N° d'ordre : 1760

THESE

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE

Pour obtenir le titre de

DOCTEUR

en Automatique, Productique et Informatique Industrielle

par

Stéphane DELSERT

CLASSIFICATION INTERACTIVE NON SUPERVISEE DE DONNEES MULTIDIMENSIONNELLES PAR RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF

Soutenue le 1er Juillet 1996 devant la commission d'examen :

MM.

P. VIDAL	Président	Professeur à l'U.S.T.L.
J.-G. POSTAIRE	Directeur de recherche	Professeur à l'U.S.T.L.
D. HAMAD	Co-Directeur de thèse	Maître de Conférence à l'U.S.T.L.
S. THIRIA	Rapporteur	Professeur à l'Université de Versailles
H. EMPTOZ	Rapporteur	Professeur à l'INSA de Lyon
C. VASSEUR	Examinateur	Professeur, Directeur de l'ENSAIT de Roubaix
T. DENOEUX	Examinateur	Maître de Conférences à l'U.T.C., Compiègne
C. LOUIS	Examinateur	Ingénieur d'études, Alcatel.

A la mémoire de Philippe et de Daniel.

AVANT PROPOS

Le travail présenté dans ce mémoire a été effectué au Centre d'Automatique de l'Université des Sciences et Technologies de Lille, dirigé par Monsieur le Professeur Pierre Vidal. Je le remercie de l'accueil qu'il m'a réservé au sein de son laboratoire et de l'honneur qu'il me fait en acceptant la présidence du jury de thèse.

J'adresse mes remerciements à Monsieur Jack-Gérard Postaire, Professeur à l'Université des Sciences et Technologies de Lille pour son excellent encadrement tout au long de mes travaux. Je tiens à lui exprimer toute ma gratitude pour son dynamisme, ses conseils permanents et sa confiance qu'il m'a accordé tout au long de ce travail.

C'est avec grand plaisir que j'adresse mes remerciements les plus sincères à Monsieur Denis Hamad, Maître de Conférences à l'Institut Agricole et Alimentaire de Lille, pour ses remarques constructives au cours de ce travail et pour avoir accepté de faire partie du jury.

J'exprime également mes sincères remerciements à Madame Sylvie Thiria, Professeur à l'Université de Versailles, pour l'intérêt qu'elle a porté à ce travail en acceptant d'être rapporteur de cette thèse.

Que Monsieur Hubert Emptoz, Professeur à l'INSA de Lyon, trouve ici toute ma reconnaissance pour avoir accepté de juger mon travail.

Je remercie également Monsieur Christian Vasseur, Professeur et Directeur de l'ENSAIT de Roubaix pour avoir accepté de participer au jury et pour m'avoir accueilli dans son école.

Je tiens aussi à exprimer mes sincères remerciements à Monsieur Thierry Denoeux, Maître de Conférences Habilité à l'UTC, et à Monsieur Christian Louis, Ingénieur d'études à Alcatel, pour l'intérêt qu'ils ont porté à ce travail en acceptant d'être examinateur.

Par ailleurs j'adresse mes remerciements les plus sincères à tout ceux qui, de prés ou de loin, m'ont aidé par leur compétence et leur amitié dans l'élaboration de ce travail, et notamment les membres de l'équipe Image et Décision du Centre Automatique de Lille.

Je ne saurais terminer cet avant propos sans remercier ma femme, Laurence, et ma fille Manon, pour leurs encouragements et leurs soutiens.

NOTATIONS UTILISEES

N	: Dimension de l'espace de représentation des observations.
Q	: Nombre d'observations de l'échantillon.
$\mathcal{X}$	: Echantillon des observations.
X_q	: $q^{ième}$ observation de l'échantillon.
x_{qn}	: $n^{ième}$ coordonnée de l'observation X_q.
M	: Nombre de neurones du réseau.
$\mathcal{W}$	: Echantillon des vecteurs poids du réseau.
W_m	: $m^{ième}$ vecteur poids synaptiques de l'échantillon des vecteurs poids.
w_{mn}	: $n^{ième}$ coordonnée du vecteur poids W_m.
K	: Nombre de classes de l'échantillon.
$\mathcal{C}$	: Ensemble des classes.
C_k	: $k^{ième}$ classe.
C^E_{m*}	: Ensemble des observations de l'échantillon $\mathcal{X}$ activant le neurone m*.
$\mathbb{R}^N$	: Ensemble des réels de dimension N.
$\mathbb{N}^N$	: Ensemble des entiers naturels de dimension N.
E	: Espace de représentation des observations.
A	: Espace de projection.
$d^E()$	: Distance de l'espace d'observation.
$d^A()$	: Distance de l'espace de projection.
t	: Variable d'itérations.
T	: Nombre total d'itérations d'un apprentissage.

m^* : indice du neurone gagnant.

m^{**} : indice du second neurone gagnant.

a_m : Etat du neurone m.

y_m : Sortie du neurone m.

v_m : Entrée totale du neurone m.

$\alpha(t)$: Coefficient d'apprentissage.

$\beta(t)$: Coefficient d'apprentissage.

$\mu_m(t)$: Coefficient de conscience du neurone m.

$h()$: Loi d'interaction.

$r()$: Rayon d'interaction.

$V()$: Ensemble des neurones voisins.

$d_m()$: Distance relative au vecteur poids W_m.

U_m : Vecteur position du neurone m sur la carte $U_m=(i,j)^T$.

I : Nombre de neurones en abscisse d'une carte.

J : Nombre de neurones en ordonnée d'une carte.

z_{MAX} : Maximum des distances inter-neurones.

z_{MED} : Médian des distances inter-neurones.

z_{MOY} : Moyenne des distances inter-neurones.

$P(X/C_k)$: Probabilité sous-jacente des observations provenant de la classe C_k.

$P(C_k)$: Probabilité a priori de la classe C_k.

$p(X)$: fonction densité de probabilité.

$\hat{p}(X)$: Estimation de la fonction densité de probabilité.

^{T} : Opérateur transposé.

$\| \|$: Norme Euclidienne.

$| |$: Valeur absolue.

Arg : opérateur argument.

min : opérateur minimum.

max : opérateur maximum.

mod : Opérateur modulo.

div : Division entière.

$\in$: Appartenance d'un élément à un ensemble.

$\cup$: Union de deux ensembles.

$\cap$: Intersection de deux ensembles.

$\varnothing$: Ensemble vide.

$\exists$: Quantificateur existenciel.

$\forall$: Quantificateur universel.

SOMMAIRE

INTRODUCTION GENERALE

Le travail que nous présentons porte sur la classification non supervisée à l'aide des réseaux de neurones à apprentissage compétitif. Les techniques de classification sont utilisées dans des domaines aussi divers que la médecine, la biologie, l'économie, la sociologie, etc... La classification apporte souvent, dans ces domaines, des réponses très utiles. En dégageant d'une masse importante de données des similitudes, elle permet la compréhension de systèmes complexes difficilement modélisables à l'aide d'équations mathématiques.

Les années 80 ont connu une évolution remarquable dans les techniques neuronales : de nouveaux réseaux et algorithmes d'apprentissage sont apparus. On peut distinguer deux différents secteurs dans ces évolutions. Le premier découle directement de l'étude du fonctionnement des cellules du cerveau. Le second est lié à l'introduction de techniques de minimisation dans les algorithmes d'apprentissage. Les réseaux de neurones à apprentissage compétitif ont bénéficié d'avancées significatives dans ces deux secteurs.

Nous présentons tout d'abord, dans le premier chapitre, différentes techniques de classification, et abordons de manière succincte les réseaux de neurones. Enfin nous citons les différentes applications des réseaux de neurones en classification et en réduction de dimension.

Le second chapitre est consacré aux différentes techniques d'apprentissage des réseaux de neurones à apprentissage compétitif. Nous présentons dans un premier temps les réseaux de neurones à apprentissage compétitif. Puis nous établissons les nombreux liens entre l'apprentissage de ces réseaux et l'algorithme des K-means, bien connu en classification. Nous présentons ensuite des techniques d'apprentissage dérivées de celle de

l'apprentissage compétitif ou interviennent des phénomènes de compétition, de conscience ou encore de rivalité entre les neurones offrant ainsi des améliorations par rapport à l'algorithme des K-means. Nous montrons enfin que certaines de ces techniques d'apprentissage permettent d'utiliser ces réseaux de neurones en classification, mais aussi en réduction de données.

Le troisième chapitre présente un réseau dérivé des réseaux à apprentissage compétitif: les cartes de Kohonen. Ces cartes ont une structure similaire aux réseaux à apprentissage compétitif, mais la technique d'apprentissage modélise l'interaction entre les neurones existant dans certaines zones du cerveau. Nous montrons comment nous pouvons, à l'aide de ce type de réseau, obtenir des représentations en deux dimensions d'un échantillon d'observations multidimensionnelles.

Le quatrième chapitre se propose d'intégrer les différents réseaux dans un cadre de classification interactive. Nous exposons d'abord la méthodologie d'un processus de classification interactif. Puis nous exploitons les différentes techniques neuronales détaillées dans les précédents chapitres pour développer des outils logiciels destinés à aider l'opérateur dans son analyse.

Le dernier chapitre présente des résultats expérimentaux obtenus en utilisant des échantillons réels et générés artificiellement. Un exemple réel est tiré d'une étude biométrique d'abeilles provenant des îles de la Guadeloupe. Deux autres exemples, générés artificiellement, visent à montrer l'intérêt de l'interface interactive s'appuyant sur les techniques neuronales pour traiter des échantillons complexes. Nous comparons les représentations par cartes de Kohonen avec d'autres techniques de projections plus classiques, à savoir l'analyse en composantes principales, l'algorithme de projection de Sammon et la projection par réseaux multicouches en mode auto-associatif.

Présentons tout d'abord la classification et les réseaux de neurones : deux domaines de recherche apparemment bien différents.

CHAPITRE I :

DE LA CLASSIFICATION AUTOMATIQUE AUX RESEAUX DE NEURONES ARTIFICIELS

I.1 PRESENTATION

Dans diverses disciplines telles que la biologie, la psychologie, la médecine, l'économie ou le marketing, l'analyste est souvent confronté à une masse importante de données multidimensionnelles qu'il doit analyser afin d'en extraire des « informations utiles » facilement exploitables. La démarche qu'il adopte pour y parvenir s'intègre dans le cadre de l'analyse de données et plus précisément dans celui de la classification.

I.1.1 D'un ensemble d'individus à un échantillon d'observations

La première étape dans la conception d'un classifieur concerne le recueil des données et leurs prétraitements. Il s'agit de la caractérisation des individus sous forme de données numériques facilement exploitables par un calculateur. Pour bien cerner le problème, considérons un exemple bien connu dans la littérature statistique : les iris de Fisher. Il s'agit d'analyser un ensemble d'individus composés d'iris de différentes variétés : Sétosa, Versicolor et Virginica [FISH 36]. L'individu correspond ici à une iris. La première étape consiste à décrire ces individus par un certain nombre de caractéristiques, appelées attributs. Cette étape est l'une des plus délicates car elle conditionne la qualité de l'analyse. Les attributs relevés

par Fisher sont la longueur et la largeur des sépales et la longueur et la largeur des pétales. Pour chaque iris on obtient un vecteur de dimension 4 dont les composantes sont constituées des valeurs de ces attributs. Un tel vecteur, également appelé observation multidimensionnelle, est associé à chaque individu. Fisher a utilisé un échantillon de 150 observations émanant d'un prélèvement des caractéristiques de 150 individus.

D'une manière générale, on dispose d'un ensemble de données constitué d'un tableau à 2 entrées *Individus×Attributs* de Q×N éléments (cf. Tableau I.1). La qième ligne du tableau représente l'ensemble des N valeurs des attributs relevés sur le qième individu qui sera considéré comme un élément de $\mathbb{R}^N$. La nième colonne représente les valeurs prises par le nième attribut a_n sur les différents individus. Nous supposons que les attributs prennent des valeurs numériques réelles.

Individus	Attributs						
	a_1	....	a_n	.....	.....		a_N
I_1	x_{11}	...	x_{1n}	...	...		x_{1N}
...	...		...				...
...	...		...				...
I_q	x_{q1}	...	x_{qn}	...	...		x_{qN}
...	...		...				...
...	...		...				...
I_Q	x_{Q1}	...	x_{Qn}	...	...		x_{QN}

Tableau I. 1:Tableau de données Individus×Attributs

Chaque observation peut être aussi représentée par un point dans l'espace $\mathbb{R}^N$, dont la position est déterminée par un vecteur noté $X_q=[x_{q1},x_{q2},...,x_{qn},...,x_{qN}]^T$. N est la dimension de l'espace $\mathbb{R}^N$, appelé aussi espace de représentation des observations. x_{qn} est la valeur du nième attribut relevé sur le $q^{ième}$ individu.

I.1.2 Définition d'une classe

Une classe est définie selon Everitt [EVER 74] par l'une ou l'autre des définitions suivantes :

- Une classe regroupe un ensemble d'observations semblables. Toutes observations d'une classe n'étant pas semblables avec les observations provenant d'une autre classe.

- Les classes peuvent être définies par des régions connexes de l'espace de représentation des observations denses en observations séparées les unes des autres par des régions de densité relativement faible.

Une définition plus précise et reposant sur l'analyse de la fonction de densité de probabilité sous-jacente à la distribution des observations (abv. fdp) consiste à associer la présence des classes à l'existence de maxima locaux de la fdp appelés modes.

I.1.3 Objectifs de la Classification

La classification se propose de résoudre trois objectifs:

- Le premier consiste à déterminer, lorsque l'on dispose d'un ensemble $\mathcal{X}$ constitué de Q observations $\mathcal{X}=\{X_1,...,X_q,...,X_Q\}$, le nombre de classes K entre lesquelles se répartissent les observations, K étant inconnu.

- Le second, appelé classification de l'échantillon, consiste à décomposer l'échantillon $\mathcal{X}$ en K sous-ensembles ou classes $\mathcal{C}=\{C_0, ..., C_k, ..., C_{K-1}\}$; $C_k \in \mathcal{P}(\mathcal{X})$, où $\mathcal{P}(\mathcal{X})$ est l'ensemble des parties de $\mathcal{X}$.

- Le troisième consiste à décomposer l'espace de représentation des observations en K sous-espaces, appelés domaines. Un domaine représente un sous-espace de l'espace des observations de telle sorte que chaque observation appartenant à ce sous-espace est assignée à une même classe. Lorsque ces domaines sont connus, la classe d'appartenance d'une observation est automatiquement déterminée en identifiant le domaine auquel cette observation appartient. Notons D_k le domaine associé à la classe k et, $\mathcal{D}=\{D_0, ..., D_k, ..., D_{K-1}\}$, l'ensemble des domaines. Cet objectif vise à élaborer un classifieur.

Souvent l'on utilise les techniques de classification en connaissant a priori le nombre de classes K en présence et en disposant d'un ensemble d'observations prototypes dont on

connaît l'appartenance à ces classes. Dans ce cas, l'analyste doit seulement concevoir un classifieur capable d'assigner correctement des observations inconnues. La classification s'effectue dans ce cas en **mode supervisé**. Par contre, si l'on ne dispose d'aucune de ces informations, la classification s'effectue en **mode non supervisé**. L'objectif est alors de déceler le nombre K de classes et de classer les observations disponibles en les assignant aux différentes classes mises en évidence. Notons que la détermination des classes et la classification sont étroitement liées.

I.1.4 Méthodologies

Les différentes méthodes de classification proposées dans la littérature diffèrent par le degré d'intervention de l'opérateur humain. Ainsi, on distingue des méthodes de **classification automatique,** dites « presse-bouton », où l'opérateur n'intervient pas dans le processus de classification, les méthodes dites **semi-automatiques** dans lesquelles l'opérateur fixe quelques paramètres et enfin la **classification interactive** où l'opérateur contrôle les différents maillons du processus de classification.

I.2 CLASSIFICATION AUTOMATIQUE NON SUPERVISEE

Nous présentons, dans ce paragraphe, les principales techniques permettant de déduire le nombre K de classes en présence dans l'échantillon d'observations ainsi que différentes techniques de classification en mode non supervisé.

I.2.1 Méthodes locales

Les procédures locales reposent sur l'étude de la fdp sous-jacente à la distribution des observations constituant l'échantillon soumis à l'analyse. L'estimation de cette fdp, supposée inconnue, fait alors appel à des **estimateurs non paramétriques** comme l'estimateur du noyau de Parzen ou l'estimateur des k plus proches voisins [DUDA 73], [PARZ 62], [LOFT 65].

La détermination du nombre de classes K s'apparente dans ce cas à la recherche des modes de la fdp estimée. On distingue 4 méthodes permettant de déceler les modes de la fdp.

I.2.1.1 Recherche des maxima locaux

Dans ce type d'approche, il s'agit d'analyser la fonction de densité sous-jacente à la distribution des observations disponibles pour extraire l'information nécessaire à leur classification. Rappelons que l'on associe la présence d'une classe à la présence d'un mode de la fdp.

Les modes peuvent être détectés en remontant les pentes de la fdp selon la direction de son gradient [KOON 76] ou en déplaçant progressivement les observations jusqu'à ce que chacune d'elles se stabilise en des endroits indiquant les modes de cette fonction [BOCK 79]. Une variante de cette approche consiste à calculer directement le gradient local à partir des observations [FUKU 75].

I.2.1.2 Analyse de la convexité

Au lieu de considérer les modes comme des maxima locaux de la fdp, Vasseur et Postaire [VASS 80] les assimilent à des régions de l'espace où cette fonction est concave. Dans ce cas, l'analyse de la convexité de la fdp est effectuée en intégrant cette dernière sur des domaines d'observation de tailles variables [POST 82a]. Cette méthode est plus robuste que les techniques faisant appel aux notions de gradient.

I.2.1.3 Extraction des contours des modes

Touzani et al. considèrent les modes comme des régions délimitées par leurs contours. Après réalisation d'un filtrage médian de la fdp, des opérateurs différentiels multidimensionnels permettent d'extraire les contours des modes [TOUZ 89].

I.2.1.4 Morphologie mathématique

Postaire et al.[POST 93] proposent une méthode qui permet la détection des modes grâce à la morphologie mathématique binaire. Sbihi a affiné cette méthode en utilisant des opérateurs morphologiques en niveaux de gris [SBIH 95].

Les résultats obtenus par ces méthodes dépendent directement de certains paramètres tels que la taille du noyau ou le nombre des k plus proches voisins pour les estimateurs non paramétriques de la fdp ou encore le pas de discrétisation pour la détection des modes par la morphologie mathématique.

I.2.2 Méthodes Globales

Ce type de méthodes peut faire appel soit à des techniques métriques soit à des techniques statistiques.

