Renaud Sirdey

Introduction à l'apprentissage par compétition

Éditions des Nik's News
www.niksnews.com/editions/

1998

Introduction

Ce document consiste en une brève présensation formelle des modèles connexionnistes à apprentissage par compétition. Le principal intérêt de ces systèmes vient de leur capacité à apprendre sans professeur : ils ne nécessitent que peu de connaissances *a priori* sur le processus à analyser. Dans un premier temps, le lecteur trouvera une présentation des réseaux de neurones à apprentissage par compétition, suivie dans un second temps d'une description des cartes de Kohonen.

1 Réseaux de neurones

1.1 Le neurone formel

Comme dans les autres types de perceptron, l'unité de traitement de l'information est appelée neurone. Un neurone est défini par un vecteur d'entrée $\vec{x}$ (vecteur forme) et par un vecteur de poids synaptiques $\vec{\omega}$. Dans le type de réseau présentement étudié la sortie d'un neurone est liée à la distance euclidienne[1] qui sépare le vecteur d'entrée du vecteur de poids. Soit

$$\nu(\vec{x}) = \frac{1}{1 + \|\vec{x} - \vec{\omega}\|^2} \tag{1}$$

Plus cette distance est faible plus la réponse du neurone est grande et réciproquement. La réponse maximale est égale à 1 lorsque le vecteur d'entrée est égal au vecteur poids. La sortie du neurone donne donc une idée de la ressemblance entre le vecteur forme et le vecteur de poids.

1.2 Le neurone gagnant

En général, les réseaux de neurones à apprentissage par compétition sont appliqués aux problèmes de discrimination i.e. décider si un vecteur d'entrée appartient à une classe C_k parmi un ensemble de classes $\{C_{i,i=1,\cdots,m}\}$. Ce type de réseau posséde une architecture monocouche dont le nombre de neurones est égal au nombre de classes (voir figure 1). Lorsque l'on présente un vecteur forme au réseau, ce dernier décide que $\vec{x} \in C_k$ si le $k^{\text{iéme}}$ neurone a la plus grande réponse, ce neurone est appelé <u>neurone gagnant</u> (*best matching unit*). La règle

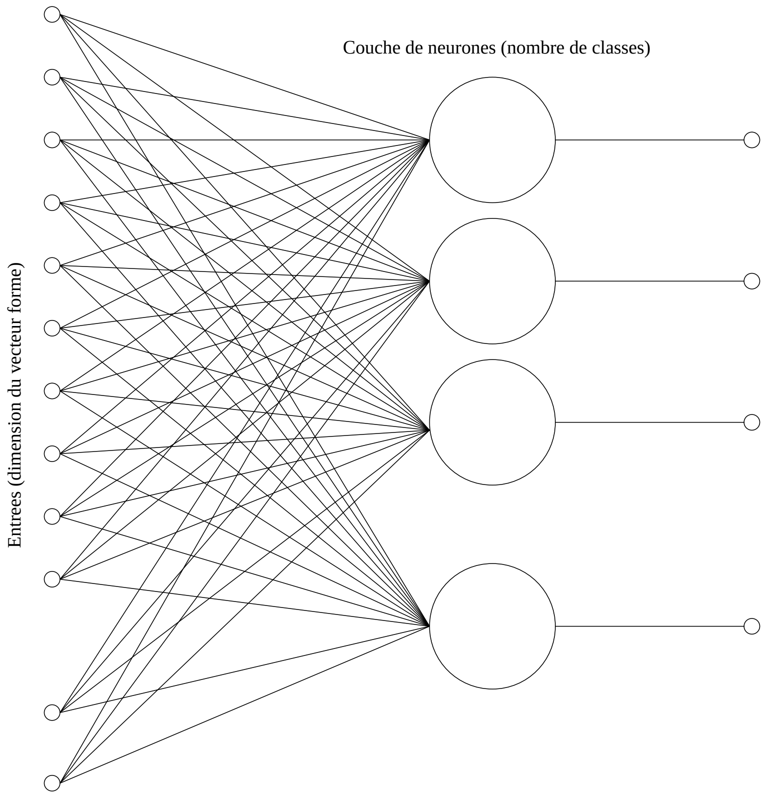
[1] Notons que d'autres distances sont utilisables mais sans connaissance *a priori* la distance euclidienne reste la plus impartiale.

de décision s'exprime donc sous la forme

$$d(\vec{x}) = k \text{ si } k = \text{argmax}_i \frac{1}{1 + \|\vec{x} - \vec{\omega}_i\|^2}$$

Les vecteurs de poids apparaissent donc comme des vecteurs prototypes qui décrivent les classes présentes dans les données. Le problème revient donc à construire une règle d'apprentissage qui permette de trouver ces prototypes à partir d'un ensemble de vecteurs d'exemple i.e. à partir d'une connaissance partielle du processus à analyser.

