

# Classification automatique

## Chapitre 4 : Cartes Auto-Organisatrices

Allou Samé  
allou.same@ifsttar.fr

# Plan

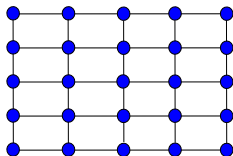
- 1 Objectifs
- 2 Structure d'une carte topologique
- 3 Rappel : version online des k-means
- 4 Algorithme SOM
- 5 Fonction de voisinage
- 6 Exemple illustratif
- 7 Représentations graphiques
- 8 Algorithme SOM dans R

# Carte topologique de Kohonen ou Self Organising Map

## Objectifs

Associer des données multidimensionnelles aux nœuds d'une grille régulière (généralement 1D ou 2D)

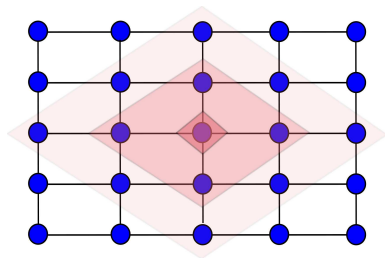
- Réduction de la dimension des données
- Respect de la topologie des données à travers la grille  
→ des données voisines correspondent à des nœuds voisins



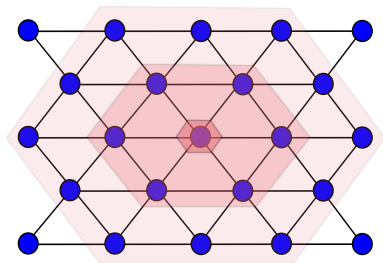
## Autres synonymes

- Carte auto-organisatrice ou réseau auto-adaptatif
- **SOM (Self Organizing Map)**

## Exemple de topologies et voisinages associés



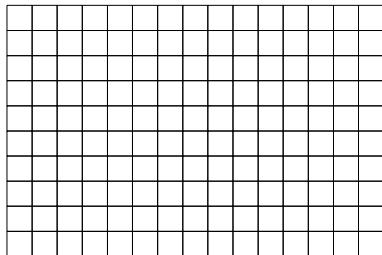
Grille rectangulaire



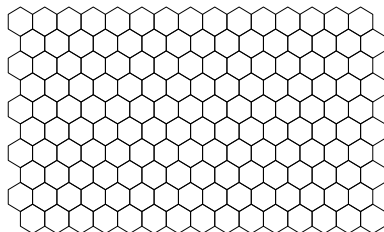
Grille hexagonale

# Exemple de topologies et voisinages associés

Représentation équivalente



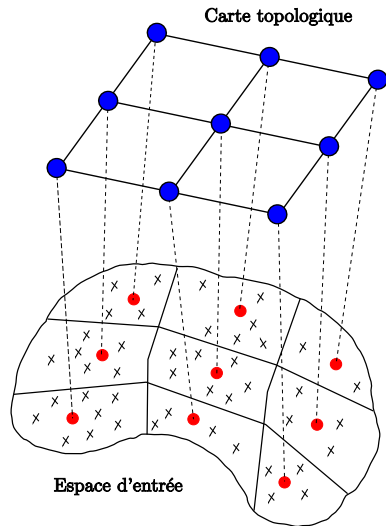
Grille rectangulaire



Grille hexagonale

# Structure d'une carte topologique

- **Espace d'entrée** : espace dans lequel se trouvent les données d'origine ( $\mathbb{R}^p$ )
- **Carte topologique** : grille décrivant les données ; appelée aussi **espace de sortie**
- A chaque nœud de la carte topologique est associé un **représentant ou centre** dans l'espace d'entrée