I.2.2.1 Techniques métriques

On distingue deux principales approches de type métrique permettant de classer les observations. La première consiste à effectuer un partitionnement de l'espace de représentations des observations tandis que la seconde établit une hiérarchie de classes. Ces méthodes supposent la définition d'un critère mesurant la proximité des individus d'une même classe et donc de la qualité d'une partition.

I.2.2.1.1 Techniques de partitionnement

De nombreux critères de proximité ont été proposés [FRIE 67],[JONE 68],[FUKU 70]. Parmi ceux-ci, on distingue ceux conduisant à un partitionnement de l'espace [ANDE 73] et les procédures itératives basées sur la notion de noyau telles que la méthode des nuées dynamiques [DIDA 71]. Cependant le résultat obtenu dépend de l'initialisation des procédures. Des techniques basées sur le concept de formes fortes ont été élaborées pour les différentes approches afin de pallier ce problème [ANDE 73],[DIDA 79].

I.2.2.1.2 Classification hierarchique

L'ensemble des classes peut être représenté par une structure hiérarchique ascendante ou descendante [JAMB 78]. La classification hiérarchique ascendante consiste à considérer initialement chaque observation comme une classe, puis à regrouper à chaque étape, les observations deux à deux suivant un critère de similarité. La démarche descendante consiste à grouper toutes les observations dans une seule classe, puis à diviser itérativement les classes suivant un critère de dissimilarité. Le nombre de classes est alors déterminé par un seuil prédéfini [BAYN 80], [LANE 67], [LUKA 79]. Une autre méthode consiste à réduire la hiérarchie des parties, ou l'arbre des classifications, sous forme d'un ensemble de noeuds significatifs, avant d'effectuer la partition de l'ensemble des objets à classer [GORD 87][LERM 91].

I.2.2.2 Techniques statistiques

Nous nous plaçons dans l'hypothèse où les modèles de distribution des observations

sont supposés connus a priori, mais leurs paramètres sont inconnus. Dans ce cas, le problème de l'analyse de données peut être ramené à celui de la détermination des paramètres d'un mélange de fonctions de densité représentant les distributions des observations provenant de chacune des classes en présence dans l'échantillon analysé.

L'estimation des paramètres d'un mélange de fdp peut être obtenue par des techniques d'apprentissage Bayésien ou par des procédures d'estimation par maximum de vraisemblance [DUDA 73].

Cependant, toutes ces techniques statistiques d'apprentissage non supervisé nécessitent, outre la possibilité d'utiliser un modèle paramétrique pour décrire la fonction de densité sous-jacente, des hypothèses restrictives. Ainsi la connaissance du nombre de classes en présence est souvent exigée [SCHR 76], ce nombre pouvant même être limité à deux classes dans certains cas [MAKO 77],[MIZO 75]. D'autres hypothèses restrictives, telles que l'égalité des matrices de covariance ou la connaissance des probabilités a priori des différentes classes sont parfois exigées.

Postaire et Vasseur proposent d'analyser la convexité de la fdp sous-jacente pour approcher tous les paramètres nécessaires à la description d'un mélange gaussien totalement inconnu. Bien que limitée aux distributions normales, cette approche ne nécessite aucune hypothèse restrictive [POST 81][POST 82b].

I.3 CLASSIFICATION AUTOMATIQUE SUPERVISEE

L'objectif des techniques de classification supervisée est de concevoir un classifieur capable d'assigner toute observation inconnue qui lui est présentée à une classe parmi K classes.

Avant d'être opérationnel, le classifieur nécessite une phase d'apprentissage et une phase de test conduisant à la séparation de l'échantillon disponible en une base d'apprentissage et une base de test. L'apprentissage consiste à utiliser séquentiellement les couples [Observation : Classe] de la base d'apprentissage pour ajuster les paramètres du classifieur. A la fin de cette phase, les paramètres sont fixés et la phase de test débute pour évaluer les performances du classifieur sur les données de la base de test non utilisées dans la première phase.

L'apprentissage d'un classifieur peut être envisagé par deux approches différentes :

métriques ou statistiques.

I.3.1 Méthodes métriques

Cette approche consiste à partitionner l'espace de représentation des observations en domaines. Ces domaines sont délimités par des surfaces. Beaucoup de surfaces de séparation peuvent être envisagées, les plus simples correspondent au cas linéaire. D'autres surfaces de séparation d'ordre supérieur à 1 peuvent également être utilisées, par exemple des surfaces quadratiques telles que hypersphères, paraboloïde, ellipsoïde, etc. Les possibilités sont alors supérieures à celles d'un classifieur linéaire, mais le défaut de cette démarche est que le nombre de coefficients à ajuster, dépend directement de la dimension de l'espace des observations. C'est la raison pour laquelle on se limite souvent à des surfaces de séparation de type hyperplan (cf. Figure I.1). Dans ce cas les techniques utilisées consistent à déterminer les équations mathématiques de ces hyperplans. Ensuite, à partir de ces équations, on construit une série de règles permettant de déterminer la classe d'appartenance de toute observation inconnue. Ainsi l'on obtient des règles de décision similaires à celle-ci :

$$\text{si} \quad d_k(X) > d_{k*}(X) \quad \forall \quad k \neq k* \quad \text{alors} \quad X \in C_{k*},$$

où $d_k(X)$ est appelée fonction discriminante.

L'équation déterminant l'hyperplan de séparation entre le domaine D_k et le domaine D_{k*} s'exprime par :

$$d_k(X) = d_{k*}(X).$$

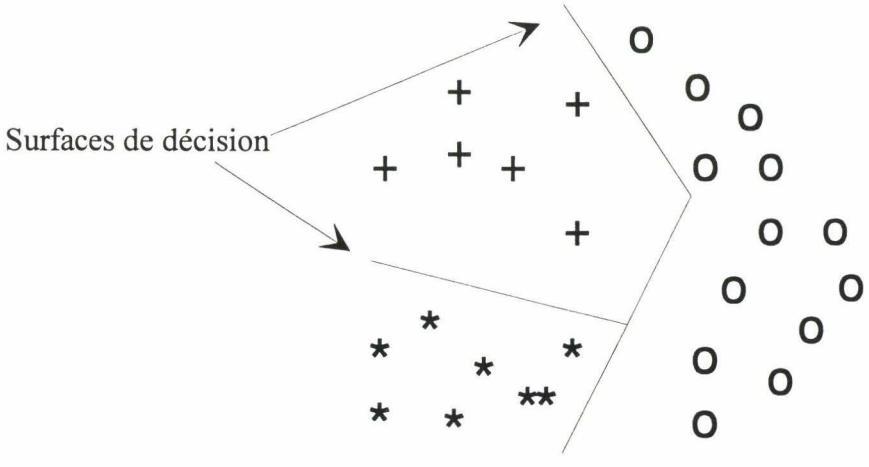


Figure I. 1: Représentation des domaines

Le classifieur assignera alors une observation inconnue X, en identifiant le domaine D_k de l'espace dans lequel se trouve cette observation [DUDA 73]. Ainsi on a :

$X \in D_k \Leftrightarrow X \in C_k$

I.3.2 Méthodes statistiques

Dans le cadre des méthodes statistiques, on fait appel aux caractéristiques statistiques de la distribution des observations. La théorie de la décision constitue une approche fondamentale des problèmes de classification qui se trouvent posés en termes probabilistes. Elle permet d'effectuer un classement optimal, basé sur la connaissance des probabilités a priori et des probabilités conditionnelles associées à chaque classe. Cependant, en pratique on ne dispose pas de ces informations. Elles doivent être estimées à partir de l'ensemble des prototypes de chaque classe.

La conception d'un classifieur statistique se décompose en deux autres approches selon que l'on considère la fdp sous une hypothèse paramétrique ou non.

Après l'estimation des paramètres statistiques, le classifieur est défini par les règles de Bayes. Ainsi une observation inconnue X est assignée à la classe C_{k*} si elle vérifie la règle suivante :

si $p(X/C_{k*})\times P(C_{k*})>p(X/C_k)\times P(C_k)\ \forall\ k\in[0;K-1],\ k\neq k^*$ alors $X\in C_{k*}$.

$P(C_k)$ est la probabilité a priori de la classe C_k.

$p(X/C_k)$ est la densité de probabilité sous-jacente des observations provenant de la classe C_k.

I.4 CLASSIFICATION INTERACTIVE

I.4.1 Objectifs de la classification interactive

Dans les méthodes de classification automatique présentées dans le paragraphe précédent, l'opérateur a un rôle limité au réglage de quelques paramètres sans pouvoir agir directement sur les résultats de la classification.

Une autre voie consiste à faire intervenir l'opérateur plus amplement dans le processus de classification en utilisant les facultés visuelles exceptionnelles de l'Homme à analyser instantanément une scène complexe à 2 ou 3 dimensions. Comme l'espace de représentation des observations est de grande dimension, on procède à une projection des données aussi fi-

dèle que possible sur l'écran d'un ordinateur.

L'avantage de la classification interactive est l'introduction implicite dans ce processus de l'expérience et des connaissances de l'analyste. Ainsi, ce dernier peut, à l'aide des outils de projection et à chaque étape du processus de classification, juger de la qualité des résultats obtenus en les comparant avec la connaissance a priori qu'il dispose sur l'échantillon. De plus, l'expérience acquise lui permet d'opter vers différentes techniques de classification ou de représentation des observations. Cette démarche confère une plus grande souplesse d'analyse que celles offertes par les techniques de classification automatique. Cette approche permet aussi de donner une plus grande confiance à l'analyste dans la qualité des résultats obtenus. En effet, l'analyste accordera plus de crédit à un résultat issu d'une analyse interactive qu'à celui émanant d'un processus automatique car cette approche lui fournit un contact visuel avec l'échantillon d'observations.

Cependant l'approche interactive présente quelques inconvénients. L'utilisation des méthodes de projection d'observations multidimensionnelles dans un espace à deux dimensions peut conduire à une distorsion de la représentation des classes. Ainsi deux classes distinctes dans l'espace d'origine peuvent se trouver confondues dans l'espace de projection. Afin de pallier cet inconvénient, la démarche interactive nécessite l'emploi de différents outils de projection. De plus, en introduisant le facteur humain, celle-ci n'offre pas l'objectivité des techniques de classification automatique. Toutefois, cette démarche offre la possibilité de combiner les techniques automatiques et les techniques interactives afin de confronter les résultats obtenus avec différents outils.

I.4.2 Présentation de la classification interactive

On peut représenter le schéma d'un processus de classification interactif par la suite d'étapes [CHIE 76][CHIE 78] (cf. Figure I.2) :

Etape 1 : Acquisition.

- L'acquisition des observations consiste dans un premier temps à déterminer les attributs caractérisant au mieux les individus. Ensuite, lorsque ces attributs sont définis, il convient, à partir des individus, d'obtenir l'échantillon d'observations multidimensionnelles. Cette étape peut nécessiter des prétraitements de l'information.

Etape 2 : Visualisation.

- L'étape suivante fournit à l'opérateur sur un écran d'ordinateur une représentation de l'échantillon d'observations multidimensionnelles.

Etape 3 : Analyse.

- L'analyse de l'échantillon consiste à déterminer le nombre de classes de l'échantillon.

Etape 4 : Classification.

- La classification de l'échantillon consiste à assigner chaque observation à l'une des classes en présence.

Les résultats de chacune de ces 4 phases peuvent conduire l'analyste à remettre en cause la phase précédente.

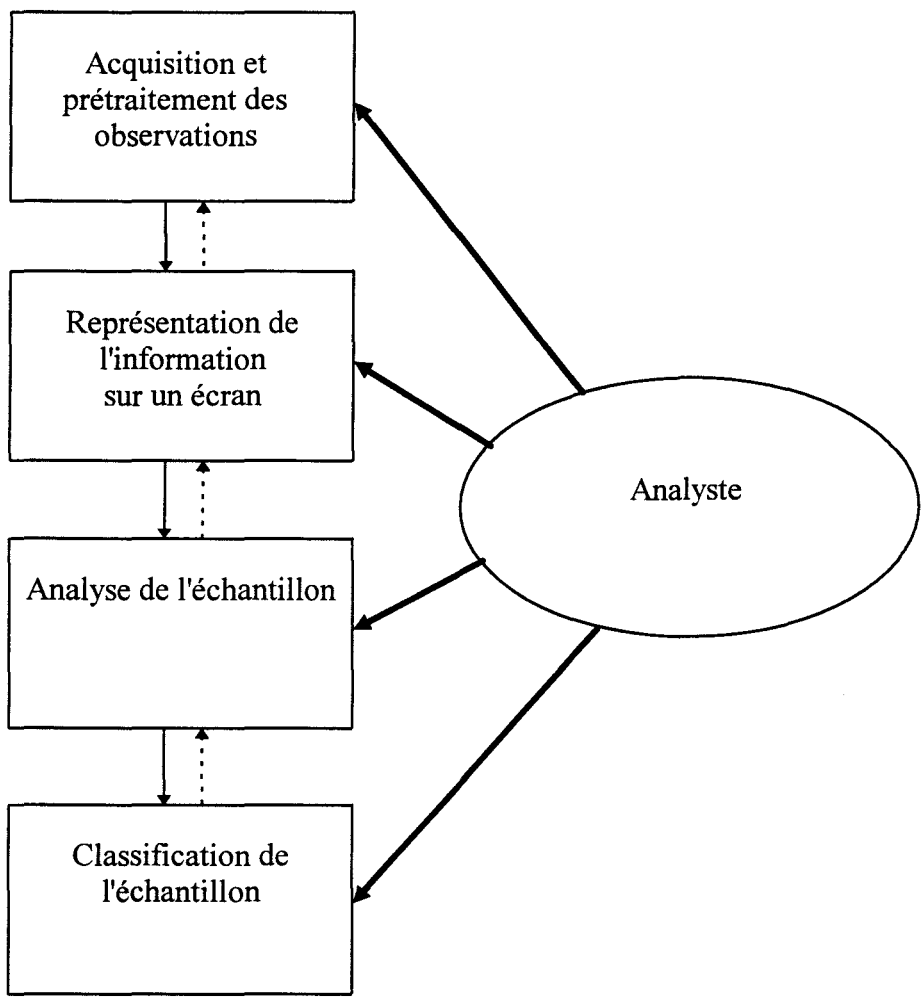


Figure I. 2 : Principe de fonctionnement d'un processus de classification interactive

L'élément essentiel d'un processus de classification interactif est la représentation de l'information. Elle sert de principal moyen de communication entre le processus et l'analyste.

I.4.3 Techniques de représentation

On distingue deux principaux moyens de représentation d'échantillons d'observations multidimensionnelles. Le premier s'attache à représenter symboliquement tous les attributs de chaque observation. C'est le cas par exemple de la méthode des visages de Chernoff où chaque attribut détermine la position ou la taille d'un élément du visage humain [CHER 73][CHIEN 76]. Ces techniques utilisent l'ensemble des valeurs des attributs de l'observation traitée afin d'effectuer la représentation. Ce mode de représentation ne permet pas de faire apparaître les similitudes entres les observations, notamment les relations de distance. De plus, lorsque le nombre d'observations est relativement important, l'analyse s'avère rapidement très délicate.

Le second moyen de représentation d'échantillons d'observations multidimensionnelles emploie des algorithmes de réduction de dimension. Il s'agit de projeter les données multidimensionnelles sur un plan. La projection obtenue découle en général d'une minimisation d'un critère d'erreur. Ainsi de nombreuses techniques de projection se distinguent par le type d'erreur minimisée.

I.4.4 Outils de projection

Siedlecki a proposé une étude détaillée des différentes techniques de projection [SIED 88a] et une analyse qualitative de ces différentes projections à des fins de classification [SIED 88b]. Il définit deux types de projections suivant qu'elles sont analytiques ou non. Une projection est dite analytique s'il existe une fonction permettant de projeter tout point de l'espace d'origine dans l'espace de projection. A l'inverse pour les techniques ou algorithmes de projection non analytiques, il n'existe pas de fonction permettant de passer de l'espace $\mathbb{R}^N$ au plan de projection. De ce fait, pour projeter une nouvelle observation, l'algorithme de projection doit reprendre l'ensemble de l'échantillon auquel on adjoint cette nouvelle donnée. Curieusement, cette propriété se confond souvent avec la linéarité ou la non linéarité de la projection.

I.4.4.1 Projections linéaires

Les projections linéaires sont basées sur la minimisation d'un critère qui permet de définir deux matrices A et B telles qu'à chaque observation X de l'échantillon on associe un vecteur $Y=[y_1,y_2]^T$ représentant la position de celle-ci dans le plan de projection selon la relation:

$$Y=A.X+B.$$

Les différentes méthodes de projection linéaire diffèrent suivant le choix du critère à minimiser.

I.4.4.1.1 Projections supervisees

Les projections supervisées sont basées sur la recherche des vecteurs propres d'une combinaison linéaire des matrices de covariance intraclasses et interclasses [WATA 67],[FUKU 70]. Ces méthodes nécessitent la connaissance au préalable des classes constituant l'échantillon. Cependant, ces projections ne garantissent pas la séparabilité des classes. Un autre type d'approche consiste à minimiser la variance intraclasse et à maximiser la variance interclasse lors de la projection. Le plan de projection est déterminé à l'aide des vecteurs R_1 et R_2 maximisant le critère suivant :

$$J_F(R) = \frac{R^T B R}{R^T S R}$$

R est un vecteur de dimension N.

B est la matrice de variance interclasse

S est la somme des matrices de covariance intraclasse.

Le pionnier de ce type de méthode fût Fisher [FISH 36]. Des variantes de cette méthode sont basées sur des modifications des matrices B et S [FEHL 78][SIED 88a].

I.4.4.1.2 Projections non supervisees

La plus connue des projections non supervisées est certainement la méthode d'analyse en composantes principales (abv. A.C.P.) [COOL 71][SEBE 84] qui est basée sur la transformation de Karhunen-Loeve [AHME 75]. L'ACP consiste à déterminer deux matrices A et B afin que les variances des observations transformées soient maximales. On obtient :

$$A = [R_1 | R_2]^T \text{ et } B = -A\overline{X}$$

où R_1 et R_2 sont les vecteurs propres associés aux plus grandes valeurs propres de la matrice

de variance de l'échantillon dans l'espace d'origine.

$\overline{X}$ est le vecteur moyenne de l'échantillon $\overline{X} = \frac{\sum_{q=1}^{Q} X_q}{Q}$

Utilisant un critère de type différent, Tuckey et Friedman ont proposé une autre méthode baptisée en anglais « projection pursuit algorithm » permettant de préserver la séparabilité des classes en n'ayant aucune connaissance a priori sur ces classes [FRIE 74]. Celle-ci, constituant une généralisation de l'ACP, est, parmi les techniques de projection linéaire non supervisée la plus apte à révéler la structure des données pour la classification non supervisée [SIED 88b].

