Classification non supervisée à deux niveaux guidée par le voisinage et la densité

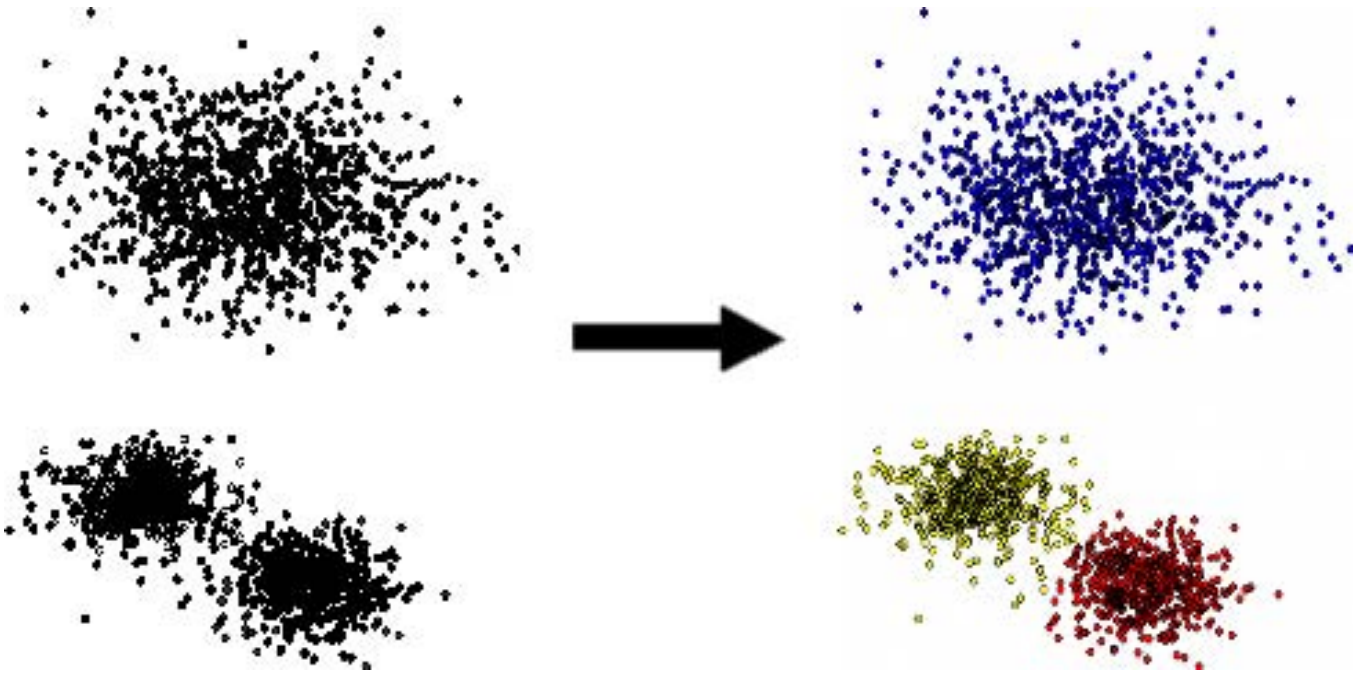
Guénaël Cabanes

LIPN UMR 7030 du CNRS
Equipe A3

Contexte : la classification non supervisée

- Définir sur un ensemble d'objets deux à deux comparables, une partition qui respecte au mieux les ressemblances entre objets.

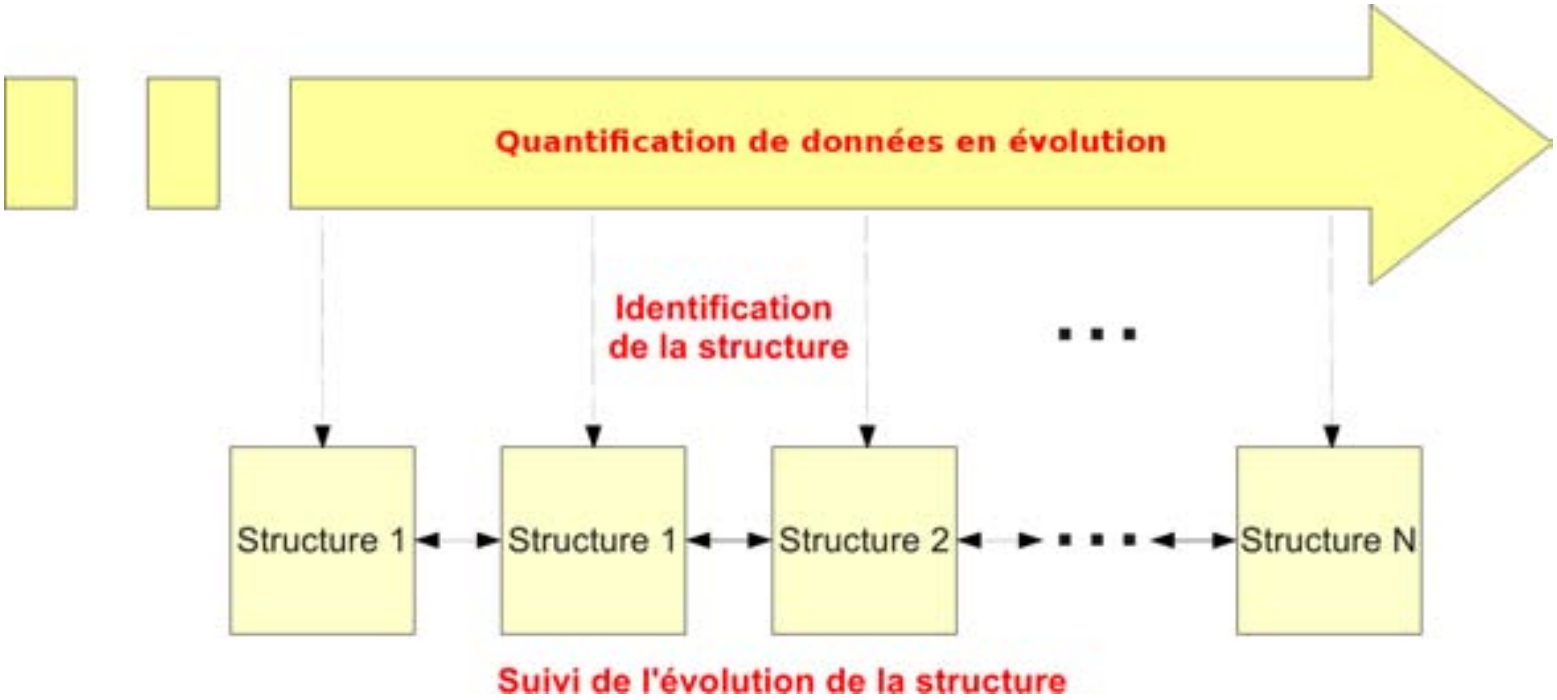
- La ressemblance entre deux objets doit être grande lorsqu'ils figurent dans le même groupe et faible dans le cas contraire.



Objectifs

1. **Existence**
   - Existe-t-il une structure dans les données ?
2. **Identification**
   - Quelle est la structure des données ?
   - Comment choisir le nombre de clusters ?
3. **Suivi**
   - Comment analyser des données en évolution ?
   - Peut-on comparer des structures ?

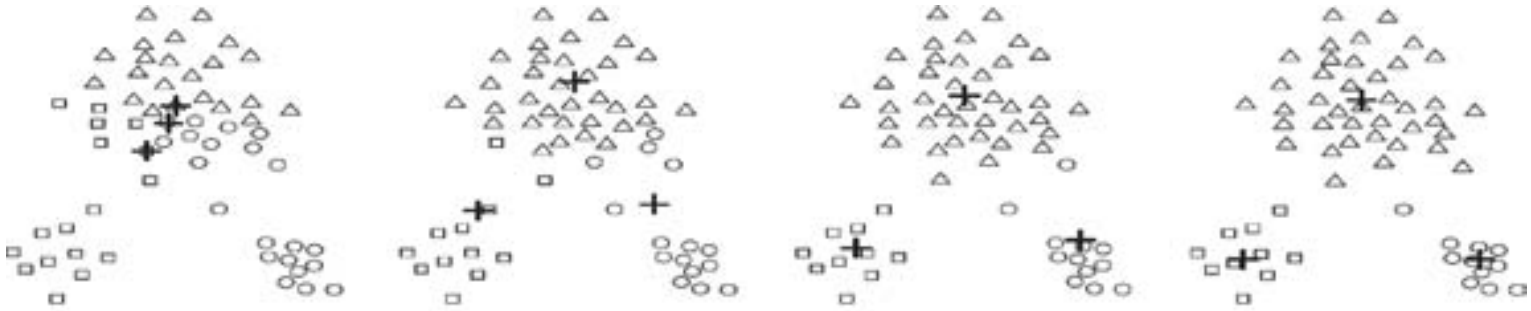
Objectifs



Modèles choisis

Méthodes à base de prototypes

Méthodes à base de prototypes : classification

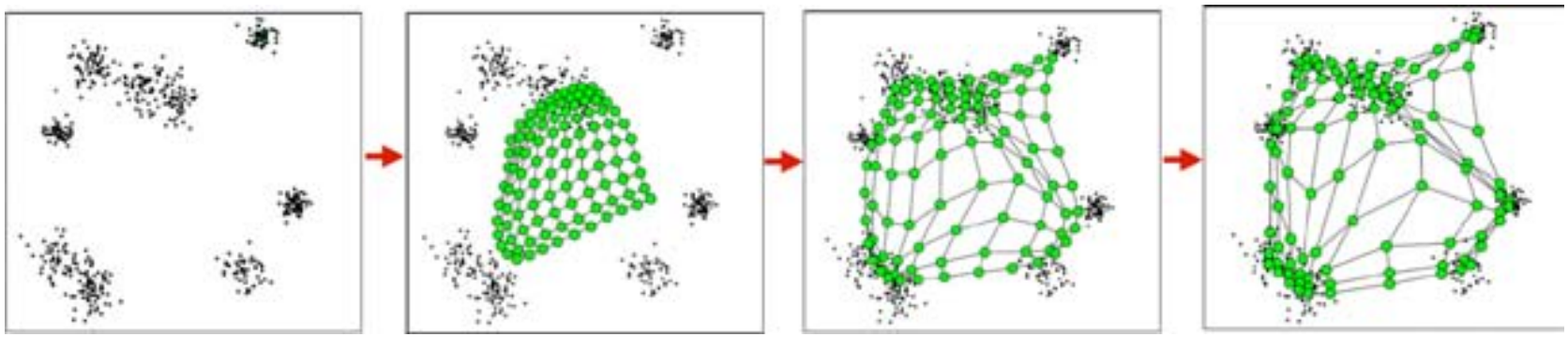


(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.

Chaque partition est représentée par un prototype

- Centres Mobiles
- K-Moyennes, K-Medoïds
- ...

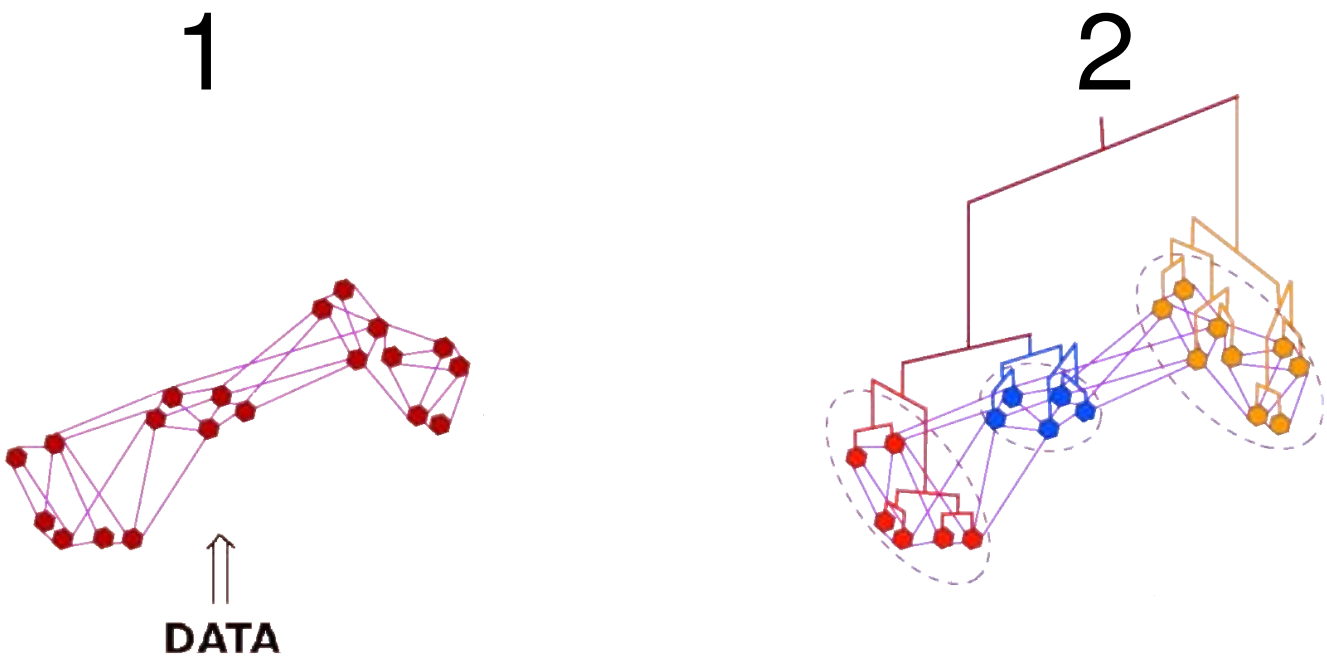
Méthodes à base de prototypes : quantification



Chaque partition est représentée par plusieurs prototypes

- Neural Gas (NG)
- Self-Organizing Map (SOM)
- ...

Méthodes à base de prototypes : à deux niveaux



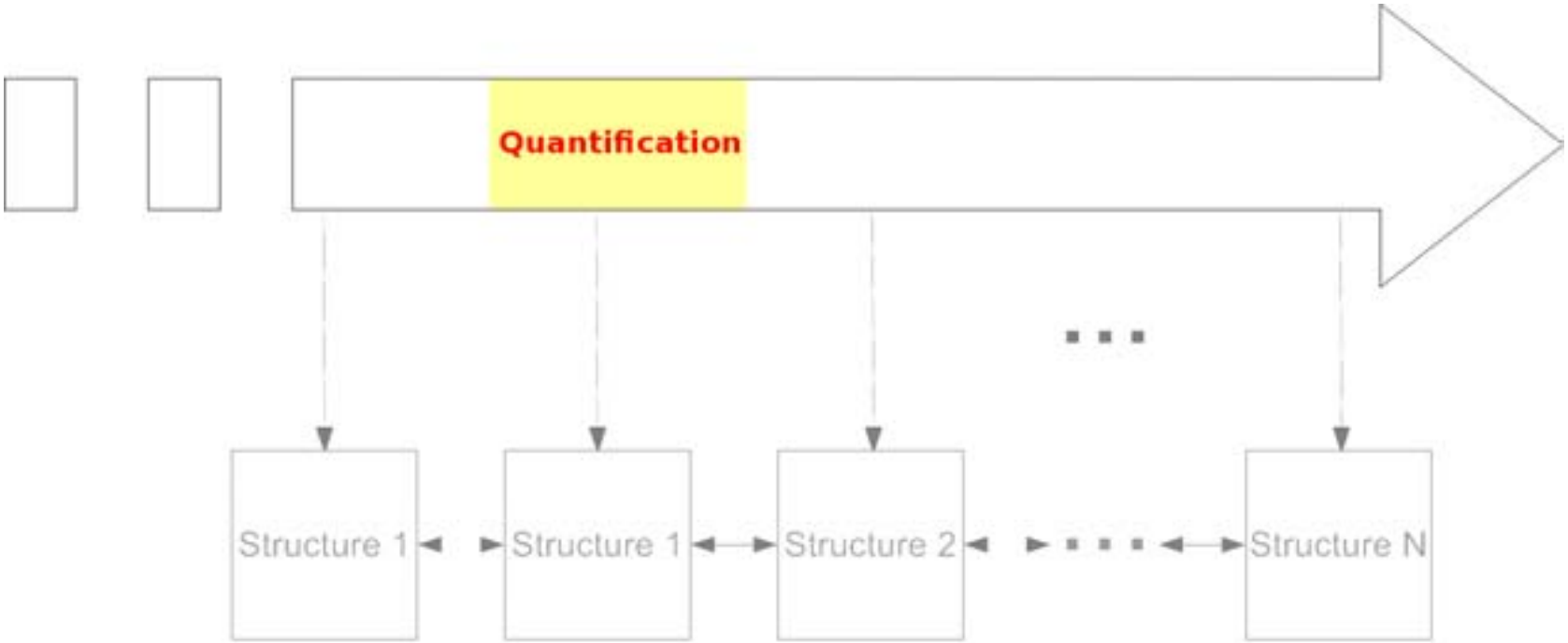
Méthodes à deux niveaux

- Premier niveau : quantification vectorielle
- Deuxième niveau : classification

Plan

Apprentissage des contraintes topologiques

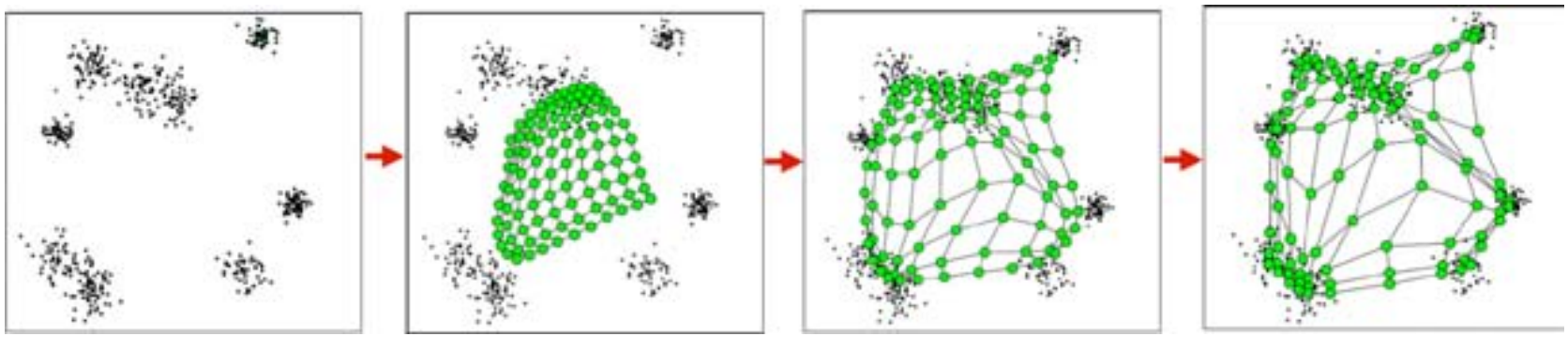
Objectifs


Quantification
Structure 1
Structure 1
Structure 2
Structure N

Objectifs

1. **Apprendre les contraintes topologiques dans les Cartes Auto-Organisatrices (SOM)**
2. **Améliorer la quantification des données**
3. **Réduire le nombre de neurones non représentatifs**
4. **Conserver la structure topologique de la SOM**

L'algorithme des Cartes Auto-Organisatrices (SOM)

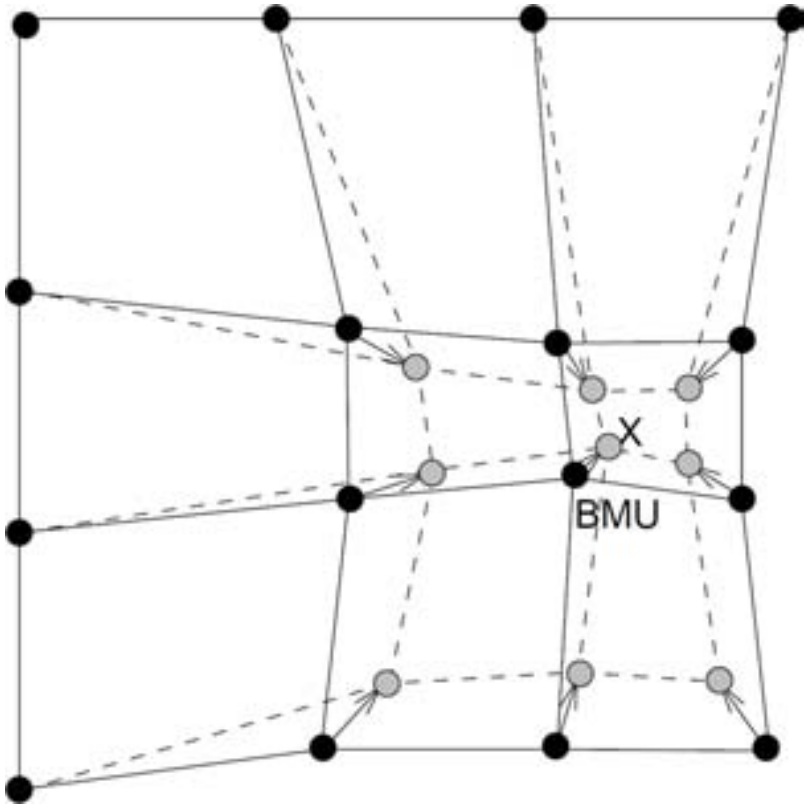


Fonction de coût :

$$\widetilde{R}(w) = \sum_{k=1}^{N} \sum_{j=1}^{M} K_{j,u^*(x^{(k)})} \| w_j - x^{(k)} \|^2$$

Compromis entre la structure topologique de la carte et l'erreur de quantification pour la représentation des données.

Conservation de la structure topologique de la SOM

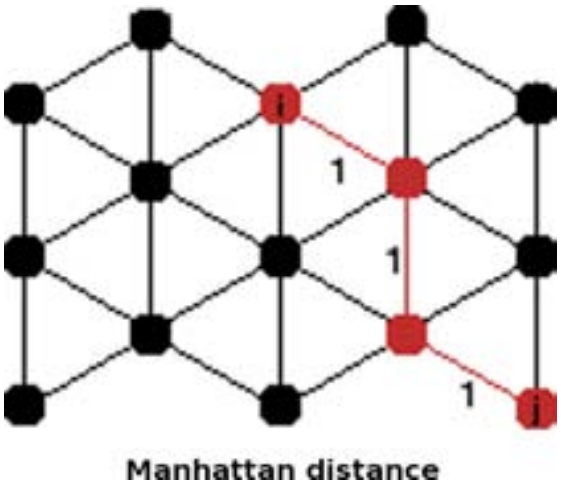


Chaque neurone est activé par les données représentées par ce neurone, mais aussi, à un degré moindre, par les données représentées par ses voisins.

Conservation de la structure topologique de la SOM

Fonction de voisinage :

$$K_{ij} = \frac{1}{\lambda(t)} \times e^{-\frac{d_M^2(i,j)}{\lambda^2(t)}}$$



Manhattan distance

Ici $d_M(i,j) = 3$, c'est la longueur du chemin le plus court dans le graphe entre i et j.

Indices de qualité des SOM

Quantization Error *Qe* :
C'est la mesure de la distance moyenne entre chaque vecteur de donnée et son meilleur représentant.

$$Qe = \frac{1}{N} \sum_{k=1}^{N} \| w_{u^*(x^{(k)})} - x^{(k)} \|^2$$

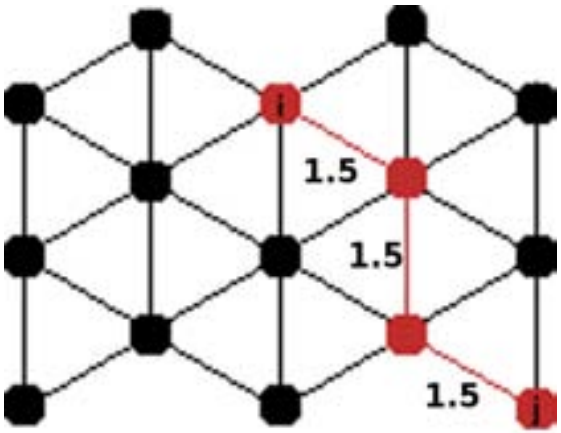
Topological Error *Te* :
C'est la proportion des données ayant les deux meilleurs représentants non adjacents. Décrit la façon dont la SOM préserve la topologie de l'ensemble des données étudiées.

Neuron Utilization *Ne* :
Mesure le pourcentage de neurones qui ne sont jamais le meilleur représentant d'une donnée de la base.

Relâchement simple des contraintes topologiques

Protocol

Nous avons testé différentes versions de SOM où chaque distance de Manhattan est multipliée par une constante.
Par exemple, SOM(α) est similaire à l'algorithme SOM, mais $d(i,j) = \alpha \times d_M(i,j)$.

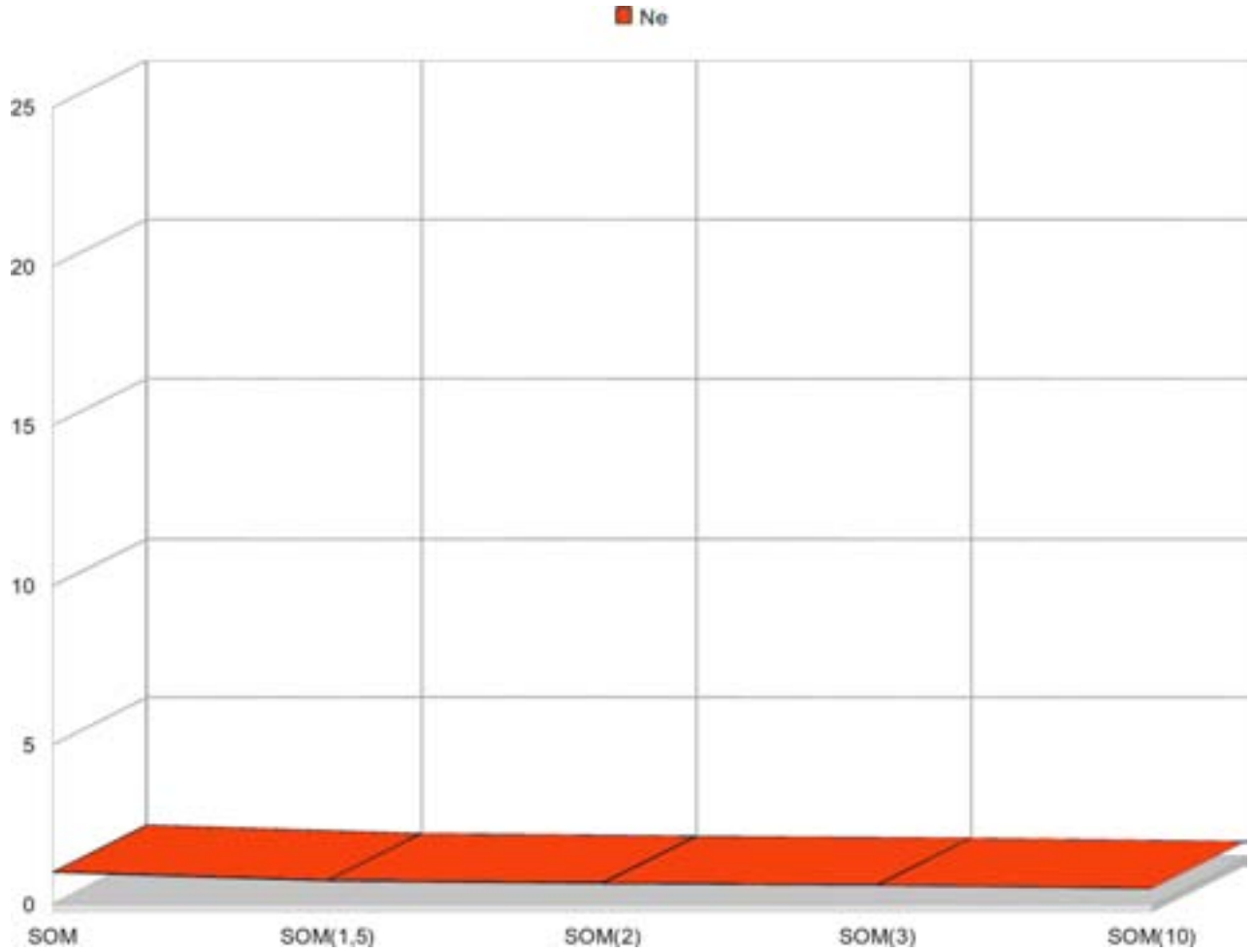


Ici $d_{\alpha=1.5}(i,j) = 4.5$, c'est la longueur du chemin le plus court dans le graphe entre i et j multipliés par 1.5.