FIG. **1** – Architecture classique d'un réseau de neurones à apprentissage par compétition.



1.3 Critère d'apprentissage

A partir d'une entrée $\vec{x}$ proposée au réseau, on définit l'erreur de quantification $\varepsilon(\vec{x})$ de la manière suivante

$$\varepsilon(\vec{x}) = \frac{1}{2}\|\vec{x} - \vec{\omega}_k\|^2$$

où k désigne l'indice du neurone gagnant.
Le critère que l'on cherche à optimiser durant la phase d'apprentissage est l'espérance mathématique de l'erreur de quantification, soit

$$E[\varepsilon(\vec{x})] = \frac{1}{2}\int_{R^n} \|\vec{x} - \vec{\omega}_k\|^2 f(\vec{x}) d\vec{x}$$

où $f(x)$ désigne la loi de probabilité multidimensionnelle du vecteur forme $\vec{X}$ et est généralement inconnue.
Minimiser ce critère est une tâche relativement compliquée mais la théorie des processus stochastiques permet de définir une règle d'apprentissage simple à partir d'un ensemble de vecteurs d'exemple (ensemble d'apprentissage).

1.4 Apprentissage en mode non supervisé

En mode non-supervisé ou sans professeur les vecteurs d'exemple ne sont pas étiquetés i.e. leur classe d'appartenance est inconnue. La règle d'apprentissage s'exprime alors de la manière suivante

$$\left\{ \begin{array}{lcl} \vec{\omega}_k^{(t+1)} & = & \vec{\omega}_k^{(t)} + \rho(t)(\vec{x} - \vec{\omega}_k^{(t)}) \\ \vec{\omega}_i^{(t+1)} & = & \vec{\omega}_i^{(t)},\ i \neq k \end{array} \right.$$

où $\vec{x}$ est le vecteur d'exemple présenté à l'instant t et k l'indice du neurone gagnant ; le terme $\rho(t)$ (taux d'apprentissage) est de la forme $\rho_0 t^{-\alpha}$ avec $\alpha \in]0, 1[$; l'algorithme ne converge que si

$$\lim_{t\to\infty} \rho(t) = 0$$

Les poids sont initialisés aléatoirement et la règle est appliquée sur l'ensemble d'apprentissage jusqu'à convergence[2]. D'un point de vue géométrique, on peut voir la règle comme consistant à rapprocher les poids du neurone gagnant des composantes du vecteur d'exemple tout en laissant les autres neurones inchangées (règle du type *winner takes all*). Au fur et à mesure de l'apprentissage le terme $\rho(t)$ augmente l'inertie du réseau et le rend de moins en moins apte à faire des changements radicaux dans ces poids.

[2]. La convergence est par ailleurs assez difficile à estimer, en général on suppose qu'un grand nombre de présentations de l'ensemble d'apprentissage conduit à une convergence *presque sûre*.

1.5 Apprentissage en mode supervisé

Le mode supervisé ou avec professeur tire partie de l'étiquetage des vecteurs d'exemple, la règle qui en découle est assez similaire au mode non supervisé

$$\begin{cases} \vec{\omega}_k^{(t+1)} & = & \vec{\omega}_k^{(t)} + \rho(t)(\vec{x} - \vec{\omega}_k^{(t)}) & \text{si } \vec{x} \in C_k \\ \vec{\omega}_k^{(t+1)} & = & \vec{\omega}_k^{(t)} - \rho(t)(\vec{x} - \vec{\omega}_k^{(t)}) & \text{si } \vec{x} \notin C_k \end{cases}$$

Cette règle peut être interprétée géométriquement de la même manière que la précédente si ce n'est que l'on introduit une « punition » du neurone gagnant si le réseau s'est trompé i.e en cas d'erreur le neurone gagnant est éloigné du vecteur d'exemple.