I.4.4.2 Projections non linéaires

I.4.4.2.1 Projections supervisees

On distingue deux principales méthodes de projection supervisée non linéaires. La première utilise les vecteurs moyennes et les matrices de covariance de deux classes de référence afin de déterminer la position de la projection d'une observation. Les coordonnées du point projeté sont déterminées par la distance de Mahalanobis par rapport à chaque vecteur moyennes des deux classes. La seconde méthode utilise l'estimateur non paramétrique des k plus proches voisins. Dans le cas où l'échantillon est composé de deux classes, les coordonnées de la position d'une observation projetée correspondent aux distances séparant cette observation des deux kième plus proches observations provenant de chaque classe. Cette méthode est généralisée aux échantillons comportant plus de deux classes. Dans ce cas, les positions des points projetés s'expriment en coordonnées sphériques [SIED 88a].

I.4.4.2.2 Projections non supervisees

La projection par triangulation est certainement la plus simple des méthodes de projections non supervisées. Elle repose sur le fait que la position d'un point dans un plan peut être déterminée à partir des distances par rapport à deux autres points. Cette méthode nécessite l'initialisation d'un point de référence. La position d'une observation est déterminée par triangulation entre ce point de référence et le point le plus proche de l'observation.

La famille de méthodes « multidimensionnal scaling » [GOWE 66] [KRUS 77], consiste à minimiser des critères basés sur les différences entre les distances interpoints dans l'espace d'origine et les distances interpoints images dans le plan. La projection de Sammon

en est un exemple. Il minimise le critère suivant:

$$J_{SAM} = \frac{1}{\sum_{q'>q} d^E(X_q, X_{q'})} \sum_{q>q'} \frac{\left[d^E(X_q, X_{q'}) - d^A(Y_q, Y_{q'})\right]^2}{d^E(X_q, X_{q'})}$$

où d^E est la distance Euclidienne dans l'espace d'origine E et d^A est la distance Euclidienne dans l'espace de projection A.

Le critère J_{SAM} est non linéaire et sa minimisation est obtenue par une technique de descente de gradient [SAMM 69]. Biswas et al. ont comparé l'algorithme de Sammon avec plusieurs algorithmes de projection et ont constaté que celui-ci préserve le mieux la structure des données [BISW 81]. Son principal défaut est le temps de calcul lorsque la taille de l'échantillon est importante et l'inexistence de fonction analytique reliant le plan de projection à l'espace d'origine.

I.5 DES OUTILS RECENTS : LES RESEAUX DE NEURONES

Les progrès réalisés dans le domaine des réseaux de neurones ces dernières années ont montré une certaine analogie entre les réseaux de neurones et les techniques de classification statistiques. Werbos a estimé que 80% des travaux effectués sur les réseaux de neurones ont un rapport avec la classification classique [WERB 91].

Les réseaux de neurones sont inspirés de l'idée simple suivante: le cerveau est un excellent classifieur. En effet, il peut presque instantanément analyser et classer les objets d'une scène compliquée en deux dimensions. L'idée est la suivante: si l'on considère un modèle simplifié du neurone, si l'on interconnecte un grand nombre de ces neurones formels et si l'on adapte les poids des connexions selon un objectif bien défini, on peut espérer approcher les performances du cerveau !

Bien que les débuts des recherches sur les réseaux neuromimétiques aient conduit à des échecs, un regain d'intérêt considérable a lieu depuis le début des années 80. Ceci est dû à une innovation dans les domaines de la conception de nouvelles architectures de réseaux et des algorithmes d'apprentissage d'une part, et aux progrès importants dans la réalisation de circuits intégrés VLSI, d'autre part. Différents prototypes d'ordinateurs neuronaux ont commencé à voir le jour [MELT 92],[WAI 92]. Dans ce paragraphe nous rappelons tout d'abord le modèle du neurone formel. Ensuite, on présente les architectures usuelles et le fonctionnement des réseaux suivant qu'ils sont destinés à un apprentissage supervisé ou non.

I.5.1 Neurone formel ou produit

Dans les années quarante, les cellules nerveuses furent modélisées par McCulloch et Pitts [MCCU 43]. D'après leur définition, le neurone formel effectue une sommation pondérée de ses entrées, suivie d'un seuillage, comme le montre la figure I.3. Le neurone formel comporte :

- Les entrées : $x_1, x_2, ..., x_n, ..., x_N$.

- L'entrée totale v_m qui est le produit scalaire du vecteur des entrées et du vecteur des poids synaptiques dont la composante w_{mj} désigne le poids synaptique de la connexion entrée j du neurone m. Dans ces conditions, on a:

$$v_m = \sum_{j=1}^{N} w_{mj}.x_j$$

- La fonction seuil : $y_m = \begin{cases} 1 & \text{si } v_m > \theta \\ 0 & \text{si } v_m \leq \theta \end{cases}$

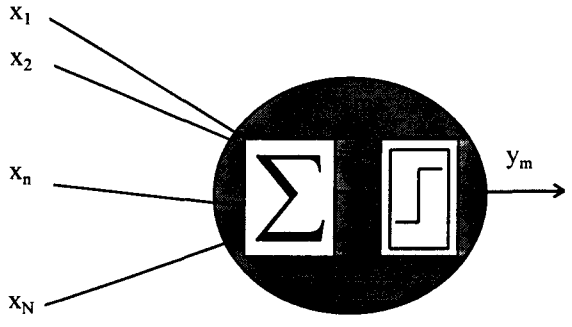


Figure I. 3 : Neurone formel ou produit

Le succès du neurone formel est dû au fait qu'il se prête bien à une interprétation géométrique. En effet, il a été montré que les poids synaptiques correspondent aux coefficients d'un hyperplan. Le neurone formel est donc un classifieur linéaire. Par ailleurs, il est facile à réaliser avec un amplificateur opérationnel [DUDA 73].

Un modèle plus général fut introduit par Mc Clelland [MCCL 86] (cf. Figure I.4). Dans ce modèle :

- $x_1, x_2, ..., x_n, ..., x_N$ représentent les stimulus d'entrée du neurone,

- $v_m = h(x_1, x_2, ..., x_n, ..., x_N)$ est l'entrée totale,

- $a_m = g(v_m)$ est l'état du neurone,

- $y_m = f(a_m)$ est la sortie du neurone.

-h désigne la fonction d'entrée totale. Elle définit le prétraitement effectué sur les en-

trées. La fonction d'entrée totale la plus courante est le produit scalaire entre le vecteur entrée du neurone et son vecteur poids synaptique.

- g est la fonction d'activation du réseau. Les fonctions d'activation les plus utilisées sont la fonction signe et la fonction sigmoïde.

-f est la fonction de sortie, souvent c'est une fonction identité, d'où $y_m = a_m$.

Notons que les valeurs des entrées et des sorties utilisées dépendent de la nature des réseaux et peuvent être binaires ou réelles.

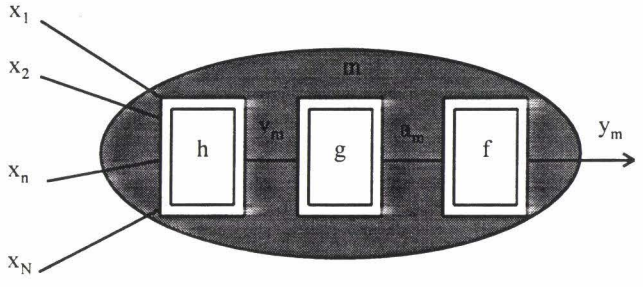


Figure I. 4 : Neurone formel selon Mc Clelland

La figure I.5 représente l'un des neurones les plus utilisés issu du modèles de Mc Clelland. L'entrée totale du réseau v_m correspond à la distance entre l'entrée X du neurone et son vecteur poids W_m :

$$v_m = \|X - W_m\| = \sqrt{\sum_{0<j\leq N} (x_j - w_{mj})^2}\ .$$

C'est le neurone distance qui est à la base des réseaux à apprentissage compétitif.

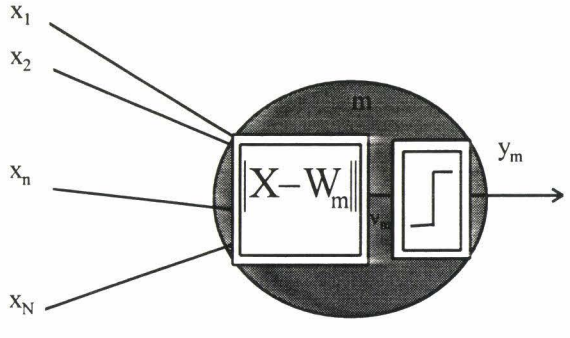


Figure I. 5 : Neurone distance

Le neurone formel entre, comme élément de base, dans une architecture plus complexe où des neurones s'interconnectent pour former des couches spécialisées. L'ensemble de ces couches forme un réseau neuronal artificiel. La bibliographie sur les réseaux de neurones

est très riche [WASS 89], [DAYH 90], [DAVA 89],[HAYK 94]. Néanmoins, on peut classer les réseaux suivant leur architecture et leur algorithme d'apprentissage.

I.5.2 Architecture des réseaux

Les architectures les plus connues sont les réseaux complètement interconnectés et les réseaux multicouches.

I.5.2.1 Les réseaux complètement interconnectés

Ce sont des réseaux de type réseau de Hopfield [HOPF 82]. Chaque neurone est connecté à tous les neurones du réseau et l'ensemble du système évolue d'états stables en états stables. La figure I.6 représente un réseau de Hopfield à 4 neurones avec connexions bidirectionnelles entre chaque paire de neurones. Le principal avantage de ce type de réseau est la possibilité de lui appliquer les résultats de la physique statistique. Ses principaux domaines d'application sont les mémoires associatives et l'optimisation combinatoire [HAYK 94].

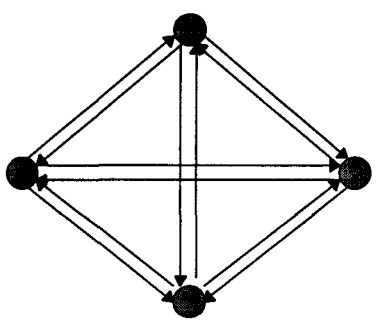


Figure I. 6 : Réseau complètement connecté

I.5.2.2 Les réseaux multicouches

Ces réseaux sont composés d'une suite de couches à circulation de l'information dirigée vers l'avant «Feed Forward», sans rétroaction. La première couche est appelée couche d'entrée, la dernière est la couche de sortie alors que les autres couches, n'ayant pas de lien direct avec les entrées et les sorties, sont appelées couches cachées (cf. Figure I.7).

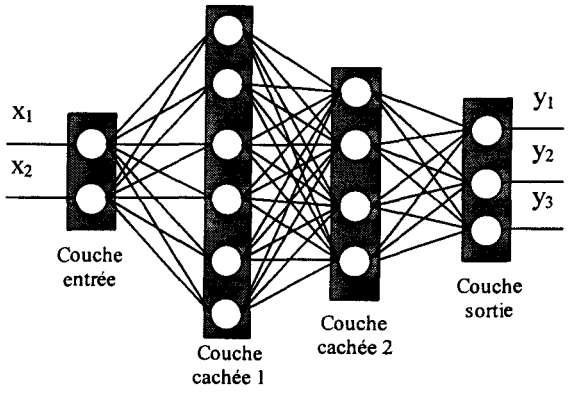


Figure I. 7 : Réseau multicouches

Les neurones d'une même couche sont exclusivement reliés à l'ensemble des neurones de la couche suivante par des connexions auxquelles sont associées des poids synaptiques. Ainsi, il n'y a pas de connexions de retour d'informations vers les couches précédentes. Le réseau fonctionne, en quelque sorte, en boucle ouverte.

Il existe différents types de réseaux ayant cette forme d'architecture. Le premier en date est le perceptron. Celui-ci, constitué d'une couche d'entrée et d'une couche de sortie, permet de classer des observations si les classes de l'échantillon sont linéairement séparables. Le perceptron multicouche à rétropropagation du gradient [RUME 85a] est une généralisation du perceptron. Il permet aussi de classer des observations, même si les classes de l'échantillon ne sont pas linéairement séparables.

L'interconnexion des neurones au sein d'une même couche dépend de la nature du réseau. Ainsi, il n'existe pas de connexions entre les neurones de la couche cachée du perceptron multicouche. Par contre, dans les réseaux à apprentissage compétitifs (abv. RNAC) les neurones de la couche cachée sont liés par des connexions d'inhibitions afin d'implanter la loi du "tout au vainqueur" [DAYH 90].

Les domaines d'application des réseaux multicouches sont très variés : reconnaissance des formes, compression des signaux et des images, optimisation, contrôle, etc...

I.5.3 Apprentissage

L'apprentissage permet au réseau de s'adapter à son environnement. Durant cette phase, où il s'agit d'apprendre par l'exemple, les vecteurs poids du réseau sont modifiés suivant un objectif bien défini dépendant des données de l'environnement. On distingue deux types d'apprentissage selon qu'il est supervisé ou non supervisé. La figure I.8 montre le prin-

cipe de ces modes d'apprentissage.

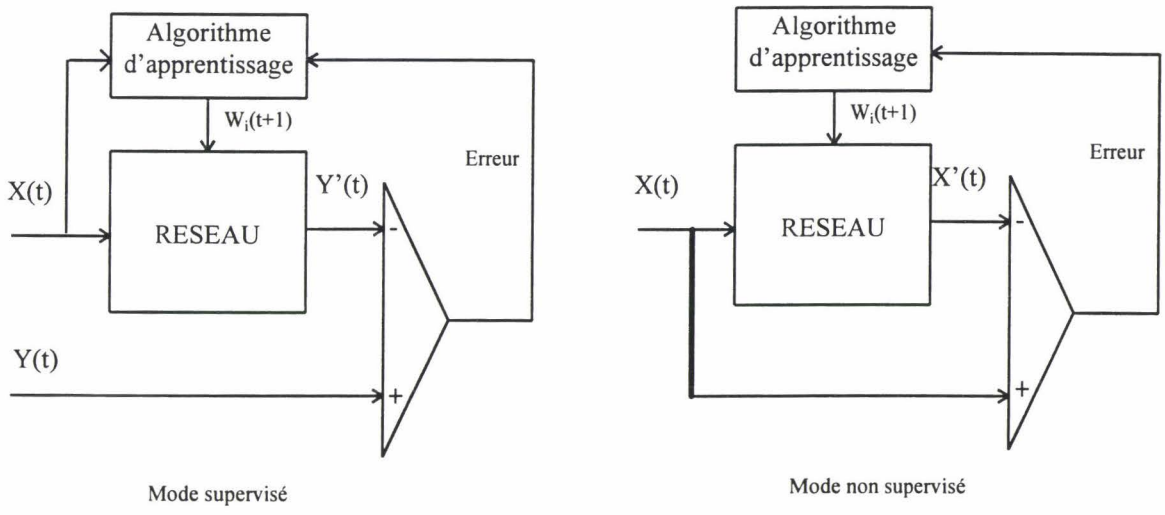


Figure I. 8 : Apprentissage des réseaux

L'apprentissage supervisé consiste à présenter séquentiellement les observations et les classes auxquelles elles appartiennent. Le réseau adapte ses poids pour mémoriser les différentes classes d'objets présents et, de ce fait, il devient capable de reconnaître la classe à laquelle appartient une observation inconnue présentée à son entrée. Notons que cette approche n'est utilisable que lorsque les classes des objets sont connues à priori.

L'apprentissage non supervisé permet de découvrir la structure des données à partir des données elles-mêmes, sans information a priori quant à leur appartenance aux différentes classes présentes dans l'échantillon analysé. Le réseau utilise une distance mesurant la similarité entre les observations pour grouper dans la même catégorie des observations semblables.

I.6 LES RESEAUX DE NEURONES DANS LA CLASSIFICATION

I.6.1 Taxonomie des réseaux de neurones classifieurs

Lippman distingue 4 groupes de réseaux de neurones classifieurs [LIPP 89] :

- Les réseaux de neurones probabilistes qui peuvent être considérés comme classifieurs statistiques basés sur l'identification des mélanges de densités de probabilité gaussiennes [DUDA 73], [FIRM 95].
- Les classifieurs à surfaces de décision hyperplanes. C'est le cas du perceptron

multicouche à apprentissage par rétropropagation du gradient, de la machine de Boltzmann et des classifieurs par arbres de décision [LIPP 87], [ACKL 85], [BREI 84].

- Les classifieurs à noyau. C'est le cas des estimateurs de noyaux de Parzen, des fonctions potentiels et les réseaux de neurones à fonction de base radiale RBF (Radial Basis Function) [SPEC 90], [MILL 90],[MOOD 89].

- Les classifieurs basés sur les exemples comme les Kppv, RCE (Reduced Coulomb Energy), ART (Adaptative Resonance Theory) et LVQ (Learning Vector Quantization) [CARP 87] [KOHO 88b].

I.6.2 La projection

La projection a été abordée sous deux aspects par les techniques neuronales. Le premier a consisté à concevoir un équivalent neuronal aux techniques de projections classiques. Ainsi des techniques de projection, telles l'ACP ou l'analyse discriminante trouvent des équivalents dans les techniques neuronales [HERT 91], [HORN 92], [MAO 93], [JAIN 92].

L'utilisation des réseaux de neurones est intéressante car elle permet d'obtenir une relation analytique, en général non linéaire, entre une observation et la position de sa projection. Ainsi, le réseau de neurones multicouche à apprentissage par rétropropagation du gradient basé sur le critère d'erreur de Sammon, permet d'obtenir une projection similaire à celle obtenue par l'algorithme classique. Dans ce cas, on dispose en plus de la relation analytique permettant de définir la position de la projection de n'importe quelle nouvelle observation [MAO 94].

Le second aspect de la projection par réseaux de neurones a consisté à adapter des réseaux existant afin d'obtenir une projection. Parmi ces réseaux on trouve les projections discriminantes non linéaires par réseaux de neurones, le réseau multicouche en mode auto-associatif et les cartes topologiques de Kohonen [MAO 93], [WEBB 90], [KOHO 88a].

La figure I.9 présente les différentes techniques de projection par réseaux de neurones.