Bases de test

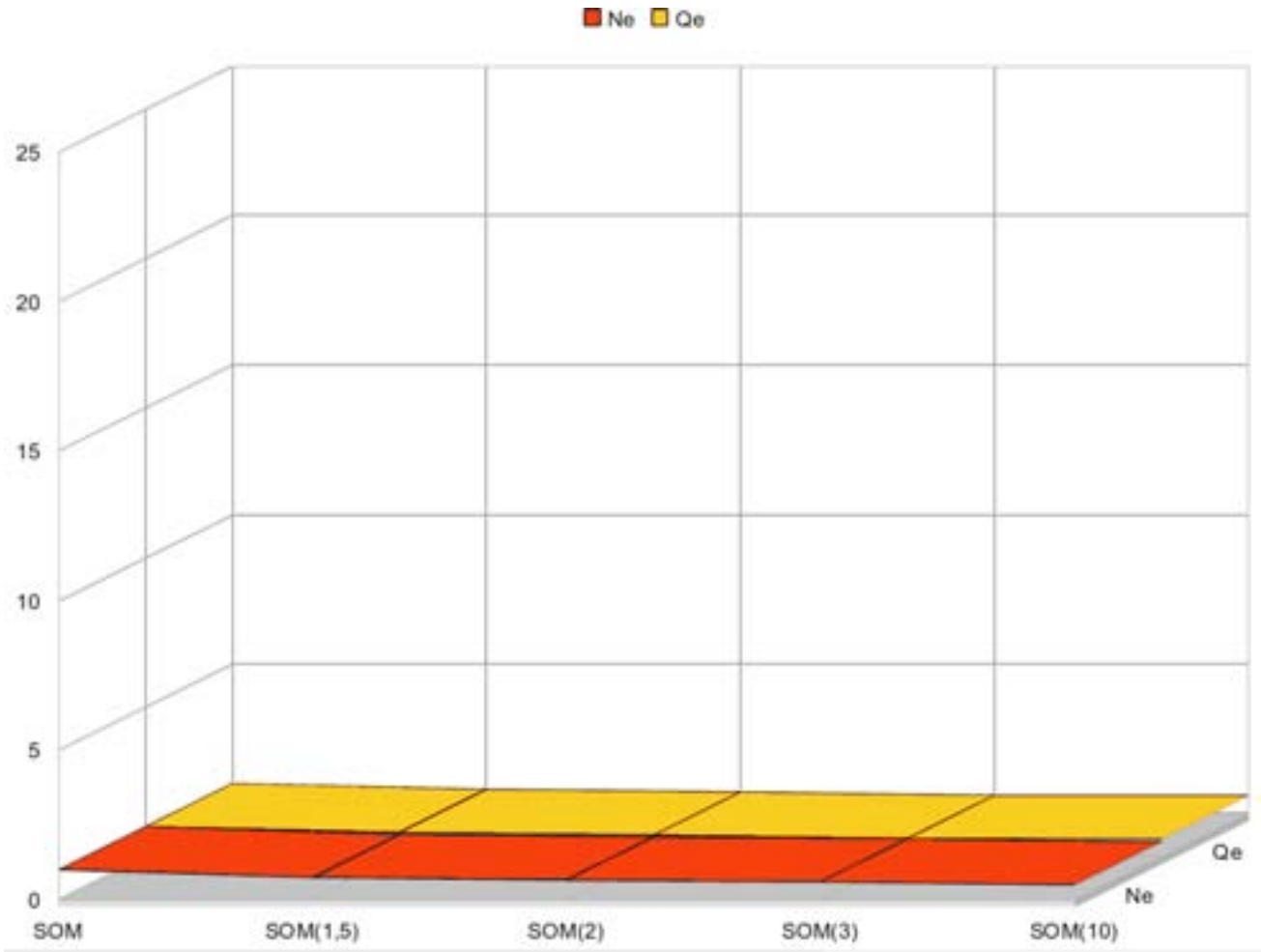
Bases	Type	Taille	Dimension
Target	Artificiel	770	2
TwoDiamonds	Artificiel	800	2
Hepta	Artificiel	212	3
Tetra	Artificiel	400	3
Iris	Réel	150	4
Harot	Réel	132	6
Housing	Réel	506	13
Wine	Réel	178	13
Cockroach	Réel	1369	3
Chromato	Réel	134	60

Résultats : Utilisation des Neurones



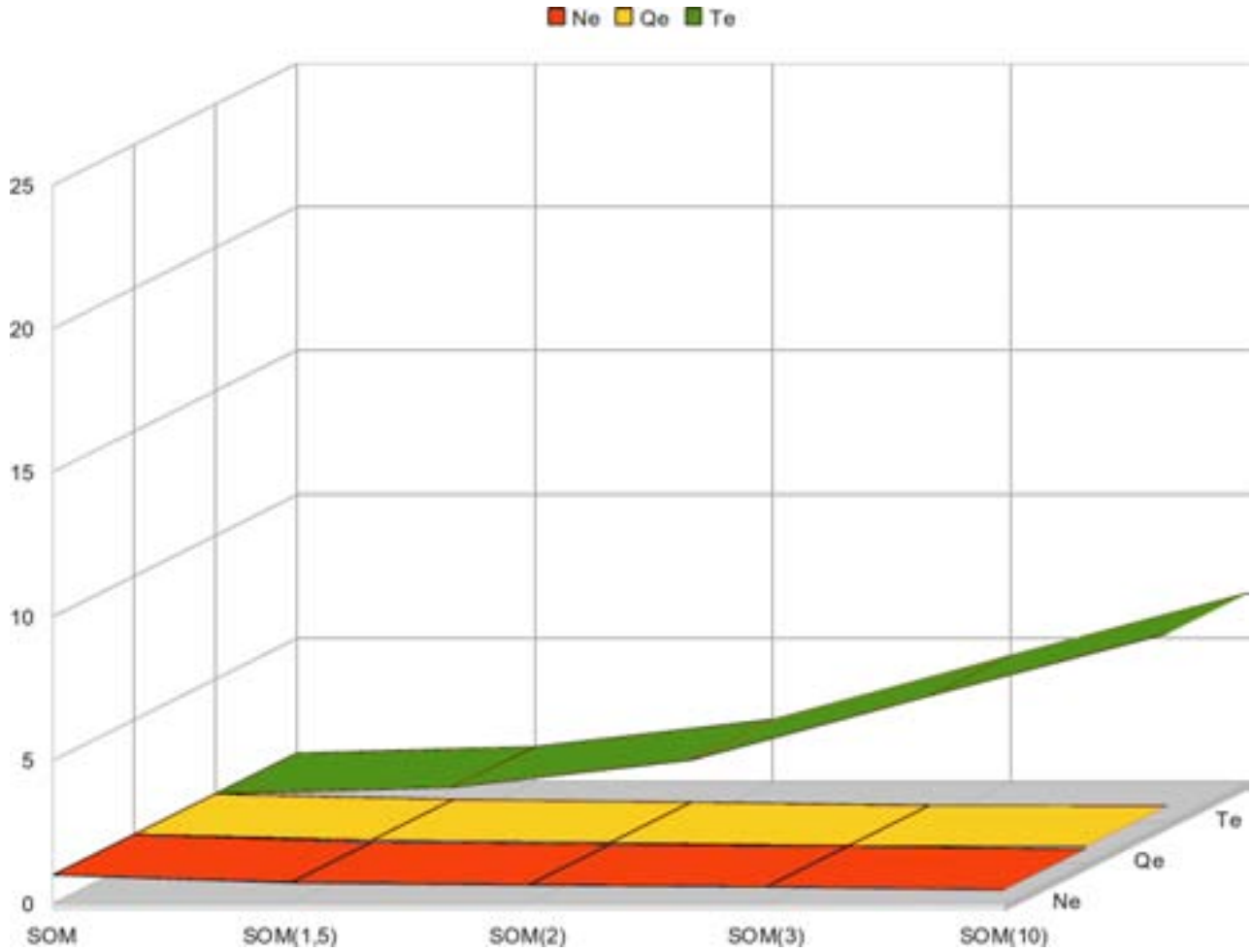
Valeur moyenne de *Ne* sur l'ensemble des bases de données selon α.

Résultats : Erreur de Quantification



Valeur moyenne de *Qe* sur l'ensemble des bases de données selon α.

Résultats : Erreur Topologique



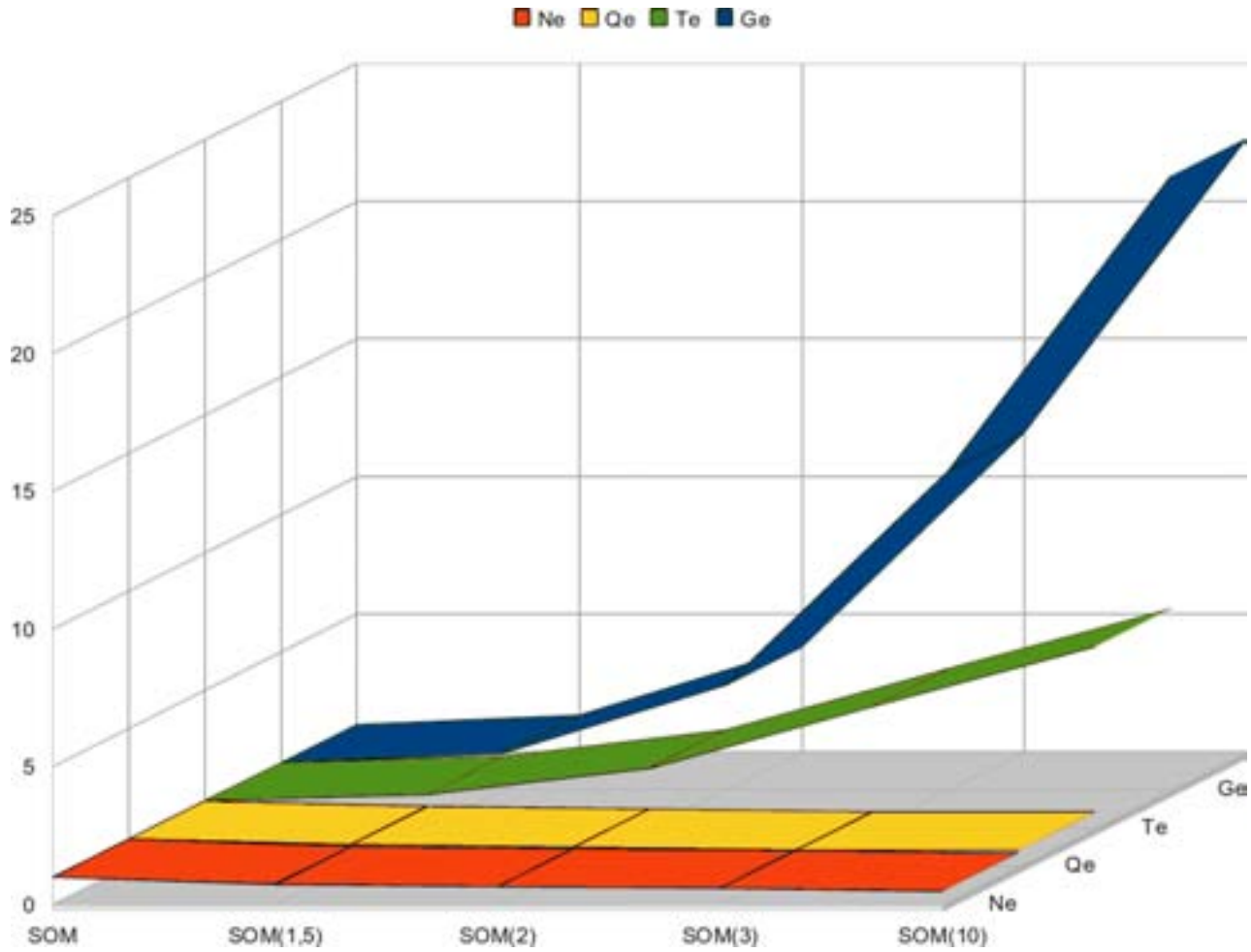
Valeur moyenne de *Te* sur l'ensemble des bases de données selon α.

Comment quantifier le compromis ?

Puisque le gain en *Ne* et *Qe* est associé à une perte en *Te*, nous proposons de définir une Erreur Générale qui reflète les compromis entre *Ne*, *Qe* et *Te* :

$$Ge = Te^2 \times Ne \times Qe$$

Résultats : Erreur Globale



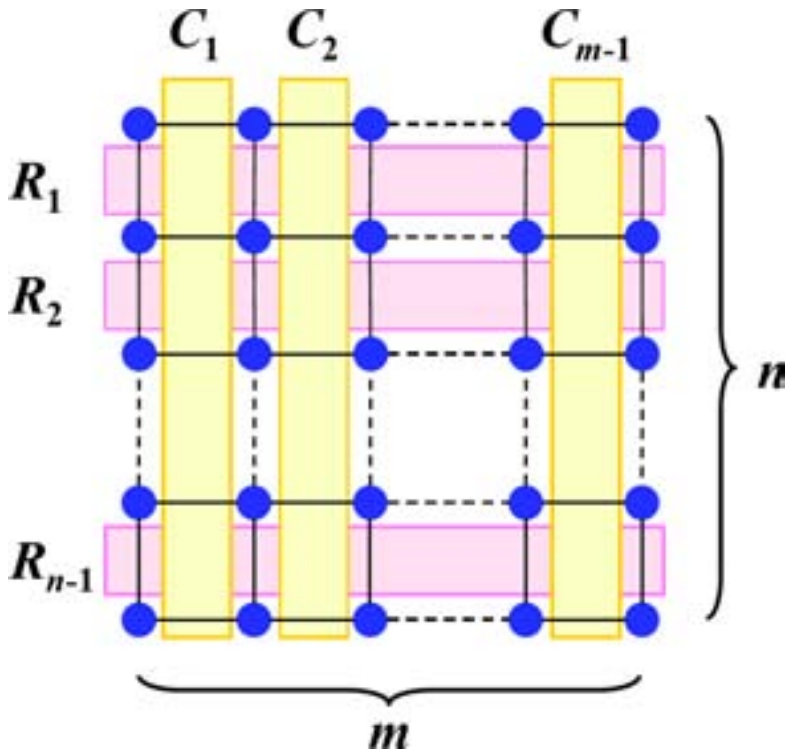
Valeur moyenne de *Ge* sur l'ensemble des bases de données selon α.

Comment faire mieux que SOM ?

Sous l'effet d'une diminution simple des contraintes topologiques, le gain en *Ne* et *Qe* ne peut pas compenser la perte en *Te*.

Peut-on utiliser les données d'apprentissage pour rendre adaptative les contraintes topologiques ?

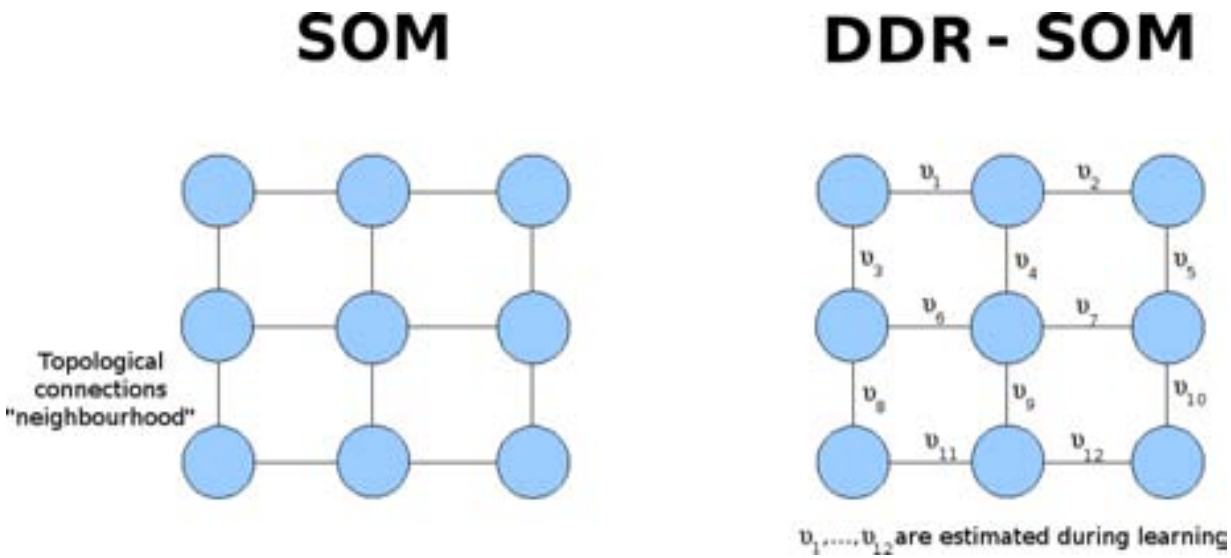
La notion de voisinage adaptatif



False Neighbors - SOM (Matsushita et Nishio 2008)

Dans FN-SOM, les auteurs proposent de conserver la topologie bi-dimensionnelle de la SOM, mais en associant à chaque “ligne” ou “colonne” de connexions un indice de voisinage qui est lié à la validité globale de la “ligne” ou de la “colonne”.

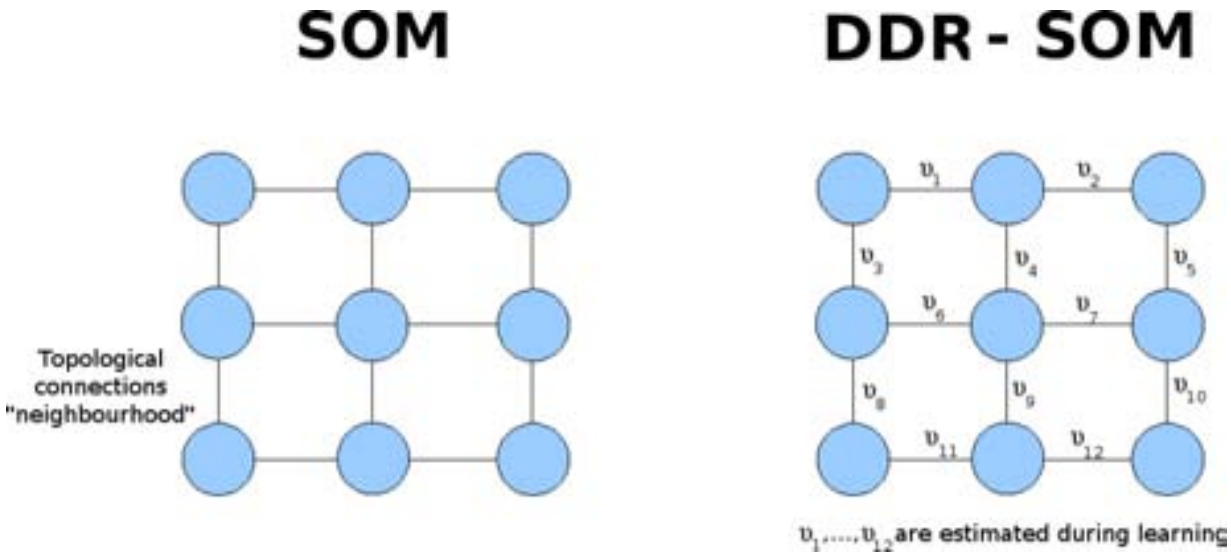
Principe du nouvel algorithme : apprendre le voisinage



Apprendre le voisinage

Dans l'algorithme DDR-SOM, nous proposons d'associer à chaque connexion de voisinage une valeur réelle *v* qui indique la pertinence de la relation entre les neurones connectés.

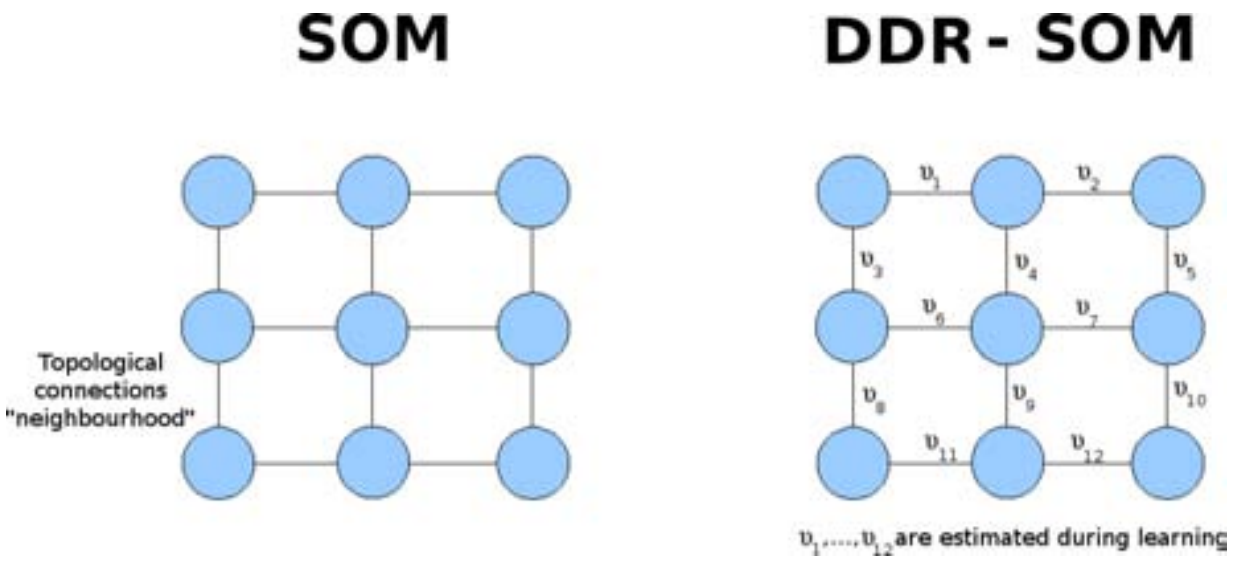
Principe du nouvel algorithme : apprendre le voisinage



Apprendre le voisinage

Une paire de neurones voisins, qui sont tout deux de bons représentants d'un même ensemble de données devraient être fortement connectés, tandis qu'une paire de neurones voisins qui ne représentent pas le même type de données doit être faiblement connectés.

Principe du nouvel algorithme : apprendre le voisinage

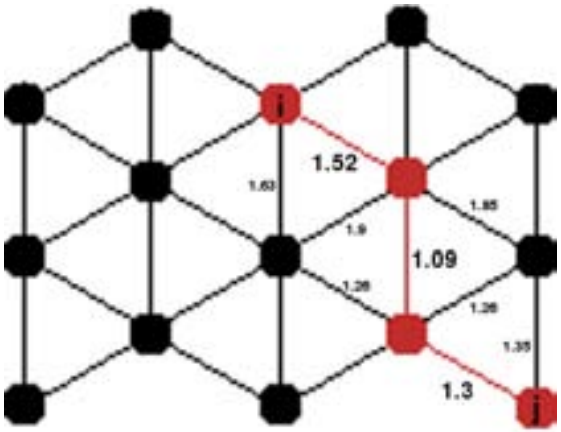


Apprendre le voisinage

On associe à chaque connexion de voisinage une valeur réelle selon une fonction logistique qui dépend du nombre de données bien représentées par chacun des deux neurones connectés. σ est le paramètre de la fonction logistique.

Utilisation des valeurs de voisinage pour le calcul d'une distance de Manhattan pondérée

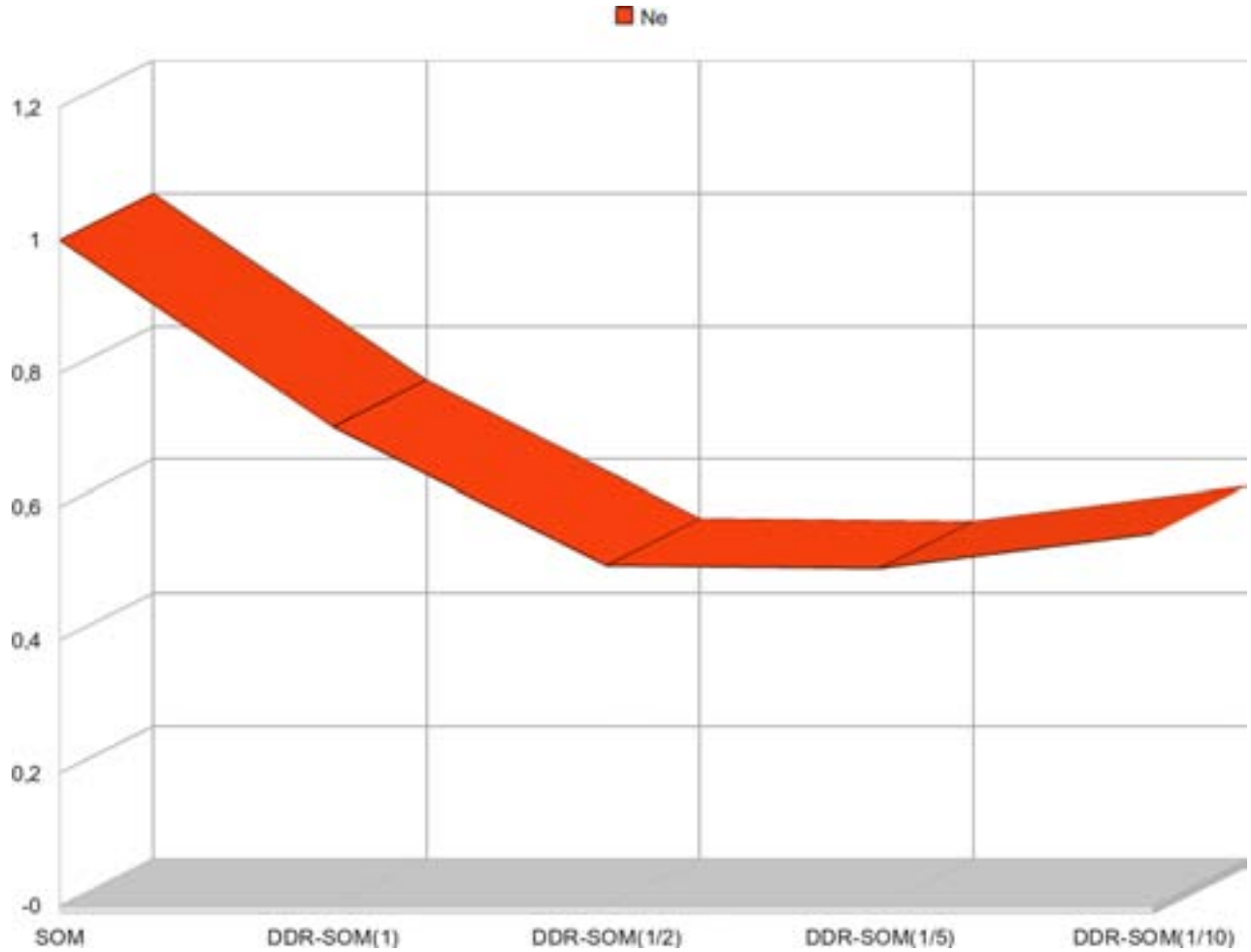
Ces valeurs sont utilisées pour estimer une distance de Manhattan pondérée $d_{WM}(i,j)$ entre chaque paire de neurones voisins i et j.