1.6 Exemple d'application

Hormis leur utilisation potentielle dans le domaine du traitement des langages naturels, les réseaux de neurones à apprentissage par compétition ont un champs d'applications relativement large (discrimination, compression et analyse de données, ...). A titre d'exemple, la figure 2 présente le résultat d'une méthode de requantification d'images basée sur l'apprentissage par compétition non supervisé[3].

2 Les cartes de Kohonen

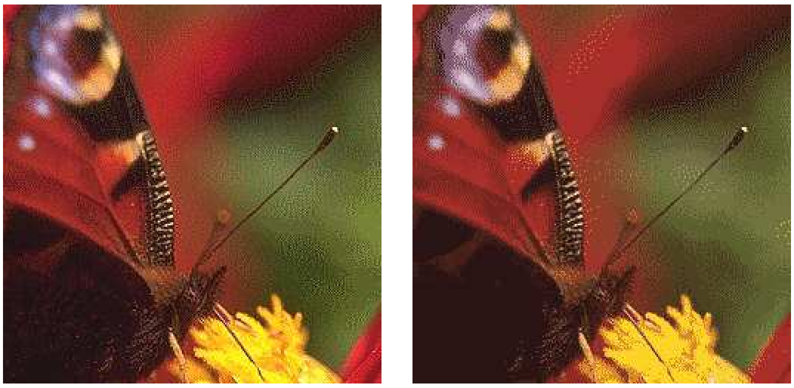
2.1 Présentation

Les cartes de Kohonen ou cartes topologiques auto-organisatrices (*self organizing feature maps*, SOM) sont une extension de l'approche présentée dans la section précédente. L'idée consiste aussi à utiliser un ensemble de neurones formels pour mémoriser (et trouver durant la phase d'apprentissage) des vecteurs prototypes mais on impose cette fois des relations de voisinage[4]. Ces relations de voisinage impliquent que les vecteurs prototypes ne sont pas positionnés aléatoirement sur la carte mais que leur agencement respecte la topologie de

3. Cette méthode a été développée par moi-même au cours de l'été 1997. Des résultats comparatifs (notament avec `Photoshop 4.0`) sont disponibles à l'adresse *Web* `http://wwwetu.utc.fr/ sirdeyre/neural/neural.html`. Un article et un logiciel de conversion sont disponibles sur simple demande par `e-mail`. Par ailleurs, l'algorithme a été utilisé dans le cadre du développement d'un `CD-ROM` pour la ville de Noyon (`http://www.noyon.com`).

4. Ce modèle est directement inspiré des systèmes biologiques où les cellules nerveuses ne sont pas des entités indépendantes mais forment des ensembles fortement connectés.

FIG. **2** – (a) Exemple d'image *true-color* quantifiée sur 3 composantes (rouge, vert, bleu) avec 8 bits par composantes. L'image contient 29196 couleurs. (b) image approximée à l'aide de seulement 16 couleurs (vecteurs prototypes fournis par la régle d'apprentissage par compétition non supervisé) et d'un algorithme de diffusion d'erreur de type Floyd. © R. Sirdey 1997.



(a) (b)

l'espace des vecteurs formes (on parle alors de projection préservant la topologie ou de *topology-preserving mapping* [Koh84]).
Les cartes les plus couramment utilisées sont de dimension 1 ou 2. En dimension 2 le réseau ou carte se présente sous la forme d'une matrice de neurones (la figure 3 présente l'architecture classique d'une carte 2D). Chaque neurone reçoit les éléments du vecteur forme en entrée et renvoit une sortie inversement proportionnelle à la distance entre ces derniers et les poids, voir équation (1). Comme précédemment le neurone gagnant est celui qui ressemble le plus au vecteur d'entrée au sens de la distance utilisée.