## Rappel : version online des k-means

- Choix de  $K$  centres au hasard :  $\mu_1(0), \dots, \mu_K(0)$
- Dès qu'un nouveau point  $\mathbf{x}_{t+1}$  est présenté :
  - affecter  $\mathbf{x}_{t+1}$  à la classe dont le centre est le plus proche (soit  $\mu_{k^*}(t)$  ce centre et  $n_{k^*}(t)$  l'effectif de sa classe)
  - mettre à jour le centre (gagnant) et l'effectif associé :

$$\begin{aligned}\mu_{k^*}(t+1) &= \mu_{k^*}(t) + \frac{1}{n_{k^*}(t) + 1} (\mathbf{x}_{t+1} - \mu_{k^*}(t)) \\ n_{k^*}(t+1) &= n_{k^*}(t) + 1\end{aligned}$$

- poser  $t = t + 1$

## Convergence

- Assurée quand le nombre de données tend vers l'infini, contrairement à l'algo. k-means classique
- Assurée dans le cadre des algorithmes de gradient stochastique :

$$\mu_k(t+1) = \mu_k(t) + \varepsilon_t (\mathbf{x}_{t+1} - \mu_k(t)),$$

avec  $\sum_{t=1}^{\infty} \varepsilon_t = \infty \quad \sum_{t=1}^{\infty} \varepsilon_t^2 < \infty$

# L'algorithme SOM (version online)

## ■ **Initialisation :**

- choix de la taille de la grille ( $K_1$  lignes et  $K_2$  colonnes)
- choix aléatoire des centres  $\mu_1, \dots, \mu_k, \dots, \mu_K$  avec  $K = K_1 \times K_2$

## ■ **Répéter tant que le nombre d'itérations maxi n'est pas atteint :**

- choix aléatoire d'un point  $\mathbf{x}$  dans le jeu de données
- **Phase de compétition :**  
Affecter  $\mathbf{x}$  au nœud dont le représentant  $\mu_{k^*}$  dans l'espace d'entrée est le plus proche de  $\mathbf{x}$  au sens de la distance euclidienne

$$k^* = \arg \min_k \|\mathbf{x} - \mu_k\|^2$$

→ le centre  $\mu_{k^*}$  est appelé **centre gagnant**

## ■ **Phase de coopération :**

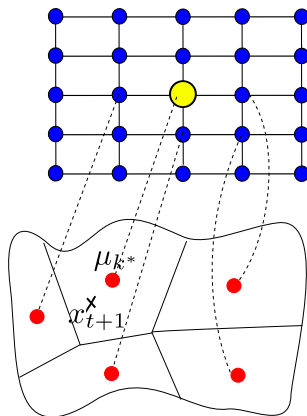
Mettre à jour le centre  $\mu_{k^*}$  et ses voisins

$$\mu_k \leftarrow \mu_k + \varepsilon_t h(k, k^*) (\mathbf{x} - \mu_k)$$

où  $h(k, k^*)$  est la **fonction de voisinage** et  $\varepsilon_t$  est le **pas d'apprentissage**.

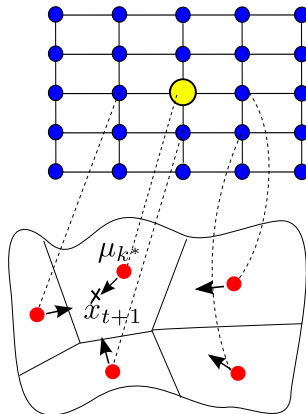


## Algorithme SOM : phase de compétition



Sélection du nœud associé au centre le plus proche de  $x$   
(centre gagnant  $\mu_{k^*}$ )

## Algorithme SOM : phase de collaboration



Le centre gagnant  $\mu_{k^*}$  et les autres centres sont rapprochés du point  $x$  avec une amplitude qui dépend de leur proximité (dans l'espace de sortie) avec le centre gagnant

## Fonction de voisinage $h(k, k^*)$

### Noyau gaussien

$$h(k, k^*) = \exp \left( -\frac{\|c_{k^*} - c_k\|^2}{2\sigma^2} \right)$$

- $\|\cdot\|$  désigne la distance euclidienne
- $c_k$  et  $c_{k^*}$  sont les coordonnées de  $\mu_k$  et  $\mu_{k^*}$  dans la grille
- $\sigma > 0$  permet de régler l'amplitude du voisinage de  $c_{k^*}$ .