Weighted Manhattan distance

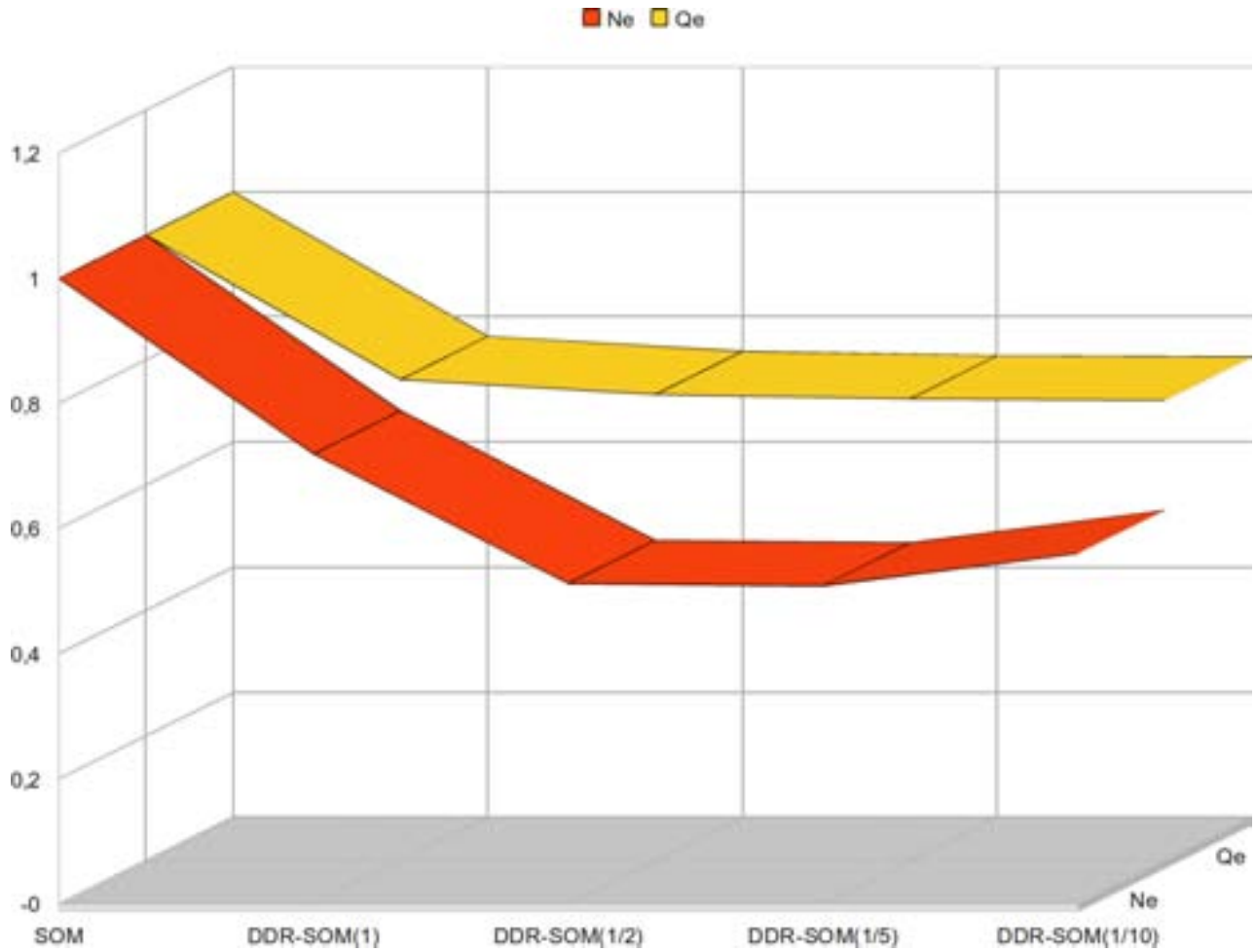
Ici $d_{WM}(i,j) = 1.52+1.09+1.3 = 3.91$, il s'agit du plus court chemin entre i et j en fonction des valeurs de connexion v.

Résultats : Utilisation des Neurones



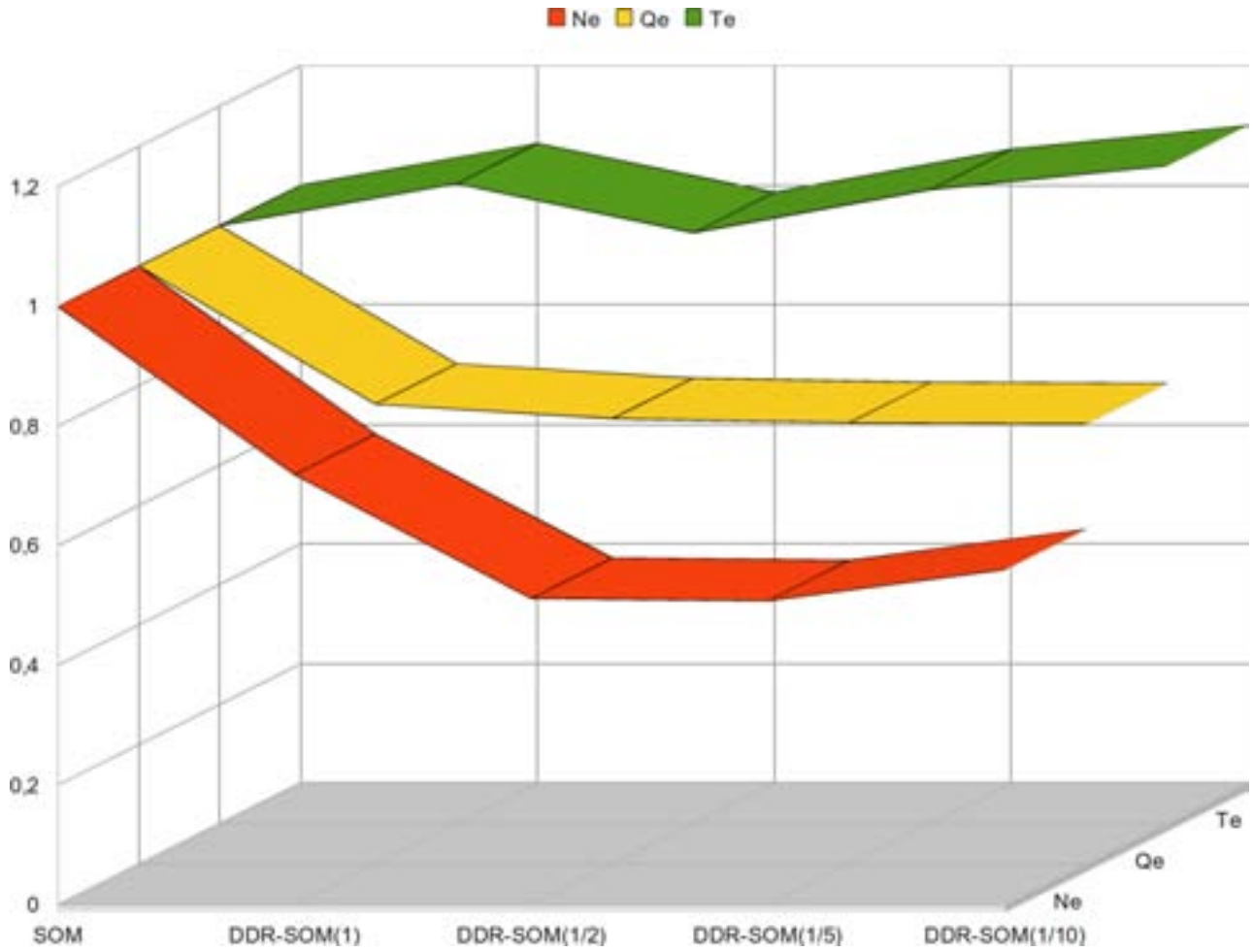
Valeur moyenne de *Ne* sur l'ensemble des bases de données selon σ.

Résultats : Erreur de Quantification



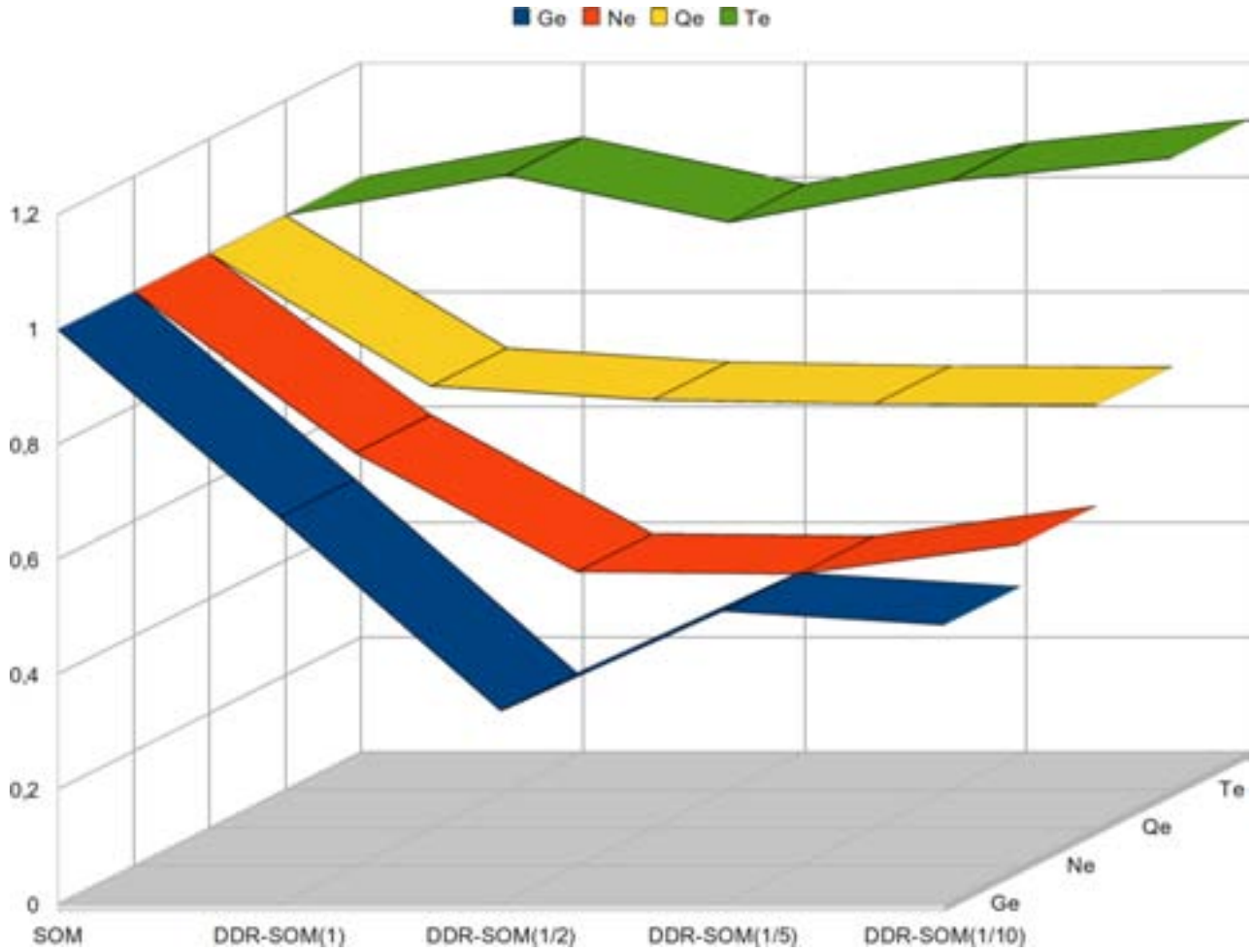
Valeur moyenne de *Qe* sur l'ensemble des bases de données selon σ.

Résultats : Erreur Topologique



Valeur moyenne de *Te* sur l'ensemble des bases de données selon σ.

Résultats : Erreur Générale



Valeur moyenne de *Ge* sur l'ensemble des bases de données selon σ.

Résultats : Erreur Générale

Valeur de *Ge* pour chaque base et chaque algorithme :

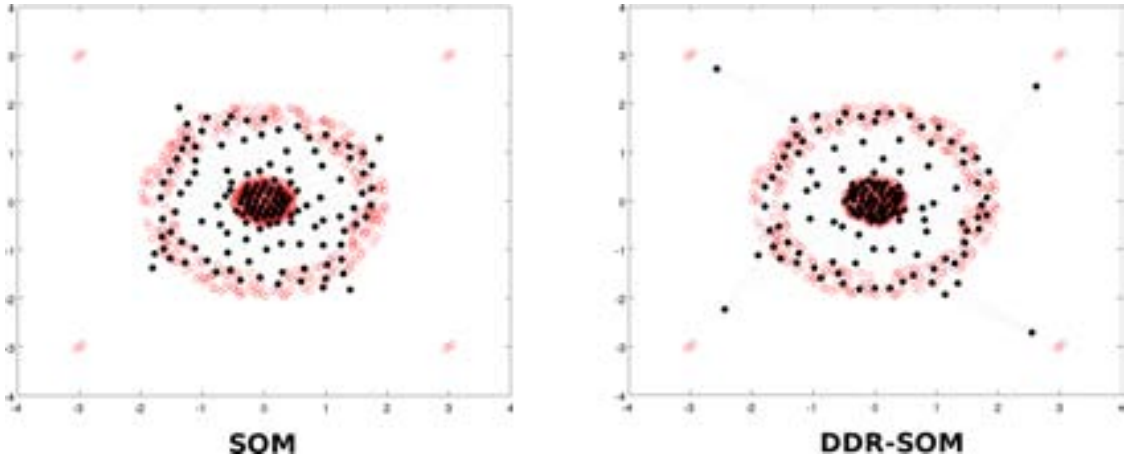
	DDR(1)	DDR(1/2)	DDR(1/5)	DDR(1/10)
Target	0,57	0,39	0,87	0,89
TwoDiamonds	0,22	0,27	0,22	0,11
Hepta	1,82	0,62	0,97	0,57
Tetra	1,17	0,53	1,61	1,12
Iris	0,09	0,21	0,29	0,28
Harot	0,79	0,17	0,06	0,38
Housing	0,83	0,54	0,35	0,55
Wine	0,51	0,14	0,13	0,33
Cockroach	0,62	0,42	0,52	0,54
Chromato	0,12	0,08	0,09	0,09

Résultats : remarques

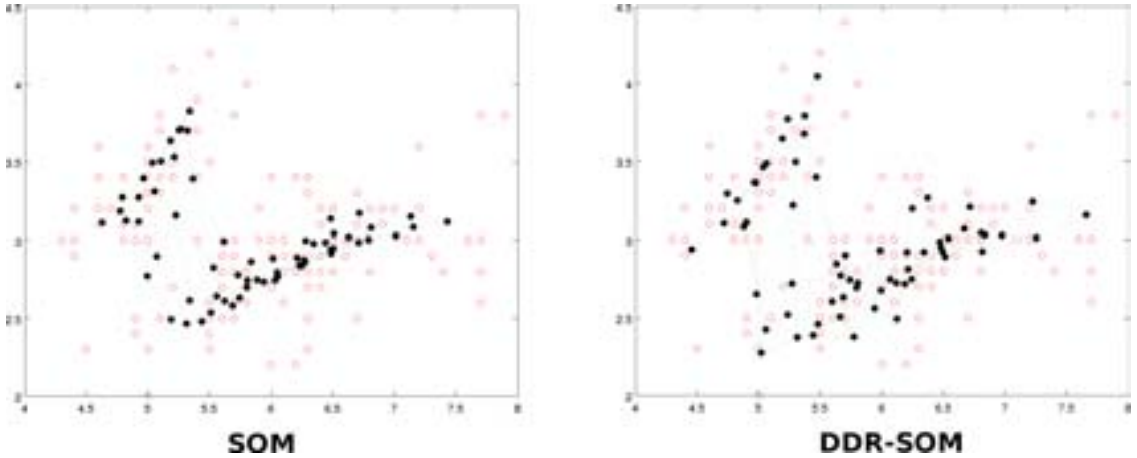
- Les performances de DDR-SOM sont meilleures que celles de SOM pour toutes les valeurs de σ, cependant une valeur de $\sigma = 1/2$ semble donner de meilleurs résultats pour ces bases de données.

- Le gain en *Ge* a tendance à être plus élevé pour les bases de données de grandes dimensions (par exemple “Chromato”, “Wine”, etc . . .).

Exemples



(a) Données “Target”.



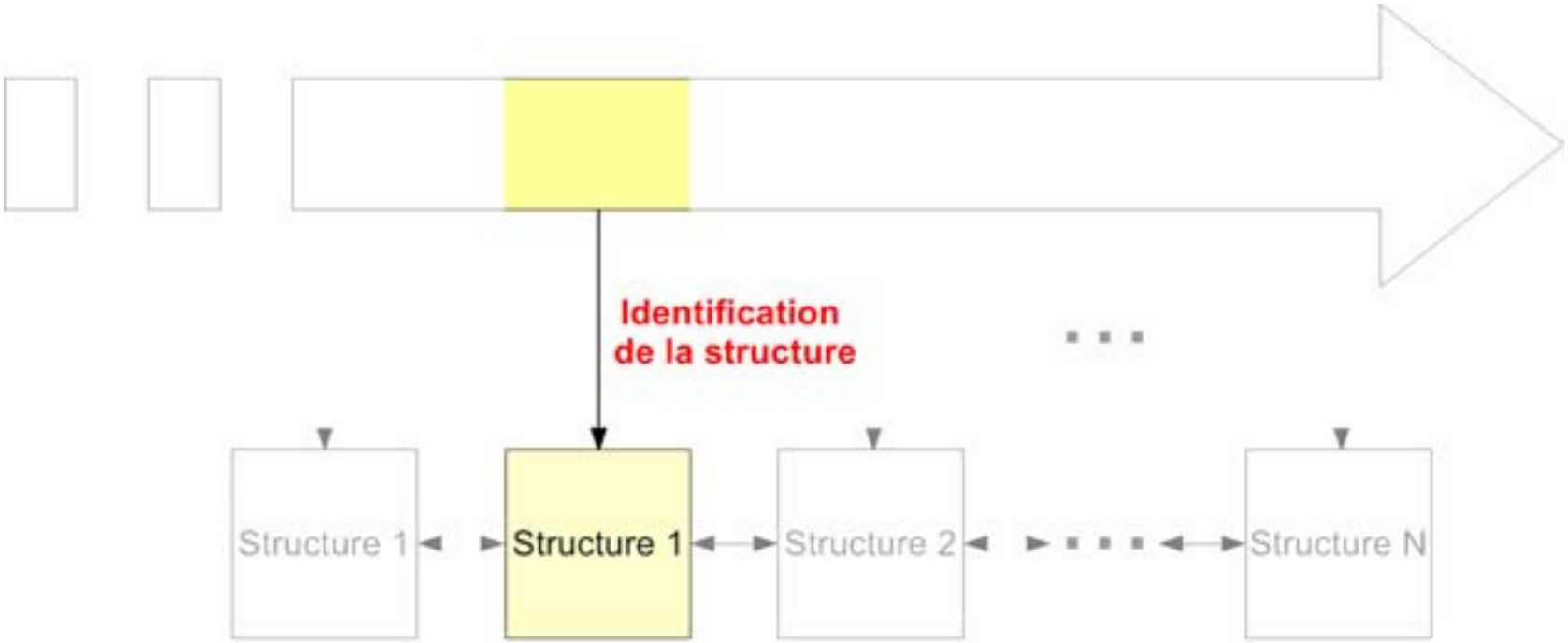
(b) Données “Iris”.

Bilan

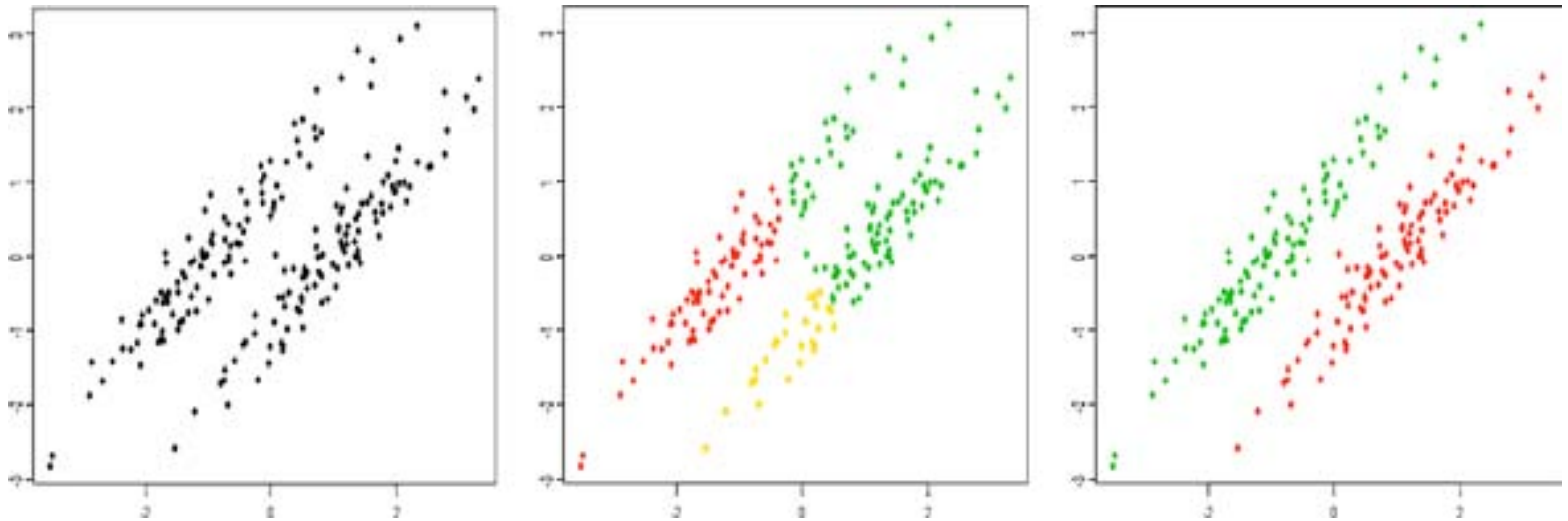
- Les expériences sur des bases de données artificielles et réelles montrent que l'algorithme DDR-SOM obtient de meilleurs résultats que l'algorithme SOM.
- Cette amélioration n'est pas obtenue avec une relaxation triviale des contraintes topologiques, en raison d'une forte augmentation de l'erreur Topologique.
- Une diminution des contraintes guidée par les données semble être une bonne solution pour améliorer le compromis $NeQe/Te$ de la SOM.

Identification de la structure des données

Objectifs


Identification
de la structure
Structure 1
Structure 1
Structure 2
Structure N

Détermination de la meilleure partition



Une recherche exhaustive pour chaque nombres de clusters est un problème combinatoire

Nombre de Stirling :

$$S_{N,K} = \frac{1}{K!} \sum_{i=1}^{K} C_k^i (-1)^{K-i} i^N$$

Nombre de Bell :

$$B_N = e^{-1} \sum_{i=1}^{\infty} \frac{i^N}{i!}$$

71 observations $B_N = 4.08^{74}$

Sélection de modèles : choix du nombre de classes

- **Méthodes de vraisemblance pénalisée :**
  - Akaike Information Criterion (AIC)
  - Bayesian Information Criterion (BIC)
- **Méthode du Gap statistic**
- **Méthodes basées sur la stabilité :**
  - On choisi la partition la plus stable après perturbation des données, des paramètres, de l'initialisation ...
- **...**
- **En pratique** : on essaye K : $K_{min} \ldots K_{max}$ et on choisi $K^* = argmin\ R_k(w)$

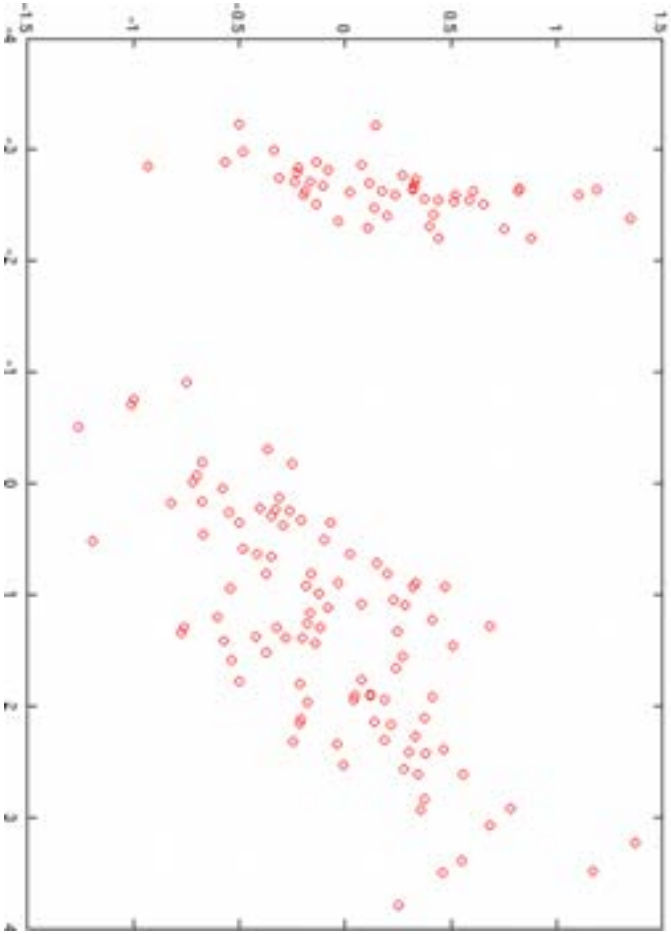
Notre approche

Détection automatique des frontières entre les groupes

1. Partitionnement selon le voisinage entre prototypes
   - Algorithme 1 : Simultaneous 2 Levels - Self Organising Map (S2L-SOM)
2. Partitionnement selon la densité au niveau des prototypes
   - Algorithme 2 : Density-based Simultaneous 2 Levels - Self Organising Map (DS2L-SOM)

L'existence d'une structure et le nombre de groupes sont détectés automatiquement