2.2 Règle d'apprentissage

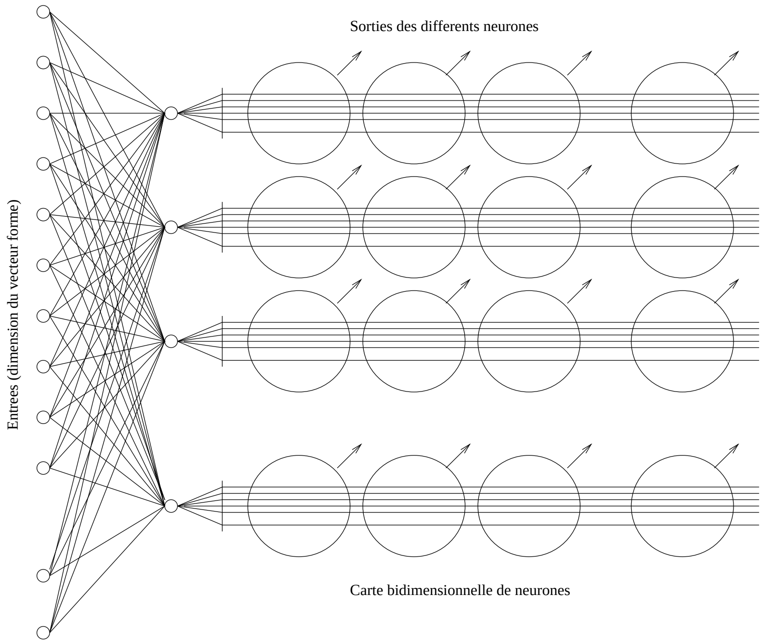
La règle d'apprentissage utilisée dans le modèle SOM possède des similarités avec celle de l'approche « réseau » mais elle doit aussi inclure le processus d'auto-organisation i.e. introduire la relation de voisinage. Elle s'exprime donc de la manière suivante[5]

$$\vec{\omega}_i^{(t+1)} = \vec{\omega}_i^{(t)} + \rho(t)\alpha_{i,k}(t)(\vec{x} - \vec{\omega}_i^{(t)}) \tag{2}$$

où k désigne l'indice du neurone gagnant.
Le terme $\rho(t)$ a la même fonction et les mêmes propriétés que dans le cas

[5] Comme dans le cas précédent les poids de chaque neurone sont initialisés de manière aléatoire.

FIG. **3** – Architecture typique d'une carte de Kohonen de dimension 2.



classique. Le terme $\alpha_{i,k}(t)$ définit par contre une fonction de voisinage centrée sur le neurone gagnant. L'idée consiste à faire changer les poids des neurones dans le voisinage proche du neurone gagnant et ce avec une amplitude d'autant plus faible que le neurone considéré est éloigné du neurone gagnant.
A partir d'une entrée $\vec{x}$ présentée à la carte, on définit la mesure de distorsion [Tra95] comme étant la somme des distances entre $\vec{x}$ et les vecteurs prototypes pondérés par la fonction de voisinage, soit

$$\varepsilon(\vec{x}) = \frac{1}{2} \sum_{i=1,\cdots,N} \alpha_{i,k} \|\vec{x} - \vec{\omega}_i\|^2$$

L'algorithme SOM peut alors être vu comme minimisant approximativement l'espérance mathématique de la mesure de distorsion

$$E[\varepsilon(\vec{x})] = \frac{1}{2} \int_{R^n} \sum_{i=1,\cdots,N} \alpha_{i,k} \|\vec{x} - \vec{\omega}_i\|^2 f(\vec{x}) d\vec{x}$$

Une fonction de voisinage couramment utilisée dans le cas à une dimension est une gaussienne centrée sur le neurone gagnant (dans ce cas on parle de règle du type *winner takes most*)

$$\alpha_{i,k}(t) = e^{-\left(\frac{i-k}{\sigma(t)}\right)^2} \tag{3}$$

où $\sigma(t)$ est une fonction décroissante du temps de la forme $\sigma_0 t^{-\beta}$ avec $\beta \in]0,1[$. On a donc

$$\lim_{t\to\infty} \sigma(t) = 0$$

En pratique et pour des raisons de temps de calcul on approxime la fonction de Gauss par une fonction de type *tophat* telle que

$$\alpha_{i,k}(t) = \begin{cases} 1 & \text{si } i-k \in [-\sigma(t), \sigma(t)] \\ 0 & \text{sinon} \end{cases}$$

La règle d'apprentissage devient donc

$$\begin{cases} \vec{\omega}_i^{(t+1)} &= \vec{\omega}_i^{(t)} + \rho(t)(\vec{x} - \vec{\omega}_i^{(t)}),\ i \in \Omega_k \\ \vec{\omega}_i^{(t+1)} &= \vec{\omega}_i^{(t)},\ i \notin \Omega_k \end{cases}$$

où Ω_k désigne la région d'activation autour du neurone gagnant.
Ces fonctions de voisinage ont leurs analogues bidimensionnels (et multidimensionnels si besoin). Dans le cas de la fonction *tophat* on peut cependant choisir une région d'activation de forme carrée, hexagonale ou circulaire [Koh84].
Une étude plus détaillée et surtout plus formelle du modèle peut être trouvée dans le livre de T. Kohonen « Self organization and associative memory » [Koh84] ainsi que dans [Tra95].