# Fonction de voisinage $h(k, k^*)$

## Noyau gaussien

$$h(k, k^*) = \exp \left( -\frac{\|c_{k^*} - c_k\|^2}{2\sigma^2} \right)$$

- $\|\cdot\|$  désigne la distance euclidienne
- $c_k$  et  $c_{k^*}$  sont les coordonnées de  $\mu_k$  et  $\mu_{k^*}$  dans la grille
- $\sigma > 0$  permet de régler l'amplitude du voisinage de  $c_{k^*}$ .

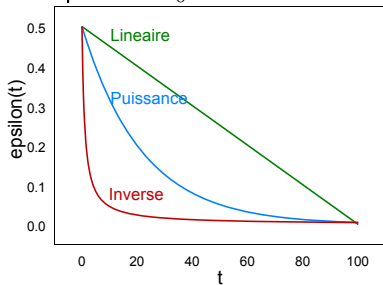
## Remarque

- $h(k, k^*)$  est une fonction décroissante de la distance entre  $c_k$  et  $c_{k^*}$
- $h(k^*, k^*) = 1$
- $h(k, k^*)$  tend vers 0 quand  $c_k$  et  $c_{k^*}$  sont très éloignés

## Pas (ou taux) d'apprentissage $\varepsilon_t$

- Le pas  $\varepsilon_t$  est une fonction décroissante du nombre d'itérations  $t$ , telle que  $0 \leq \varepsilon_t \leq 1$
- Les pas généralement utilisés sont :
  - fonction linéaire  $\varepsilon_t = \varepsilon_0(1 - \frac{t}{T})$
  - fonction puissance  $\varepsilon_t = \varepsilon_0 \exp(-\frac{t}{T})$
  - fonction inverse  $\varepsilon_t = \frac{\varepsilon_0}{1 + \frac{t}{T}}$

Exemple avec  $\varepsilon_0 = 0.5$  et  $T = 100$

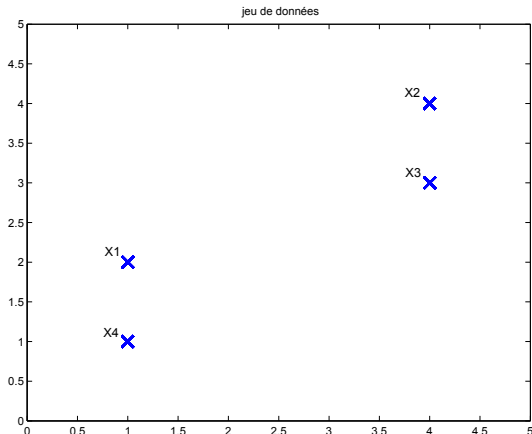


où  $\varepsilon_0$  est un pas initial et  $T$  est le nombre total d'itérations

## Exemple illustratif

Jeu de données : 4 points de  $\mathbb{R}^2$

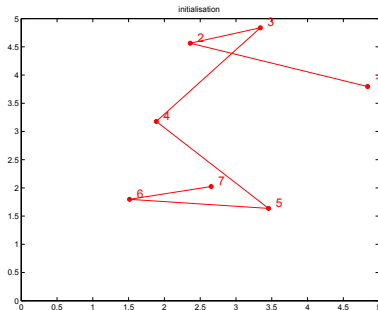
$\mathbf{x}_1 = (1 \ 2)$ ,  $\mathbf{x}_2 = (4 \ 4)$ ,  $\mathbf{x}_3 = (4 \ 3)$ ,  $\mathbf{x}_4 = (1 \ 1)$  formant deux groupes



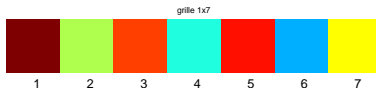
On se propose de décrire ces données à l'aide d'une grille rectangulaire  $1 \times 7$ .