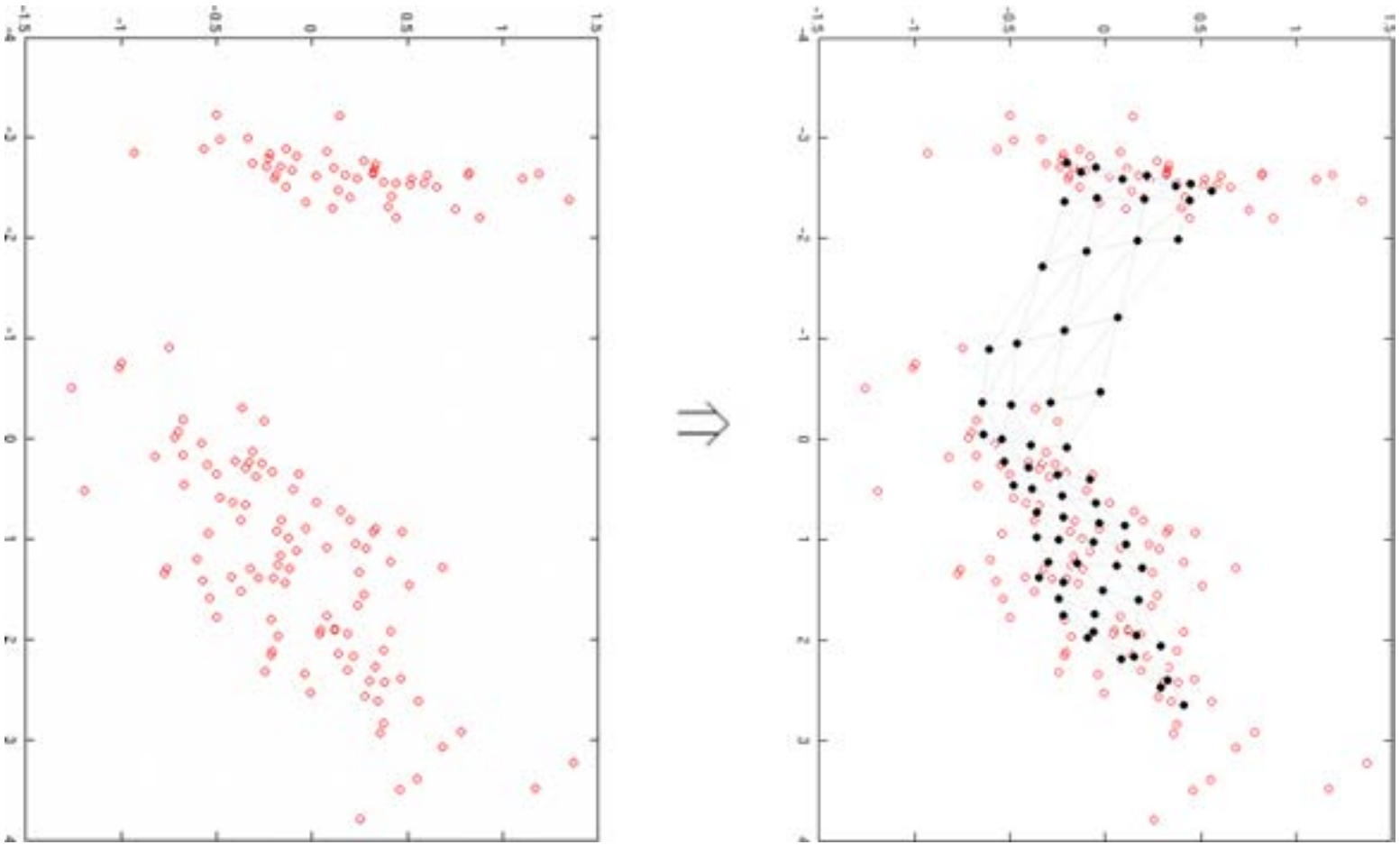
Algorithme 1 : Principe



Entrée

Données initiales

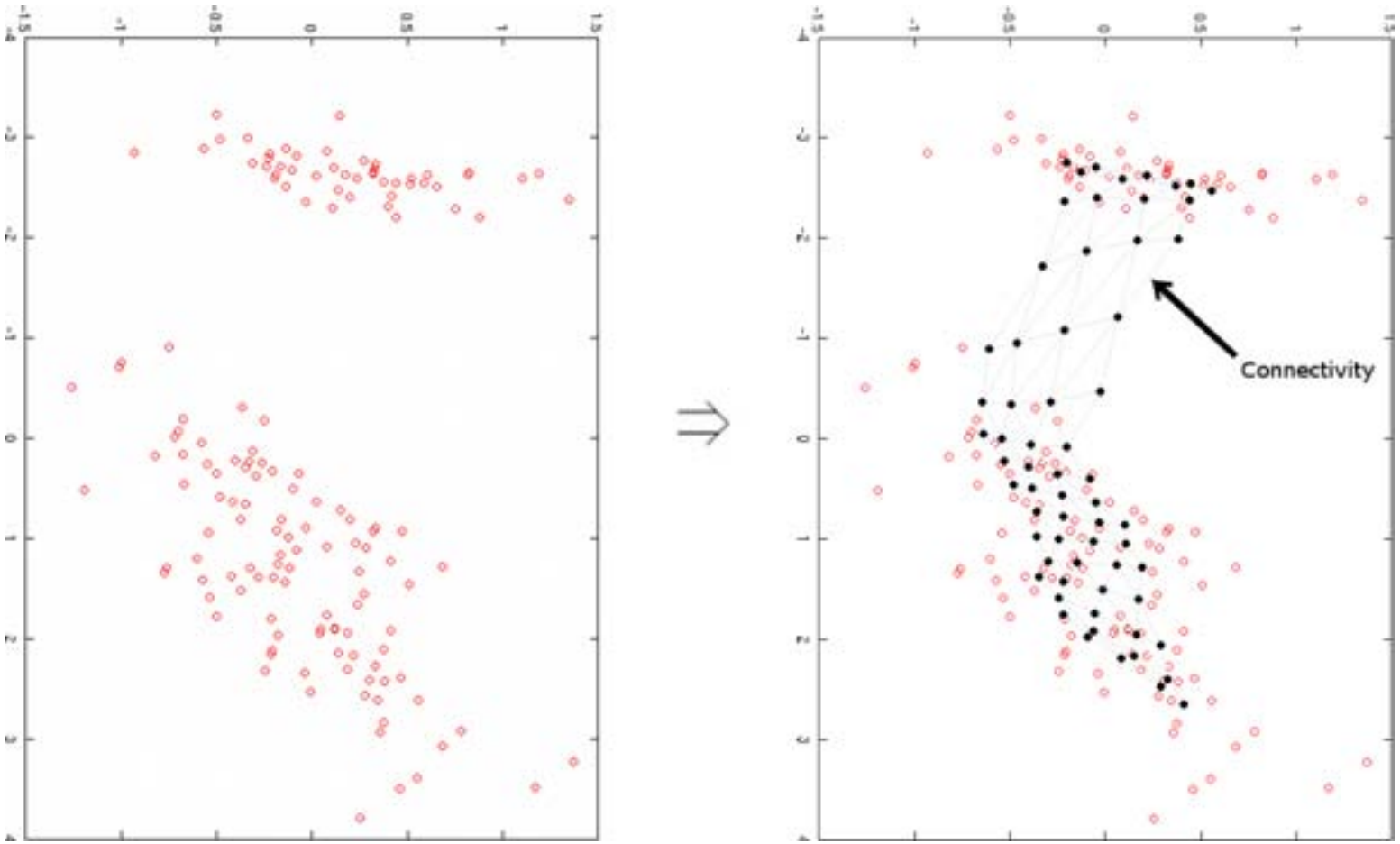
Algorithme 1 : Principe



Étape 1

Prototypes à la fin de l'apprentissage

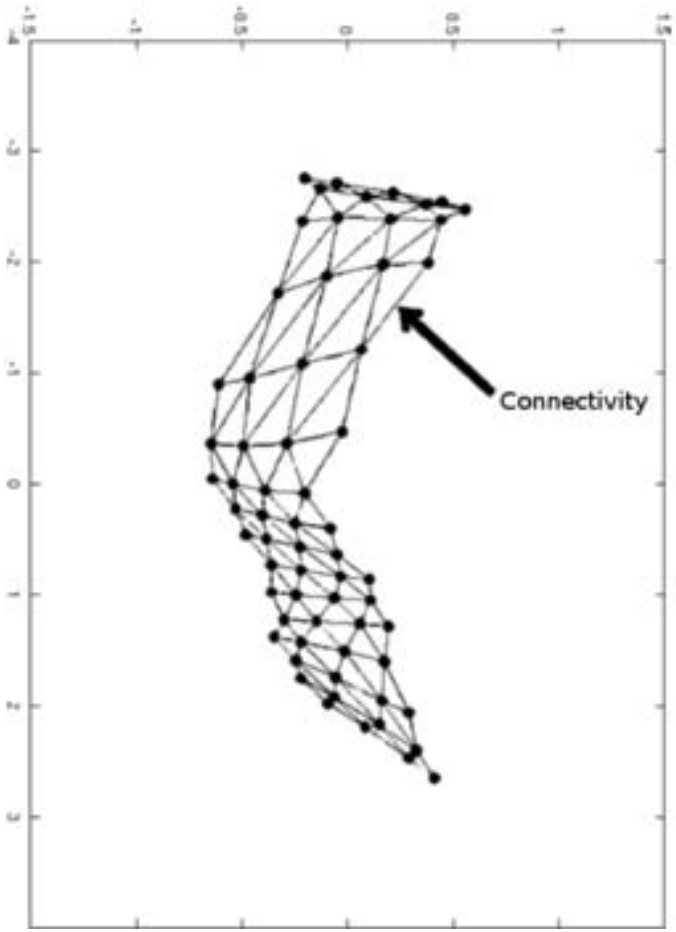
Algorithme 1 : Principe

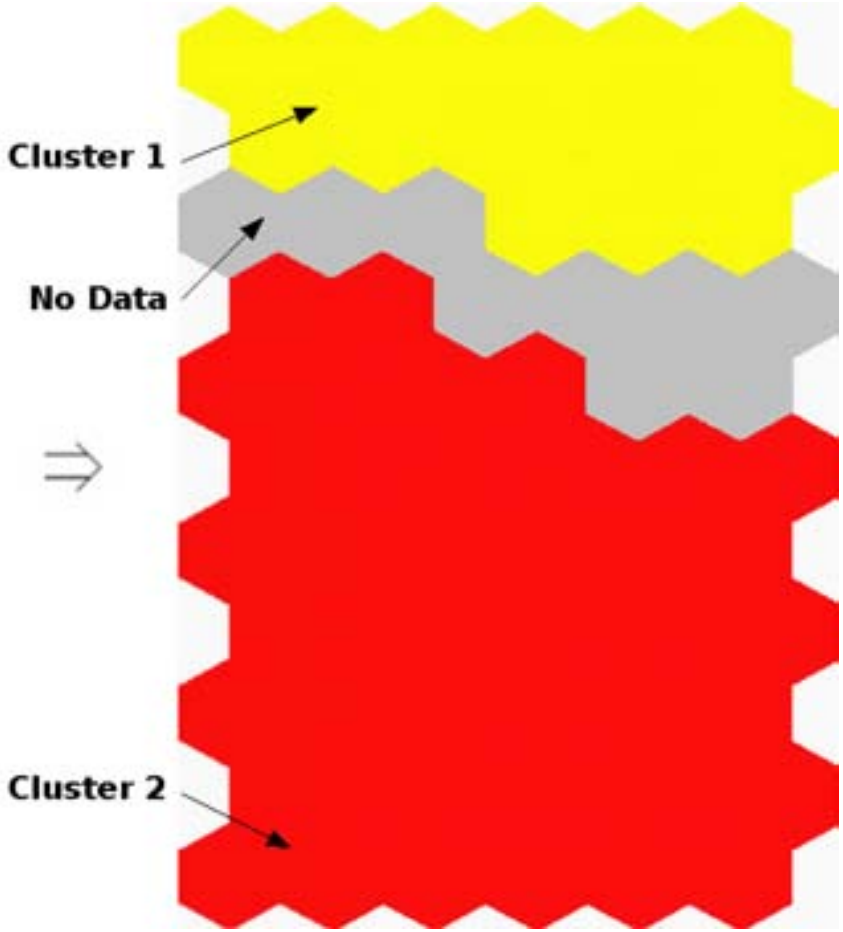


Étape 2

Calcul des informations de connectivité

Algorithme 1 : Principe

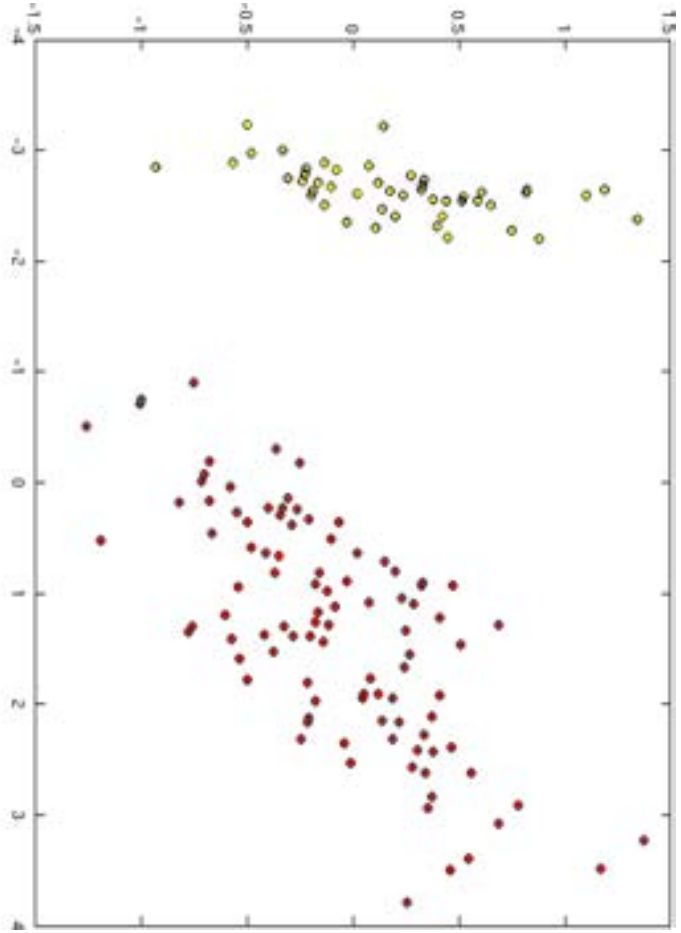




Étape 3

Partitionnement des prototypes et visualisation

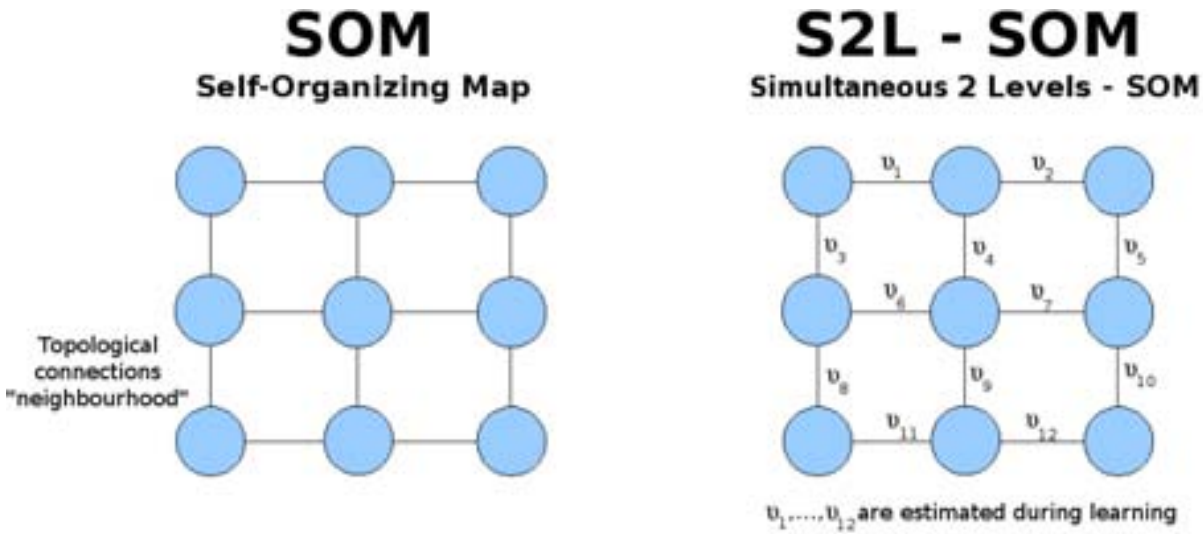
Algorithme 1 : Principe



Étape 4

Classification des données

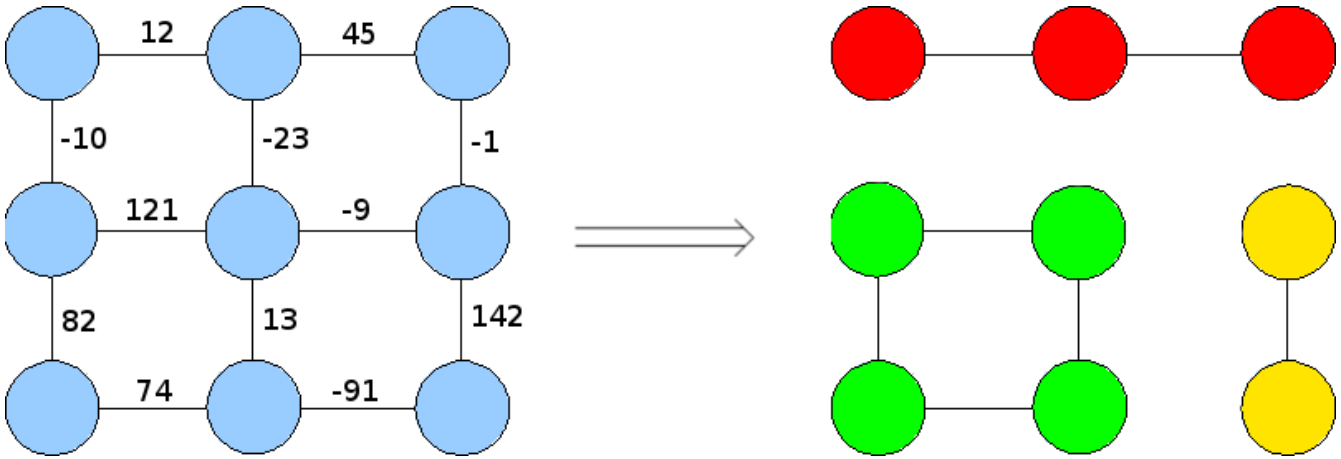
Algorithme 1 : Apprentissage de la connectivité



Apprentissage des valeurs

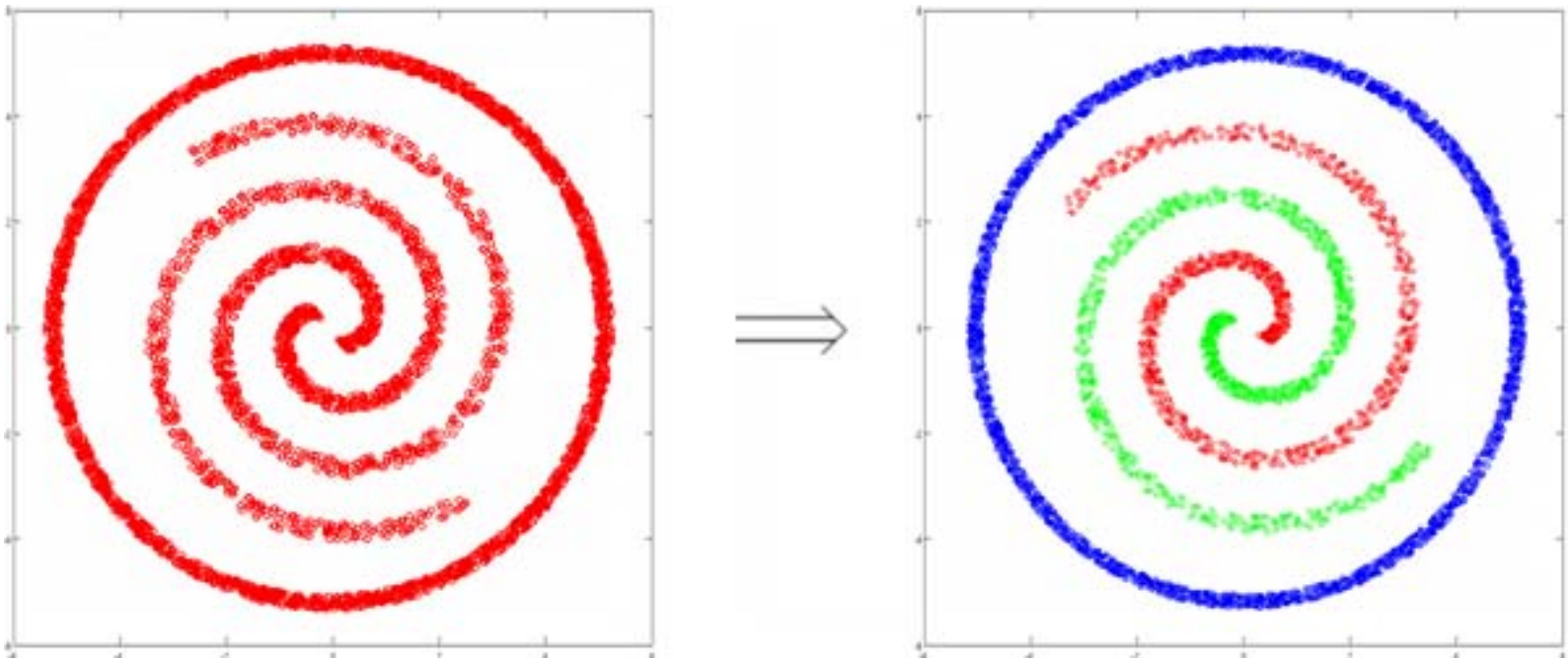
Pour chaque donnée, nous proposons d'augmenter, pendant l'apprentissage, la valeur de la connexion liant les deux meilleurs prototypes et de diminuer les valeurs des autres connexions partant du meilleur prototype.

Algorithme 1 : Partitionnement selon la connectivité



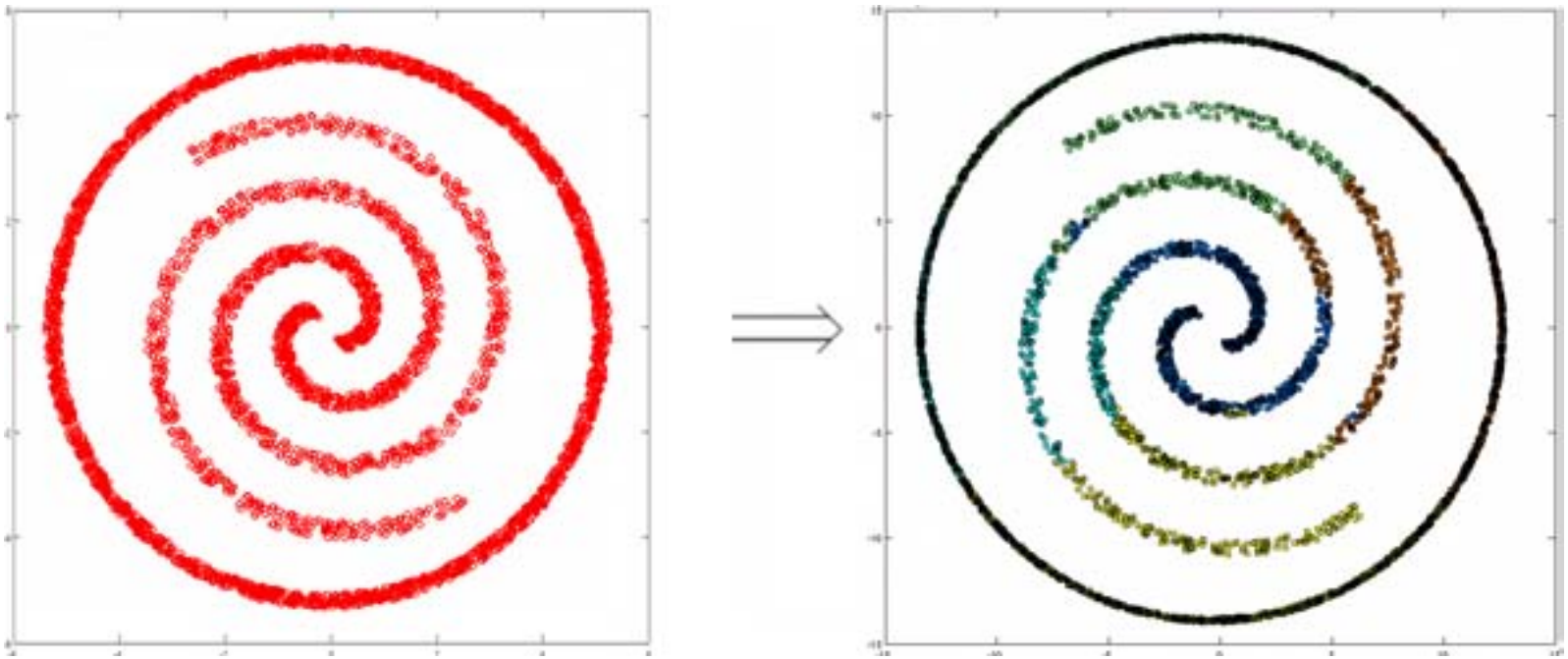
- A la fin de l'apprentissage, un ensemble de prototypes connectés par des connexions de valeurs positives sera représentatif d'un sous-groupe pertinent de l'ensemble des données.
- Ainsi, le nombre de groupes est facile à déterminer.

Algorithme 1 : Exemple



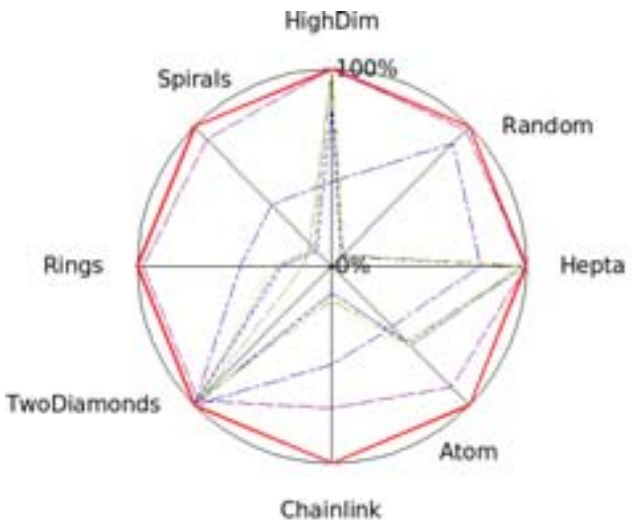
Données “Spirales” et partitionnement selon S2L-SOM

Algorithme 1 : Exemple

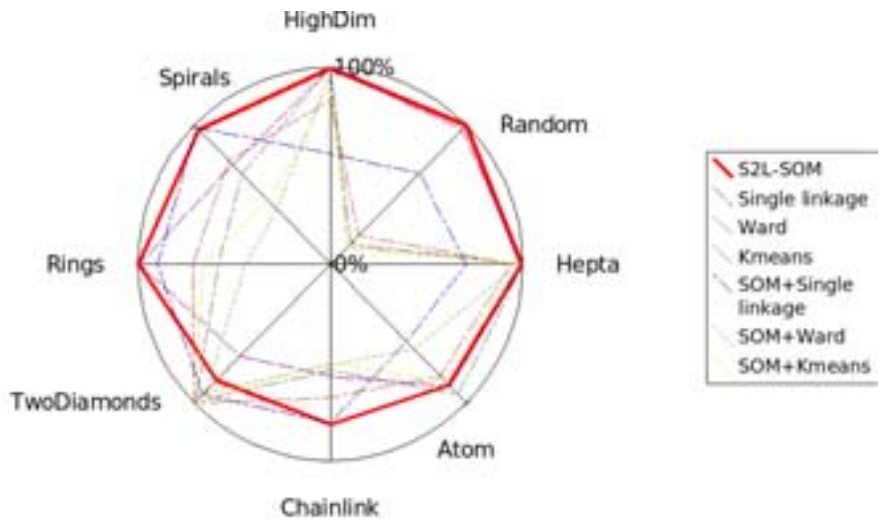


Données “Spirales” et partitionnement selon une méthode classique à deux niveaux (SOM+CAH)

Algorithme 1 : Validité et stabilité



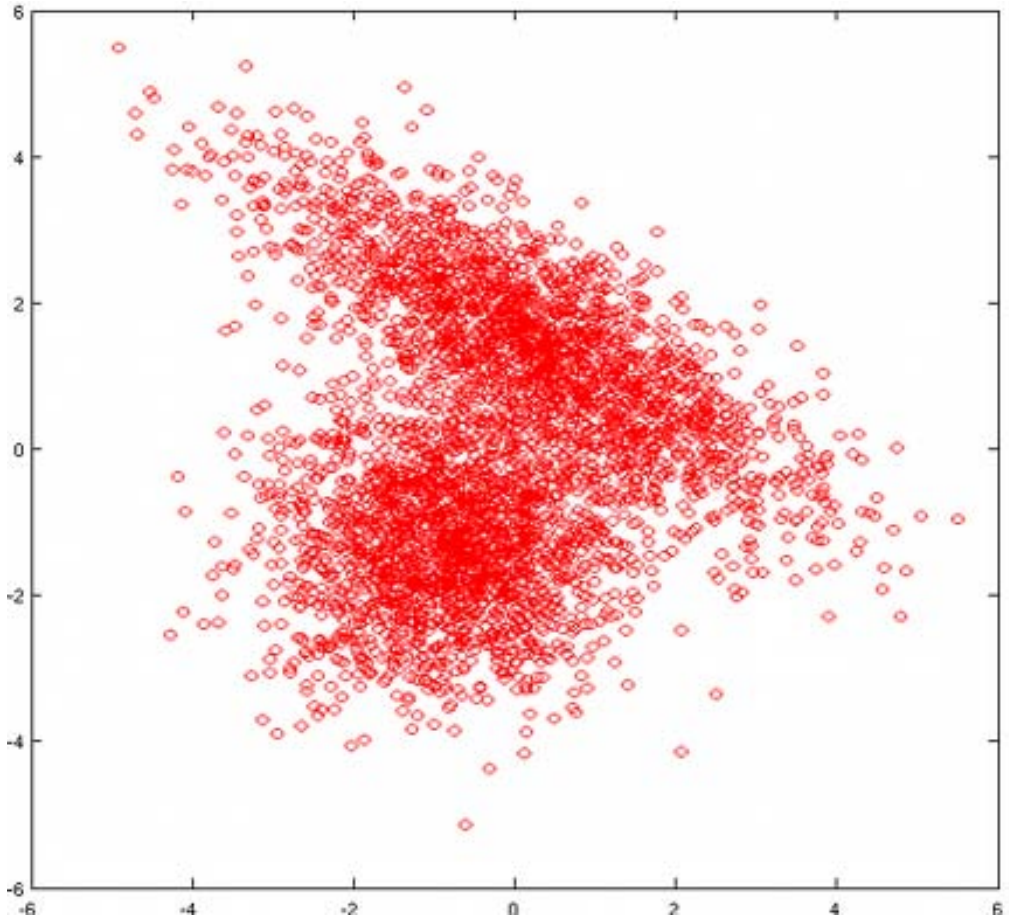
Clustering quality (Jaccard index)



Clustering stabiliy (Sub-sampling based method)

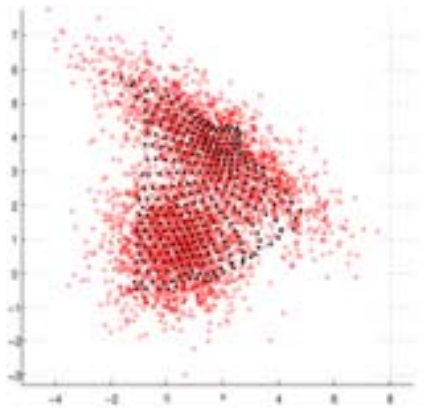
Évaluation du partitionnement en fonction de la base de données et de la méthode utilisée

Algorithme 1 : limites



Le partitionnement selon le voisinage ne permet pas de détecter des partitions en contact.