2.3 Exemples et applications

A titre d'exemple[6] les figures 4 et 5 illustrent les apprentissages respectifs d'une SOM monodimensionnelle et d'une SOM bidimensionnelle sur un ensemble de vecteurs d'exemple issus d'une distribution uniforme bidimensionnelle à valeurs dans $[0, 180[^2$. L'ensemble d'apprentissage est composé de 1000 vecteurs. Les paramètres du modèle[7] sont $\rho(t) = t^{-\frac{1}{2}}$ et $\sigma(t) = t^{-\frac{1}{8}}$ (voir équations (2) et (3)). Dans ce cas, la dimension de l'espace des vecteurs formes est suffisamment faible pour que les poids soient représentables et l'on observe clairement que la carte tend vers une projection maillée et organisée de la loi de probabilité des vecteurs d'exemple. Toutefois, l'algorithme peut ne pas converger convenablement et peut donner des configurations à torsions (voir

[6] Cette expérience a déjà été réalisée maintes fois, notament par Kohonen [Koh84]. Les résultats présentés sur les figures 4 et 5 ont été obtenus à l'aide des routines dévellopées dans le cadre de ce travail et ont donc permis leur validation.

[7] Ces paramètres ont été choisis par tatonnement en fonction des résultats donnés. Il n'existe actuellement pas de méthode automatique d'optimisation de ces paramètres et les critères de qualité des cartes sont souvent mal définis (une étude détaillée est disponible dans [Tra95]).

FIG. 4 – Convergence d'une SOM monodimensionnelle (256 neurones) sur une densité de probabilité uniforme bidimensionnelle. (a) carte initiale, (b) 10, (c) 100, (d) 1000, (e) 10000 itérations. Voir texte.

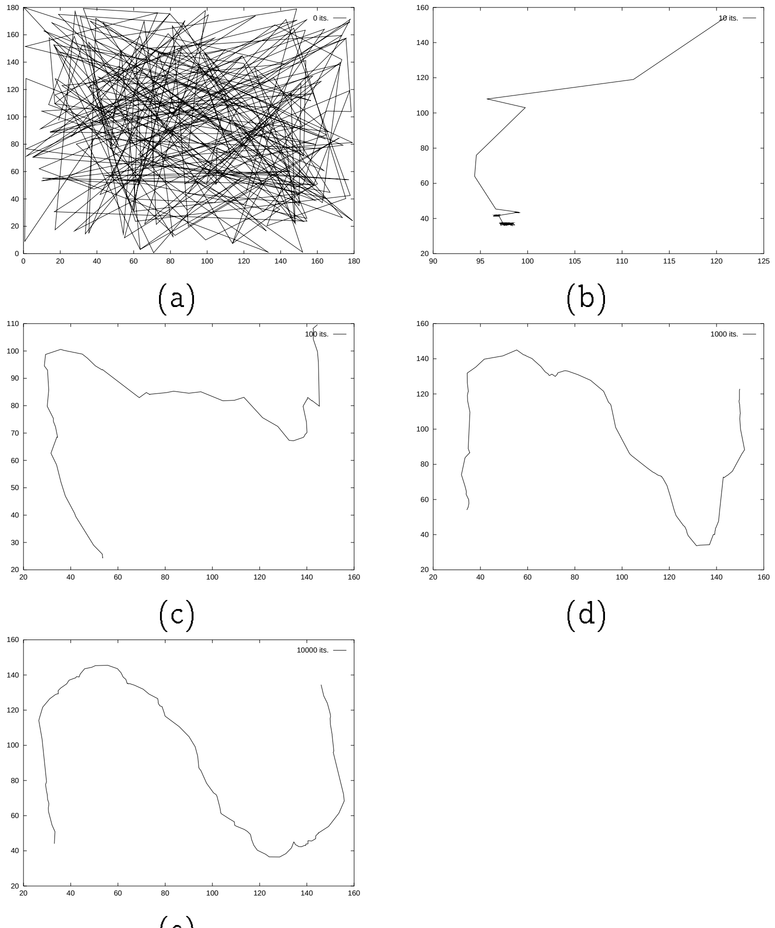
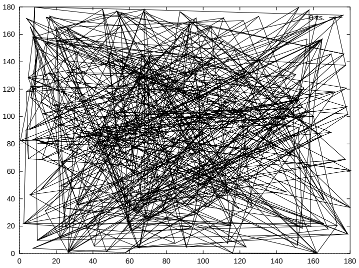
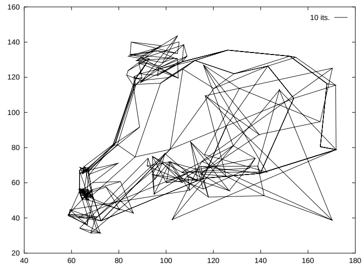