CHAPITRE II :

RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF POUR LA CLASSIFICATION AUTOMATIQUE ET LA REDUCTION DE DONNEES

II.1 INTRODUCTION

Nous avons vu dans le chapitre précédent qu'il existe une grande diversité de réseaux de neurones. Cependant tous ces réseaux nécessitent une phase d'apprentissage lors de laquelle on corrige adaptativement leurs paramètres afin qu'ils fournissent les réponses désirées. Les paramètres corrigés sont les poids des connexions reliant les neurones. La règle permettant de modifier ces poids est appelée règle d'apprentissage. Selon Kosko [KOSK 92], il existe essentiellement deux règles permettant d'adapter les poids dans un contexte non supervisé, c'est-à-dire sans fournir d'autres informations supplémentaires que celles que l'on présente à l'entrée du réseau. La première de ces règles est la règle de Hebb [HEBB 49] qui consiste à modifier le poids de l'arc reliant deux neurones en lui ajoutant un terme proportionnel au produit des sorties des neurones reliés. La seconde est la règle d'apprentissage compétitif. Dans cette règle, les arcs reliant deux neurones sont orientés. On distingue ainsi le neurone entrée et le neurone sortie. Cette règle, modifie le poids d'un arc

reliant deux neurones de façon à approcher la valeur de sortie du neurone entrée. Cette différence revêt une importance capitale car, en choisissant des neurones entrées spécifiques, ce mécanisme d'adaptation permet au réseau de mémoriser l'information que l'on présente à son entrée. De plus, cette règle est très simple. Elle est utilisée pour l'apprentissage de réseaux de neurones très divers et elle est à la base d'autres règles d'apprentissage.

Nous présentons tout d'abord, au cours de ce chapitre, les réseaux les plus simples utilisant la règle d'apprentissage compétitif. Ceux-ci sont appelés réseaux à apprentissage compétitif. Ils nous permettent de présenter le rôle essentiel que jouent les poids des neurones et de dégager les domaines d'utilisation possibles dans le cadre général de l'analyse de données.

Le premier domaine d'utilisation dans ce cadre est la classification automatique non supervisée. Nous montrons comment utiliser ces réseaux afin de classer des échantillons d'observations multidimensionnelles. Nous montrons ensuite les analogies de cette technique de classification avec l'algorithme classique des K-means. Nous proposons ensuite des algorithmes d'apprentissage dérivés de la règle d'apprentissage compétitif permettant de résoudre certains problèmes de convergence, bien connus des utilisateurs de l'algorithme des K-means.

Le second domaine d'utilisation de ces techniques neuronales est la réduction de données qui est utilisée quand on désire extraire un échantillon de taille réduite d'un échantillon trop important pour être analysé. On peut ainsi analyser l'échantillon réduit par des techniques qu'il serait impossible d'employer sur l'échantillon d'origine. Nous montrons comment obtenir une réduction de données en utilisant certaines techniques d'apprentissage dérivées de l'apprentissage compétitif.

Nous concluons enfin ce chapitre en évaluant les algorithmes d'apprentissage en termes de coûts de calcul lors d'une implantation de ces réseaux sur une machine séquentielle.

II.2 RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF

Rumelhart et Zisper [RUHM 85b] ont défini les réseaux de neurones à apprentissage compétitif (abréviation : R.N.A.C.) suite à l'étude des principes d'auto-organisation. Etant donné les multiples versions de ces réseaux, nous proposons une version adaptée à notre problème, mais cependant suffisamment générale pour englober les différentes variantes étudiées au cours de ce travail.

II.2.1 Architecture

L'architecture des RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF est généralement représentée par deux couches de neurones comme le montre la figure II.1.

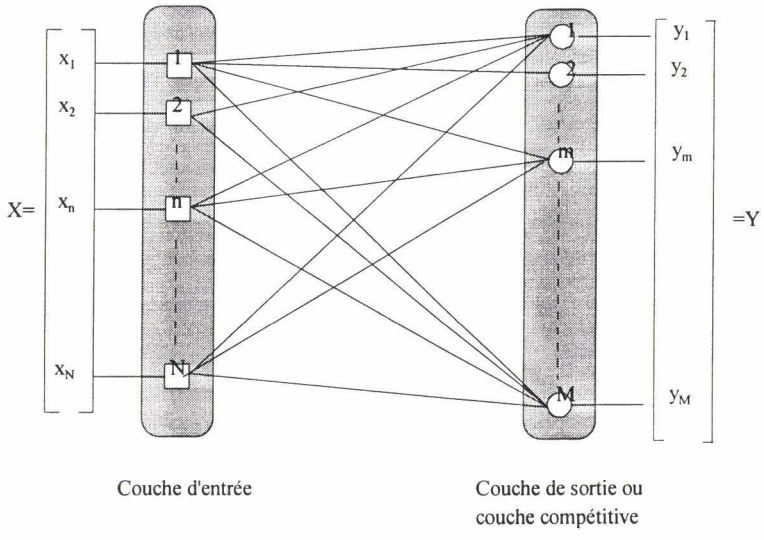


Figure II. 1 : Représentation des réseaux à apprentissage compétitif

La couche d'entrée est composée de N neurones, N étant la dimension de l'espace de représentation des observations. La couche de sortie, ou couche compétitive, est composée d'un nombre M de neurones.

Chaque neurone m, m=0, ..., M-1, de la couche de sortie est relié à l'ensemble des neurones de la couche d'entrée par une connexion pondérée. Le vecteur $W_m=(w_{m1}, ..., w_{mn}, ..., w_{mN})^T$, où w_{mn} est le poids de la connexion entre le neurone de sortie m et le neurone d'entrée n, est appelé vecteur poids du neurone m. Ce vecteur poids est de même dimension que le vecteur X constitué par les entrées du réseau. Ainsi, on peut représenter l'entrée X et le vecteur poids W_m dans un même espace de dimension N par deux points dont les positions par rapport à l'origine sont données par les composantes des vecteurs X et W_m.

La représentation interne des neurones diffère selon leur appartenance à la couche d'entrée ou à la couche de sortie.

Neurones de la couche d'entrée

Les neurones de cette couche ne font aucun calcul et ont pour unique rôle de transmettre les informations présentes à l'entrée du réseau à l'ensemble des neurones de la couche de sortie. Etant donné la particularité de ces neurones, on les symbolise sur des schémas sous forme de carrés. Le neurone de la figure II.2 transmet la valeur de x_n aux neurones de la couche suivante.

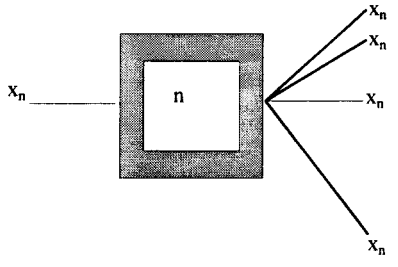


Figure II. 2 : Représentation d'un neurone de la couche d'entrée

Neurones de la couche de sortie

Les entrées d'un neurone de la seconde couche correspondent aux composantes du vecteur d'entrée X représenté sur la figure II.1. Le nombre d'entrées d'un neurone de la couche compétitive est donc égal à la dimension des observations constituant l'échantillon. La présentation d'une observation à l'entrée du réseau implique une compétition entre les neurones de la seconde couche. Un neurone m* de cette couche est gagnant si la distance d_{m*} séparant l'observation présentée à l'entrée du réseau de son vecteur poids est la plus faible sur l'ensemble des distances d_m associées aux autres neurones de la couche. La sortie y_{m*} du neurone gagnant est mise à 1. Les autres neurones ont leurs sorties y_m nulles. La figure II.3 représente les neurones de cette couche.

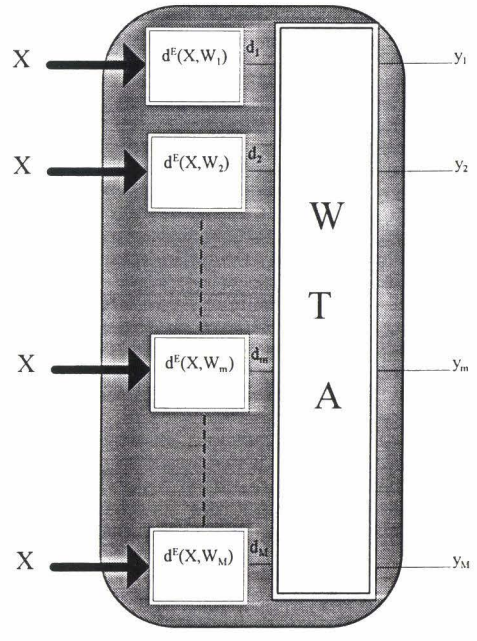


Figure II. 3 : Représentation des neurones de la couche compétitive

La première fonction réalise le calcul de la distance d_m entre le vecteur d'entrée du neurone m et son vecteur poids associé.

$d_m=d^E(X,W_m)=(X-W_m)^T(X-W_m)$.

La seconde fonction est la fonction d'activation. Pour les réseaux à apprentissage compétitif elle suit la loi dite « Tout au vainqueur » (en anglais « Winner Takes All » ou WTA). Ainsi, on définit m* tel que :

$$m^* = \underset{0\le m<M}{\text{Arg min}}(d_m)$$

m* est l'indice du neurone dont le vecteur poids W_{m^*} est le plus proche de l'observation X (cf. Figure II.4).

II.2.2 Algorithme d'Apprentissage Compétitif

L'apprentissage d'un réseaux de neurones à apprentissage compétitif s'effectue à partir d'un échantillon de Q observations $\mathcal{X}=\{X_1, ..., X_q, ..., X_Q\}$, $X_q=(x_{q1}, ..., x_{qn}, ..., x_{qN})^T$ réparties dans l'espace de représentation des observations $\mathbb{R}^N$ de dimension N. Le nombre de neurone de la couche d'entrée est égal à la dimension de cet espace.

Les vecteurs poids du réseau sont de même dimension que les observations présentées à l'entrée du réseau. On note l'échantillon constitué par les M vecteurs poids $\mathcal{W}=\{W_0, ..., W_m, ..., W_{M-1}\}$ avec $W_m=(w_{m1}, ..., w_{mn}, ..., w_{mN})^T$. L'apprentissage compétitif consiste à modifier les poids du réseau afin qu'ils s'adaptent au mieux aux observations de l'échantillon d'apprentissage $\mathcal{X}$.

Durant l'apprentissage compétitif, on présente les observations une à une à l'entrée du réseau. Aussi, nous définissons t comme variable d'itération et X(t) comme l'observation présentée à l'entrée du réseau à l'itération t.

Lors de la présentation d'une observation X(t) à l'itération t, chaque neurone calcule la distance $d_m(t)$ entre son vecteur poids $W_m(t-1)$ et X(t).

Posons : $d_m(t)=(X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$.

Le neurone dont la distance est la plus faible est dit gagnant et sa sortie est mise à 1. C'est le neurone dont le vecteur poids est le plus proche de l'observation X(t). Les sorties des autres neurones sont mises à 0. Nous appelons loi d'activation la relation déterminant l'état de sortie des neurones. Ainsi la loi d'activation d'un réseaux de neurones à apprentissage compétitif s'exprime selon la formule :

$$\begin{cases} y_m = 1 & si \quad m = m^* \\ y_m = 0 & si \quad m \neq m^* \end{cases} \text{ avec } m^* = \underset{0 \le m < M}{\operatorname{Arg\,min}} \left(d_m(t)\right)$$

Le vecteur poids du neurone gagnant est modifié afin de minimiser l'erreur $e_{AC}(t)$ telle que :

$$e_{AC}(t) = \frac{1}{2} \sum_{0 \le m < M} y_m(t) d_m(t) = \frac{1}{2} \sum_{0 \le m < M} y_m(t) \left(X(t) - W_m(t-1)\right)^T \left(X(t) - W_m(t-1)\right) .$$

Notons que seul le terme d'indice m* est différent de 0. Sa sensibilité par rapport à W_m est donnée par :

$$\frac{\partial e_{AC}(t)}{\partial W_m} = -y_m(t)\left(X(t) - W_m(t-1)\right) .$$

L'adaptation des neurones est effectuée selon la méthode de descente du gradient :

$$W_m(t) = W_m(t\text{-}1) - \alpha(t)\frac{\partial e_{AC}(t)}{\partial W_m} \quad .$$

En remplaçant $\frac{\partial e_{AC}(t)}{\partial W_m}$ dans l'équation ci-dessus, nous obtenons :

$$W_m(t) = W_m(t\text{-}1) - \alpha(t).y_m(t).(X(t)-W_m(t\text{-}1)).$$

Nous pouvons aussi remplacer $y_m(t)$ ce qui donne :

$$W_m(t) = W_m(t\text{-}1) + \alpha(t).(X(t)-W_m(t\text{-}1)) \qquad \text{si } m=m^*$$

$$W_m(t) = W_m(t\text{-}1) \qquad \text{si } m \neq m^*.$$

Les équations déterminant la mise à jour des vecteurs poids sont appelées règles d'apprentissage. La figure II.4 illustre le mécanisme d'adaptation des vecteurs poids. Nous représentons les vecteurs poids ainsi que les observations par des points dont les coordonnées par rapport à l'origine correspondent aux composantes des vecteurs. Sur cette figure, l'observation présentée à l'entrée du réseau apparaît sous la forme d'une étoile. Les vecteurs poids du réseau sont représentés par des points noirs. Le point le plus proche de l'observation correspond à celui dont la distance $d_{m^*}(t)$ est la plus faible. Le vecteur poids $W_{m^*}(t\text{-}1)$ est modifié et par conséquent les coordonnées du point sont modifiées. Ce point, représenté en gris sur la figure II.4, est déplacé vers l'observation X.

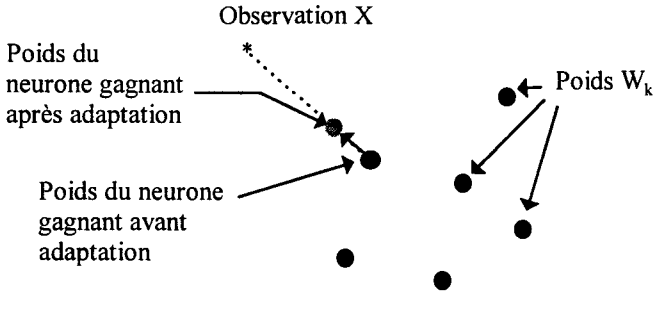


Figure II.4 : Mécanisme d'adaptation des poids de l'apprentissage compétitif

Le paramètre $\alpha(t)$ est appelé le coefficient d'apprentissage. Il dépend de la variable d'itération t. Ce terme est inférieur à 1 et décroît selon les itérations. Ainsi, au début de l'apprentissage, le réseau apprend avec un coefficient d'apprentissage élevé. Puis, il apprend de moins en moins au cours des itérations : le réseau s'accoutume à l'échantillon. Le

coefficient d'apprentissage doit satisfaire aux conditions stochastiques de convergence [GROS 82] :

Condition de plasticité : $\sum_{t=0}^{\infty} \alpha(t) = \infty$,

Condition de stabilité : $\sum_{t=0}^{\infty} (\alpha(t))^2 < \infty$.

La condition de plasticité garantit la convergence des poids vers les observations. La condition de stabilité permet d'éviter la divergence de l'algorithme. La fonction la plus connue vérifiant ces hypothèses est la suivante :

$$\alpha(t) = \frac{\alpha_0}{t} \quad \text{avec} \quad t>0.$$

Algorithme d'apprentissage compétitif

1. Initialisation aléatoire des poids $W_m(0)$ du réseau $m=0, ..., M-1$.

Rang d'itération $t=1$.

2. Tirage d'une observation $X(t)$ dans l'échantillon $\mathcal{X}$.

3. Calcul pour chaque neurone de la distance $d_m(t)$

$$d_m(t)=(X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$$

4. Activation du neurone m^* et inhibition des autres tel que :

$$m^* = \underset{0\le m<M}{Argmin}\left(d_m(t)\right)$$

$y_m=1$ *si* $m=m^*$

$y_m=0$ *si* $m\neq m^*$.

5. Modification des vecteurs poids

Calcul de $\alpha(t)$

Adaptation des vecteurs poids :

$W_m(t)=W_m(t-1)+\alpha(t).(X(t)-W_m(t-1))$ *si* $m=m^*$

$W_m(t)=W_m(t-1)$ *si* $m \neq m^*$

6. Test du critère d'arrêt,

si la condition est vérifiée : fin de l'apprentissage.

sinon reprise du procédé à partir de l'étape 2 avec $t=t+1$.

Remarque :

Une loi de sélection ou de tirage des observations, souvent utilisée, est celle dite du « Bootstrap ». Celle-ci consiste à effectuer un tirage avec remise des Q observations de cet échantillon [SAPO 90]. Nous avons préféré utiliser un tirage sans remise. Autrement dit, à chaque itération, l'on effectue un tirage de telle sorte que les Q observations sont sélectionnées une fois et une seule. Au bout de Q itérations, on reprend le tirage à partir de l'échantillon initial. Chaque période de Q itérations est appelée époque. Le tirage sans remise nous garantit la relation suivante :

$E(X) = \overline{X}$ pour t multiple de Q, avec Q=card($\mathcal{X}$)

où E(X) est l'espérance des observations de l'échantillon $\mathcal{X}$. Elle s'exprime par :

$$E(X) = \frac{\sum_{q=1}^{Q} X_q}{Q}.$$

$\overline{X}$ représente le vecteur moyenne de l'ensemble des observations présentées à l'entrée du réseau à l'instant t.

$$\overline{X} = \frac{\sum_{t'=0}^{t} X(t')}{t+1}$$

Cette technique nous assure qu'à chaque époque l'ensemble des observations de l'échantillon a été « appris » par le réseau.

Nous allons maintenant voir de quelle façon utiliser ces réseaux afin d'effectuer une classification non supervisée d'un échantillon d'observations.

II.3 APPLICATION DES RNAC A LA CLASSIFICATION AUTOMATIQUE NON SUPERVISEE

On dispose d'un échantillon de Q observations $\mathcal{X}=\{X_1, ..., X_q, ..., X_Q\}$, $X_q=(x_{q1}, ..., x_{qn}, ..., x_{qN})^T$ réparties dans l'espace de représentation des observations $\mathbb{R}^N$. On désire assigner les observations à l'une des K classes $\{C_0, ..., C_k, ...C_{K-1}\}$.

Pour effectuer une classification à l'aide d'un réseaux de neurones à apprentissage

compétitif, nous considérons que chaque vecteur poids est représentatif d'une classe. Ainsi on fixe le nombre de neurones du réseau M au nombre de classes K. Chaque classe C_k est caractérisée par le vecteur poids $W_k=(w_{k1}, ..., w_{kn}, ..., w_{kN})^T$.