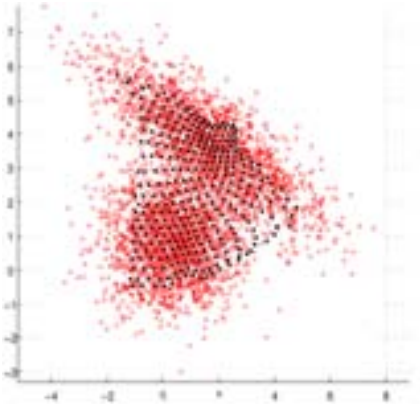
Algorithme 2 : principes

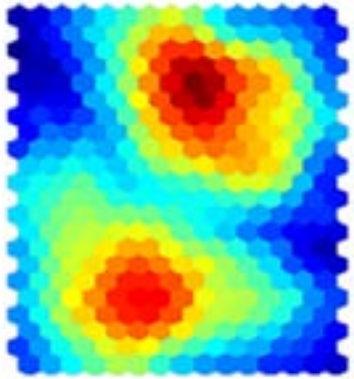


Étape 1

Prototypes à la fin de l'apprentissage

Algorithme 2 : principes

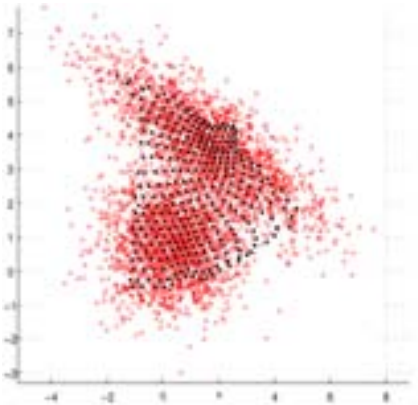


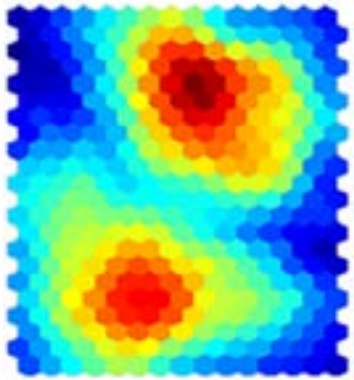


Étape 2

Calcul des informations de densité

Algorithme 2 : principes

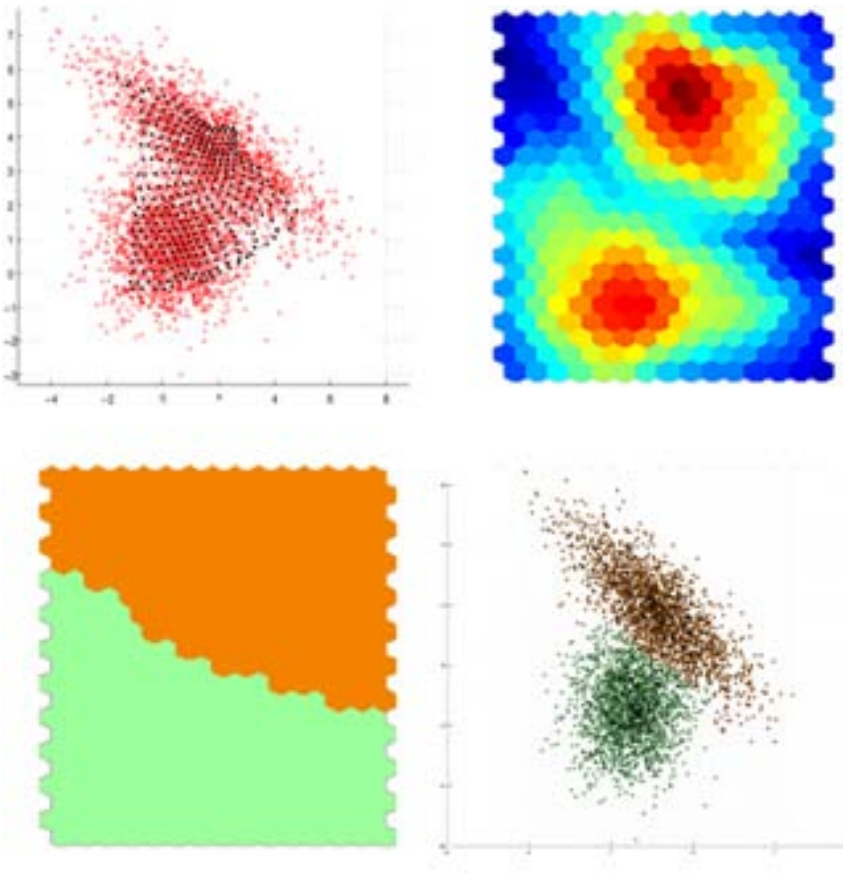




Étape 3

Partitionnement des prototypes et visualisation

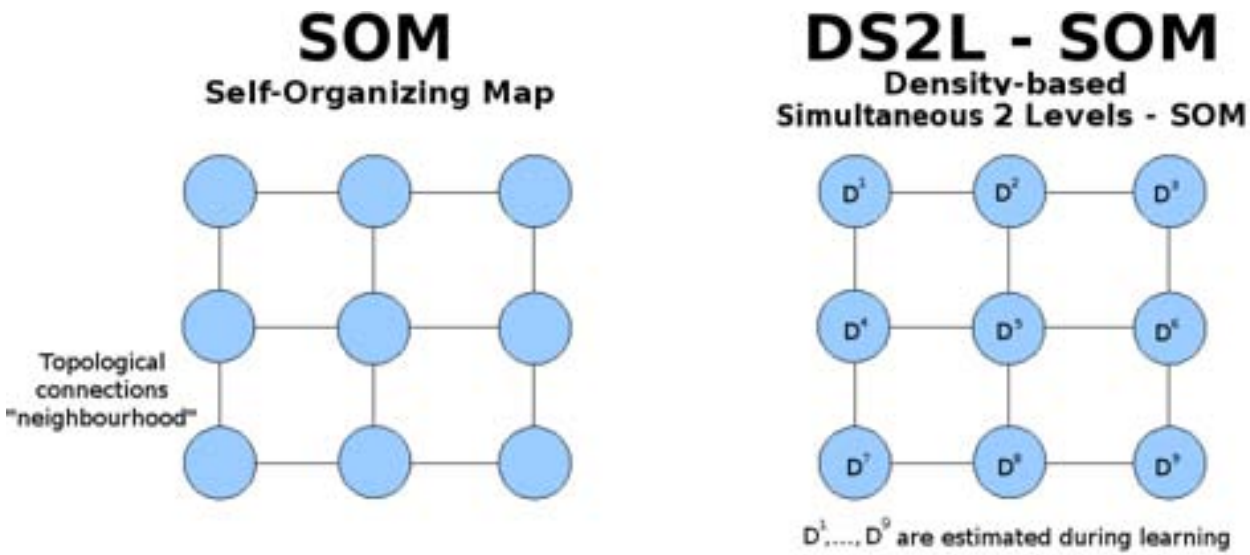
Algorithme 2 : principes



Étape 4

Classification des données

Algorithme 2 : Apprentissage de la densité

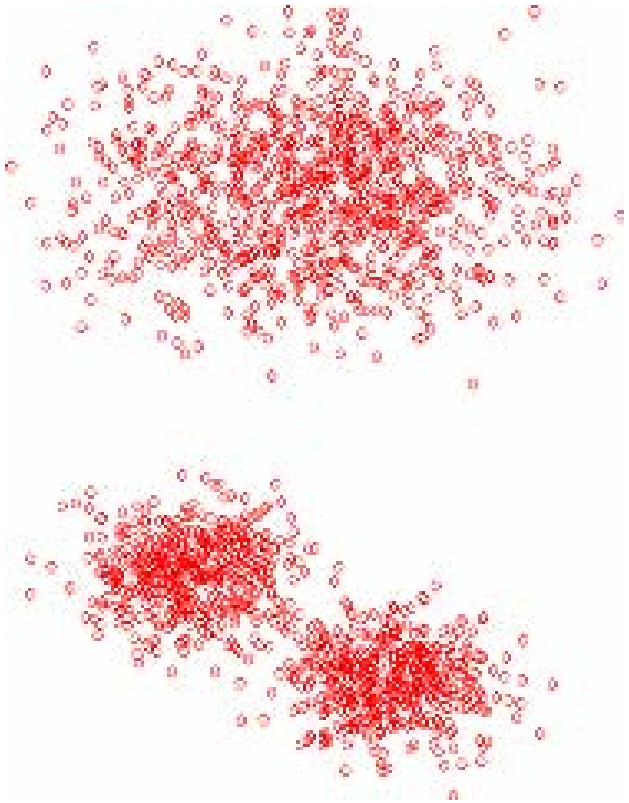


Estimation de la densité

La densité D_j de chaque prototype j est calculée en plaçant une Gaussienne sur chaque point de donnée $x^{(k)} = 1..N$ et en faisant la somme de toutes ces Gausiennes sur j.

$$D_j = \sum_{k=1}^{N} e^{-\frac{\|w_j - x^{(k)}\|^2}{2\sigma^2}}$$

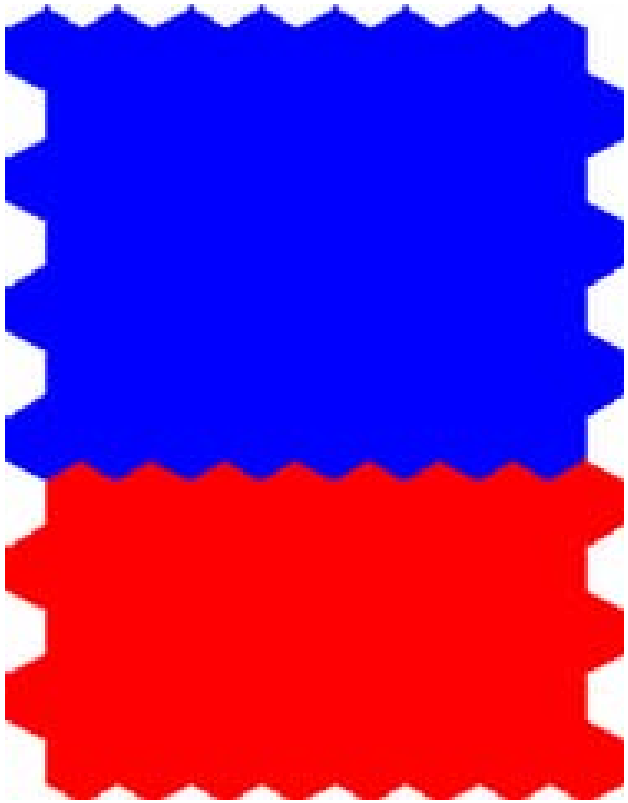
Algorithme 2 : Partitionnement selon le voisinage et la densité



Initial state

Données initiales

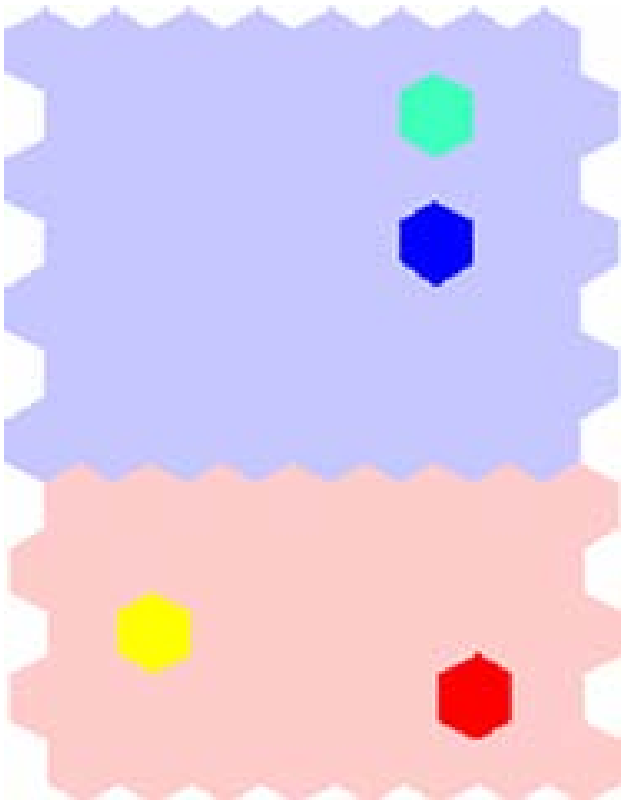
Algorithme 2 : Partitionnement selon le voisinage et la densité



Step 1

Détection des partitions bien séparées en utilisant le voisinage

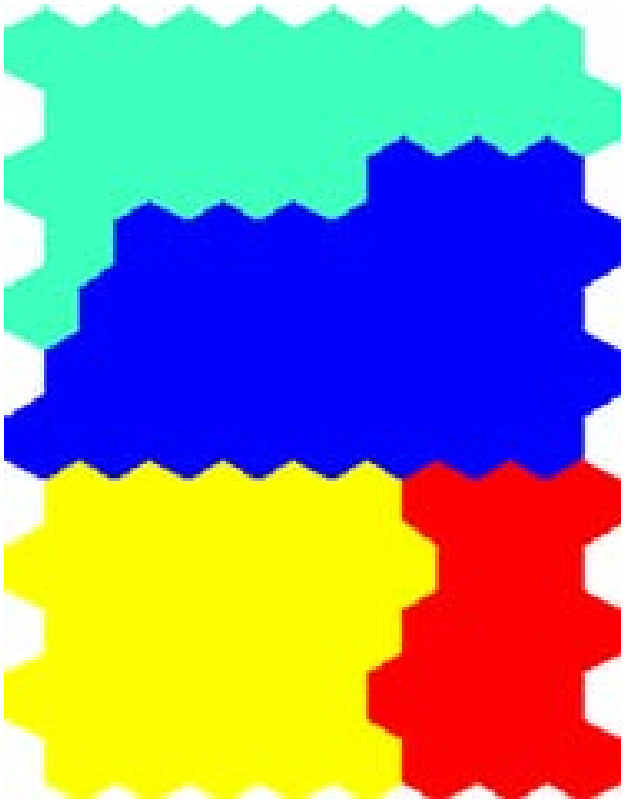
Algorithme 2 : Partitionnement selon le voisinage et la densité



Step 2

Calcul des pics de densités

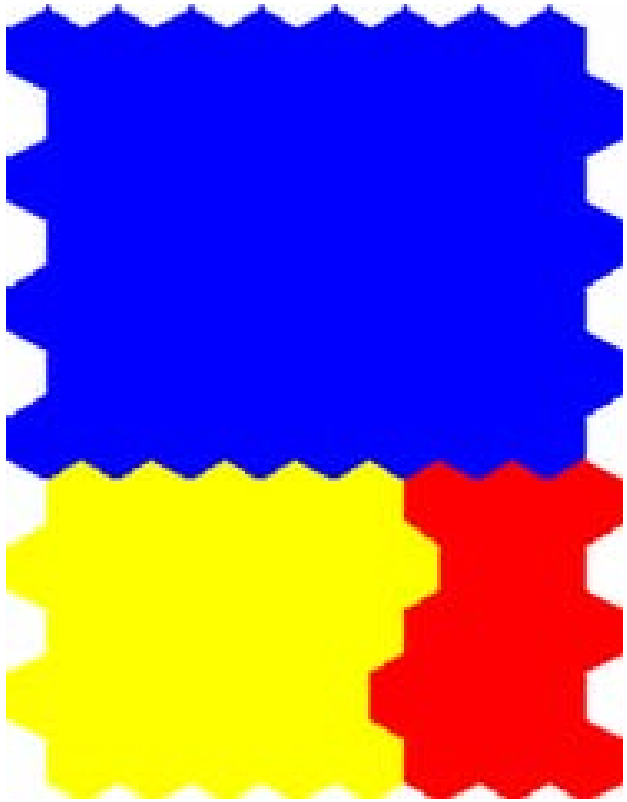
Algorithme 2 : Partitionnement selon le voisinage et la densité



Step 3

Chaque prototype est associé à un pic de densité

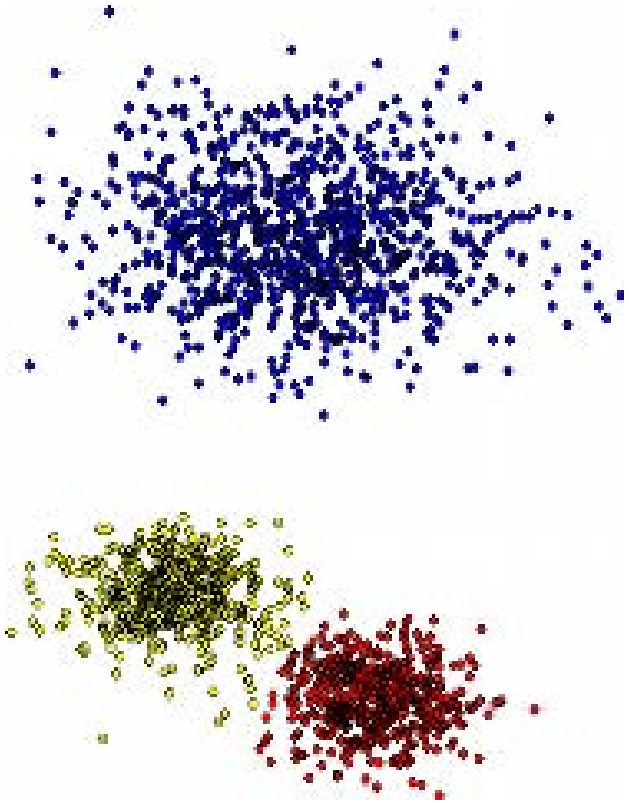
Algorithme 2 : Partitionnement selon le voisinage et la densité



Step 4

Fusion des sous-groupes non pertinents

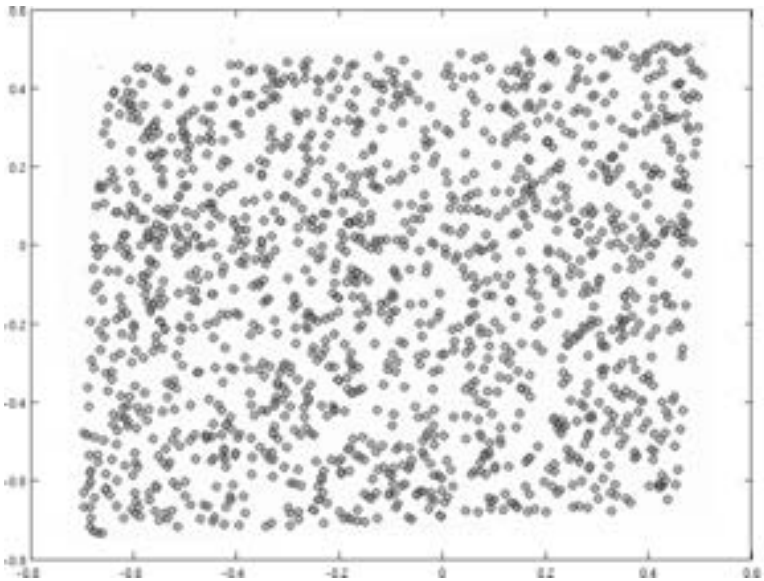
Algorithme 2 : Partitionnement selon le voisinage et la densité

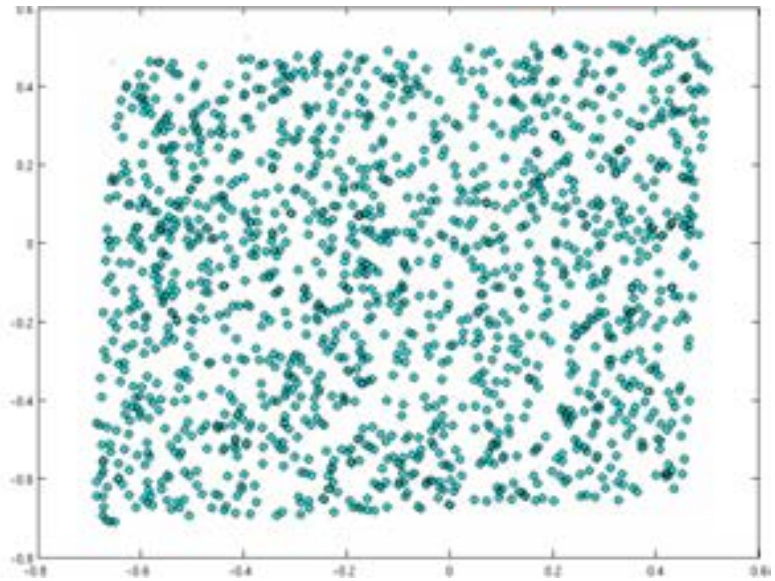


Step 5

Partitionnement des données

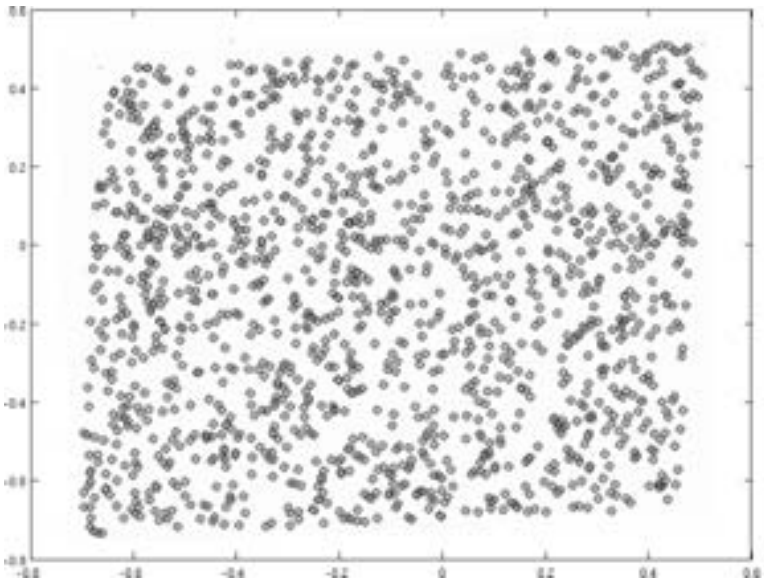
Algorithme 2 : Exemple 1

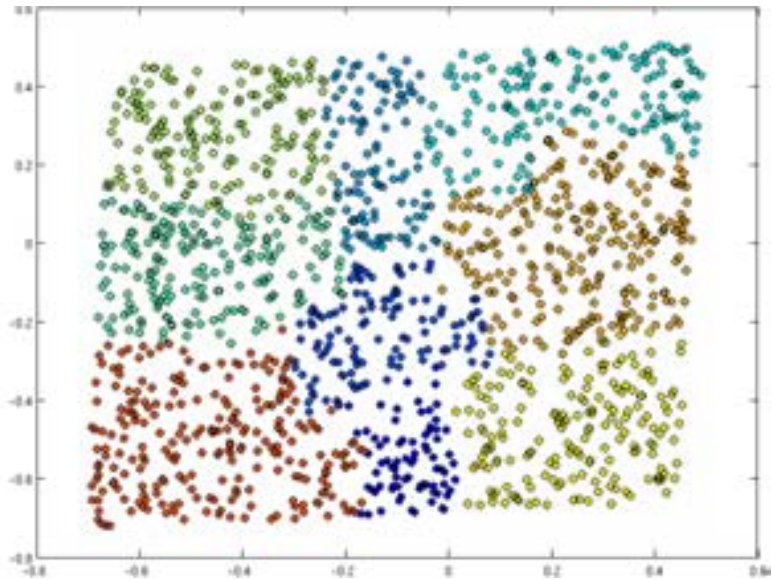




Données non structurées : partitionnement selon DS2L-SOM (détection de l'existence d'une structure)

Algorithme 2 : Exemple 1





Données non structurées : partitionnement selon une méthode à deux niveaux classique

Algorithme 2 : Exemple 2

12000 data points

500 dimensions

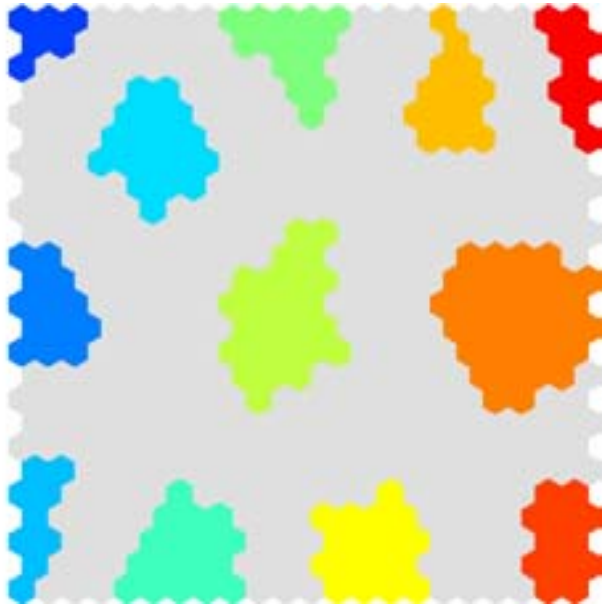
12 clusters

Algorithme 2 : Exemple 2

12000 data points

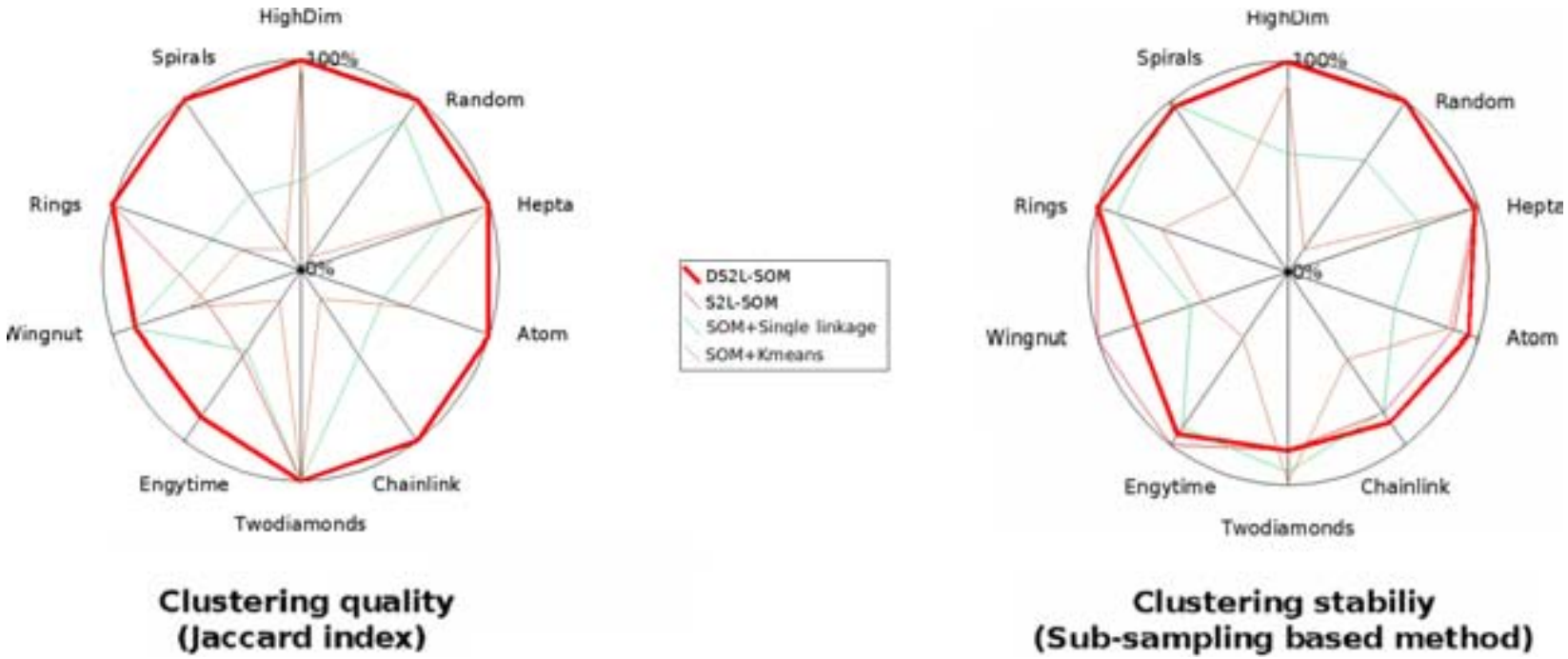
500 dimensions

12 clusters



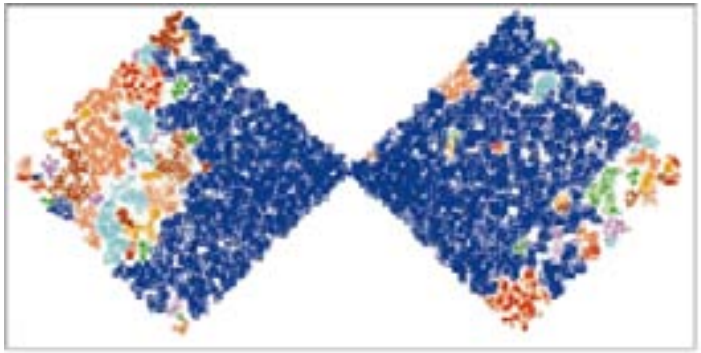
Données “HighDim” : partitionnement selon DS2L-SOM

