \mu_0 \alpha_\mu^t, \quad 0 < \alpha_\mu < 1$$

$$\rho(t) = \rho_0 \alpha_\rho^t, \quad 0 < \alpha_\rho < 1$$

Réseaux auto-organisateurs

SOM - Règle d'apprentissage (séquentiel)

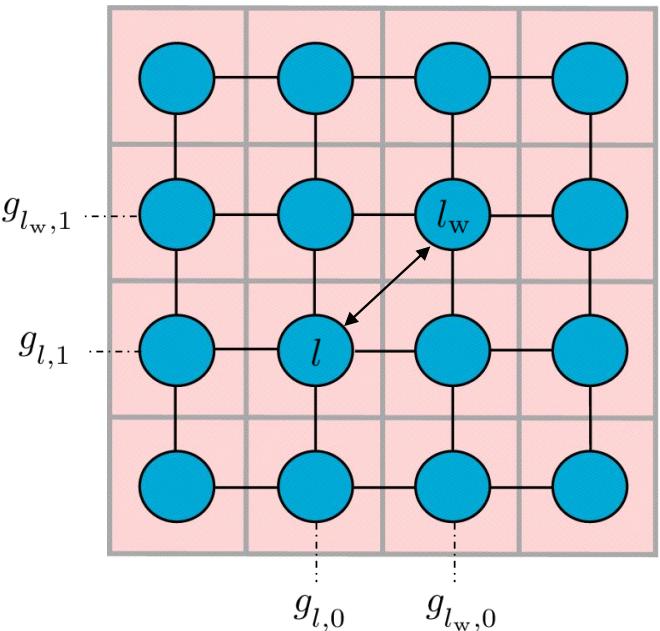
Fonction de voisinage ou de topologie: $f_{\text{nb}}(x)$

- Une fonction commune est la fonction Gaussienne suivante:

$$f_{\text{nb}} \left(d_{\text{nb}}^{(l, l_w)}(t), \rho(t) \right) = \exp \left(\frac{\| \mathbf{g}_l - \mathbf{g}_{l_w}(t) \|^2}{2 \rho(t)^2} \right),$$

$$\mathbf{g}_l = [g_{l,0}, g_{l,1}]^T, l \in L_{\text{out}}$$

g_l représente la position du neurone sur la grille (pas l'espace d'entrée), et $g_{l_w}(t)$ indique la position du neurone gagné à l'itération t

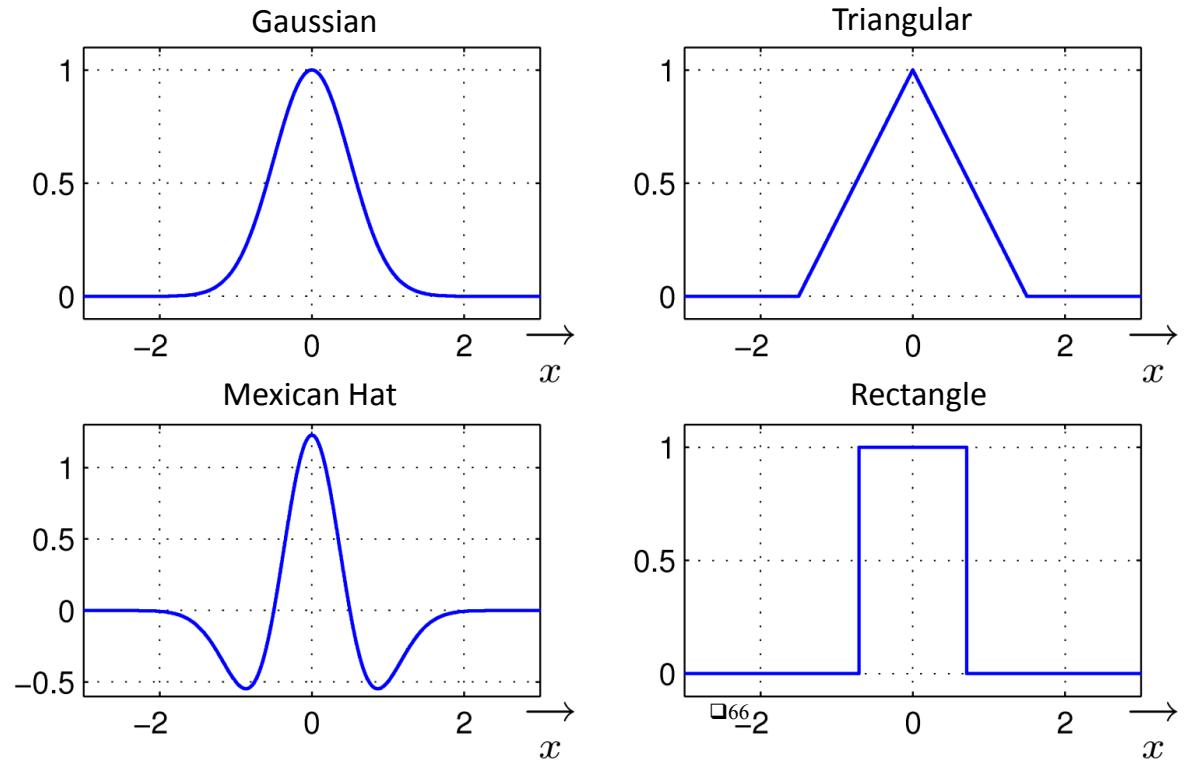


Réseaux auto-organisateurs

SOM - Règle d'apprentissage (séquentiel)