Pour effectuer une classification de l'échantillon, nous procédons à un apprentissage du réseaux de neurones à apprentissage compétitif avec l'échantillon d'observations. Au début de l'apprentissage, les vecteurs poids sont initialisés aléatoirement. Durant l'apprentissage, les vecteurs poids vont se rapprocher des observations si on les représente dans le même espace $\mathbb{R}^N$. On constate que, durant cette phase, les K vecteurs poids se dirigent vers les centres des K classes. Après cette phase d'apprentissage, les poids des neurones sont fixes et ne changent plus en fonction du rang d'itération. On assigne alors les observations suivant la règle :

$$X \text{ est assignée à la classe } C_{k^*} \text{ si } k^* = \underset{0\le k<K}{\text{Arg min}}\left((X - W_k)^T (X - W_k)\right)$$

k* est aussi l'indice du neurone gagnant lorsque l'on présente X à l'entrée du réseau.

Algorithme de classification par réseaux de neurones à apprentissage compétitif

1. *Fixer le nombre M de neurones égal au nombre K de classes*
2. *Effectuer l'apprentissage du réseau avec l'échantillon d'observations*
3. *Initialisation des classes :*

   $C_k(0)=\varnothing$ *avec* $k=0, ..., K-1$.

   rang d'itération $t=1$.

3. *Tirage sans remise d'une observation X(t) dans l'échantillon* $\mathcal{X}$
4. *Calcul pour chaque neurone de la distance :*

   $d_k(t)=(X(t)-W_k)^T(X(t)-W_k)$

5. *Assigner X à la classe* C_{k^*} *vérifiant :*

   $$k^* = \underset{0\le k<K}{\text{Arg min}}\, d(X(t), W_k)$$

6. *Mise à jour des classes :*

   $C_k(t)=C_k(t-1)\cup\{X(t)\}$ *si* $k=k^*$

   $C_k(t)=C_k(t-1)$ *si* $k\neq k^*$

7. *Test du critère d'arrêt :*

   si $t=Q$ *alors fin de la classification*

   sinon reprise du procédé à partir de l'étape 3 avec $t=t+1$.

Cette technique de classification ne permet pas toujours d'obtenir une classification correcte. Suivant l'initialisation des vecteurs poids, il arrive que certains poids ne se déplacent vers aucune classe ou encore que plusieurs vecteurs poids se dirigent vers une même classe. Ce problème de convergence se rencontre aussi dans la convergence des vecteurs moyennes dans l'algorithme K-means.

II.4 CLASSIFICATION PAR RESEAUX DE NEURONE A APPRENTISSAGE COMPETITIF ET ALGORITHME DES K-MEANS

II.4.1 Rappel de l'algorithme des K-means

Comme pour la classification par réseaux de neurones à apprentissage compétitif, on dispose d'un échantillon de Q observations $\mathcal{X}=\{X_1, ..., X_q, ..., X_Q\}$, $X_q=(x_{q1}, ..., x_{qn}, ..., x_{qN})^T$ réparties dans l'espace d'observations $\mathbb{R}^N$. On désire assigner les observations à l'une des K classes $\{C_0, ..., C_k, ...C_{K-1}\}$. Chaque classe C_k est caractérisée par un vecteur $W_k=(w_{k1}, ..., w_{kn}, ..., w_{kN})^T$ et contient q_k observations de telle sorte que :

$$\sum_{0\le k<K} q_k = Q$$

Mac Queen suppose que les observations arrivent séquentiellement [ANDE 73]. Il propose d'étiqueter une observation se présentant à l'instant t en l'affectant à la classe $C_k(t-1)$ contenant alors $q_k(t-1)$ observations et dont le vecteur $W_k(t-1)$ est le plus proche. Ce vecteur est alors réactualisé en tenant compte de cette observation. Notons :

$$d(X(t),W_k(t-1)) = (X(t)-W_k(t-1))^T(X(t)-W_k(t-1)),$$

la distance entre X(t) et $W_k(t-1)$.

L'observation X(t) est assignée à la classe $C_{k*}(t-1)$ contenant $q_k(t-1)$ observations, si :

$$d(X(t),W_{k*}(t-1))\le d(X(t),W_k(t-1)) \ \forall\ k=0, ..., K-1.$$

Nous pouvons formuler cette règle d'assignation autrement :
L'observation X(t) est assignée à la classe $C_{k*}(t-1)$ contenant $q_k(t-1)$ observations, si :

$$k^* = \underset{0 \le k < K}{\text{Arg min}}\, d(X(t), W_k(t-1))$$

La classe $C_{k^*}(t)$ contient alors une observation de plus. Ainsi on a:

$q_{k^*}(t)=q_{k^*}(t-1)+1$

Les prototypes $W_k(t)$ sont réactualisés selon le schéma :

$$W_k(t) = W_k(t-1) + \frac{1}{q_k(t)}\left(X(t) - W_k(t-1)\right) \qquad \text{si } k=k^*$$

$$W_k(t) = W_k(t-1) \qquad \text{si } k \neq k^*.$$

Cette règle résulte de la minimisation de l'erreur spatiale totale suivante :

$$e_{KM}(t) = \frac{1}{2} \sum_{0 \le k < K} \sum_{X \in C_k(t)} \left(X - W_k(t)\right)^T \left(X - W_k(t)\right)$$

La sensibilité de $e_{KM}(t)$ par rapport à $W_k(t)$ est :

$$\frac{\partial e_{KM}(t)}{\partial W_k} = - \sum_{X \in C_k(t)} \left(X - W_k(t)\right)$$

L'annulation du gradient conduit à :

$$\sum_{X \in C_k(t)} \left(X - W_k(t)\right) = 0$$

d'où : $$W_k(t) = \frac{1}{q_k(t)} \left(\sum_{X \in C_k(t)} X \right)$$

Autrement dit, $W_k(t)$ correspond au vecteur moyenne des observations appartenant à la classe $C_k(t)$. Cette équation peut aussi s'écrire d'une manière itérative. Ainsi pour la présentation de X(t) à l'instant t, on obtient :

si $k=k^*$:

$$W_k(t) = \frac{1}{q_k(t)} \left(\sum_{X \in C_k(t-1)} X + X(t) \right)$$

avec $q_k(t)=q_k(t-1)+1$

si $k \neq k^*$:

$W_k(t)=W_k(t-1)$

avec $q_k(t)=q_k(t-1)$

Cette équation peut encore s'écrire, en tenant compte de $W_k(t-1)$, sous la forme :

$$W_k(t) = W_k(t-1) + \frac{1}{q_k(t)}\left(X(t) - W_k(t-1)\right) \qquad \text{pour } k=k^*$$

$$W_k(t)=W_k(t-1) \qquad \text{pour } k\neq k^*$$

Algorithme des K-means

1. Initialisation des vecteurs moyennes $W_k(0)$

On prend $W_k(0)$ aléatoirement parmi l'échantillon $\mathcal{X}$, avec k=0, ..., K-1.

$C_k(0)=\{W_k(0)\}$, $q_k(0)=1$ avec k=0, ..., K-1.

t=1.

2. Sélection aléatoire d'une observation X(t) de l'échantillon non encore assignée à une classe.

3. Calcul pour chaque k, k=0, ..., K-1, de :

$$d(X(t),W_k(t-1))=((X(t)-W_k(t-1))^T((X(t)-W_k(t-1))$$

4. Recherche de la classe k telle que :*

$$k^* = \underset{0\leq k<K}{Arg\,min}\, d(X(t),W_k(t-1))$$

5. Assignation de X à la classe C_{k^*} :

$C_{k^*}(t)=C_{k^*}(t-1)\cup\{X(t)\}$ *et* $q_{k^*}(t)=q_{k^*}(t-1)+1$

$C_k(t)=C_k(t-1)$ *et* $q_k(t)=q_k(t-1)$ *pour* $k\neq k^*$

6. Réactualisation des vecteurs prototypes

$$W_k(t) = W_k(t-1) + \frac{1}{q_k(t)}\left(X(t) - W_k(t-1)\right) \qquad si\ k=k^*$$

$W_k(t)=W_k(t-1)$ *si* $k\neq k^*$

7. Incrémentation du rang d'itération.

8. Réitération des 6 dernières étapes jusqu'à ce que toutes les observations soient assignées à une classe.

9. Réeffectuer l'étiquetage des observations avec les vecteurs moyennes finaux

10. Recalculer les vecteurs moyennes des classes avec le nouvel étiquetage selon:

$$W_k = \frac{1}{q_k}\sum_{X\in C_k} X \qquad avec\ k=0, ..., K-1.$$

II.4.2 Comparaisons entre l'apprentissage des réseaux de neurones à apprentissage compétitif et l'algorithme des K-means

L'algorithme d'apprentissage des réseaux de neurones à apprentissage compétitif ne classe pas les observations. La classification est effectuée dans une seconde phase après cette phase d'apprentissage. Cependant cet algorithme et celui des K-means se déroulent selon des procédés qui ne diffèrent que par quelques détails.

Le premier, par ordre de déroulement de ces algorithmes, concerne l'initialisation de ce que l'on appelle les vecteurs poids pour l'apprentissage et les vecteurs moyennes pour l'algorithme des K-means. En effet, dans l'algorithme des K-means, on initialise les vecteurs moyennes par des observations de l'échantillon tirées aléatoirement. Par contre, les vecteurs poids peuvent être initialisés par d'autres méthodes.

L'algorithme des K-means nécessite de plus de mémoriser l'ensemble des K coefficients $q_k(t)$ alors qu'un seul coefficient est nécessaire pour l'algorithme d'apprentissage compétitif.

La loi d'activation, bien qu'elle n'assigne aucune observation à une classe, est similaire à la règle d'assignation dans l'algorithme des K-means.

A part la différence entre les coefficients $q_k(t)$ et $\alpha(t)$, la règle d'apprentissage est similaire à celle du calcul des vecteurs moyennes.

Contrairement à l'algorithme des K-means qui ne suppose qu'une seule passe ou époque [TOU 74], l'apprentissage compétitif nécessite le choix d'un critère d'arrêt. Celui-ci peut être défini par un nombre maximum d'itérations. On peut aussi utiliser le taux de décroissance de la somme des distances quadratiques entre les poids des neurones gagnants et les observations.

Malgré ces petites différences au niveau des algorithmes, Kosko montre que si les vecteurs poids se stabilisent au cours de la phase d'apprentissage alors ceux-ci convergent vers le centre des classes [KOSK 92]. Ainsi considérons la condition limite :

$\Delta w_m = W_m(t)-W_m(t-1)=0$,

Ceci engendre la condition réciproque :

$\Leftrightarrow X(t) - W_m(t-1) = 0$

En intégrant cette équation sur l'ensemble des observations C_m ayant activé le neurone m, on obtient :

$$\int_{C_m} (X - W_m) p(X) dX = 0 \Leftrightarrow \quad W_m = \frac{\int_{C_m} X p(X) dX}{\int_{C_m} p(X) dX}$$

W_m correspond alors au vecteur moyenne des observations activant le neurone m. Ainsi, si l'on fixe le nombre de neurones égal au nombre de classes, alors les vecteurs poids constituent une approximation des vecteurs moyennes des classes.

Les similitudes entre ces deux algorithmes se traduisent aussi par les similitudes des résultats dans la classification des échantillons. Ainsi la classification par réseaux de neurones à apprentissage compétitif hérite des mêmes défauts que la classification par l'algorithme des K-means. On obtient souvent des solutions sous-optimales : certains vecteurs poids ne convergent vers aucune classe. De même, comme les K-means, cette technique de classification ne permet pas toujours d'obtenir une solution satisfaisante lorsque les classes ont des formes non globulaires. Des techniques récentes d'apprentissage ont été proposées pour améliorer la convergence des vecteurs poids vers les observations. Celles-ci sont basées sur de légères modifications au niveau de la loi d'activation et de la règle d'apprentissage.

II.5 TECHNIQUES RECENTES D'APPRENTISSAGE DES RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF

II.5.1 Apprentissage Compétitif Sensible à la Fréquence

Il a été montré, dès 1985, que la convergence de l'apprentissage compétitif est assurée grâce aux conditions stochastiques de convergence (plasticité et stabilité), mais que ces dernières n'évitent pas la convergence des vecteurs poids vers une solution sous-optimale [RUME 85b]. Ce problème est dû à l'initialisation des poids. De plus, un mauvais choix du nombre de neurones détériore considérablement les résultats de la classification : certains neurones, loin de toute observation, ne sont jamais activés. Ils sont appelés en anglais « dead units ».

L'apprentissage compétitif sensible à la fréquence, également appelé "Frequency Sensitive Competitive Learning", a été introduit par Ahalt et al. [AHAL 90], suite aux travaux de De Sieno [DESI 88] sur l'ajout d'un mécanisme de conscience à l'apprentissage compétitif classique. Une approche similaire avait été entreprise par Grossberg [GROS 87], qui suggérait d'utiliser un seuil pour contourner le problème des "deads-units" [RUME 85b]. Dans le modèle de Grossberg, un neurone devient plus sensible aux vecteurs d'entrée lorsqu'il perd la compétition et moins sensible lorsqu'il la gagne.

Chaque neurone m possède son propre coefficient de conscience $\mu_m(t)$ qui est égal au nombre de fois où il a été gagnant. Dans ce modèle, un neurone n'est donc plus seulement caractérisé par son vecteur poids W_m, mais également par μ_m.

La conscience va agir sur le neurone qui gagne le plus souvent, en "lui donnant du remords", autorisant ainsi les autres neurones à gagner la compétition. Plus précisément, la loi de sélection du neurone gagnant est modifiée selon le schéma :

$$\mu_{m^*} d\left(X, W_{m^*}\right) < \mu_m d\left(X, W_m\right), \quad \text{pour tout } m = 0, \ldots, M-1, \; m \neq m^*.$$

Un neurone qui gagne souvent la compétition voit son coefficient μ_m augmenter et, grâce à cette loi, a de moins en moins de chance de gagner, laissant ainsi à d'autres neurones, moins fréquemment activés, la possibilité de gagner. De ce fait, la conscience permet de contourner le problème de l'initialisation des vecteurs poids, en donnant des chances à tous les neurones d'être excités. On obtient donc une meilleure répartition spatiale des vecteurs poids. La figure II.5 illustre ce mécanisme d'adaptation des vecteurs poids.

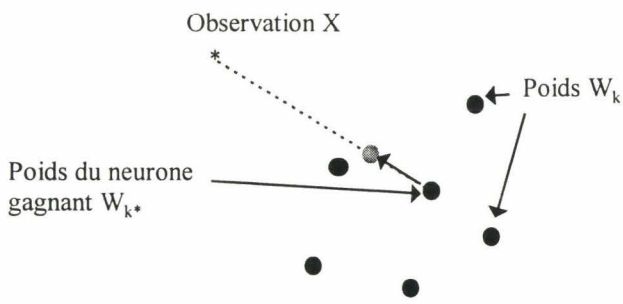


Figure II. 5 : Mécanisme d'adaptation des vecteurs poids dans l'apprentissage compétitif sensible à la fréquence

Algorithme d'apprentissage compétitif sensible à la fréquence

1. Initialisation aléatoire des poids $W_m(0)$ du réseau m=0, ..., M-1.

Initialisation des $\mu_m(0)$ à 1, pour m = 0, ..., M-1,

Rang d'itération t=1.

2. Tirage d'une observation X(t) dans l'échantillon $\mathcal{X}$.

3. Calcul pour chaque neurone de :

$$d_m(t)=\mu_m(t-1).(X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$$

4. Activation du neurone m et inhibition des autres tel que :*

$$m^* = \underset{0\le m<M}{Arg\ min}\left(d_m(t)\right)$$

$y_m=1$ *si* $m=m^*$

$y_m=0$ *si* $m\neq m^*$.

5. Modification des vecteurs poids

Calcul de $\alpha(t)$

Adaptation des vecteurs poids :

$W_m(t)=W_m(t-1)+\alpha(t).(X(t)-W_m(t-1))$ *si* $m=m^*$

$W_m(t)=W_m(t-1)$ *si* $m \neq m^*$

Mise à jour des coefficients de conscience :

$\mu_m(t)=\mu_m(t-1)+1$ *si* $m=m^*$

$\mu_m(t)=\mu_m(t-1)$ *si* $m\neq m^*$.

6. Test sur la condition d'arrêt,

si la condition est vérifiée : fin de l'apprentissage.

sinon reprise du procédé à partir de l'étape 2 avec t=t+1.

Remarques :

L'introduction d'un terme de fréquence dans la loi d'activation modifie le terme d'erreur à minimiser :

$$e_{ACSF}(t) = \frac{1}{2}\sum_{0\le m<M}\mu_m(t-1)y_m(t)(X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$$

La sensibilité de l'erreur par rapport aux vecteurs poids W_m est :

$$\frac{\partial}{\partial W_m} e_{ACSF}(t) = -\mu_m(t-1)(X(t)-W_m(t-1)) \quad \text{pour } m=m^*$$

$$\frac{\partial}{\partial W_m} e_{ACSF}(t)=0 \quad \text{pour } m\neq m^*$$

La règle de modification des poids s'écrit alors :

$W_m(t)=W_m(t-1)+\alpha(t).\mu_m(t-1)(X(t)-W_m(t-1))$ pour $m=m^*$.

$W_m(t)=W_m(t-1)$ pour $m\neq m^*$.

On peut remarquer que la règle d'apprentissage est la même que celle utilisée dans l'apprentissage de base et ne fait pas intervenir $\mu_m(t)$ [DESI 88],[AHAL 90]. Dès lors, on ne peut plus considérer cette technique d'apprentissage comme une technique de minimisation de l'erreur mais comme un moyen de minimiser l'effet de sous-utilisation des neurones. Cela suppose donc qu'après cette minimisation, on termine l'apprentissage avec un apprentissage compétitif classique afin de minimiser réellement l'erreur entre l'observation présentée à l'entrée du réseau et le vecteur poids le plus proche.

Pour obtenir une loi d'activation classique à la fin de l'apprentissage compétitif sensible à la fréquence, De Sieno propose un seuil maximum pour les coefficients de conscience. On obtient alors la loi d'évolution suivante :

$$\mu_m(t) = \begin{cases} \mu_m(t-1)+1 & \text{si} \quad m=m^* \quad \text{et} \quad \mu_m(t-1)<\mu_{max} \\ \mu_m(t-1) & \text{sinon} \end{cases}$$

μ_{max} représente le coefficient maximal de conscience des neurones.

Dans le cas idéal, au bout d'un certain nombre d'itérations, tous les neurones ont le même coefficient de conscience et la loi d'activation s'apparente alors à la loi d'activation compétitive.

Une solution plus simple que celle proposée par De Sieno consiste à effectuer deux apprentissages. Ainsi nous effectuons un apprentissage en utilisant l'apprentissage compétitif sensible à la fréquence puis nous effectuons un apprentissage compétitif. Lors de ce deuxième apprentissage nous initialisons les vecteurs poids initiaux par ceux obtenus à la fin du premier apprentissage. Cette technique nous garantit que chaque neurone subit une activation compétitive classique lors du deuxième apprentissage.