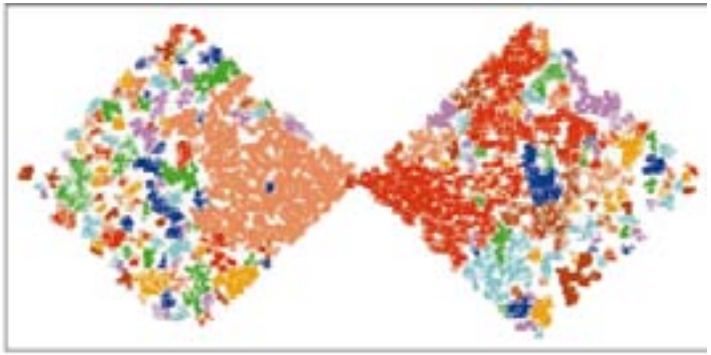
Algorithme 2 : Validité et stabilité



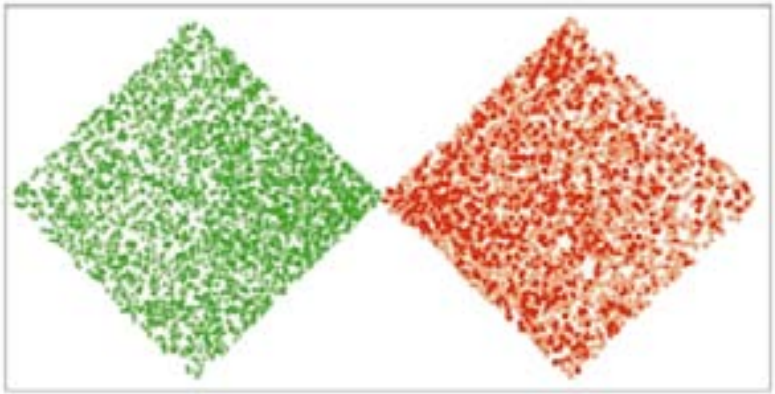
Évaluation du partitionnement en fonction de la base de données et de la méthode utilisée

Algorithme 2 : Comparaisons



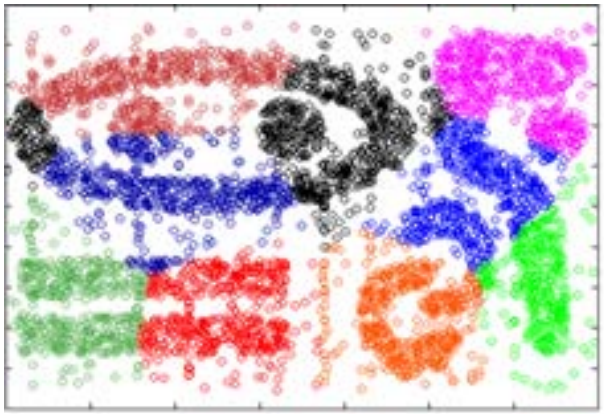


DBSCAN



DS2L-SOM

Algorithme 2 : Comparaisons



Spectral

CURE

DS2L-SOM

Extensions des algorithmes

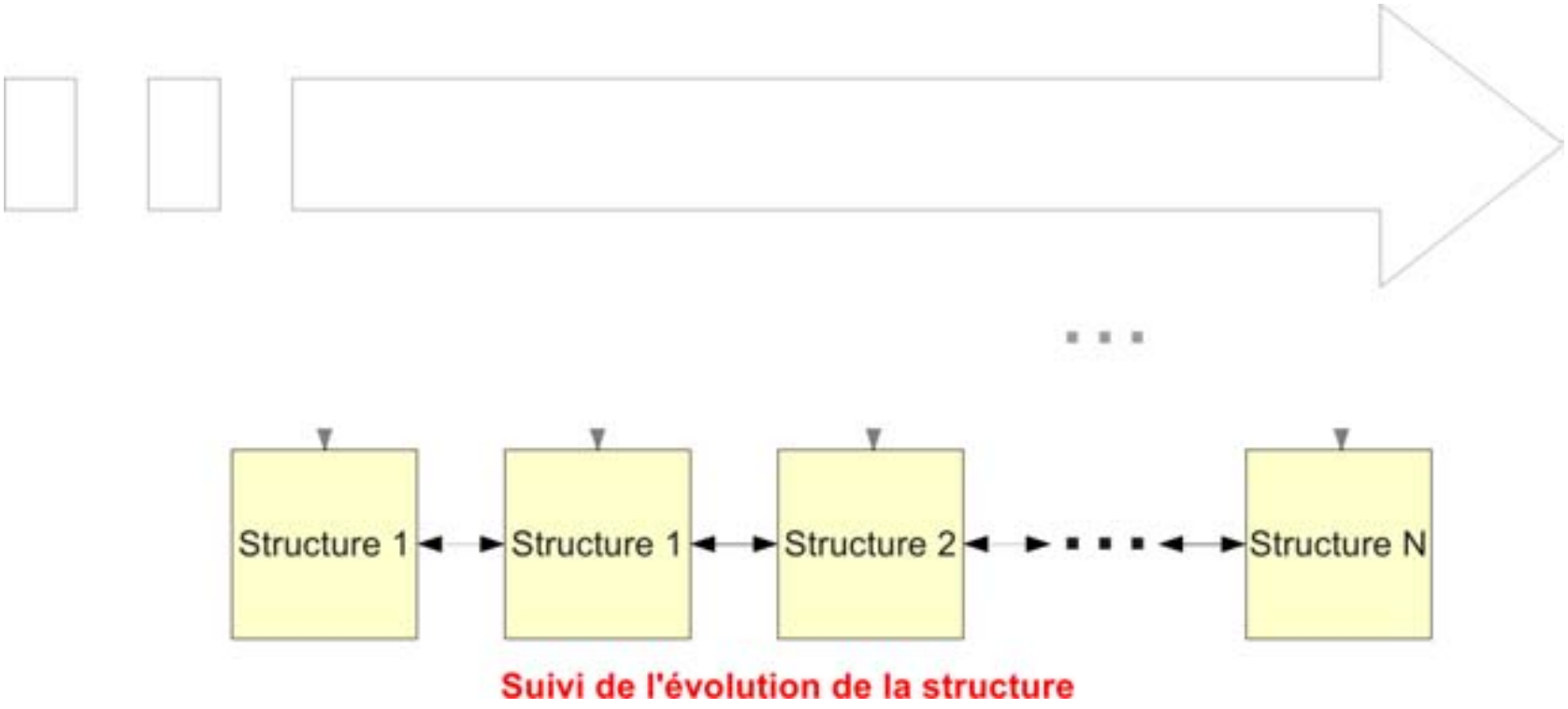
- Versions Batch VS. Stochastique
- Bi-clustering
- Extensions à différents types de données :
  - Données intervalles
  - Matrices de similarité

Propriétés des algorithmes :

- Les étapes de quantification et de partitionnement se font en parallèle
  - Prise en compte directe de la structure des données
  - Bonne qualité du partitionnement obtenu
- Détection automatique du nombre de groupe
- Complexité linéaire en nombre de données

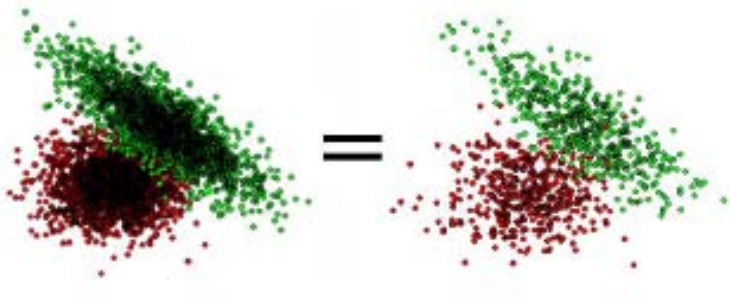
Suivi de la structure des données

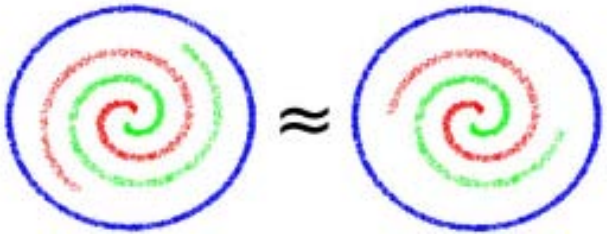
Objectifs



Suivi de l'évolution de la structure

Objectifs





Analyser et comparer la structure des données

Principe général

1. **Représentation des données par un ensemble de prototypes enrichis**

2. **Modélisation de la distribution des données à partir de l'estimation d'une fonction de densité**

   - **Estimation de cette fonction de densité à partir des informations associés aux prototypes**
   - **La méthode doit avoir une complexité linéaire pour suivre l'évolution des données**

3. **Comparaisons de la densité des données**

Enrichissement des prototypes

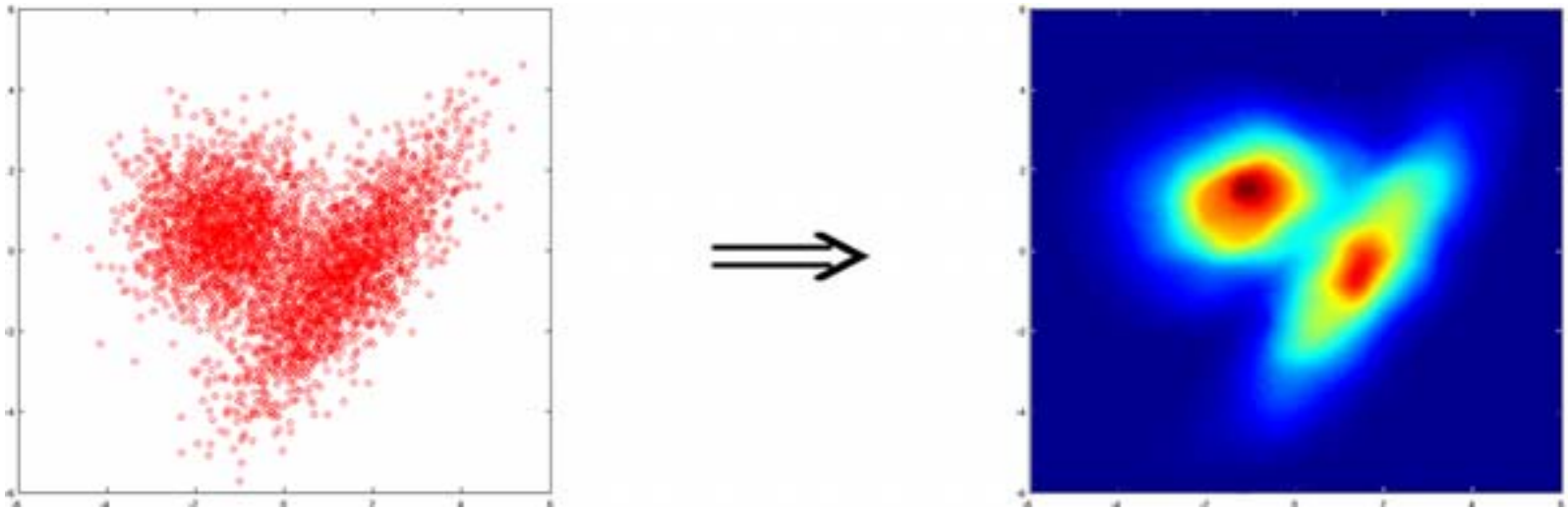
Pendant l'apprentissage, chaque prototype est associé à de nouvelles informations structurelles extraites des données.

Informations associées aux prototypes

- Densité locale
- Variabilité locale
- Voisinage

Estimation de la distribution des données

Estimer la fonction de densité qui associe à chaque point de l'espace de représentation des données une densité.



Estimation de la distribution : Principe

Nous connaissons la densité au niveau des prototypes. Il faut en déduire une approximation de la fonction.

Hypothèse

Cette fonction peut être approximée sous la forme d'un mélange de noyaux Gaussiens sphériques centrés sur les prototypes :

$$f(x) = \sum_{i=1}^{M} \alpha_i K_i(x) \qquad \text{avec} \qquad K_i(x) = \frac{1}{\sqrt{2\pi}.h_i} e^{-\frac{|w_i - x|^2}{2{h_i}^2}}$$

Problème

Il faut déterminer les paramètres h et α.

Estimation de la distribution : Principe

Nous connaissons la densité au niveau des prototypes. Il faut en déduire une approximation de la fonction.

Hypothèse

Cette fonction peut être approximée sous la forme d'un mélange de noyaux Gaussiens sphériques centrés sur les prototypes :

$$f(x) = \sum_{i=1}^{M} \alpha_i K_i(x) \qquad \text{avec} \qquad K_i(x) = \frac{1}{\sqrt{2\pi}.h_i} e^{-\frac{|w_i - x|^2}{2{h_i}^2}}$$

Problème

Il faut déterminer les paramètres h et α.

Estimation de la distribution : Choix du paramètre h

L'idée est que h_i estime l'étendue des données représentées par la Gaussienne K_i, ces données étant captées par le neurone i et par ses voisins.

$$h_i = \frac{1}{\sum_j v_{i,j}} \sum_j v_{i,j} \frac{s_i N_i + d_{i,j} N_j}{N_i + N_j}$$

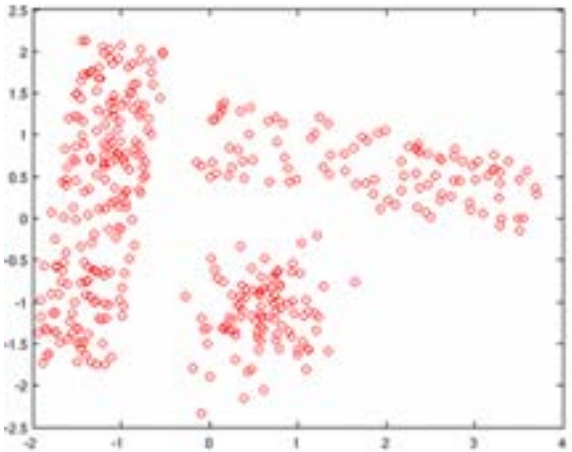
Estimation de la distribution : Choix du paramètre α

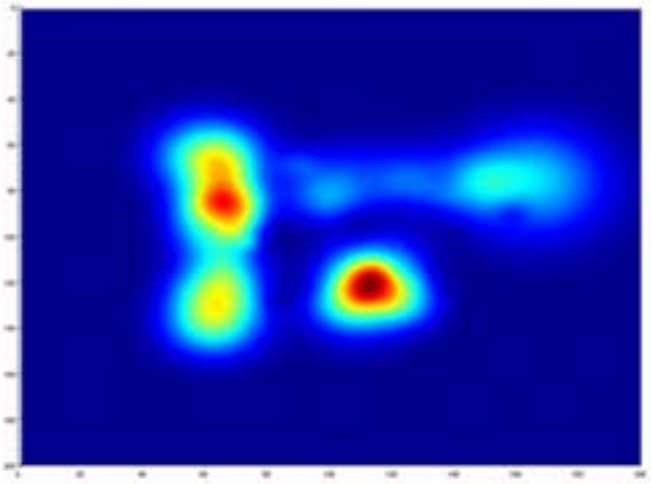
Principe

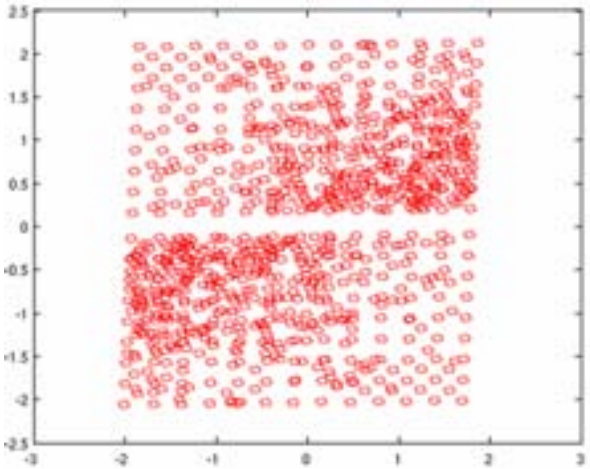
- La densité D de chaque prototype w est connue ($f(w_i) = D_i$).
- Nous pouvons utiliser une méthode de descente de gradient pour déterminer α_i.

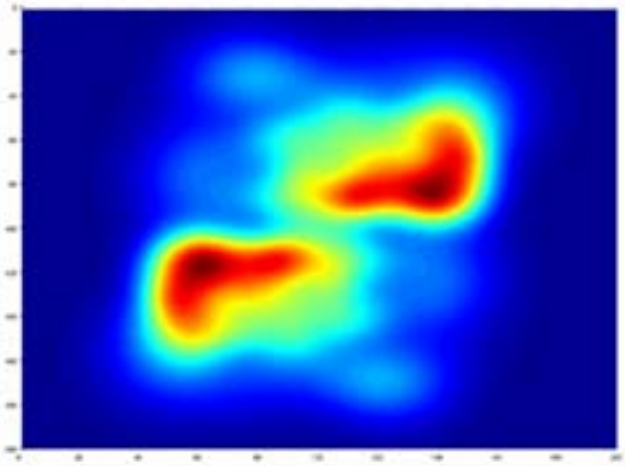
$$\alpha = \arg\min_{\alpha} \frac{1}{M} \sum_{i=1}^{M} \left[\sum_{j=1}^{M} (\alpha_j K_j(w_i)) - D_i \right]^2$$

Estimation de la distribution : Exemples

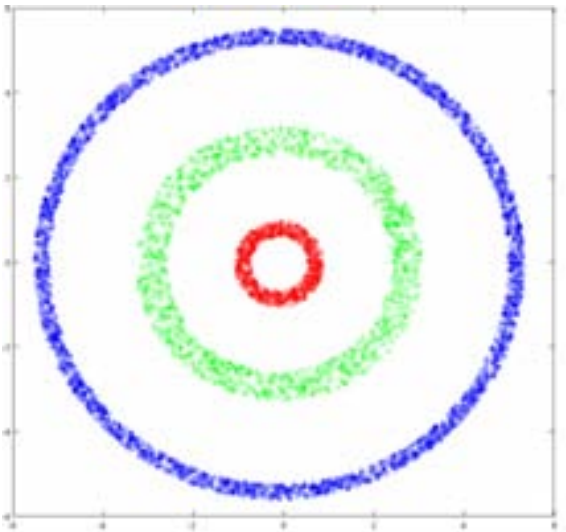


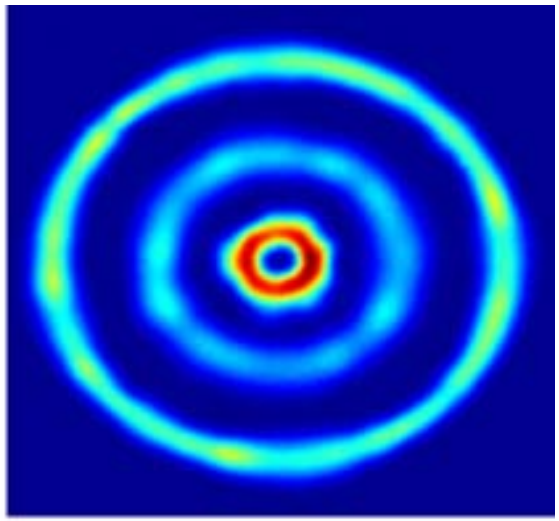


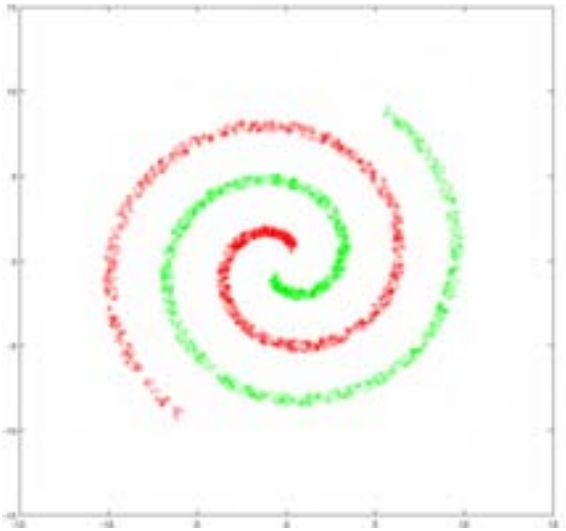


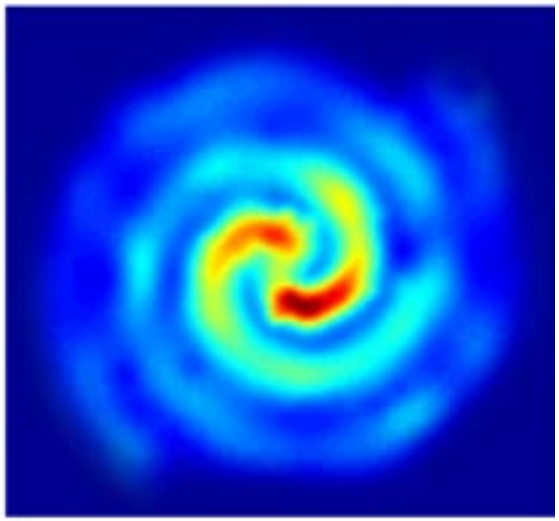


Estimation de la distribution : Exemples









Objectif

Définir une mesure de dissimilarité entre deux ensembles de données A et B, représentés par deux ensembles :

$$E_A = \left[\{w_i^A\}_{i=1}^{M^A}, f^A\right] \quad \text{et} \quad E_B = \left[\{w_i^B\}_{i=1}^{M^B}, f^B\right]$$

Comparaison de structures : Mesure proposée

$$CBd(A,B) = \frac{\sum_{i=1}^{M^A} f^A\left(w_i^A\right) log\left(\frac{f^A\left(w_i^A\right)}{f^B\left(w_i^A\right)}\right)}{M^A} + \frac{\sum_{j=1}^{M^B} f^B\left(w_j^B\right) log\left(\frac{f^B\left(w_j^B\right)}{f^A\left(w_j^B\right)}\right)}{M^B}$$

Cette mesure est adaptée de l'approximation de Monte Carlo de la mesure symétrique de Kullback–Leibler.

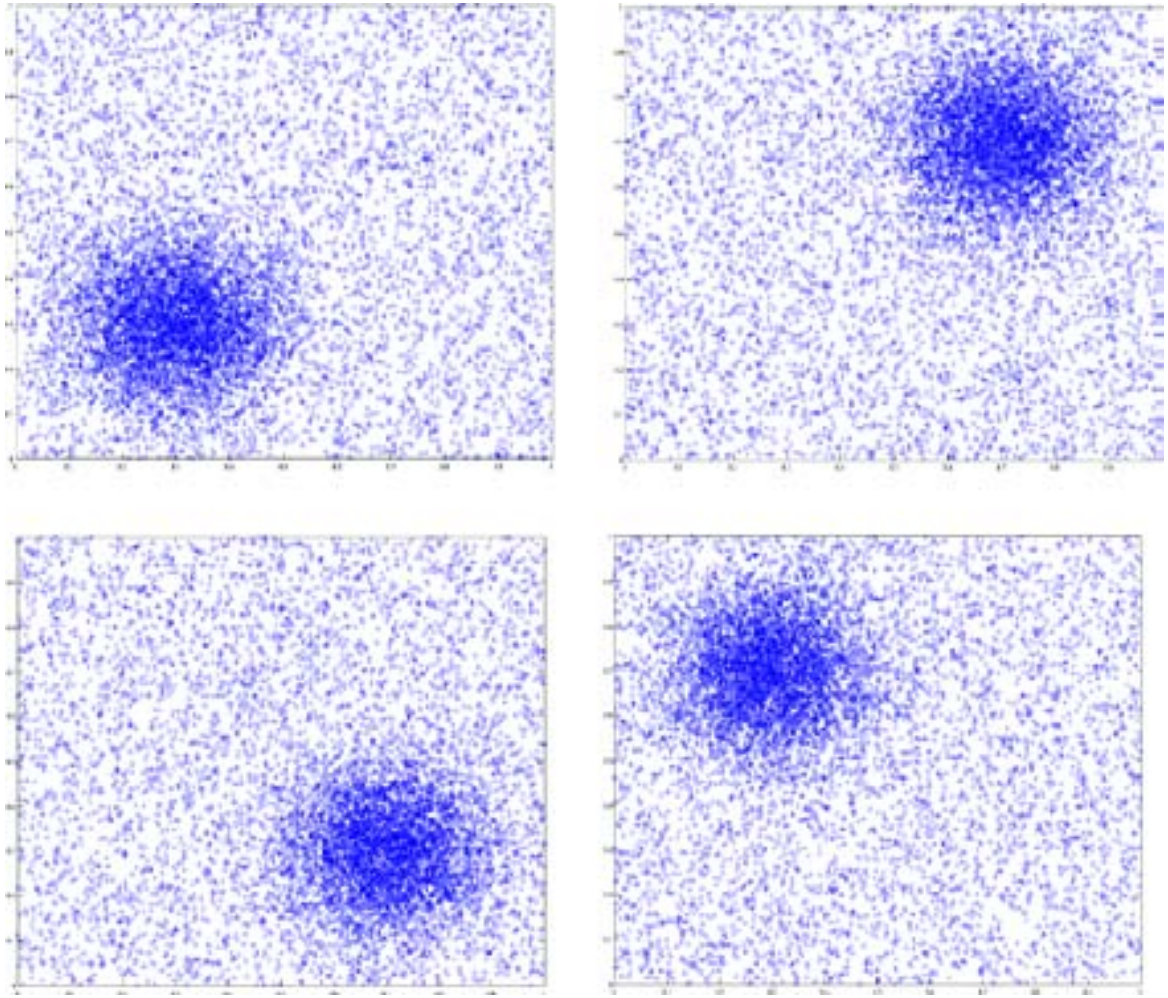
Validation expérimentale : Principe

Ce qu'il faut montrer

- Les données proviennent de distributions différentes $\Rightarrow$ grande dissimilarité entre deux modèles.
- Les deux distributions sont similaires $\Rightarrow$ faible dissimilarité entre deux modèles.

