Cartes auto-organisatrices

Nicolas P. Rougier

Master 2 - Sciences Cognitives
Université de Bordeaux

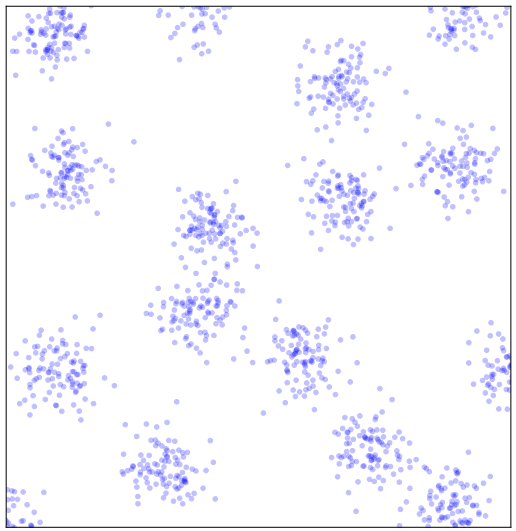
Quantification vectorielle

Soit un ensemble de données dans un espace quelconque

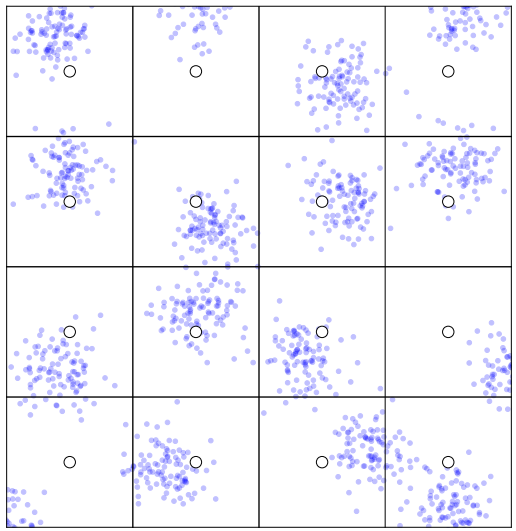
- Comment regrouper les données qui sont similaires ?
- Combien y a t-il de groupes ?
- Quel sont les meilleurs représentants de chaque groupe ?
- Peut-on reconnaître les groupes proches ?

Exemple

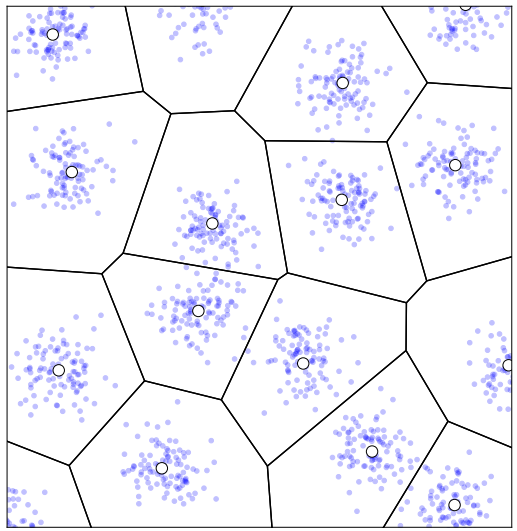
Ensemble de points



Mauvaise quantification



Bonne quantification



Quantification vectorielle

Définition

Soit un ensemble de données E de taille et de dimension quelconque. Une quantification vectorielle de E se définit par une fonction f et un ensemble $Q \subset E$ telle que $\forall x \in E, f(x) \in Q$.

Exemple

On considère des données réelles à une seule dimension ($E = \mathbb{R}$) et on veut les quantifier dans un ensemble Q:

- $Q = \mathbb{N}$
- $f(x) = int(x)$
- $1.2 \rightarrow 1$, $3.9 \rightarrow 3$, ...
- Infinité de représentants

- $Q = \{-1, +1\}$
- $f(x) = sign(x)$
- $1.2 \rightarrow +1$, $-3.9 \rightarrow -1$, ...
- Deux représentants

Problème général

Peut on automatiser la procédure pour des données quelconques ?
(trouver Q et f)

Quantification vectorielle

Algorithmes standards

- Regroupement dynamique (Y. Linde, A. Buzo & R.M. Gray, 1980)
- Moindres carrés (S.P. Lloyd, 1982)
- Growing Neural Gas (Fritzke, 1995)
- Cartes auto-organisatrices (T. Kohonen, 1982)