II.5.2 Apprentissage Compétitif avec Pénalisation du Rival

L'apprentissage compétitif avec pénalisation du rival est particulièrement bien adapté à la classification non supervisée quand on ne connaît pas le nombre de classes en présence. Il permet d'ajuster le nombre M de neurones au nombre K de classes [XU 93]. La version de base est modifiée cette fois-ci au niveau de la loi d'activation et de la règle d'adaptation des vecteurs poids.

L'idée de base de cette règle d'apprentissage est de considérer que le neurone gagnant est l'unique représentant d'une classe dès lors les autres neurones sont représentatifs d'autres classes. Il faut donc les éloigner de l'observation présentée au réseau. La technique de l'apprentissage compétitif avec pénalisation du rival consiste à repousser le vecteur poids du second neurone le plus proche de l'observation, appelé neurone rival. Ce mécanisme permet la convergence d'un unique vecteur poids vers le centre d'une classe et repousse éventuellement le vecteur poids du neurone rival vers une autre classe.

Pour renforcer la convergence des autres neurones vers les autres classes, Xu et al. reprennent le mécanisme de conscience proposé par De Sieno [XU 93]. L'algorithme possède, de ce fait, l'avantage de déterminer automatiquement le nombre de classes. En effet, les neurones éloignés de toute classe sont repoussés à l'infini. Cette procédure n'est évidemment valable qu'à condition de disposer d'un nombre de neurones supérieur ou égal au nombre de classes.

Ainsi le neurone gagnant m* vérifie :

$$\mu_{m*} d(X, W_{m*}) < \mu_m d(X, W_m), \quad \text{pour tout } m = 0, \ldots, M-1, \; m \neq m*$$

Le neurone rival m** est déterminé par :

$$\mu_{m**} d(X, W_{m**}) \leq \mu_m d(X, W_m), \quad \text{pour tout } m = 0, \ldots, M-1, \; m \neq m* \text{ et } m* \neq m**$$

Les règles d'apprentissage sont alors les suivantes :

$W_m(t) = W_m(t-1) + \alpha(t)(X(t)-W_m(t-1))$ si m=m*

$W_m(t) = W_m(t-1) - \beta(t)(X(t)-W_m(t-1))$ si m=m**

$W_m(t) = W_m(t-1)$ si m≠m* et m≠m**.

Les coefficients d'apprentissage $\alpha(t)$ et $\beta(t)$ sont des suites non croissantes respectant les conditions stochastiques de plasticité et de stabilité. De plus, on ajoute la contrainte $\alpha(t) >> \beta(t)$ pour éviter le rejet trop rapide de certains centres, hors du nuage de données.

Notons que ce choix est très délicat et conditionne largement les résultats [DOOZ 95].

Les coefficients de conscience sont modifiés de la même manière que dans l'apprentissage compétitif sensible à la fréquence, à savoir :

$\mu_m(t)=\mu_m(t-1)+1$ si $m=m^*$

$\mu_m(t)=\mu_m(t-1)$ si $m\neq m^*$.

Il convient également de noter que le mécanisme introduit dans l'apprentissage compétitif avec pénalisation du rival permet, dans certaines conditions, de faire converger l'algorithme plus rapidement. En effet, il se peut que le rival soit repoussé vers une classe "abandonnée" qui "adoptera" son vecteur poids comme centre (cf. Figure II.6).

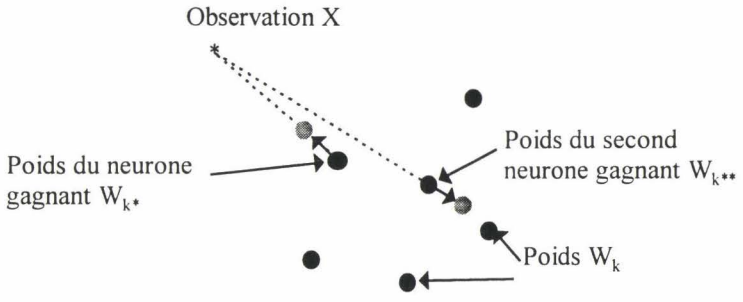


Figure II. 6 : Exemple du mécanisme de pénalisation du rival dans l'algorithme d'apprentissage compétitif avec pénalisation du rival

Algorithme d'apprentissage compétitif avec pénalisation du rival

1. Initialisation aléatoire des M vecteurs poids,

Initialisation des $\mu_m(0)$ à 1, pour m = 0, ..., M-1,

Rang d'itération t=1.

2. Tirage d'une observation X(t) dans l'échantillon $\mathcal{X}$.

3. Calcul pour chaque neurone de :

$$d_m(t)=\mu_m(t-1).(X(t)-W_m(t-1))^T(X(t)-W_m(t-1)).$$

4. Activation du neurone m et inhibition des autres tel que :*

$$m^* = \underset{0\leq m<M}{Argmin}\left(d_m(t)\right)$$

$y_m=1$ *si* $m=m^*$

$y_m=0$ *si* $m\neq m^*$.

5. Recherche du second gagnant ou rival :

$$m^{**} = \underset{\substack{0 \le m < M \\ m \neq m^*}}{Argmin}\left(d_m(t)\right) .$$

6. Modification des vecteurs poids

Calcul de $\alpha(t)$*et* $\beta(t)$*.*

Adaptation des vecteurs poids :

$W_m(t) = W_m(t-1) + \alpha(t).(X(t)-W_m(t-1))$ *si* $m=m^*$

$W_m(t) = W_m(t-1) - \beta(t).(X(t)-W_m(t-1))$ *si* $m=m^{**}$

$W_m(t)=W_m(t-1)$ *si* $m \neq m^*$ *et si* $m \neq m^{**}$

Mise à jour des coefficients de conscience :

$\mu_m(t)=\mu_m(t-1)$ *si* $m \neq m^*$

$\mu_m(t)=\mu_m(t-1)+1$ *si* $m=m^*$.

8. Test sur la condition d'arrêt,

si la condition est vérifiée : fin de l'apprentissage.

sinon reprise du procédé à partir de l'étape 2 avec $t=t+1$.

Cet algorithme hérite des défauts de convergence des apprentissages compétitifs sensibles à la fréquence. Il convient, comme dans l'algorithme d'apprentissage compétitif sensible à la fréquence, de définir un coefficient maximal de conscience des neurones ou alors d'effectuer un apprentissage compétitif classique après l'apprentissage compétitif avec pénalisation du rival. La convergence de l'apprentissage compétitif avec pénalisation du rival est très sensible au choix des paramètres $\alpha(t)$ et $\beta(t)$. De plus, la modification du vecteur poids du rival ne découle pas d'une minimisation de l'erreur. Cependant, l'utilisation de l'apprentissage compétitif avec pénalisation du rival en classification non supervisée constitue une alternative intéressante à l'algorithme des K-means car il permet de déterminer le nombre K de classes en présence [DOOZ 95].

Le phénomène de « dead unit » a été abordé par les techniques neuronales en modifiant la loi d'activation des neurones. Ainsi, en introduisant un terme de conscience dans cette loi, l'apprentissage compétitif sensible à la fréquence et l'apprentissage compétitif avec pénalisation du rival permettent de limiter ce phénomène. L'algorithme suivant pallie ce problème en modifiant la loi d'apprentissage.

II.5.3 Apprentissage Compétitif Généralisé

L'apprentissage compétitif généralisé, également appelé "Generalized Learning Vector Quantization" a été développé par Pal et al. [PAL 93] et se propose de généraliser l'apprentissage compétitif. Pour cela, dans la phase d'apprentissage, on modifie non seulement le vecteur poids du neurone gagnant, mais l'ensemble des autres vecteurs poids en fonction de la distance les séparant de l'observation présentée à l'entrée du réseau (cf. Figure II.7).

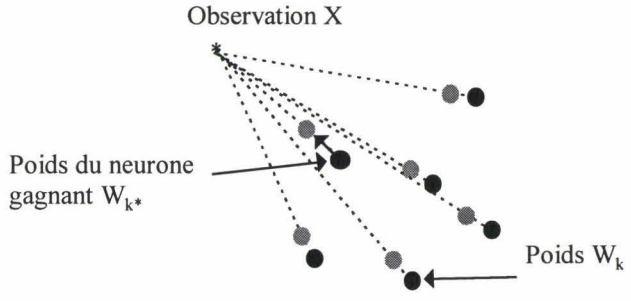


Figure II. 7 : Mécanisme d'adaptation des poids dans l'apprentissage compétitif généralisé

Cette règle d'apprentissage est issue de la minimisation de l'erreur suivante :

$$e_{ACG}(t) = \frac{1}{2} \sum_{0 \le m < M} g_m(t) \left(X(t) - W_m(t-1)\right)^T \left(X(t) - W_m(t-1)\right)$$

$$\text{avec : } \begin{cases} g_m(t) = 1 & \text{si } m = m^* \\ g_m(t) = \dfrac{1}{\sum\limits_{0 \le m < M} \left(X(t) - W_m(t-1)\right)^T \left(X(t) - W_m(t-1)\right)} & \text{si } m \neq m^* \end{cases}$$

Dans la suite, pour simplifier l'écriture, nous omettrons le paramétrage en t.

Pour établir la règle de mise à jour des poids, nous remettons en forme e_{ACG} en y introduisant g_m comme suit :

$$e_{ACG} = \frac{1}{2} \left\{ (X - W_{m^*})^T (X - W_{m^*}) + \frac{\sum\limits_{m \neq m^*} (X - W_m)^T (X - W_m)}{\sum\limits_{0 \le m < M} (X - W_m)^T (X - W_m)} \right\} .$$

En ajoutant et retranchant le terme $(X\text{-}W_{m^*})^T(X\text{-}W_{m^*})$ au numérateur du rapport à l'intérieur des accolades, cette expression prend la forme :

$$e_{ACG} = \frac{1}{2}\left\{ (X - W_{m^*})^T (X - W_{m^*}) + 1 - \frac{(X - W_{m^*})^T (X - W_{m^*})}{\sum_{0 \le m < M} (X - W_m)^T (X - W_m)} \right\} .$$

Calculons la sensibilité de e_{ACG} par rapport au vecteur poids du neurone gagnant :

$$\frac{\partial e_{ACG}}{\partial W_{m^*}} = \frac{1}{2}\left(-2(X - W_{m^*}) - \left\{ \frac{-2(X - W_{m^*}) \sum_{0 \le m < M} \|X - W_m\|^2 + 2(X - W_{m^*}) \|X - W_{m^*}\|^2}{\left(\sum_{0 \le m < M} \|X - W_{m^*}\|^2 \right)^2} \right\} \right) .$$

Pour simplifier l'écriture, nous avons adopté la notation :

$$\|X - W_m\|^2 = (X - W_m)^T (X - W_m).$$

de telle sorte que :

$$\frac{\partial e_{ACG}}{\partial W_{m^*}} = -\left\{ 1 + \frac{\|X - W_{m^*}\|^2 - \sum_{0 \le m < M} \|X - W_m\|^2}{\left(\sum_{0 \le m < M} \|X - W_m\|^2 \right)^2} \right\} (X - W_{m^*}) .$$

On obtient donc :

$$\frac{\partial e_{ACG}}{\partial W_{m^*}} = -\left\{ 1 + \frac{-\sum_{\substack{0 \le m < M \\ m \ne m^*}} \|X - W_m\|^2}{\left(\sum_{0 \le m < M} \|X - W_m\|^2 \right)^2} \right\} (X - W_{m^*})$$

Pour les autres neurones m, m≠m*, on obtient :

$$\frac{\partial e_{ACG}}{\partial W_m} = -\frac{\|X - W_{m^*}\|^2}{\left(\sum_{0 \le m < M} \|X - W_m\|^2 \right)^2} (X - W_m) .$$

On définit les coefficients β_{m^*} et β_m de la façon suivante :

$$\beta_{m^*} = \left\{ 1 - \frac{\sum_{\substack{0 \le m < M \\ m \ne m^*}} \|X - W_m\|^2}{\left(\sum_{0 \le m < M} \|X - W_m\|^2 \right)^2} \right\} \qquad \text{si } m = m^*$$

$$\beta_m = \frac{\|X - W_{m^*}\|^2}{\left(\sum_{0\le m<M}\|X - W_m\|^2\right)^2} \qquad \text{si } m\neq m^*.$$

On obtient finalement les lois d'apprentissage :

$W_m(t)=W_m(t-1)+\beta_{m^*}(t)(X(t)-W_m(t-1))$ si $m=m^*$

$W_m(t)=W_m(t-1)+\beta_m(t)(X(t)-W_m(t-1))$ si $m\neq m^*$.

Les coefficients d'apprentissage sont obtenus selon une technique bien établie de minimisation d'une fonction d'erreur quadratique et ne sont plus imposés, comme c'est le cas pour les règles d'apprentissage précédentes. Notons que l'erreur minimisée est très différente de la fonction $d_m(t)$ utilisée dans la loi d'activation. On peut considérer cette technique d'apprentissage comme une technique pour limiter les « dead unit » et appliquer, comme pour l'apprentissage compétitif sensible à la fréquence, un second apprentissage compétitif après un apprentissage compétitif généralisé. Cependant, le principal inconvénient de cette technique d'apprentissage est, qu'à chaque itération, tous les vecteurs poids sont remis à jour, ce qui augmente considérablement le temps de calcul sur une machine séquentielle. De plus, les coefficients d'apprentissage ne sont pas déterminés par une relation simple comme dans les autres techniques d'apprentissage. Cet apprentissage de type apprentissage compétitif généralisé nécessite en l'occurrence la sommation de toutes les distances $d_m(t)$ des neurones, ce qui accroît la difficulté d'une implantation matérielle.

Algorithme d'apprentissage compétitif généralisé

1. *Initialisation aléatoire des M vecteurs poids,*

   Rang d'itération t=1.

2. *Tirage d'une observation X(t) dans l'échantillon $\mathcal{X}$.*

3. *Calcul pour chaque neurone de la distance :*

   $d_m(t)=(X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$

4. *Activation du neurone m* et inhibition des autres tel que :*

$$m^* = \underset{0\le m<M}{Argmin}\left(d_m(t)\right)$$

$y_m=1$ *si* $m=m^*$

$y_m=0$ *si* $m\neq m^*$.

5. Modification des vecteurs poids

Calculs de β_m (t) et $\beta_{m*}(t)$:

$$\beta_m(t) = \left\{ 1 - \frac{\sum_{\substack{0 \le m < M \\ m \neq m^*}} \|X(t) - W_m(t-1)\|^2}{\left(\sum_{0 \le m < M} \|X(t) - W_m(t-1)\|^2 \right)^2} \right\} \qquad si\ m = m^*$$

$$\beta_m(t) = \frac{\|X(t) - W_{m^*}(t-1)\|^2}{\left(\sum_{0 \le m < M} \|X(t) - W_m(t-1)\|^2 \right)^2} \qquad si\ m \neq m^*.$$

Adaptation des vecteurs poids :

$W_m(t) = W_m(t-1) + \beta_{m*}(t).(X(t) - W_m(t-1))$ *si* $m = m^*$

$W_m(t) = W_m(t-1) + \beta_m(t).(X(t) - W_m(t-1))$ *si* $m \neq m^*$.

6. Test sur la condition de fin.

si la condition est vérifiée : fin de l'apprentissage

sinon reprise du procédé à partir de l'étape 2 avec t=t+1.

On peut noter que, pour le neurone vainqueur, le coefficient $\beta_m(t)$ est inférieur à 1 et qu'il dépend de la distance de l'observation au centre. Ce terme est d'autant plus important que la distance séparant le vecteur poids du neurone gagnant de l'observation présentée à l'entrée du réseau est importante. Cela se vérifie très simplement :

Soit d_{m*} la distance entre l'observation X et le centre vainqueur m* qui s'exprime sous la forme :

$$d_{m^*} = \|X(t) - W_{m^*}(t-1)\|^2$$

On note D la somme de toutes les distances de X(t) aux différents vecteurs poids. Ce terme s'exprime par :

$$D = \sum_{0 \le m < M} \|X(t) - W_m(t-1)\|^2$$

$\beta_{m*}(t)$ et $\beta_m(t)$ se mettent alors sous la forme :

$$\beta_{m^*}(t) = \left(1 - \frac{D - d_{m^*}}{D^2}\right) \qquad \beta_m(t) = \left(\frac{d_{m^*}}{D^2}\right)$$

Le terme β_m est proportionnel à la distance d_{m*} séparant l'observation du vecteur poids du neurone gagnant.

II.5.4 Conclusion

Les techniques présentées au cours de ce paragraphe, permettent de résoudre certains problèmes de convergence de l'algorithme d'apprentissage compétitif. Elles interviennent à deux niveaux dans l'algorithme d'apprentissage de base : la loi d'activation et la règle d'adaptation des vecteurs poids. Ainsi l'apprentissage compétitif sensible à la fréquence, par l'introduction de coefficients de conscience dans la loi d'activation, permet de limiter le phénomène de « dead-units ». L'apprentissage avec pénalisation du rival reprend la technique de l'apprentissage compétitif sensible à la fréquence pour éviter le phénomène de « dead-units » et introduit un mécanisme de pénalisation du rival dans la règle d'adaptation des poids afin d'éviter que deux vecteurs poids convergent vers la même classe. Enfin, l'apprentissage compétitif généralisé se propose de limiter le phénomène de « dead-units » en modifiant l'erreur minimisée et donc la règle d'adaptation des vecteurs poids. Toutes ces techniques d'apprentissage apportent une amélioration dans la convergence des vecteurs poids. L'utilisation ultérieure de ces vecteurs pour effectuer la classification de l'échantillon d'observations ne peut fournir que de meilleurs résultats que ceux obtenus en utilisant un apprentissage compétitif classique.

Nous avons vu que ces techniques récentes, sauf l'apprentissage compétitif avec pénalisation du rival, modifient un ensemble de vecteurs poids de telle sorte que le plus grand nombre d'entre eux s'approche des observations de l'échantillon. Cette caractéristique est très intéressante dans le domaine de la réduction de données. Pour aborder ce problème, on choisit un nombre de neurones supérieur au nombre de classes K et inférieur au nombre d'observations de l'échantillon Q. Les vecteurs poids ne convergent alors plus vers les centres des classes mais vers des zones de l'espace denses en observations.