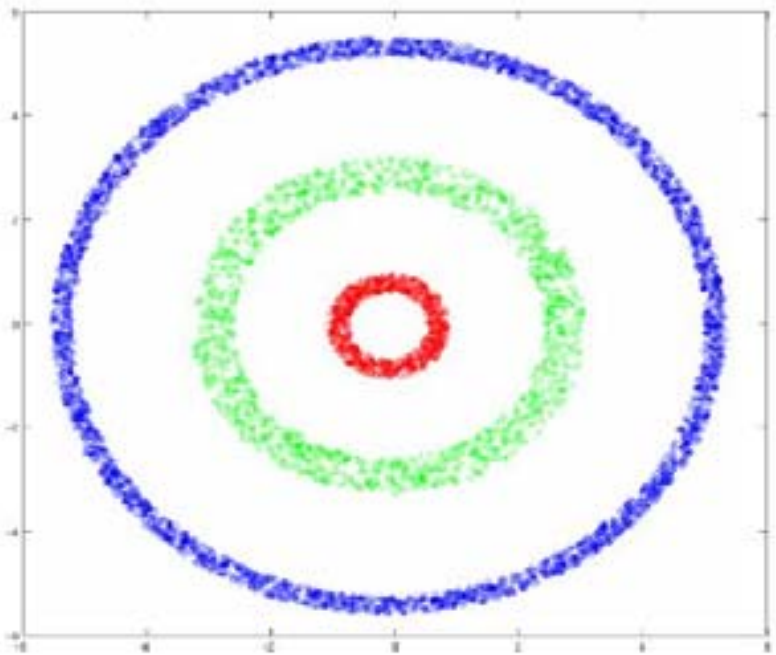
La méthode est performante si la dissimilarité est bien plus faible lorsque les données suivent la même loi de distribution que lorsque les données suivent des distributions très différentes.

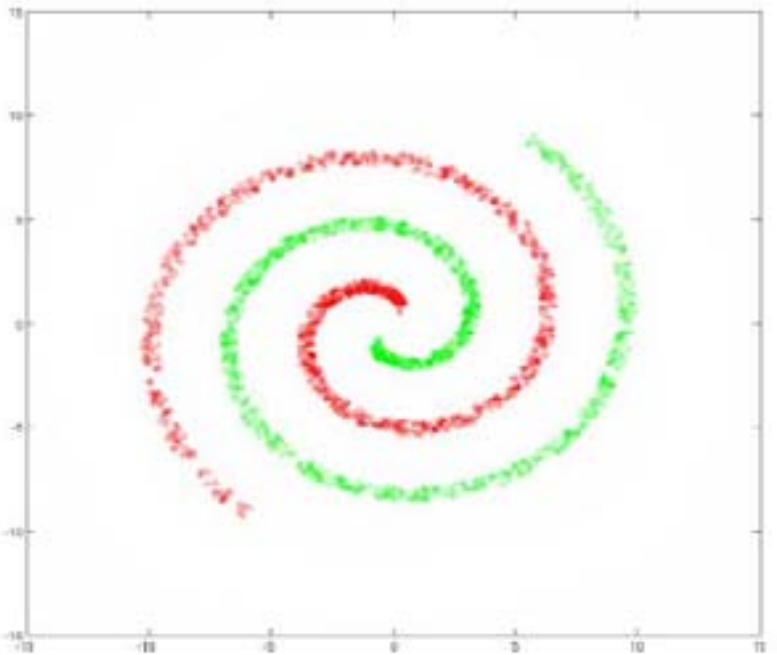
Validation expérimentale : Bases de test



Distributions Gaussiennes très bruitées.

Validation expérimentale : Bases de test





Distributions non convexes.

Validation expérimentale : Bases de test

Base “Shuffle”

1. Base de données réelle issue de l’UCI repository.
2. 5800 instances.
3. 9 variables.
4. Divisée en deux classes de distributions différentes.

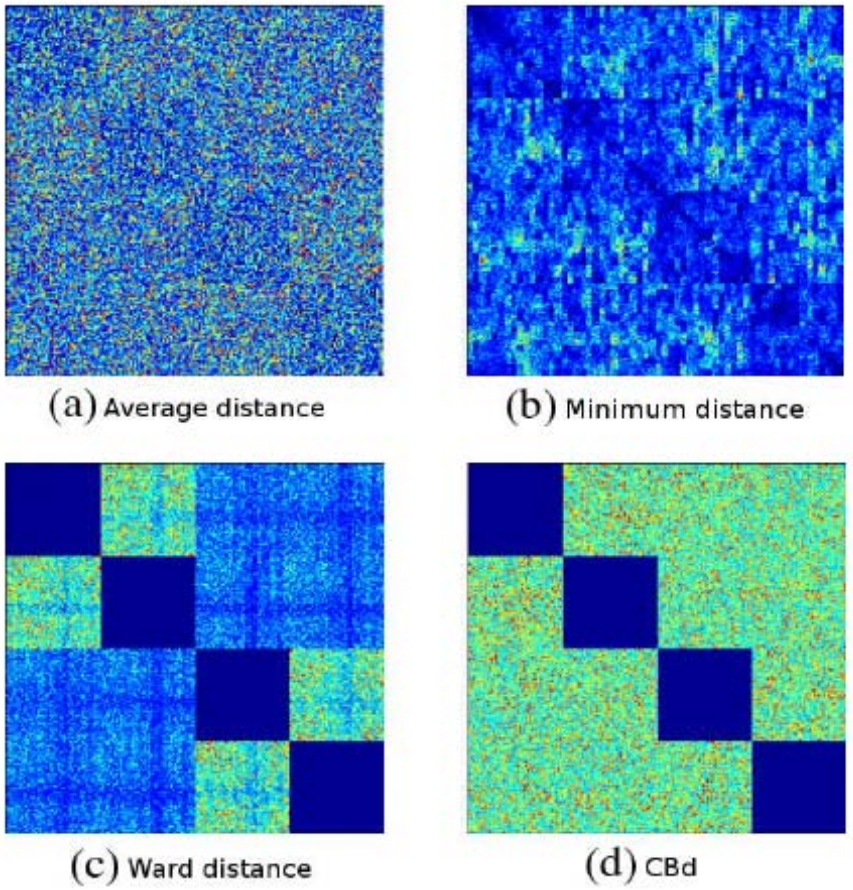
Validation expérimentale : Résultats

Distributions à comparer	Ad	Md	Wd	**CBd**
Ring 1 à 3 + Spiral 1 et 2	0.4	0.9	0.5	**1.6**
Noise 1 à 4	1.1	1.4	22.0	**115.3**
Shuffle 1 et 2	1.1	16.5	6.3	**27.6**

$$\text{Indice de Dunn} = \frac{\min(d_{inter})}{\max(d_{intra})}$$

La distance utilisant la densité est toujours meilleure que les mesures de similarité basées sur les distances entre prototypes.

Validation expérimentale : Résultats



(a) Average distance (b) Minimum distance

(c) Ward distance (d) CBd

Très bonnes performances pour la comparaison de données Gaussiennes bruitées, par rapport à des mesures de similarité basées sur les distances entre prototypes.

Avantages de la méthode :

- Rapide : mise à jour “en ligne”. Complexité linéaire.
- Faible quantité d’information à stocker pour chaque modèle.
- Modèles obtenus fiables.

Visualisation

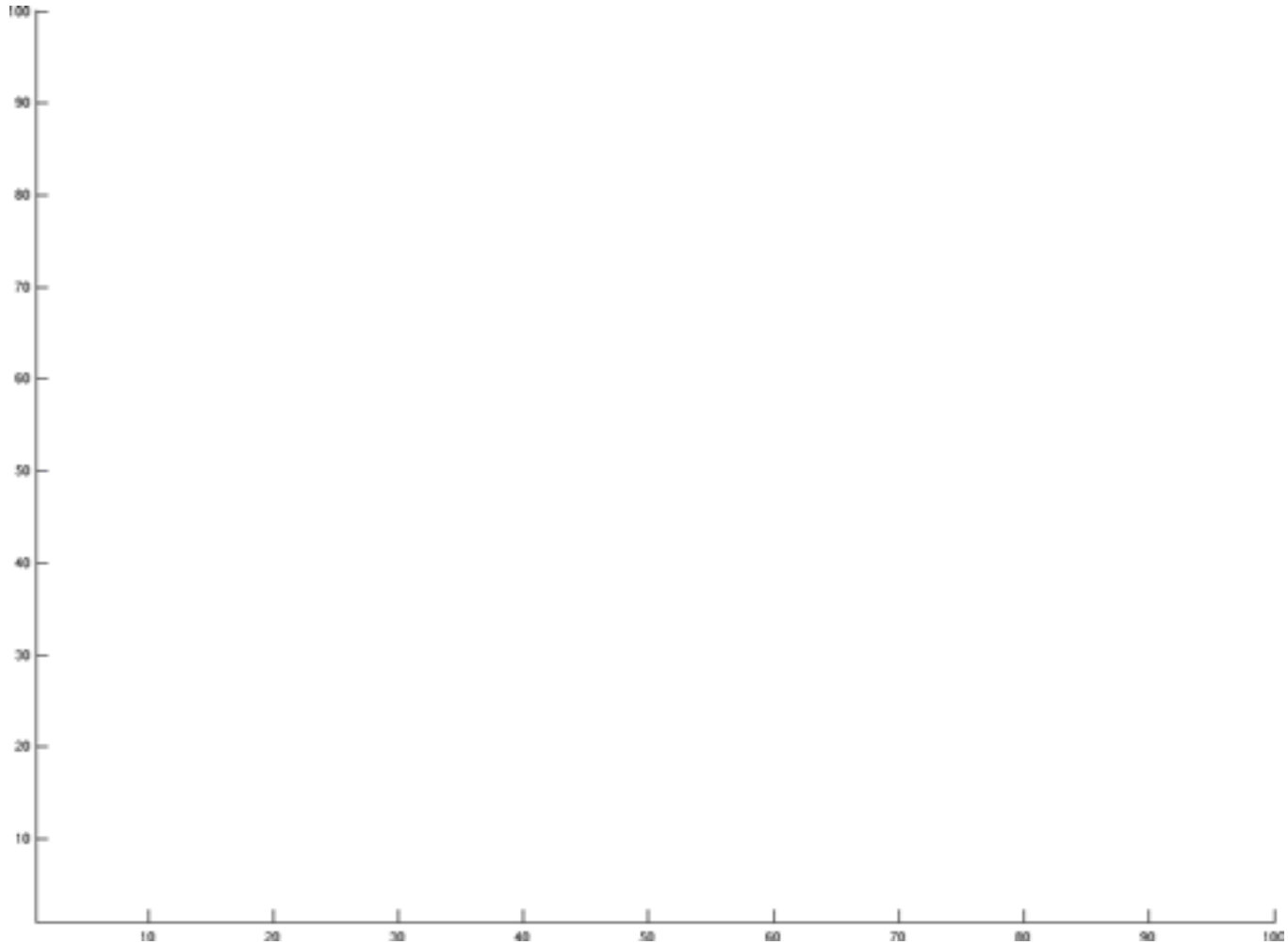
Prototypes enrichis

Pendant l'apprentissage, chaque prototype est associé à de nouvelles informations structurelles extraites des données.

Informations associées aux prototypes

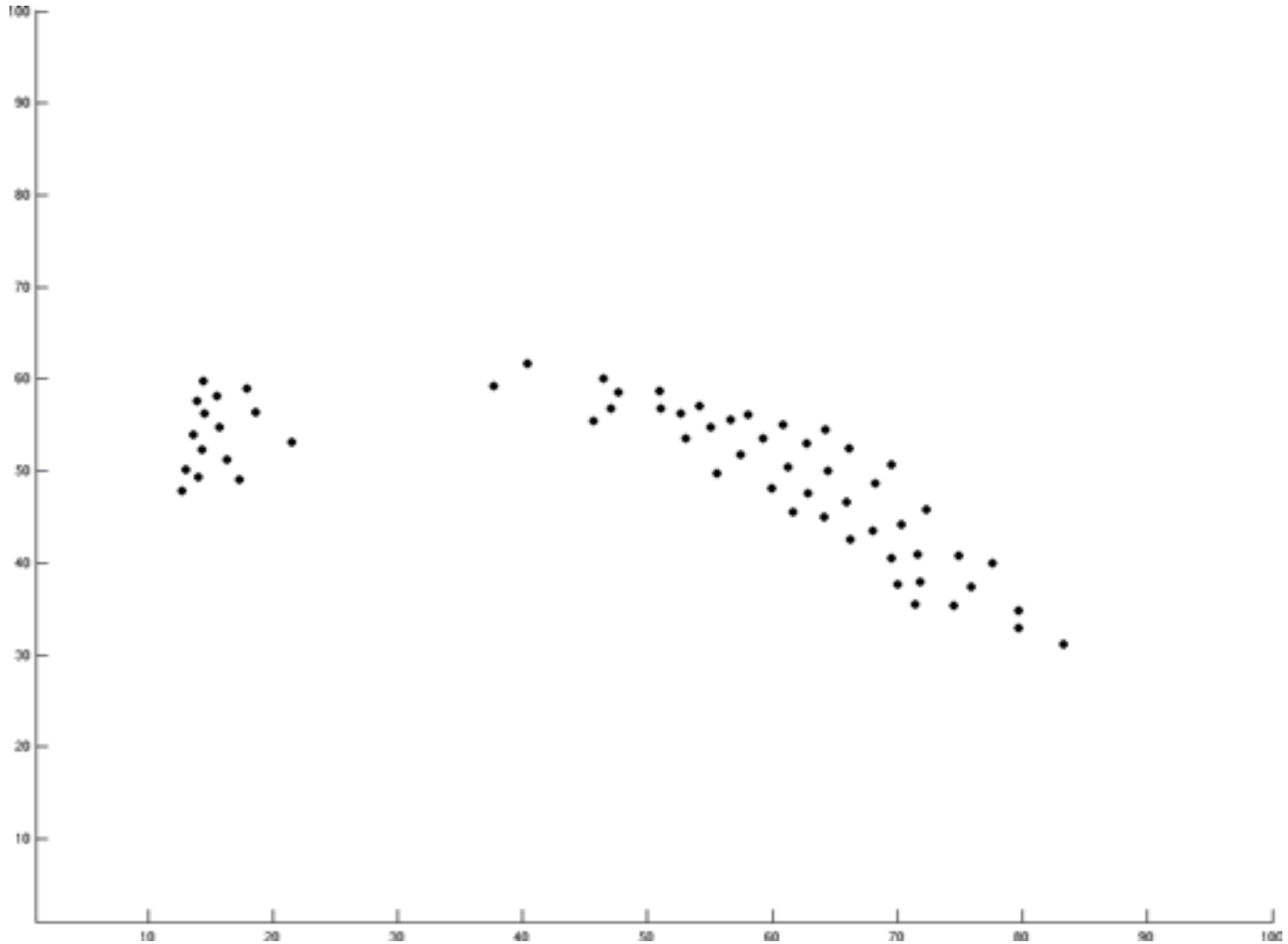
- Voisinage
- Densité locale
- Variabilité locale

Principe de la visualisation



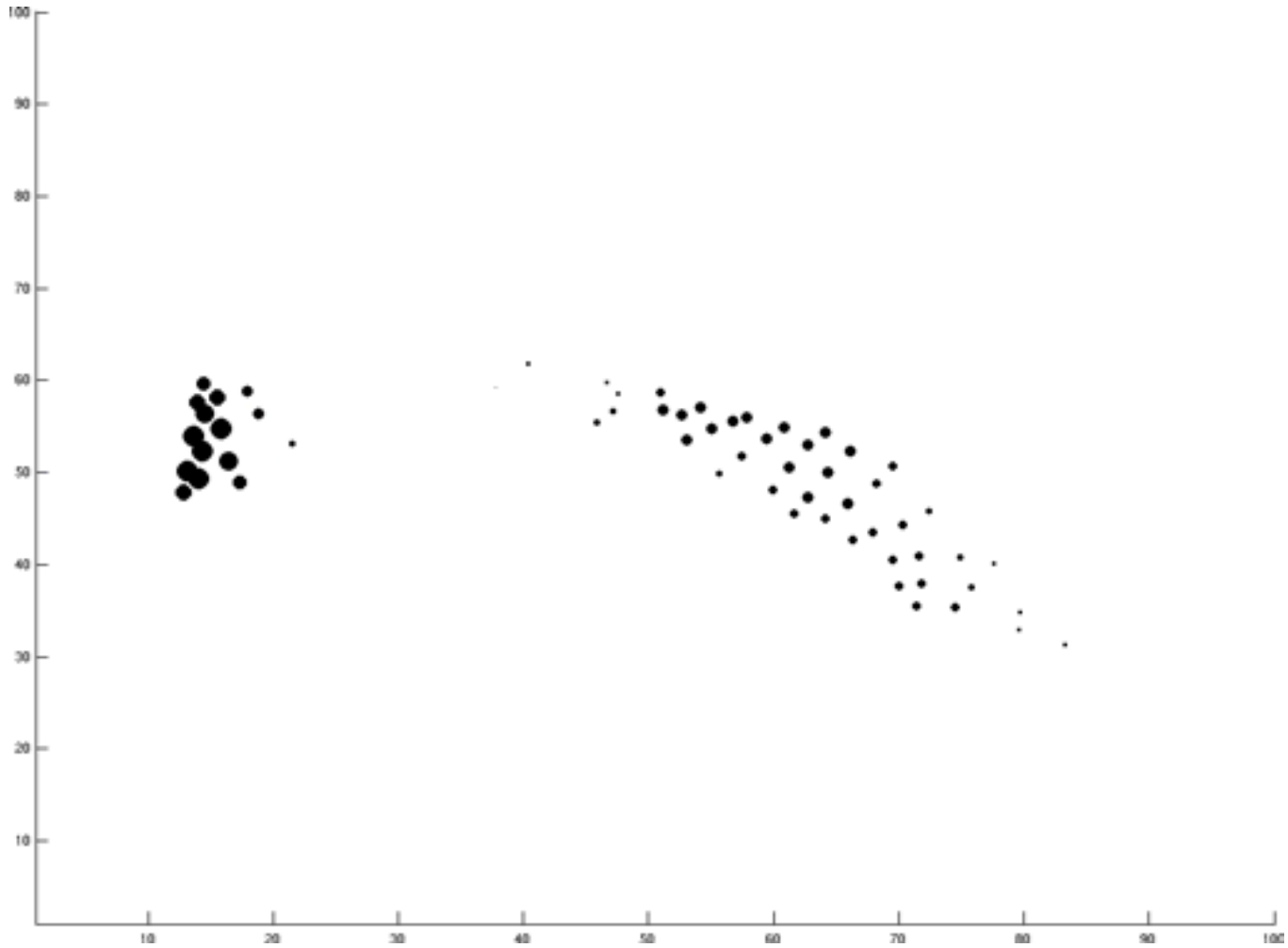
État initial

Principe de la visualisation



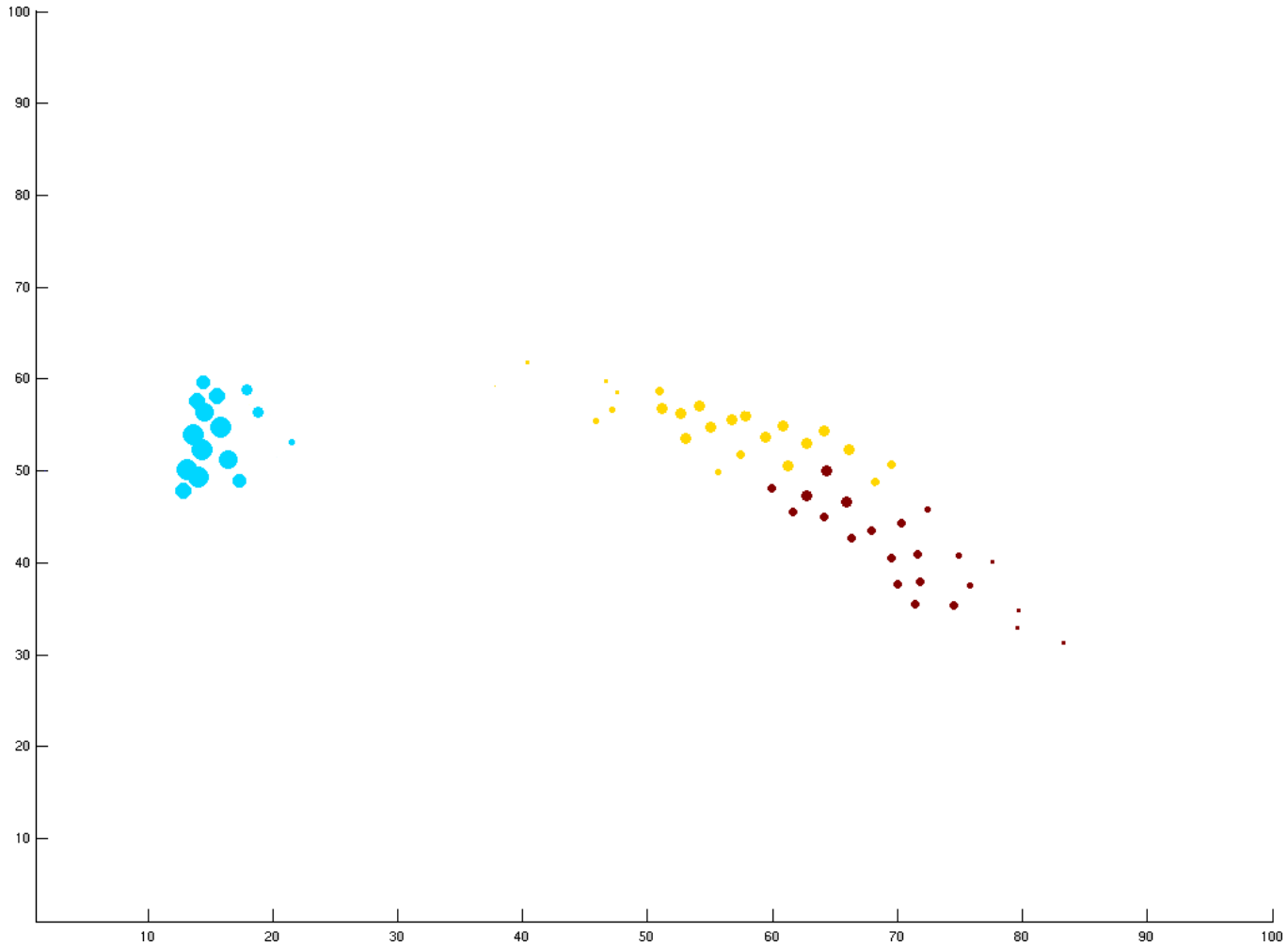
Visualisation des prototypes (sammon)