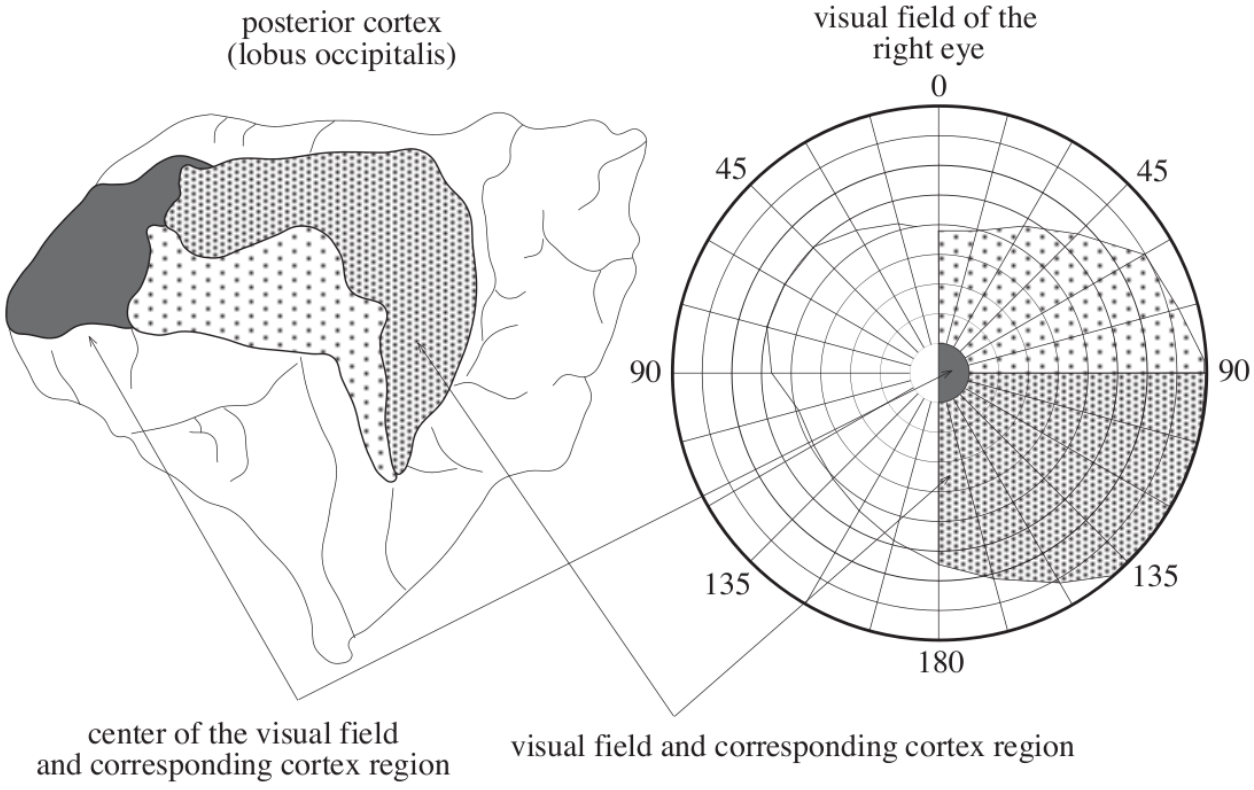
Utilisations standards

- Compression de données
- Classifications de données
- Catégorisation de données

Le cortex visuel

Rétinotopie

Les aires visuelles sont organisées (via l'apprentissage) de telle façon que deux neurones physiquement proches dans le cortex visuel traitent des entrées physiquement proches dans la rétine. On parle d'organisatiuon rétinotopique.