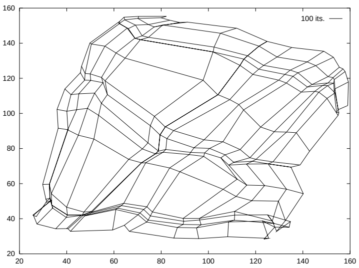
FIG. 5 – Convergence d'une SOM bidimensionnelle 16 sur une densité de probabilité uniforme bidimensionnelle. (a) carte initiale, (b) 10, (c) 100, (d) 1000, (e) 10000 itérations. Cependant, l'algorithme SOM peut ne pas converger correctement et il arrive que la carte ait des torsions irrécupérables (f). Voir texte.



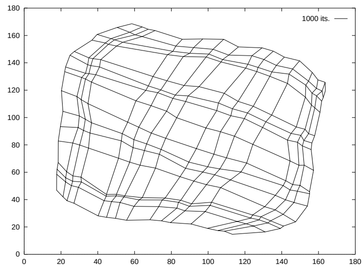
(a)



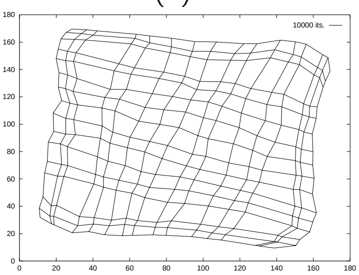
(b)



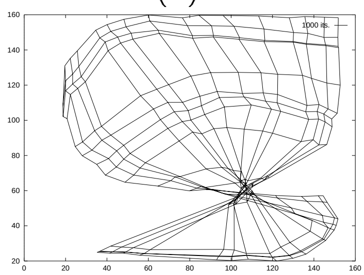
(c)



(d)



(e)



(f)

figure 5 (f)).
Cette propriété qu'ont les SOM de générer des projections de lois de probabilité en fait un outil de choix pour la représentation de données multidimensionnelles[8] ou analyse exploratoire des données (c'est d'ailleurs à cette fin qu'elles seront utilisées ici). On note aussi des applications en discrimination, en classification et en optimisation [Tra95].

2.4 Représentation des cartes

Lorsque les vecteurs d'entrées sont de dimension 2 ou 3, la carte peut directement être représentée dans l'espace des vecteurs formes (voir figures 4 et 5). Lorsque que la dimension devient supérieure il est nécessaire de faire appel à d'autres méthodes de visualisation; un inventaire relativement exhaustif est disponible dans [Tra95]. Ici, nous retiendrons la méthode dite du *local distance display* qui (dans le cas d'une SOM bidimensionnelle) consiste à produire une image dont les pixels ont une intensité inversement proportionnelle à la distance moyenne entre le vecteur prototype considéré et ses voisins immédiats sur la carte. On obtient donc une image dont les zones claires ou « montagnes » indiquent des amas de vecteurs prototypes dans l'espace des vecteurs formes (classes) alors que les zones sombres ou « vallées » renseignent sur les séparations entre les classes.

Références

[Koh84] Teuvo Kohonen. Self Organization and Associative Memory. Springer-Verlag, 1984.

[Tra95] Thomas Trautman. Développement d'un modèle de cartes topologiques auto-organisatrice à architecture dynamique, application au diagnostique. PhD thesis, Université de Technologie de Compiègne, 1995.

8. Comme l'analyse en composantes principales par exemple.