# Exemple illustratif

Initialisation des centres dans l'espace d'entrée



Grille dans l'espace de sortie

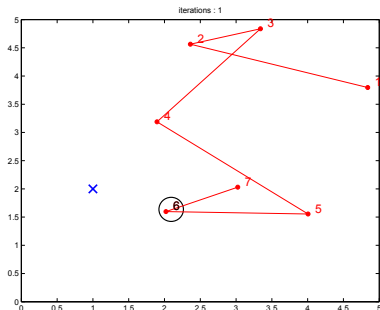


abscisse des points codée en couleur : bleu (0) → jaune (2.5) → rouge (5)

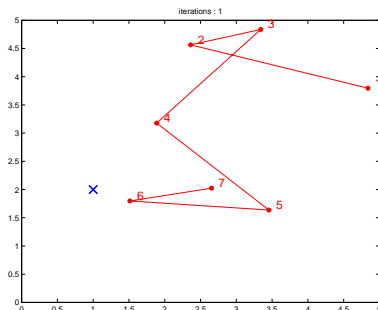
# Exemple illustratif

Tirage aléatoire d'un point : soit  $x_1$  ce point

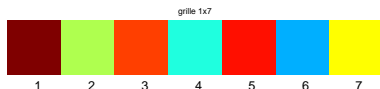
Phase de compétition  
centre gagnant :  $g_6$



Phase de coopération  
mise à jour des centres



Grille dans l'espace de sortie

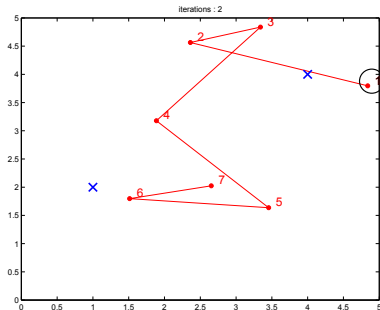




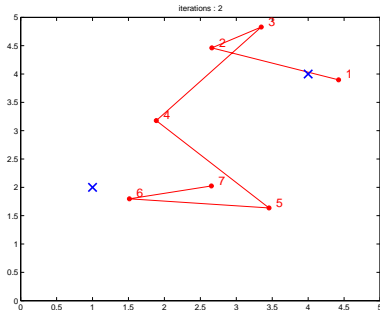
# Exemple illustratif

Tirage aléatoire d'un point : soit  $x_2$  ce point

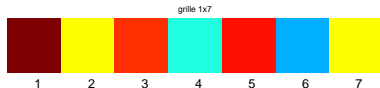
Phase de compétition  
centre gagnant :  $g_1$



Phase de coopération  
mise à jour des centres



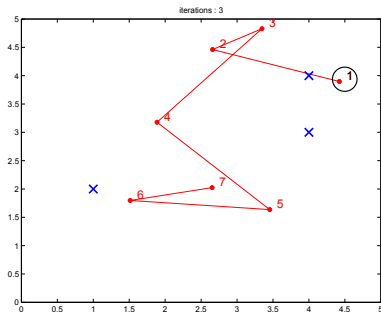
Grille dans l'espace de sortie



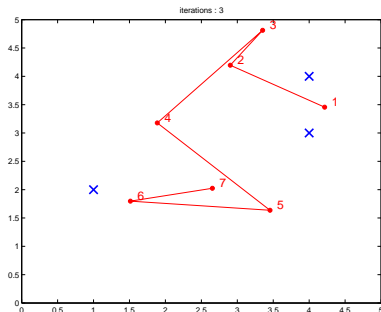
# Exemple illustratif

Tirage aléatoire d'un point : soit  $x_3$  ce point

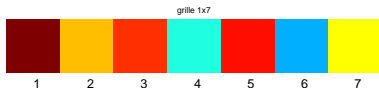
Phase de compétition  
centre gagnant :  $g_1$