Principe de la visualisation



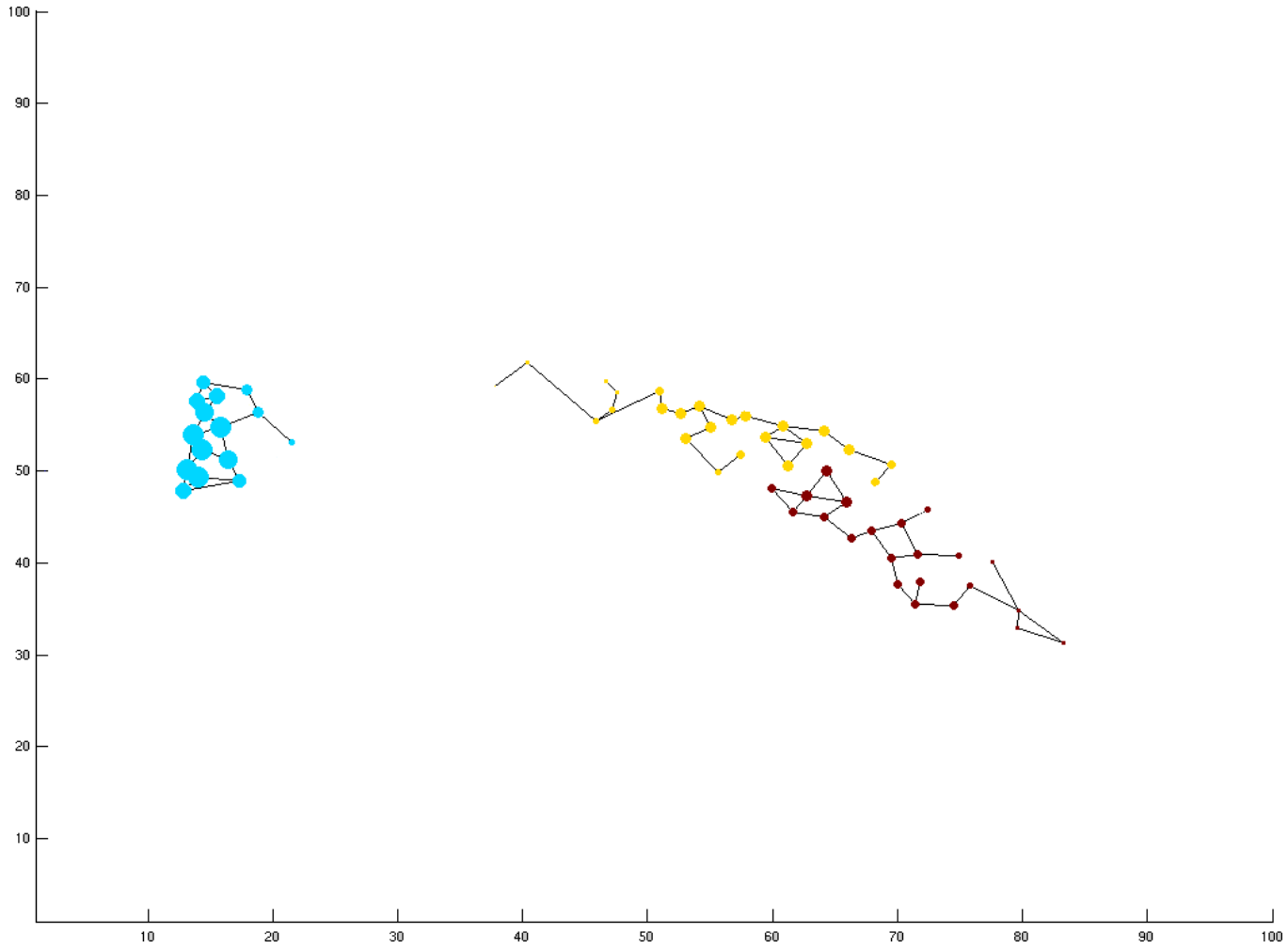
Visualisation de la densité locale

Principe de la visualisation



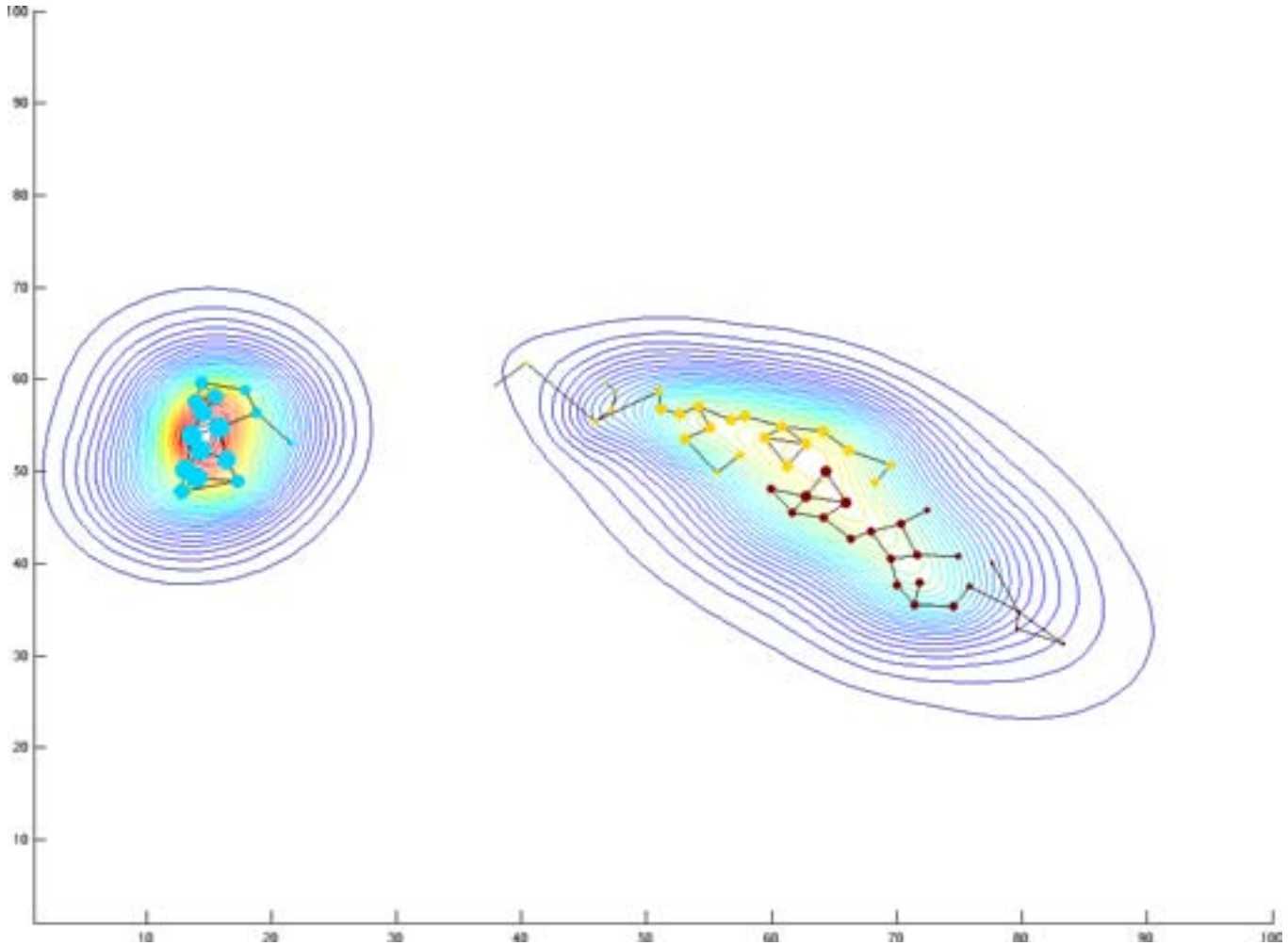
Visualisation des partitions

Principe de la visualisation



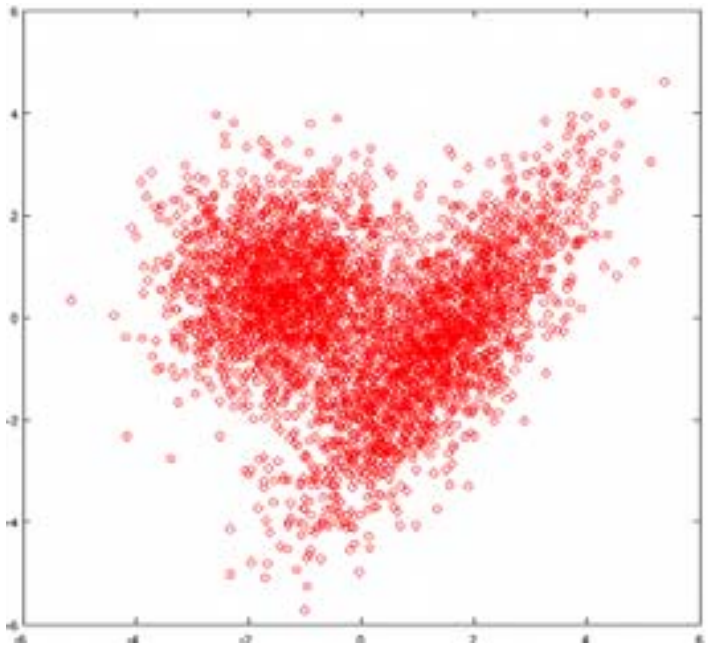
Visualisation de la connectivité

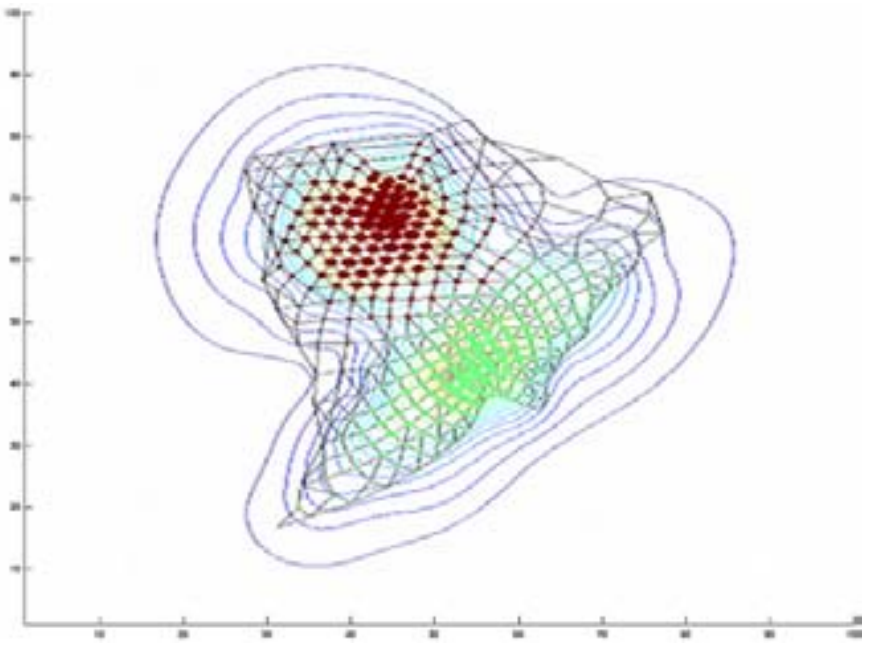
Principe de la visualisation



Visualisation de la distribution

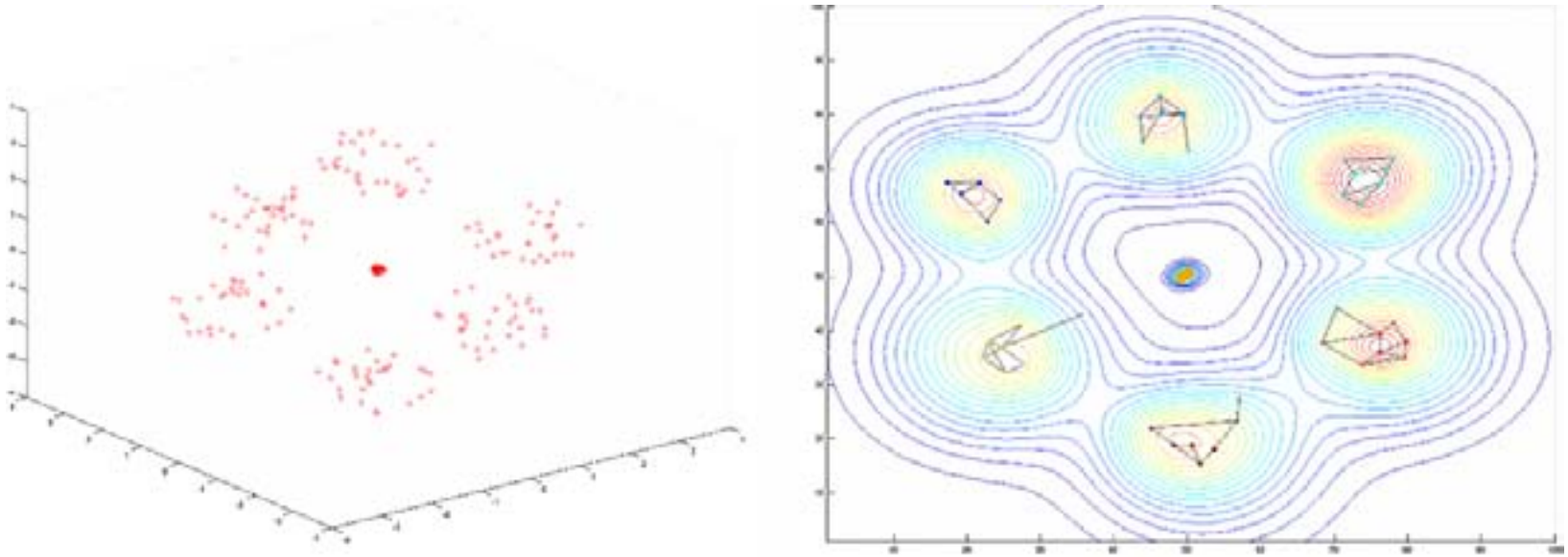
Exemples





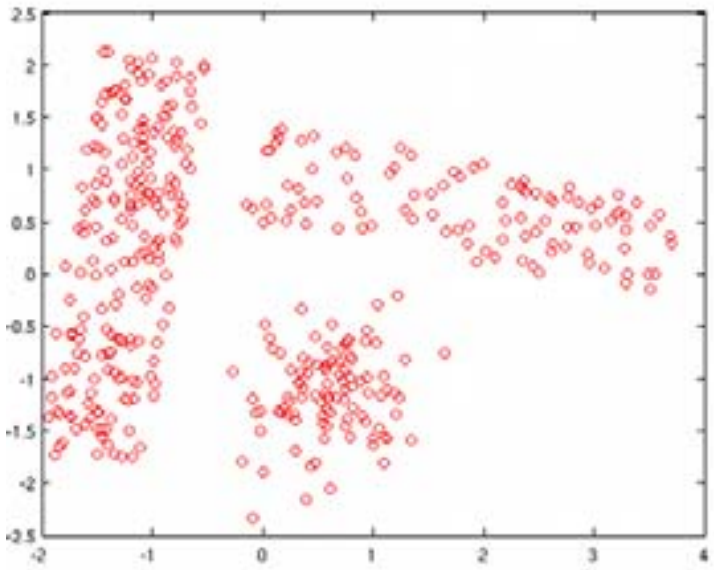
Visualisation des données “Engytime”

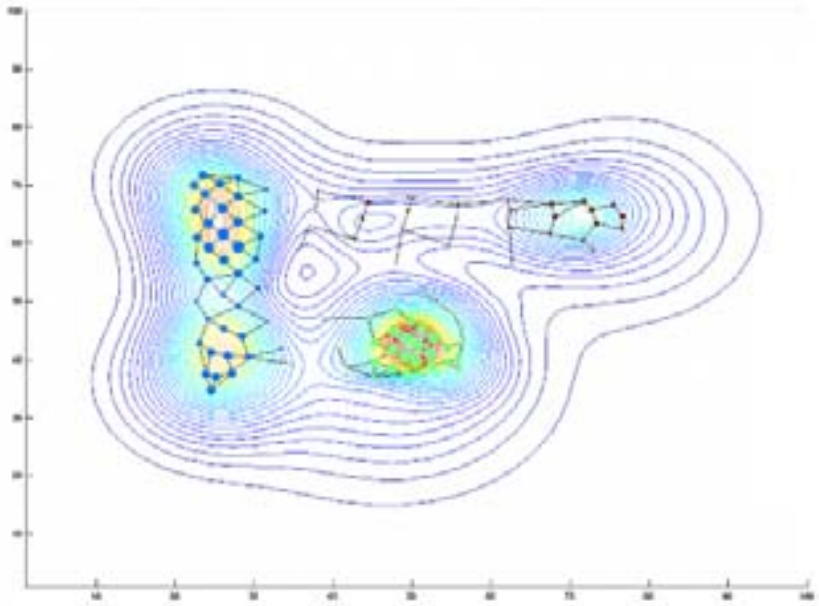
Exemples



Visualisation des données “Hepta”

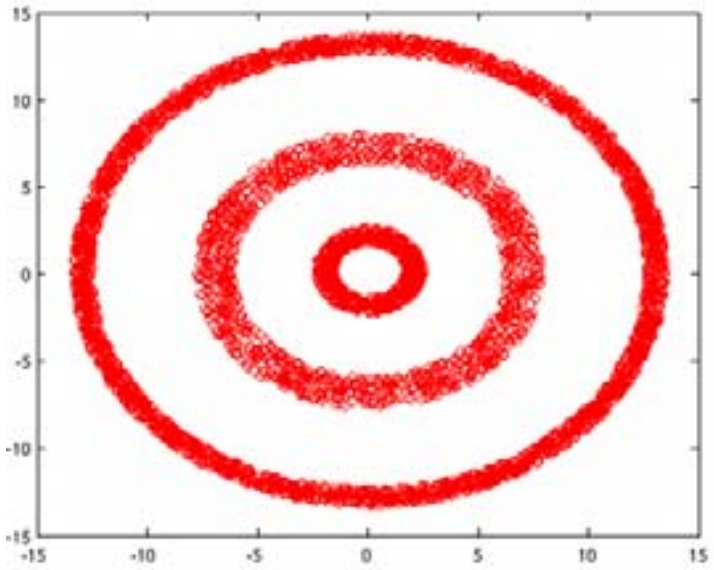
Exemples

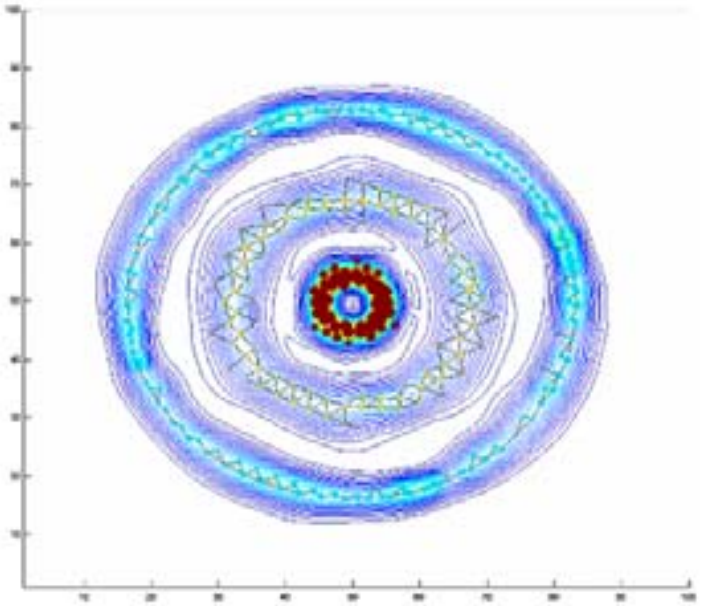




Visualisation des données “Lsun”

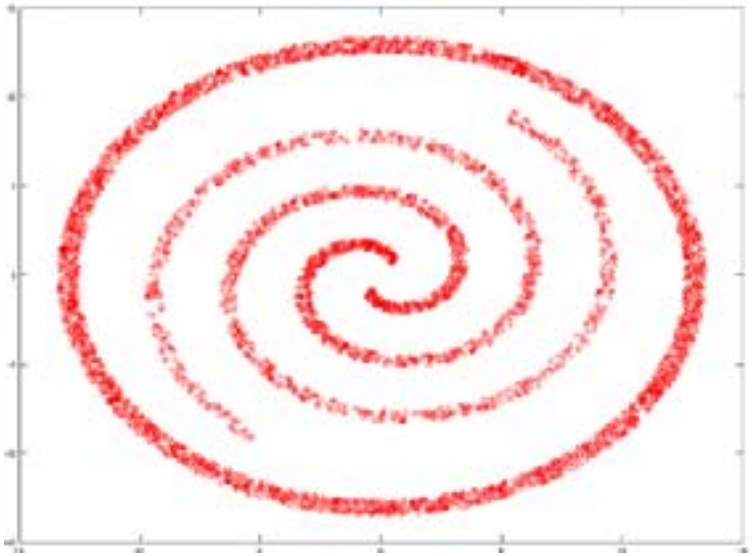
Exemples

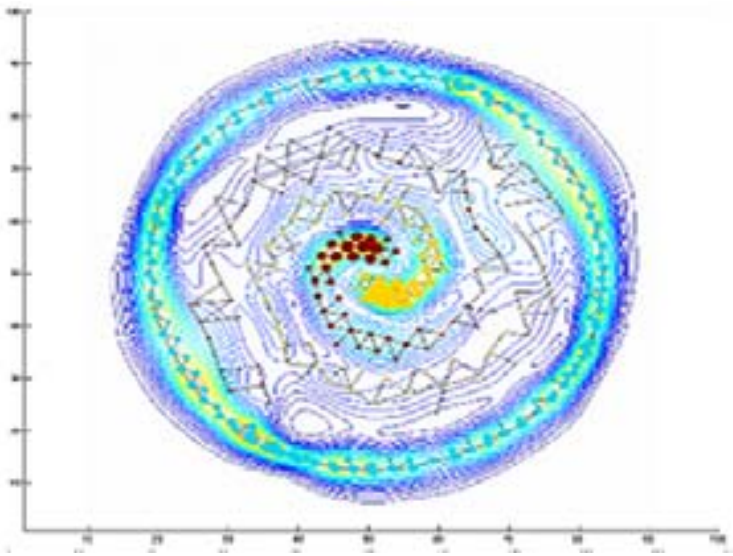




Visualisation des données “Rings”

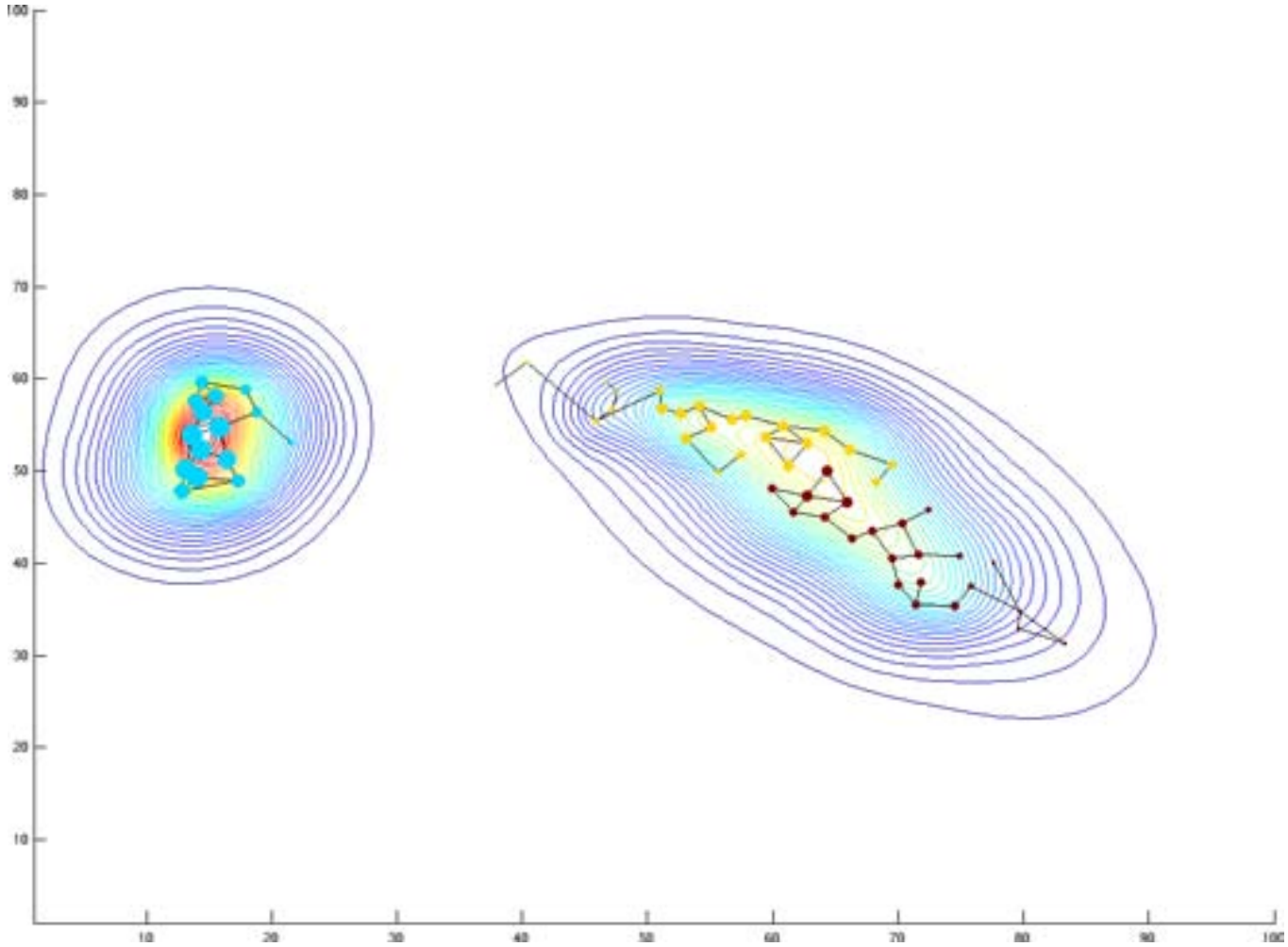
Exemples





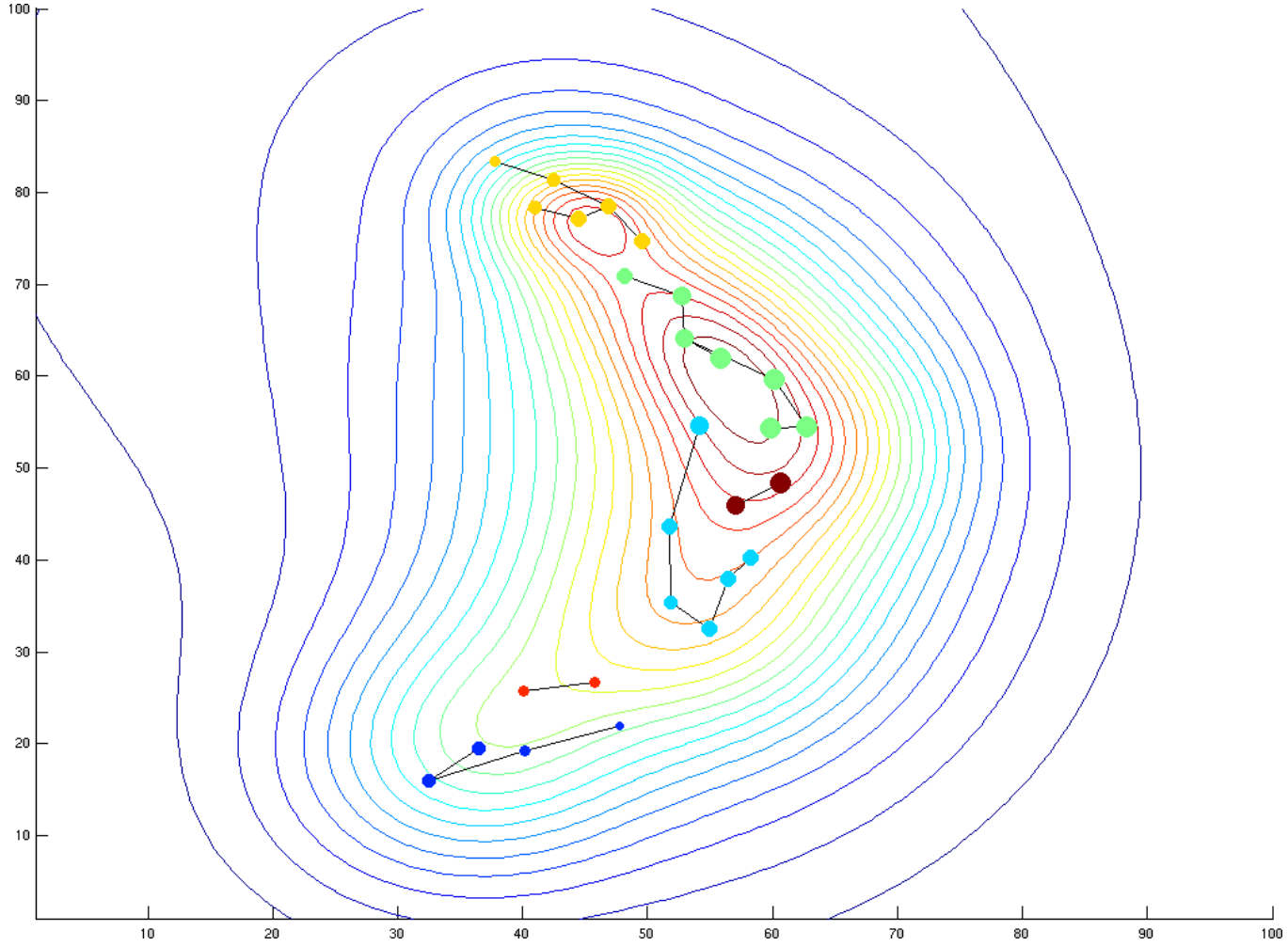
Visualisation des données “Spirals”

Exemples



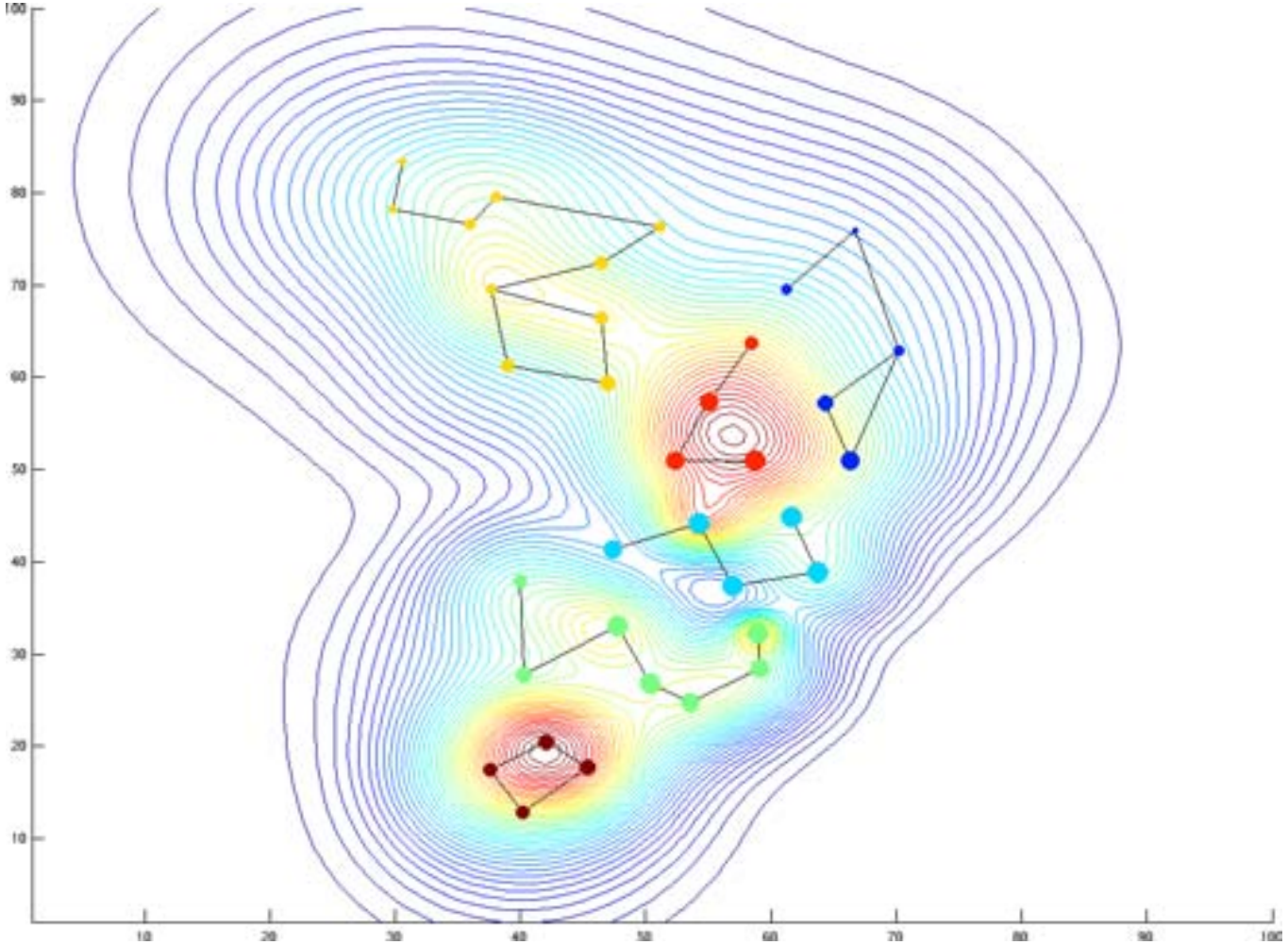
Visualisation des données “Iris”

Exemples



Visualisation des données “Ants”

Exemples



Visualisation des données “Children”

Bilan

- Nouvelle méthode de modélisation de la structure des données.
- Procédé de visualisation de cette structure, capable de mettre en valeur la structure intra et inter-groupes des données.
- La visualisation proposée se montre pertinente sur un ensemble d'exemples artificiels et réels.
- Perspectives : visualisation interactive ?

Conclusion

Contributions

- Nouveaux algorithmes de classification non supervisée pour données vectorielles.
- Extensions aux données non vectorielles.
- Amélioration de la quantification des données par une SOM.
- Estimation et comparaison de la densité des données.
- Outils de visualisation.

Perspectives

- Test et validation de l'algorithme de quantification de données évolutives.
- Analyse de données non vectorielles selon la densité.
- Version à noyaux et matrice de similarité.
- Étude plus approfondie de l'estimation de la distribution.
- Sélection de modèles.