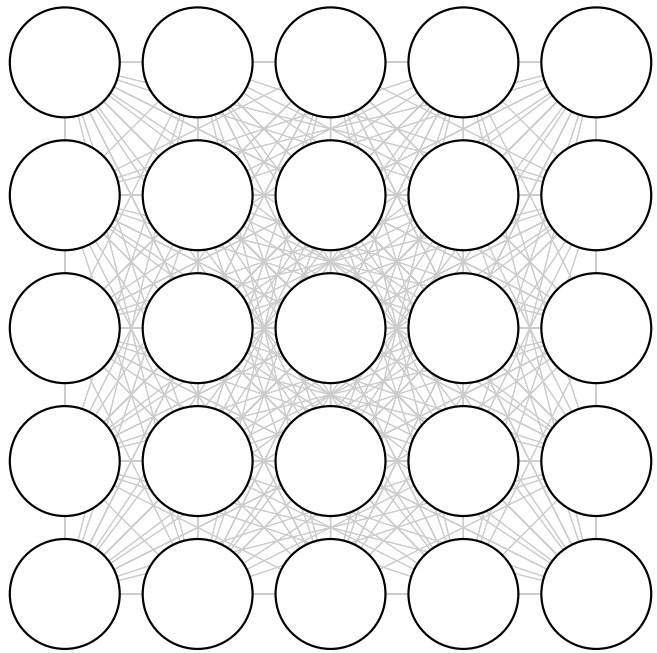
Suite à ces observations, Teuvo Kohonen a cherché à rendre compte de l'organisation spatiale du cortex.

Principe de fonctionnement

- Soit une carte de n neurones entièrement connectés
  (chaque neurone est relié à tous les autres)
- On ajoute une topologie à la carte
  (il y a une notion de distance entre chaque neurone)
- Chaque neurone est relié à l'ensemble des entrées
  (le vecteur de poids est le *prototype* du neurone)
- A chaque nouvelle entrée, le neurone ayant le prototype le plus proche est déclaré vainqueur.
- Les prototypes du vainqueur et de ses voisins sont changés afin de se rapprocher de l'entrée présenté.

Architecture

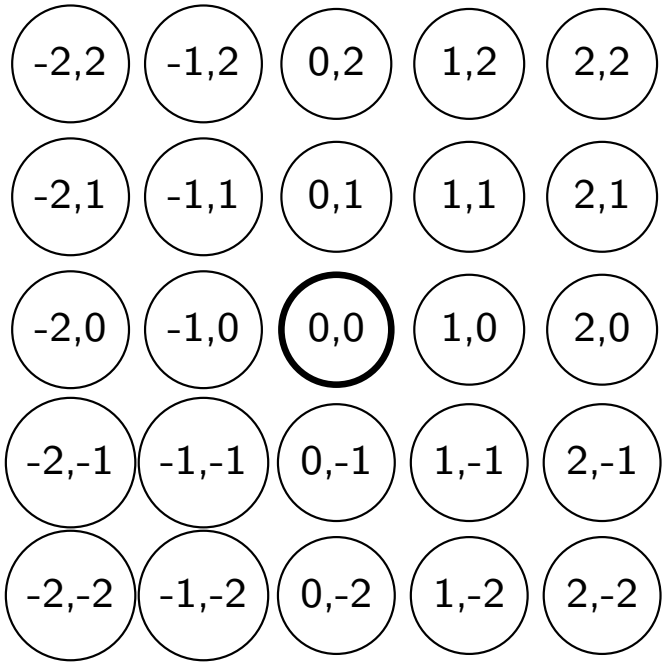
Soit une carte de n neurones entièrement connectés.
$\rightarrow$ chaque neurone est relié à tous les autres



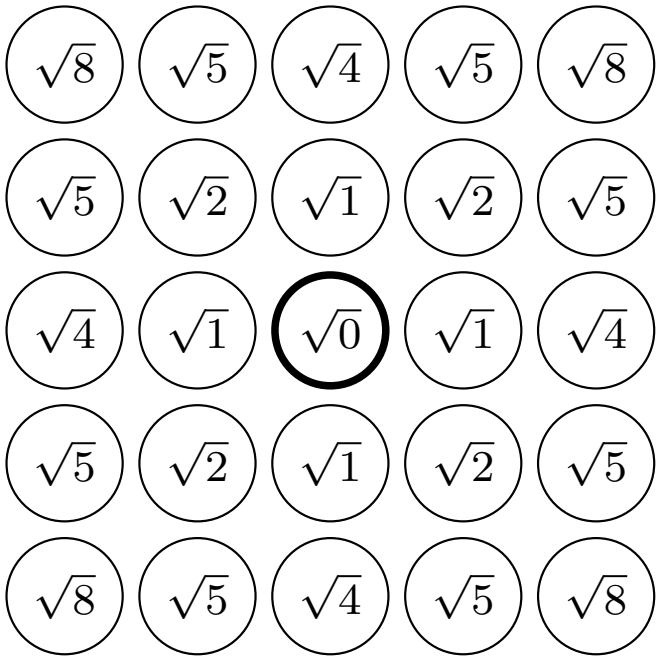
Topologie

On ajoute une topologie à la carte
$\rightarrow$ il y a une notion de distance entre chaque neurone