Phase de coopération  
mise à jour des centres



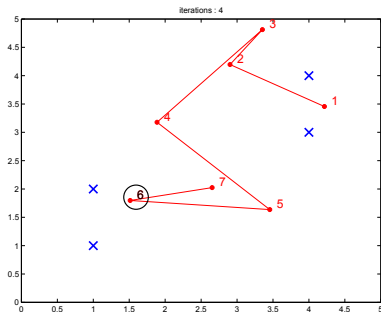
Grille dans l'espace de sortie



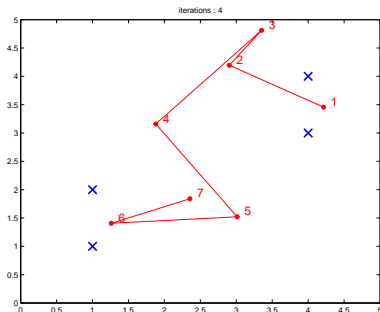
# Exemple illustratif

Tirage aléatoire d'un point : soit  $\mathbf{x}_4$  ce point

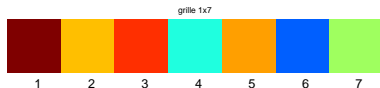
Phase de compétition  
centre gagnant :  $\mathbf{g}_6$



Phase de coopération  
mise à jour des centres

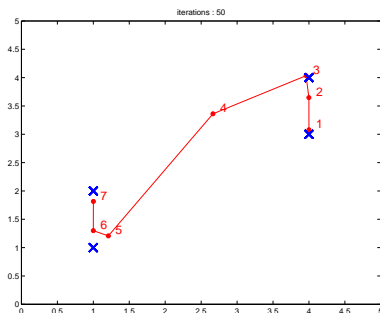


Grille dans l'espace de sortie

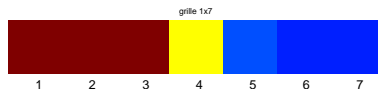


# Exemple illustratif

Centres obtenus au 50<sup>e</sup> tirage (après la phase de coopération)

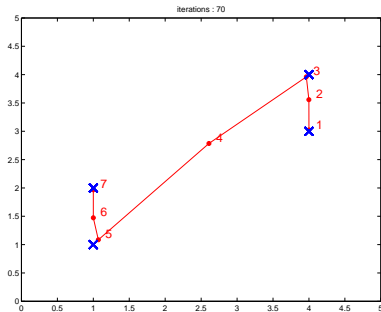


Grille dans l'espace de sortie



## Exemple illustratif

Centres obtenus au 70<sup>e</sup> tirage (après la phase de coopération)



Grille dans l'espace de sortie



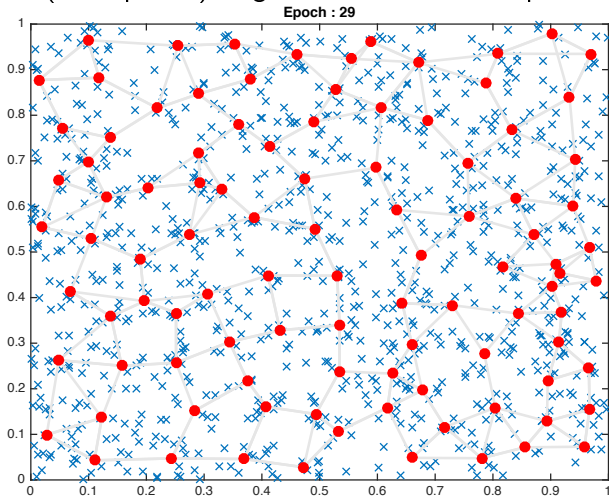
On retrouve bien la structure initiale (deux classes) des données

# Utilisation des cartes de Kohonen en classification automatique

- L'algorithme SOM fonctionne en affectant itérativement chaque individu (de l'espace d'entrée) à un nœud de la carte topologique.
- Ainsi chaque nœud de la grille peut être considéré comme une classe.
- Néanmoins, si l'objectif est de classifier les observations, ceci peut conduire généralement à un trop grand nombre de classes. Dans ce cas, les nœuds de la grille doivent être eux-aussi regroupés en classes.
- La mise en œuvre de l'algorithme SOM nécessite généralement un re-bouclage cyclique sur le même ensemble de données pour que l'algorithme converge.