Exemple: $f_{\text{nb}}(x)$

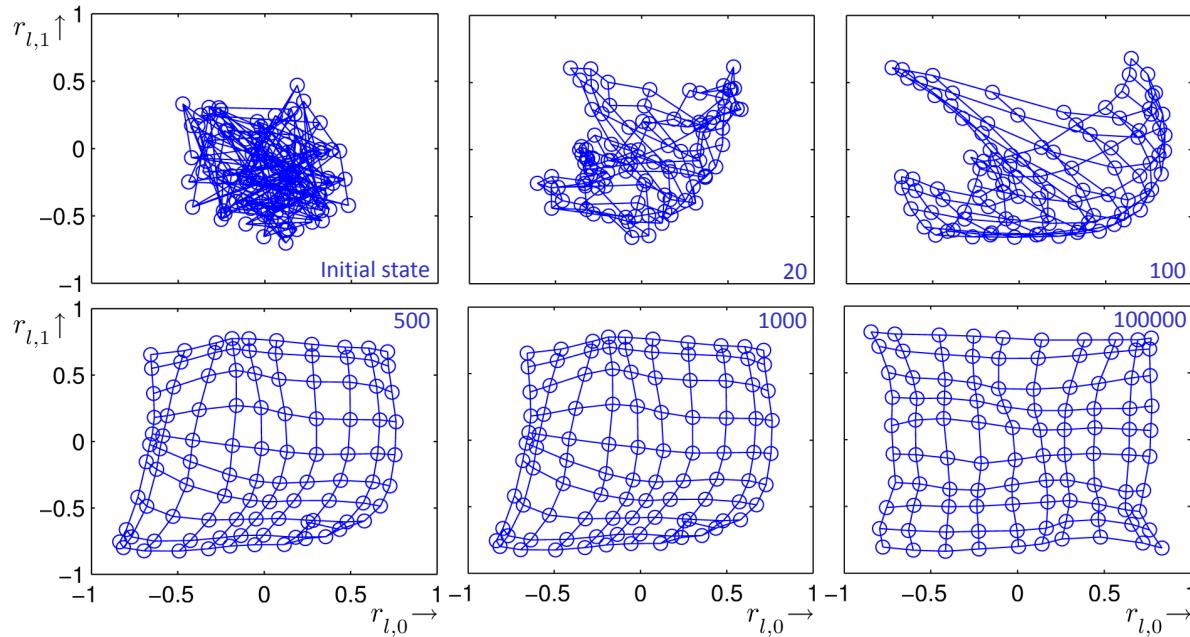
► Fonction communes:



Réseaux auto-organisateurs

SOM - Exemple avec topologie 2D

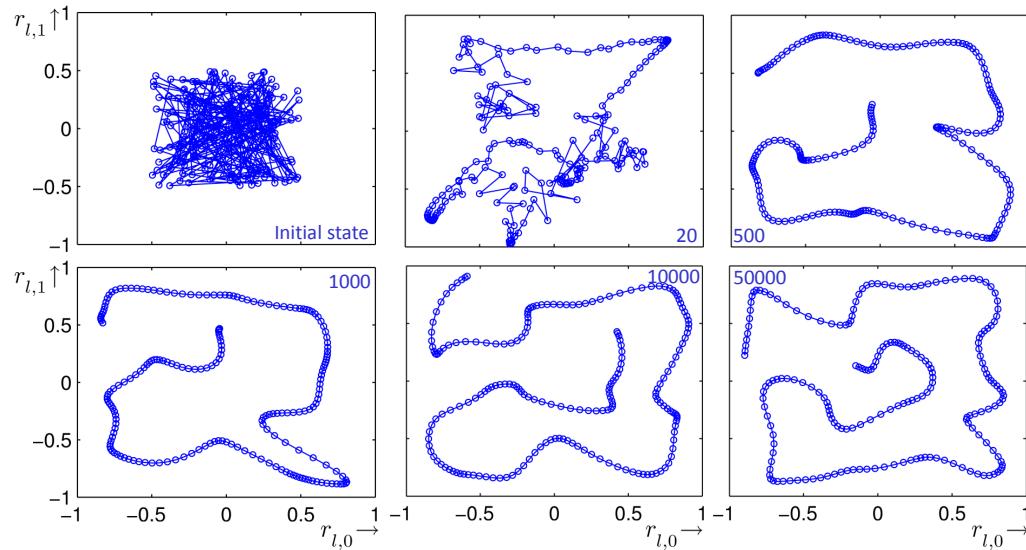
- Topologie: 10x10 neurones de sortie et fonction Gaussienne
- Initialisation aléatoire des prototypes
- Choix aléatoire de patrons d'entrées distribuées uniformément dans l'intervalle $[-1, 1]^2$



Réseaux auto-organisateurs

Exemple avec topologie 1D

- **Topologie:** 10x10 neurones de sortie et fonction Gaussienne



- **Limitations:** le SOM ne converge pas s'il est mal initialisé:
- le taux d'apprentissage initial trop petit
 - le rayon initial pour la fonction de voisinage est trop petit

Références

- ▶ Roth et al., Optimal cluster preserving embedding of nonmetric proximity data, IEEE TPAMI 2003
- ▶ Dhillon et al.. Kernel k-means, spectral clustering and normalized cuts. In KDD, 2004.
- ▶ Comaniciu and Meer. Mean shift: a robust approach toward feature space analysis. IEEE TPAMI 2002
- ▶ Girolami. Mercer kernel-based clustering in feature space. IEEE TNN, 2002.
- ▶ M. Tang, I. B. Ayed, D. Marin, and Y. Boykov. Secrets of grabcut and kernel k-means. In ICCV 2015.
- ▶ Shi and Malik. Normalized cuts and image segmentation, IEEE TPAMI 2000