II.6 APPLICATION DES RESEAUX DE NEURONES A APPRENTISSAGE COMPETITIF A LA REDUCTION DE DONNEES

La réduction consiste à construire, à partir d'un échantillon comportant un nombre très important d'observations, un autre échantillon composé d'un nombre restreint d'observations et possédant des caractéristiques statistiques aussi proches que possible de celles de l'échantillon d'origine. Il est intéressant d'utiliser les techniques de réduction de données dans le domaine de la classification automatique. En effet, les coûts de calcul de

certaines techniques de classification augmentent très rapidement avec le nombre d'observations de l'échantillon. Ces techniques deviennent inutilisables lorsque l'échantillon est très volumineux. Dans ce cas, seul le traitement sur un échantillon réduit, aussi représentatif que possible de l'échantillon de départ, est envisageable. En classification interactive, on utilise des techniques de projection afin de représenter les échantillons d'observations multidimensionnelles en deux dimensions. Les coûts de calcul et de stockage de certaines de ces techniques de projections sont très sensibles au nombre d'observations de l'échantillon et il préférable de traiter des échantillons réduits. Comme nous utilisons certaines de ces méthodes interactives dans le chapitre V, nous proposons une technique de réduction de données basée sur les réseaux de neurones à apprentissage compétitif.

Dans la réduction de données par réseaux de neurones à apprentissage compétitif, le nombre de neurones correspond au nombre d'observations désirées dans l'échantillon réduit et l'ensemble des vecteurs poids, noté $\mathcal{W}$, constitue cet échantillon réduit. Il convient, lors de la phase d'apprentissage, d'utiliser un algorithme permettant de réduire la sous-utilisation des neurones. Nous faisons appel pour cela à l'apprentissage compétitif sensible à la fréquence. L'apprentissage compétitif avec pénalisation du rival n'est pas utilisable car il repousse les neurones rivaux loin des classes, ce qui est incompatible avec les objectifs de la réduction de données. Nous avons testé l'apprentissage compétitif généralisé en réduction de données. Il apparaît, d'après nos simulations, que cette technique d'apprentissage conduit à des résultats médiocres dans ce domaine.

Afin d'illustrer l'utilisation des réseaux de neurones à apprentissage compétitif en réduction de données, nous présentons l'exemple 1.

Exemple 1:

Cet exemple, représenté sur la figure II.8, est composé de deux classes comprenant chacune 500 observations bidimensionnelles dont les paramètres statistiques sont indiqués dans le tableau II.1.

Classe	Nombre d'observations	Vecteur moyenne	Matrice de covariance
Classe 0	500	$\begin{pmatrix} 1,0 \\ 2,0 \end{pmatrix}$	$\begin{pmatrix} 0,1 & 0,0 \\ 0,0 & 0,02 \end{pmatrix}$
Classe 1	500	$\begin{pmatrix} 3,0 \\ 4,0 \end{pmatrix}$	$\begin{pmatrix} 0,06 & 0,0 \\ 0,0 & 0,03 \end{pmatrix}$

Tableau II. 1: Paramètres statistiques de l'exemple 1

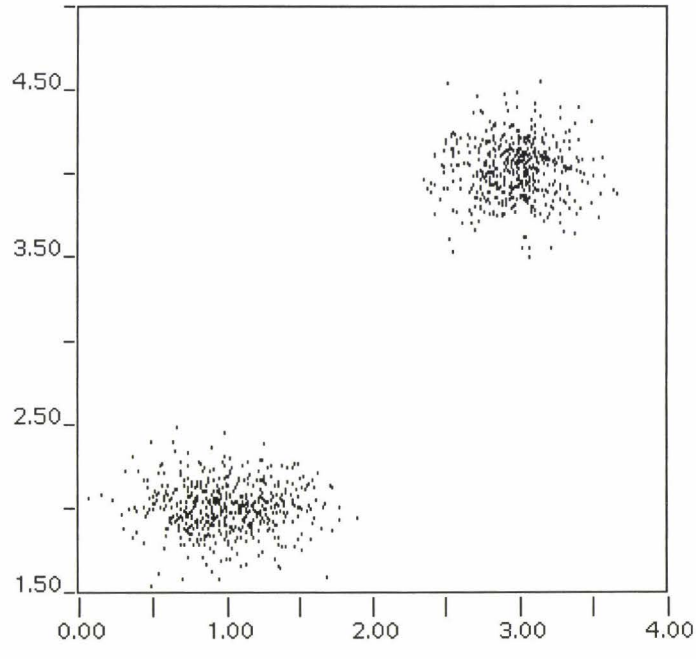


Figure II. 8 : Représentation de l'échantillon d'observations

Nous réduisons cet échantillon à 100 observations. Pour cela nous effectuons un apprentissage compétitif sensible à la fréquence d'un réseau comprenant 100 neurones avec un nombre d'époques égal à 100. Nous effectuons ensuite un apprentissage compétitif classique également pendant 100 époques. La figure II.9. représente l'échantillon composé des vecteurs de ce réseau après l'apprentissage. Tous les neurones ont convergé aux endroits denses en observations. Le tableau II.2 montre les paramètres statistiques de l'échantillon réduit.

Classe	Nombre d'observations	Vecteur moyenne	Matrice de covariance
Classe 0	53	$\begin{pmatrix} 1,01 \\ 1,99 \end{pmatrix}$	$\begin{pmatrix} 0,15 & 0,0 \\ 0,0 & 0,04 \end{pmatrix}$
Classe 1	47	$\begin{pmatrix} 2,98 \\ 4,04 \end{pmatrix}$	$\begin{pmatrix} 0,09 & 0,0 \\ 0,0 & 0,05 \end{pmatrix}$

Tableau II. 2: Paramètres statistiques de l'échantillon réduit $\mathcal{W}$ issu de l'exemple 1

Nous pouvons constater que les vecteurs moyennes des deux classes de l'échantillon réduit sont sensiblement les mêmes que ceux de l'échantillon d'origine. De même, les proportions des observations de chaque classe dans l'échantillon réduit sont sensiblement les mêmes que celles de l'échantillon original. Par contre les éléments des matrices de covariances de chaque classe diffèrent de ceux issus de l'échantillon d'origine. Nous pouvons observer sur la figure II.9 que les poids se sont répartis suivant les zones denses en observations. Cependant comme le nombre des vecteurs poids est moins important que celui des observations, les variances sont plus importantes que celles issues des observations.

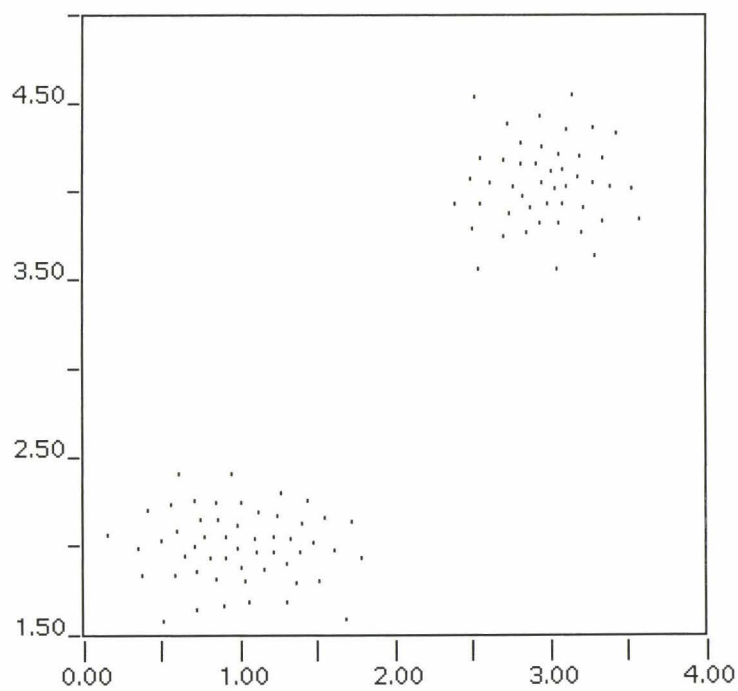


Figure II.9 : Représentation de l'échantillon des vecteurs poids après l'apprentissage

Algorithme de réduction par réseaux de neurones à apprentissage compétitif

1. Fixer le nombre de neurones égal au nombre d'observations désiré de l'échantillon réduit $\mathscr{W}$

2. Effectuer un apprentissage de l'échantillon à réduire $\mathscr{X}$ par l'algorithme apprentissage compétitif sensible à la fréquence.

3. Effectuer ensuite un apprentissage compétitif de l'échantillon à réduire .

Remarque :

L'utilisation de l'apprentissage compétitif sensible à la fréquence donne de bons résultats lorsque la réduction est très importante. Lorsque la réduction est moins importante, il est nécessaire d'augmenter le nombre d'époques afin que chaque neurone soit utilisé. Les apprentissages peuvent alors nécessiter un grand nombre d'époques. La figure II.10 représente l'évolution du pourcentage du nombre de neurones utilisés au cours des époques pour trois réseaux comportant 100, 400 et 900 neurones. Les trois apprentissages ont été effectués sur l'exemple 1. On constate que le premier réseau, comportant 100 neurones, atteint les 100% d'utilisation des neurones après 30 époques. Cela indique que tous les neurones ont été activés de telle sorte que leurs vecteurs poids sont déplacés vers les observations. Lorsque le nombre de neurones du réseau augmente, le pourcentage d'utilisation des neurones évolue moins rapidement en fonction des époques. On obtient ainsi pour un réseau comportant 900 neurones un pourcentage d'utilisation de 60% au bout de 1000 époques. L'utilisation de cette technique de réduction de données n'est donc intéressante que lorsque le taux de réduction est important.

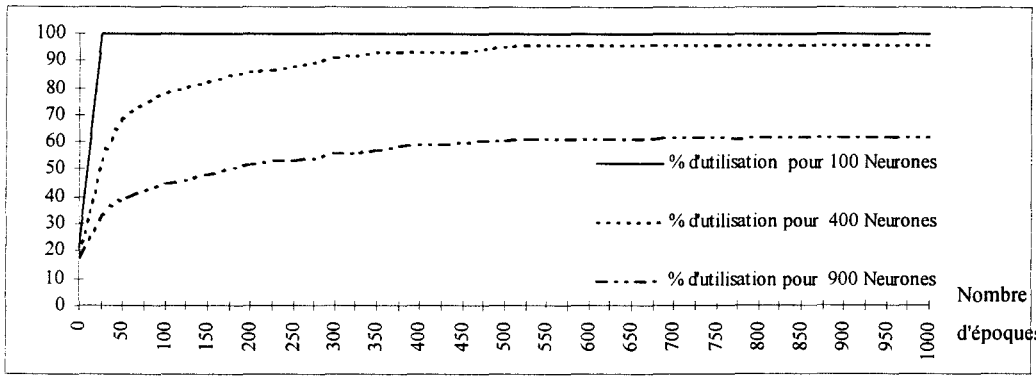


Figure II. 10 :Evolution du pourcentage d'utilisation des neurones en fonction du nombre d'époques

Nous avons utilisé cette technique de réduction de données multidimensionnelles et nous avons élaboré un critère quantitatif de distorsion basé sur le critère de divergence de Kullback [KULL 59] pour juger de la qualité de la réduction obtenue [DELS 93]. Le lecteur intéressé par ce travail peut se reporter à l'annexe I.

II.7 EVALUATION DES COUTS DE CALCUL DES DIFFERENTES TECHNIQUES D'APPRENTISSAGE

L'algorithme des K-means, comme les techniques d'apprentissage détaillées dans ce chapitre, se décomposent en deux étapes. Lors de la présentation d'une observation, la première étape consiste à rechercher le ou les neurones à adapter. Ainsi la loi d'activation détermine le neurone gagnant et éventuellement le neurone rival. La seconde étape consiste à adapter les vecteurs poids. Cette adaptation est déterminée par la règle d'apprentissage. La différence entre les techniques d'apprentissage se situe au niveau de ces deux étapes.

Souvent, on utilise les techniques d'apprentissage à partir d'une machine monoprocesseur. Dans ce cadre d'utilisation, ces techniques se différencient énormément en termes de coûts de calcul. Les tableaux II.4 et II.5 comptabilisent le nombre d'opérations élémentaires qu'exige chacune de ces techniques lors de la sélection du neurone gagnant et de l'adaptation des vecteurs poids.

Types d'opérations	K-Means	Apprentissage compétitif	Apprentissage compétitif sensible à la fréquence	Apprentissage compétitif avec pénalisation du rival	Apprentissage compétitif généralisé
comparaisons	2×Q×(K-1)	T×(K-1)	T×(K-1)	T×(K-1)	T×(K-1)
additions / soustractions	2×Q×(N-1) ×N×K	T×(N-1) ×N×K	T×(N-1)×N×K+ T	T×(N-1) ×N×K+ T	T×(N-1)×N×K
multiplications / divisions	2×N×Q×K	T×N ×K	T×(N×K+1)	T×(N×K+1)	T×Q×K

Tableau II. 3 : Coûts de calcul de la loi d'activation

Types d'opérations	K-Means	Apprentissage compétitif	Apprentissage compétitif sensible à la fréquence	Apprentissage compétitif avec pénalisation du rival	Apprnetissage compétitif généralisé
additions / soustractions	2×N×Q+(Q-K)	2×T×N	2×T×N	4×T×N	2×K×T×N
multiplications / divisions	N×Q+K	T×N	T×N	2×T×N	K×T×N

Tableau II. 4 : Coûts de calcul des règles d'apprentissage

Dans le tableau II.4, nous n'avons pas comptabilisé les opérations nécessaires au calcul des coefficients d'apprentissage.

Bien entendu, la technique d'apprentissage compétitif généralisé est la plus coûteuse en temps de calcul. Bien qu'elle tende à limiter le phénomène de sous-utilisation des neurones, on préférera utiliser l'apprentissage sensible à la fréquence suivi d'un apprentissage compétitif classique. Nous n'utilisons pas l'apprentissage compétitif avec pénalisation du rival car le réglage des coefficients d'apprentissage est très délicat.

II.8 CONCLUSION

L'algorithme des K-means a été repris sous un nouvel aspect stochastique sous le nom d'apprentissage compétitif.

Nous avons étudié 4 méthodes d'apprentissage compétitif récentes: l'apprentissage compétitif, l'apprentissage compétitif sensible à la fréquence, l'apprentissage compétitif avec

pénalisation du rival et l'apprentissage compétitif généralisé. Ces méthodes se différencient par la loi de sélection du neurone gagnant et par les règles de mise à jour des vecteurs poids des neurones.

Les méthodes d'apprentissage compétitif, d'apprentissage compétitif sensible à la fréquence et d'apprentissage compétitif avec pénalisation du rival introduisent des concepts de la nature : compétition, remords, conscience, rivalité, etc. L'apprentissage compétitif généralisé est plutôt proche des phénomènes physiques d'attraction gravitationnelle inversement proportionnels à la distance.

Nous avons exploré les capacités des réseaux de neurones à apprentissage compétitif dans les domaines de la classification automatique et de la réduction de données. Sous une même architecture et en modifiant quelque peu la loi d'activation et la règle d'apprentissage, nous pouvons obtenir de très bons résultats dans ces deux domaines. Nous disposons de plus des avantages bien connus des réseaux de neurones, à savoir leur mode de fonctionnement parallèle et leur robustesse. De plus, ils sont capables de traiter les échantillons très volumineux.

Une autre approche constituant une généralisation de l'apprentissage compétitif a été proposée par Kohonen. Celle-ci, inspirée de la modélisation de certaines zones du cortex, est plus riche que les précédentes et est détaillée dans le chapitre suivant.

CHAPITRE III :

PROJECTIONS ET REPRESENTATIONS DE DONNEES MULTIDIMENSIONNELLES PAR CARTES DE KOHONEN

III.1 INTRODUCTION

Dans le chapitre précédent, nous avons étudié les différentes règles d'apprentissage des réseaux de neurones à apprentissage compétitif récemment développées. Ceux-ci sont essentiellement adaptés à la réduction de données et à la classification automatique, où, en général, un neurone représente un groupement d'observations, ou classe, et son vecteur poids le centre du groupement ou de la classe.

Les cartes auto-adaptatives de Kohonen, présentées dans ce chapitre, constituent une généralisation des réseaux à apprentissage compétitif. Les neurones de sortie sont disposés les uns à côtés des autres, régulièrement espacés sur une grille. Ainsi chaque neurone est caractérisé par son vecteur poids dans l'espace de représentation des observations et, de plus, par sa position relative sur la grille. L'apprentissage du réseau consiste, à chaque présentation d'une observation, à sélectionner le neurone gagnant, autrement dit celui dont le vecteur poids est le plus proche de cette observation. Le vecteur poids du neurone gagnant et ceux des neurones voisins sur la grille sont alors modifiés en fonction de l'observation présentée au

réseau. Cette disposition des neurones et cette technique d'apprentissage permettent aux neurones voisins sur la grille d'être sensibles à des observations voisines dans l'espace des données : c'est le phénomène d'auto-organisation décrit par Kohonen [KOHO 82a].

A la fin de l'apprentissage, chaque neurone devient sensible à une zone de l'espace de représentation des observations et son vecteur poids converge vers le barycentre des observations présentes dans cette zone. Ainsi nous pouvons utiliser, de la même manière que dans le chapitre II, les vecteurs poids à des fins de classification et de réduction de données. De plus, deux neurones voisins sur la grille deviennent réceptifs à deux zones voisines de l'espace de représentation des observations. La grille de neurones constitue en quelque sorte une image plane des données multidimensionnelles que l'on appelle « carte ». Nous représentons chaque neurone par un pixel d'une image dont le niveau de gris reflète des propriétés de l'ensemble des distances relatives séparant son vecteur poids de ceux des neurones voisins sur la grille. Nous obtenons ainsi une « image » qui constitue une représentation des données multidimensionnelles. Nous utiliserons cette représentation dans le chapitre suivant afin de découvrir la structure d'échantillons multidimensionnels. Ces techniques de représentation constituent en effet l'un des éléments essentiels du processus de classification interactive faisant l'objet du chapitre IV.

Nous présentons maintenant l'architecture des cartes de Kohonen et ses lois d'apprentissage. Nous exploitons ensuite la faculté d'auto-organisation du réseau comme outil de projection non linéaire sur un plan. Différentes techniques de visualisation d'échantillons d'observations multidimensionnelles sont enfin détaillées et leurs performances sont évaluées sur des exemples artificiels.

III.2 CARTES TOPOLOGIQUES AUTO-ADAPTATIVES DE KOHONEN (CK)

III.2.1 Fondements biologiques du réseau de Kohonen

Il a été établi que le cerveau est organisé selon différentes zones associées au traitement de tâches spécifiques. Par exemple, dans le cerveau humain nous connaissons parfaitement les zones où se localisent les neurones traitant les informations sensorielles. Or, dans certaines de ces zones, les cellules nerveuses sont organisées en fonction de

l'information traitée. Ainsi on constate que les zones réceptrices du cortex visuel sont ordonnées selon un ordre identique à celui des unités de l'organe sensoriel lui-même : deux zones proches dans le cortex visuel sont également proches dans la rétine. La figure III.1 nous montre quelques unes de ces zones sensorielles.