Coordonnées relatives au neurone (0,0)

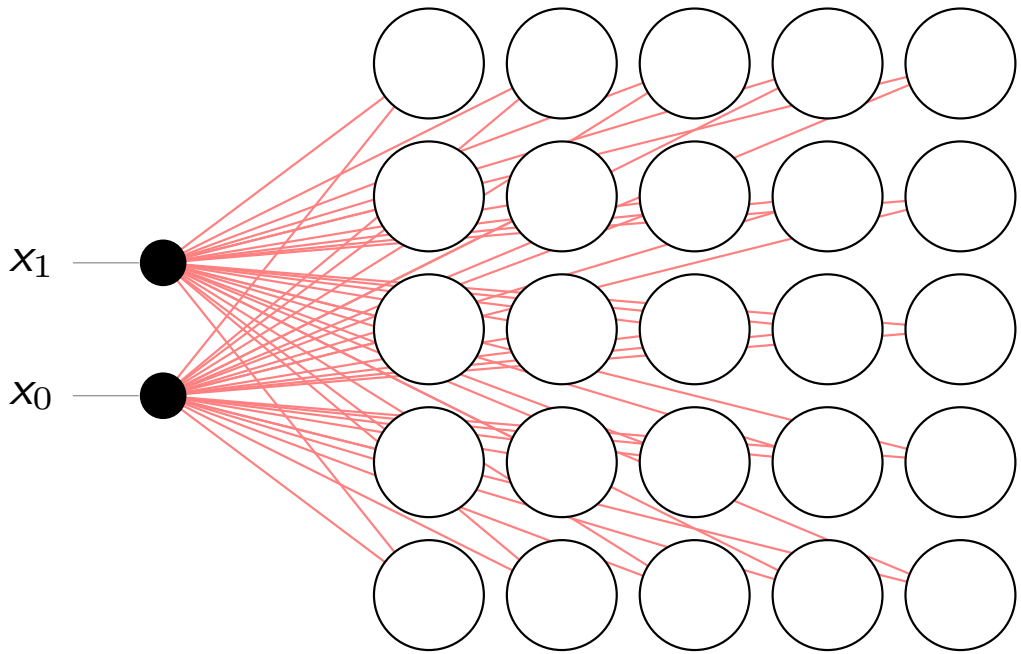


Distances relatives au neurone (0,0)



Entrées

Chaque neurone est relié à l'ensemble des entrées
$\rightarrow$ le vecteur de poids correspondant est le prototype du neurone.



Algorithme

Recherche du vainqueur

Soit une donnée $\mathbf{x} = \{x_1, x_2, ...x_n\}$, on cherche le neurone $i_{vainqueur}$ tel que la distance entre $\mathbf{x}$ et $\mathbf{w_{i_{vainquer}}}$ soit minimale. C'est à dire:

$$i_{vainqueur} = argmin_i(d(\mathbf{x}, \mathbf{w_i}))$$

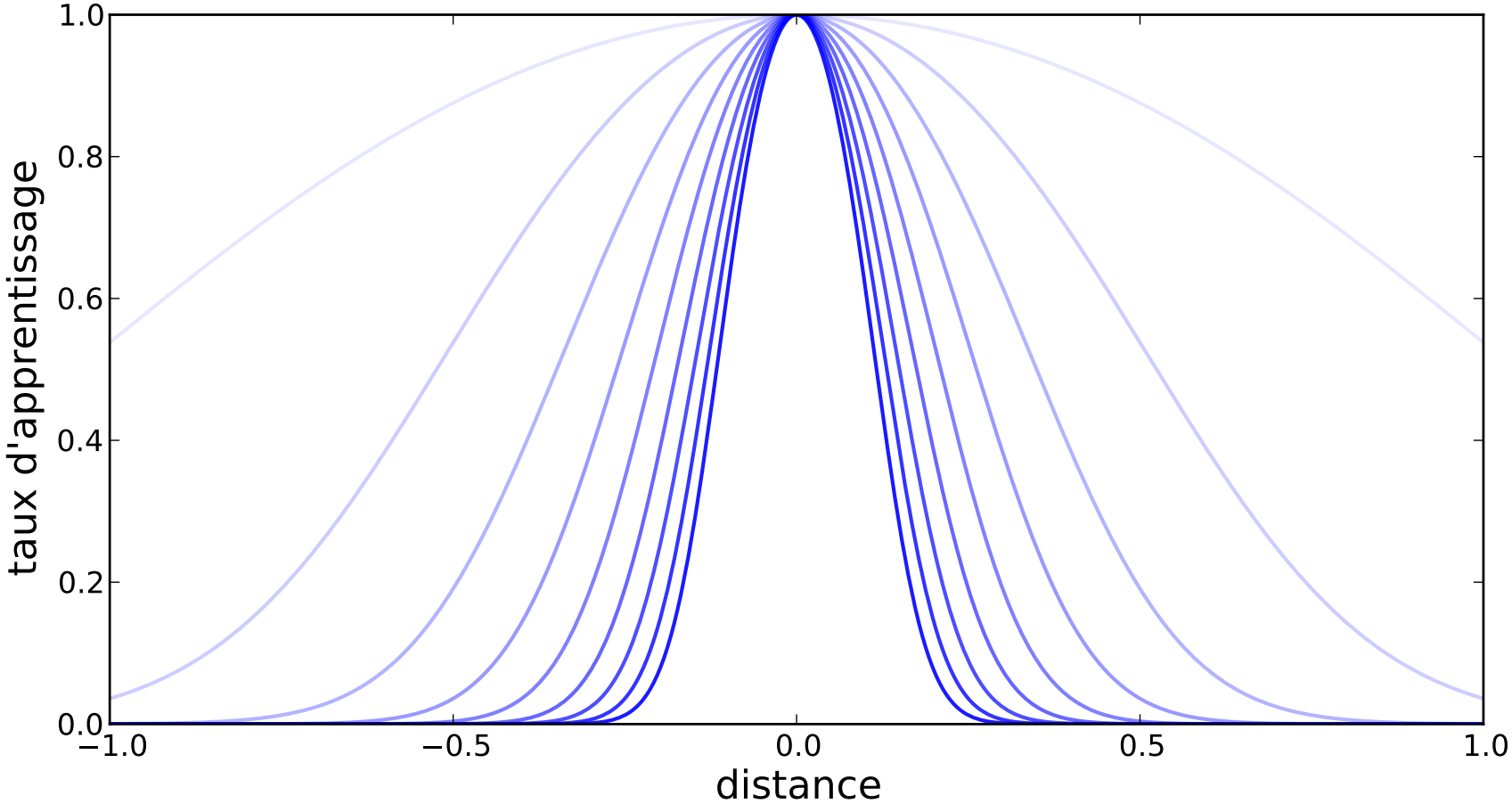
Apprentissage

A chaque exemple présenté, le vainqueur ainsi que ses voisins "les plus proches" vont modifier leur vecteur de poids selon la formule:

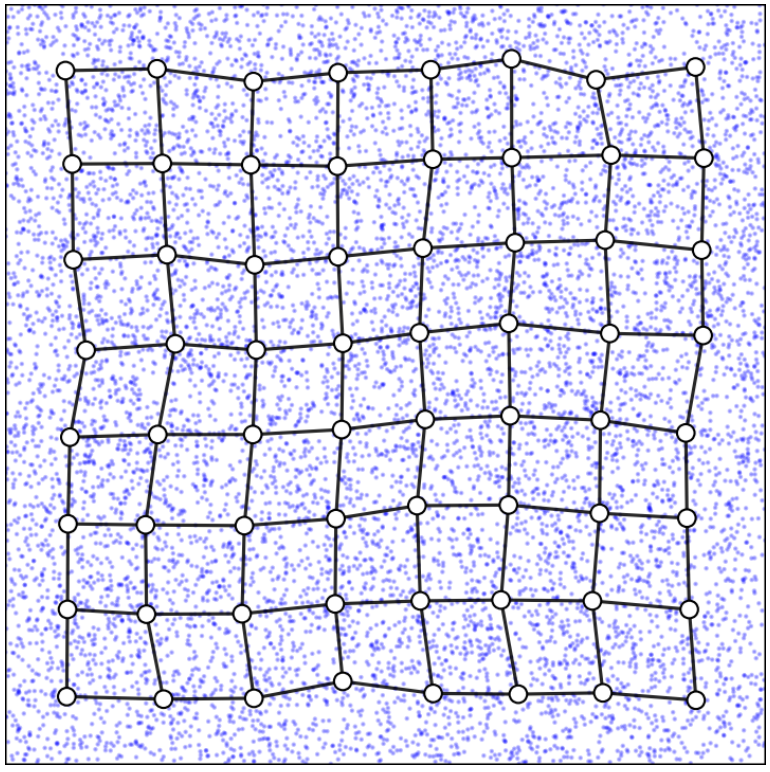
$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \mu f(\mathbf{w}_i(t) - \mathbf{x})$$