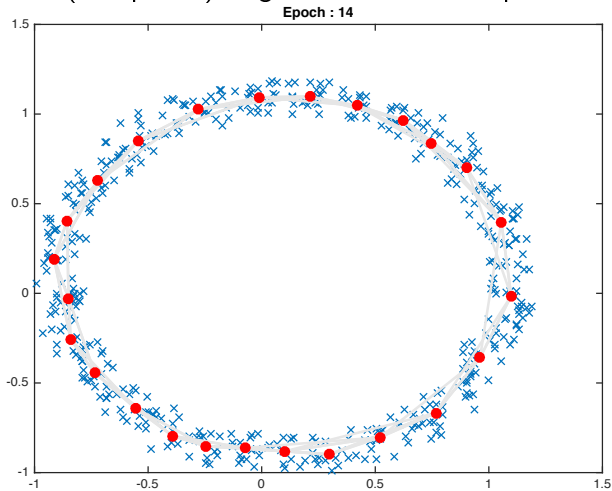
# Exemple sur des données uniformément distribuées dans le plan

Données (1000 points) et grille  $10 \times 10$  dans l'espace d'entrée



## Exemple sur des données en forme de cercle

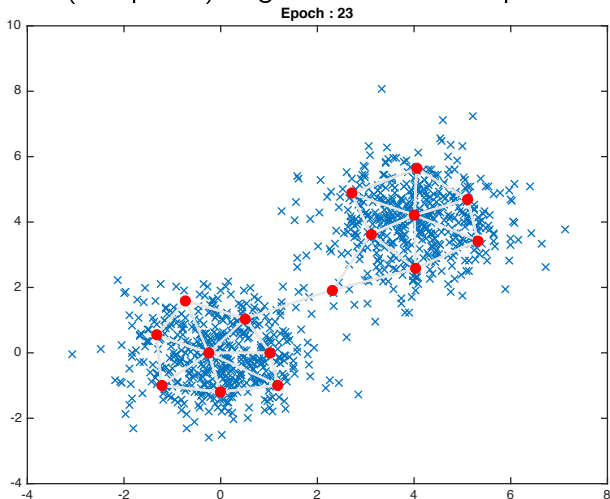
Données (500 points) et grille  $5 \times 5$  dans l'espace d'entrée





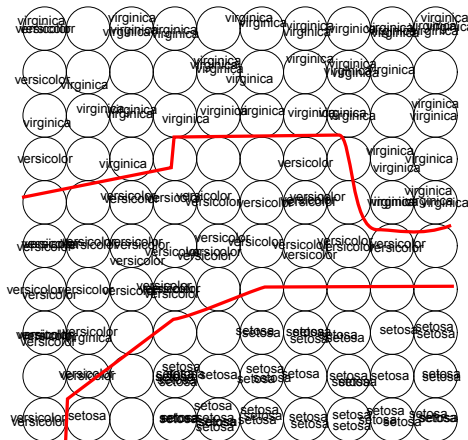
# Exemple sur des données constituées de classes

Données (500 points) et grille  $4 \times 4$  dans l'espace d'entrée



# Exemple des données IRIS de Fisher

Visualisation des données sur une grille (à l'intérieur des grilles les vrais labels des données sont affichés)

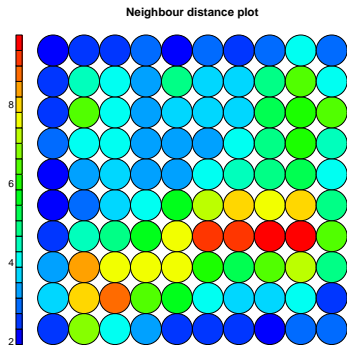


## Visualisation des résultats : Unified-Distance Matrix ou U-matrix

- Chaque entrée de la matrice représente la moyenne des distances euclidiennes avec ses neurones adjacents.
- Une valeur faible indique que les nœuds voisins sont proches dans l'espace d'entrée alors qu'une valeur élevée indique des nœuds voisins éloignés dans l'espace d'entrée, même s'ils sont proches dans l'espace de sortie.