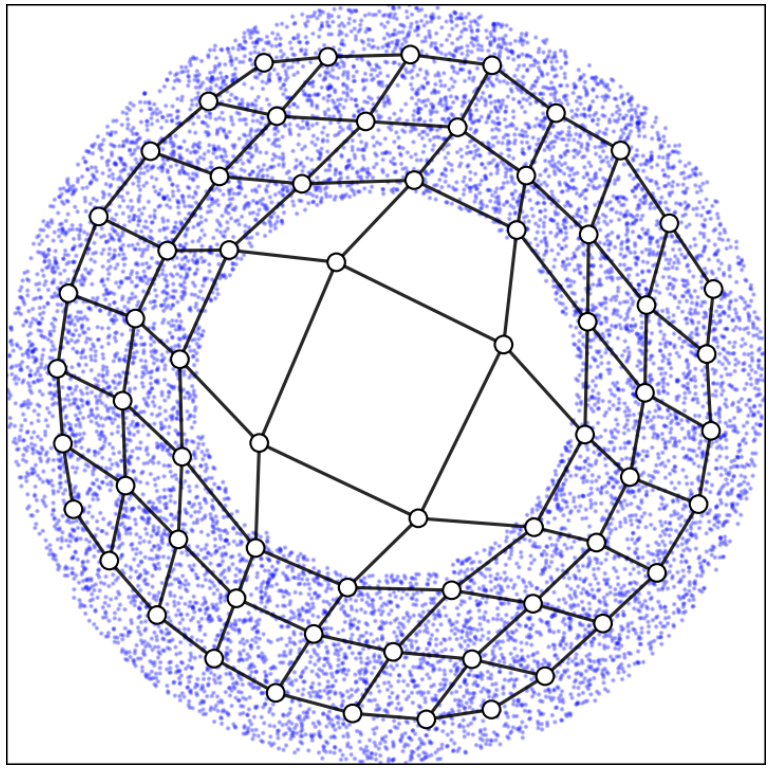
avec f qui est la fonction de voisinage et μ le pas d'apprentissage qui décroit au cours du temps.