# Visualisation des résultats : Unified-Distance Matrix ou U-matrix

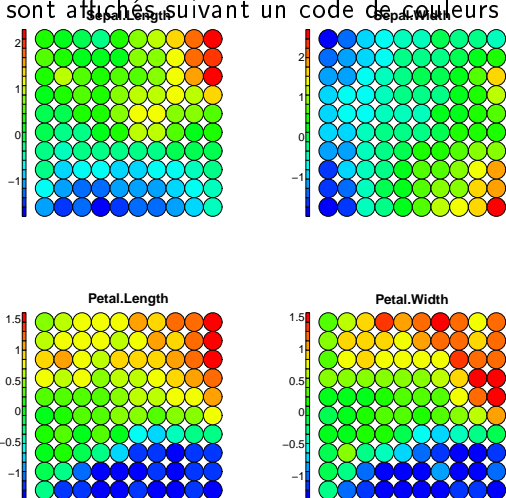
U-matrix obtenue pour les donnée IRIS



La bande jaune-rouge représente une frontière entre classes (espèces).

# Visualisation des résultats : cartes par variable appelées aussi « components planes »

Pour chaque variable, les coordonnées des centres obtenus à l'issue de l'algorithme SOM sont affichés suivant un code de couleurs



## Différences par rapport à l'algorithme des k-means

- L'algorithme SOM fait penser à la version séquentielle de l'algorithme des k-means.
- Dans la méthode des k-means, l'introduction d'une nouvelle observation conduit au recalcul du centre de sa classe.
- Dans l'algorithme SOM, on modifie non seulement le centre gagnant, mais aussi les centres voisins.
- Contrairement à la méthode des k-means, la méthode de Kohonen opère en plus une réduction du nombre de dimensions de l'espace d'entrée, comme dans l'ACP.
- A la différence de l'ACP, la projection résultante est non linéaire.

# Convergence de l'algorithme SOM

- Pas de résultats théoriques sur la convergence de l'algorithme SOM.
- Cependant si la distribution des données est discrète, on peut montrer que l'algorithme SOM est un algorithme de gradient stochastique qui minimise, quand le nombre d'itérations tend vers l'infini, un critère analogue à l'inertie intraclasse.

## Algorithme SOM dans R

```
library(class)
library(MASS)
library(kohonen)

x = scale(iris[,-5], center = TRUE, scale = TRUE)

SOM.IRIS = som(x, grid = somgrid(10,10, "rectangular"))
SOM.IRIS$codes # coordonnees des centres (espace d'entrée)
SOM.IRIS$distances # distances entre chaque donnée et son centre
SOM.IRIS$changes # ecart moyen entre les donnees et les centres

plot(SOM.IRIS, type="mapping", labels=iris[,5], pchs=2)
plot(SOM.IRIS,type="quality") # qualité de représentation
plot(SOM.IRIS,type="counts") # qualité de représentation
plot(SOM.IRIS,type="codes")
```



# Algorithme SOM dans R

```
# Modification de la palette de couleurs (bleu -> rouge)
coolBlueHotRed = function(n, alpha = 1)
{
  rainbow(n, end=4/6, alpha=alpha)[n:1]
}

# U-matrix (matrice de voisinage)
plot(SOM.IRIS, type="dist.neighbours",palette.name=coolBlueHotRed)

# Component planes
par(mfrow=c(2,2))
for (i in 1:4)
{
  plot(SOM.IRIS, type="property", palette.name=coolBlueHotRed,
  property=SOM.IRIS$codes[,i], main=colnames(SOM.IRIS$codes)[i])
}
```