Fonction de voisinage

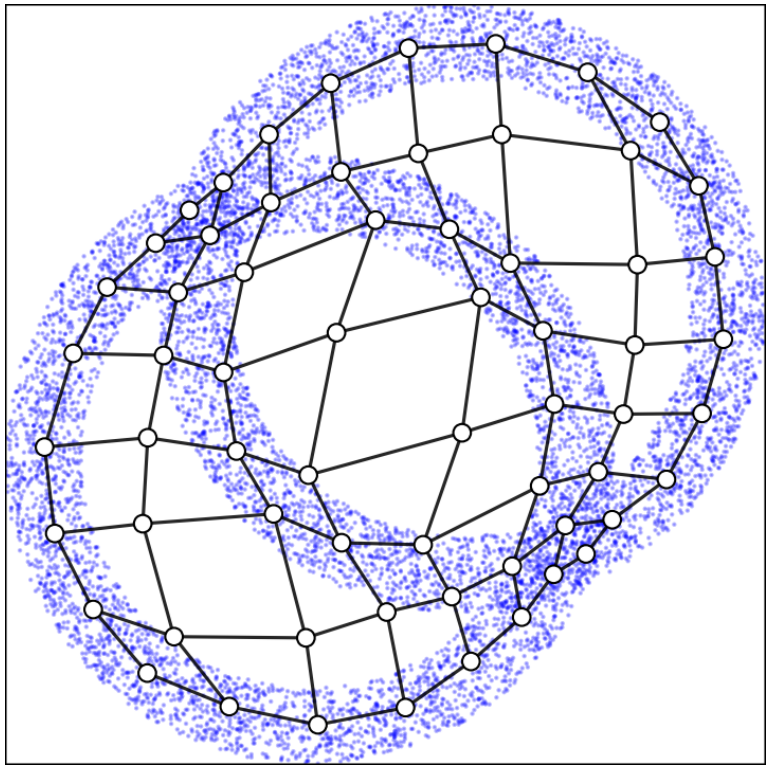


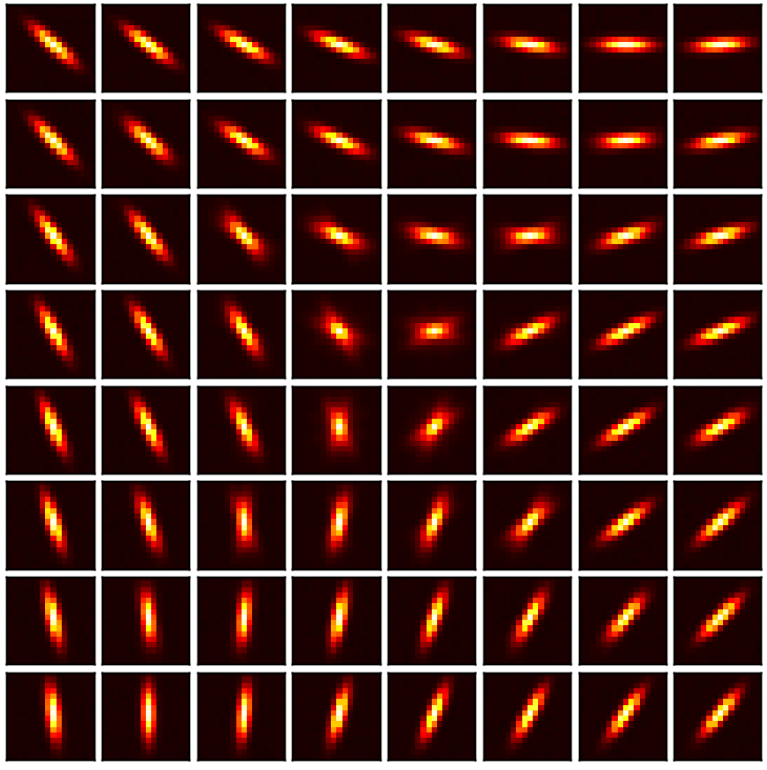
Exemples





Exemples





Difficultés

- La topologie du réseau peut ne pas correspondre à la topologie des données et cela peut poser des problèmes lors de l'apprentissage.
- La carte peut ne pas se déployer correctement en début d'apprentissage et on obtient des "neouds"
- Le réseau est figé après apprentissage puisque le pas d'apprentissage est nul.
- Il faut connaître par avance le temps d'apprentissage, c'est à dire le nombre d'exaples que l'on présente au réseau.

Références

Livres et cours

- Self-Organizing Maps, Third Edition
  Teuvo Kohonen, 2001.
- Some Competitive Learning Methods
  Bernd Fritzke, 1997.
- Neural Computation and Self-Organizing Maps - An Introduction
  Helge Ritter, Thomas Martinetz & Klaus Schulten, 1992.

Démos et Vidéos

- DemoGNG at
  `http://sund.de/netze/applets/gng/full/GNG-U_0.html`
- Dynamic Self Organization at
  `http://www.loria.fr/~rougier/research/DSOM.html`

Exercice 1

Compression d'une image

On souhaite réduire le nombre de couleurs d'une image à l'aide d'une carte de Kohonen. Soit une image constituée de 600x800 pixels décrits dans le codage RGB. Le codage RGB propose de coder sur un octet (i.e. 8 bits) chaque composante de couleur (rouge, vert, bleu).

1. Quel est le nombre total de couleurs que l'on peut virtuellement coder ?
2. On souhaite réduire le nombre de couleurs différentes à 256 couleurs pour diminuer la taille de stockage de l'image. Quelle sera la taille de l'image finale ?
3. Quelle architecture proposez-vous pour réduire automatiquement le nombre de couleurs à l'aide d'une carte auto-organisatrice de Kohonen ?
4. Tester votre architecture

Exercice 2

Nuage de points

On souhaite regrouper des données similaires à partir d'un jeu de données de dimensions 2.

1. Quelle architecture proposez-vous ?
2. Après apprentissage, que représente les poids de chaque neurone ?
3. Si les données ne sont pas homogènes, quelle influence cela a t-il sur l'oargnisation finale ?
4. Tester votre architecture