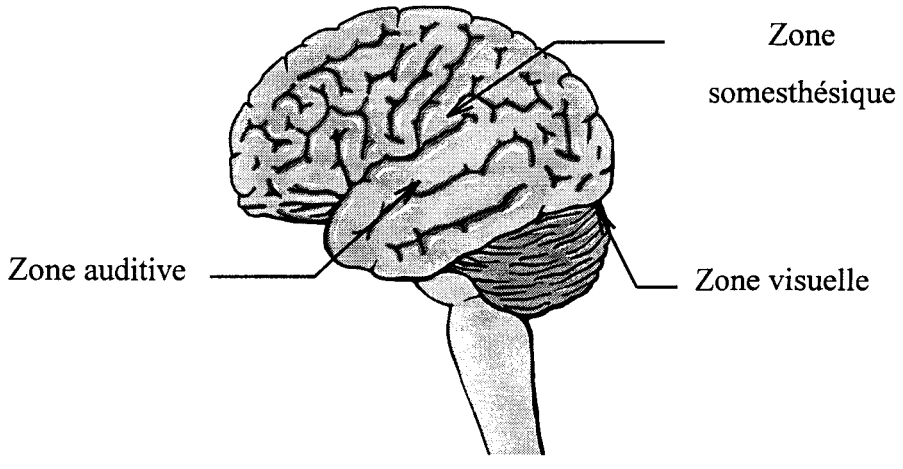


Figure III.1 :Zones sensorielles du cortex

Par ailleurs, une cellule excitée agit sur les cellules voisines dans un rayon compris entre 50 et 100 µm et inhibe les cellules dans un rayon compris entre 200 et 500 µm. Cette interaction latérale a été modélisée par la fonction dite du chapeau mexicain [KOHO 88a](cf. Figure III.2) :

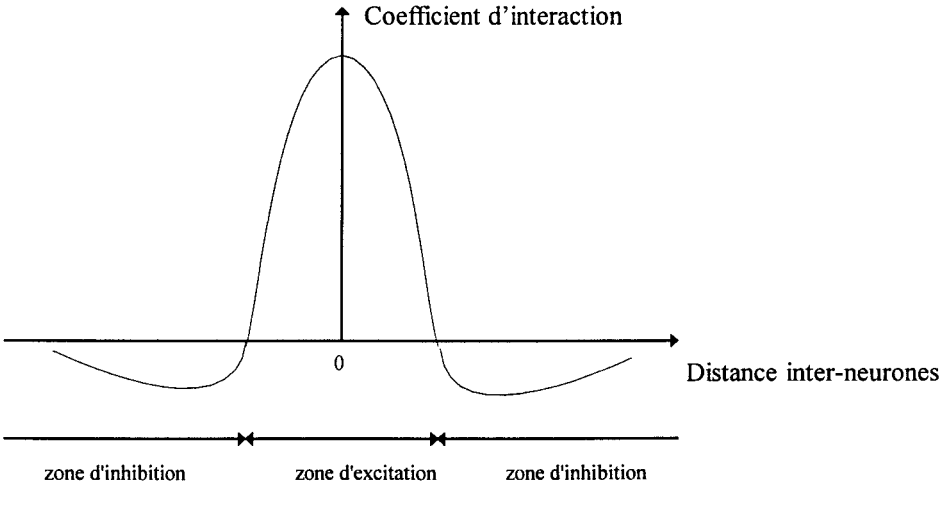


Figure III. 2 : Fonction d'interaction latérale à une dimension du type chapeau mexicain

Ainsi on constate que des cellules très proches localement dans le cortex auditif de la chauve-souris manifestent un regain d'activité lors d'émissions de sons de fréquences proches. L'étude de ces cellules a aussi révélé la présence d'interactions sur les neurones voisins lorsqu'un neurone est excité par un stimulus extérieur.

Kohonen s'inspirant de l'organisation spatiale du cortex a conçu un réseau de neurones connu sous le nom de carte auto-organisatrice de Kohonen. En utilisant des règles d'apprentissage similaires aux lois d'interaction du cortex, il a pu mettre en évidence le même phénomène d'auto-organisation [RITT 92], [KOHO 93]. Ce réseau de Kohonen a été appliqué avec succès dans différents domaines, tels que la reconnaissance de la parole, la robotique [RITT 92], la compression d'images [AHAL 90] ou le traitement d'images [SNYD 91].

III.2.3 Architecture du réseau de Kohonen

L'architecture du réseau de Kohonen diffère de celle des réseaux à apprentissage compétitif dans l'organisation des neurones à l'intérieur de la couche de sortie (cf. Figure III.3).

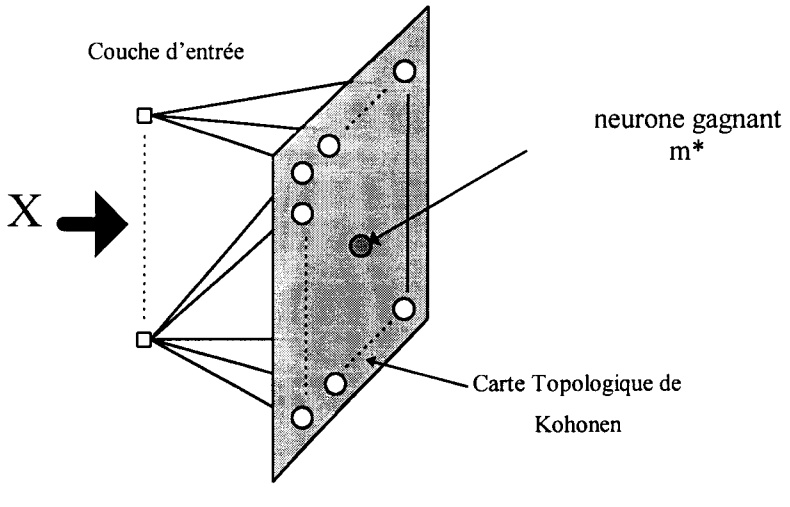


Figure III.3 : Carte de Kohonen

Kohonen prend en compte, dans sa modélisation, la position physique des neurones à l'intérieur du réseau. De plus, il définit différents types d'organisations topologiques des neurones au sein d'une couche. Ainsi, il distingue des couches compétitives de différents

ordres topologiques.

Ordre topologique 0 : Cette architecture correspond aux réseaux à apprentissage compétitif. Il n'y a pas d'ordonnancement des neurones.

Ordre topologique 1 : Les neurones sont disposés les uns à côté des autres uniformément sur un axe. Chaque neurone a un indice de position sur l'axe et possède au maximum deux voisins immédiats (cf. Figure III.4).

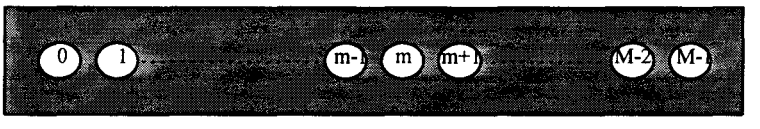


Figure III. 4 : Représentation de la carte de Kohonen d'ordre 1

Ordre topologique 2 : Les neurones sont ordonnancés selon une grille sur un plan. A chaque neurone est associé un couple d'indices de position sur la grille (cf. Figure III.5).

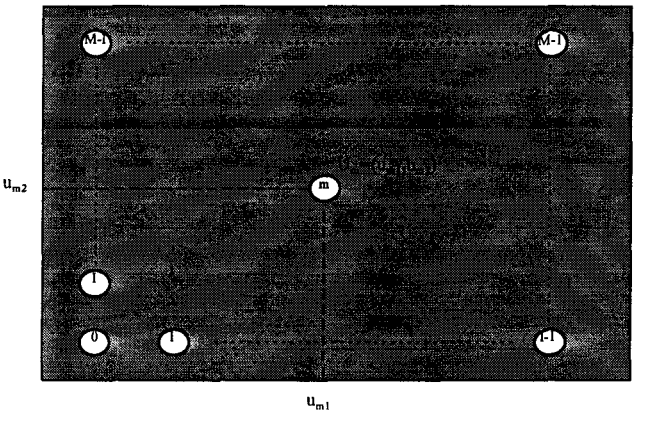


Figure III. 5 : Représentation de la carte de Kohonen d'ordre 2

Ordre topologique L : Les neurones sont uniformément répartis à l'intérieur d'un hypercube de dimension L. A chaque neurone est associé un L_uplé d'indices de position.

III.2.4 Relations indicielles entre neurones

Pour des raisons pratiques de visualisation plane, l'étude de ces réseaux se limitera à l'ordre topologique 2. Dans ce cas, les neurones sont uniformément répartis sur une grille plane. Grâce à ces relations d'ordre, un neurone est caractérisé par sa position sur la grille et son vecteur poids dans l'espace de représentation des observations. On peut définir des relations de voisinage entre neurones de manière similaire à celles se produisant à l'intérieur du cortex. Notons que les positions des neurones sur la carte sont fixes et ne varient ni au

cours de l'apprentissage, ni au cours de la phase d'exploitation.

Soient I et J, les nombres de neurones sur chaque axe de la grille. Le nombre total de neurones, noté M, s'exprime alors par : M=I×J.

Dans une carte de Kohonen, on note $U_m=(i,j)^T$ la position du neurone m, $0 \leq m < M$, dont les coordonnées sont i et j sur le plan et $U_{m^*}=(i^*,j^*)^T$ la position du neurone gagnant.

La répartition des neurones sur un plan peut se faire soit suivant un maillage carré, soit suivant un maillage hexagonal (cf. Figure III.6).

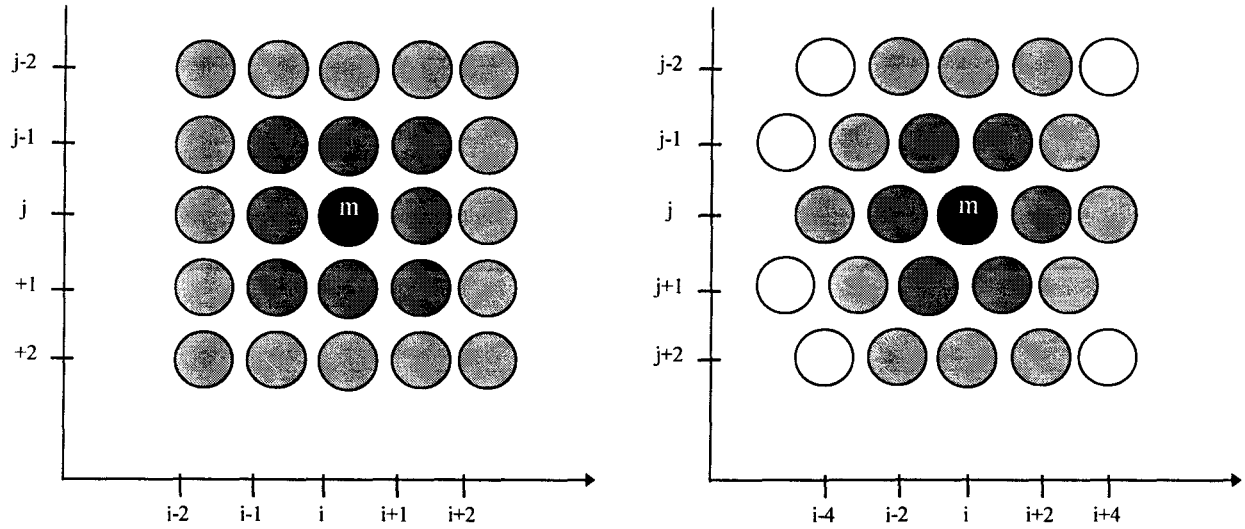


Figure III. 6 : Indexation des neurones dans les maillages carré et hexagonal d'un réseau de Kohonen d'ordre topologique 2

En fonction du maillage, on peut déterminer pour chaque neurone d'indice m sa position U_m sur la grille (cf. Tableau III.1). Dans ce tableau, « mod » désigne l'opérateur modulo et « div » désigne l'opérateur de la division entière. Le neurone d'indice 0 est positionné à l'origine sur la carte. Ainsi sa position U_0 est définie par : $U_0 = (0,0)^T$.

Nous désignons par le terme neurone m le couple de vecteurs (U_m,W_m).

Maillage carré	Maillage hexagonal
m=i+j*I	m=((i - (j mod 2)) div 2) + j*I - (j mod 2)
$\begin{cases} i = m \bmod I \\ \text{et} \\ j = m \operatorname{div} I \end{cases}$	$\begin{cases} i = 2*(m \bmod I)+(j \bmod 2) \\ \text{et} \\ j = (m \operatorname{div} I) \end{cases}$

Tableau III.1 : Relations entre l'indice d'un neurone et sa position sur la grille

III.2.5 Notion de voisinage

On définit un voisinage V(m,r) de rayon r autour d'un neurone m par l'ensemble des neurones m' tel que :

$$V(m,r) = \left\{ m' \in [0, M[, m' \neq m \quad / \quad d^A(U_m, U_{m'}) \leq r \right\}$$

La forme du voisinage est déterminée par la métrique d^A employée. Pour un espace à deux dimensions, la métrique de Minkowski s'exprime par :

$$d^A(U_m, U_{m'}) = \left(|i - i'|^\beta + |j - j'|^\beta \right)^{\frac{1}{\beta}} \quad \text{avec } \beta \geq 1.$$

Selon la valeur de β, on obtient différents types de distances.

- Distance de Manhattan ou triangulaire :

La distance de Manhattan, ou distance triangulaire, correspond à la distance de Minkowski avec β=1 :

$$d^A(U_m, U_{m'}) = |i - i'| + |j - j'|$$

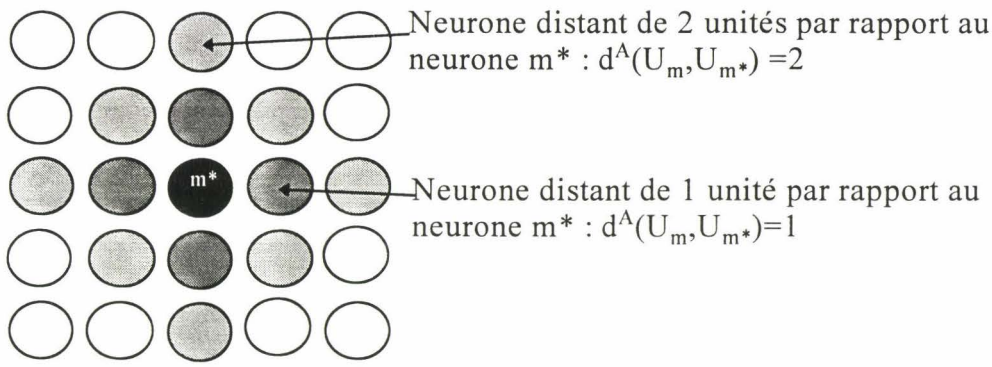


Figure III. 7 : Voisinage en carré avec la distance triangulaire

Lorsque l'on dispose les neurones suivant un maillage carré, cette métrique crée des voisinages en forme de carrés. Chaque neurone a alors au plus quatre voisins immédiats, comme le montre la Figure III.7.

- Distance Euclidienne ou sphérique :

La distance Euclidienne, ou distance sphérique, correspond à la distance de

Minkowski avec $\beta=2$:

$$d^A(U_m, U_{m'}) = \sqrt{(i-i')^2 + (j-j')^2}$$

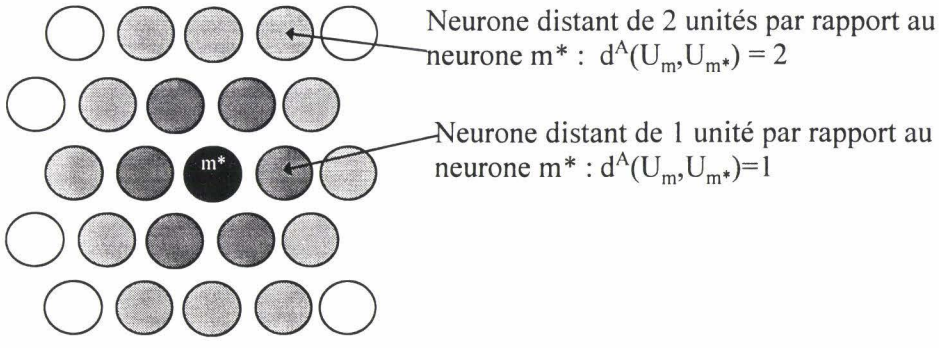


Figure III. 8 : Voisinage hexagonal

Lorsque l'on dispose les neurones suivant un maillage hexagonal, cette métrique crée des voisinages en forme d'hexagones. Chaque neurone a alors au plus six voisins immédiats, comme le montre la Figure III.8.

- Sup distance ou carrée :

La sup distance, ou distance carrée, correspond à la distance de Minkowski lorsque β tend vers l'infini. Elle s'exprime sous la forme :

$$d^A(U_m, U_{m'}) = \max(|i-i'|, |j-j'|)$$

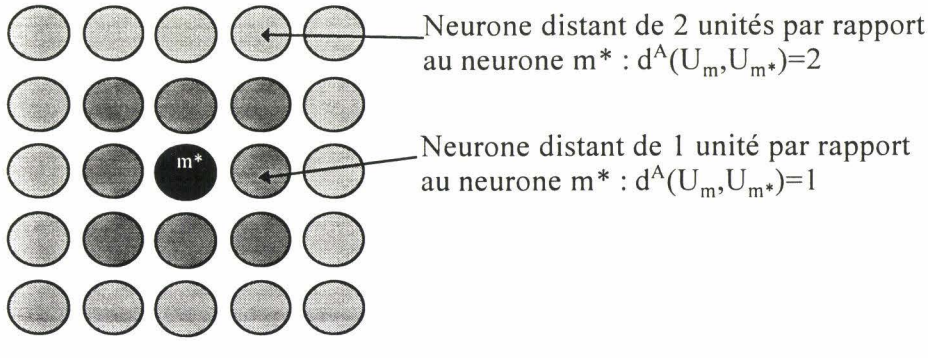


Figure III. 9 : Voisinage carré

Lorsque l'on dispose les neurones suivant un maillage carré, cette métrique crée des voisinages en forme de carrés. Chaque neurone a alors au plus huit voisins

immédiats, comme le montre la Figure III.9.

Dans la suite de ce travail, nous utilisons des réseaux à maillage hexagonal avec la distance sphérique et des réseaux à maillage carré avec la distance triangulaire et la distance carrée.

III.2.6 Lois d'apprentissage

L'apprentissage des cartes de Kohonen est similaire à l'apprentissage compétitif. A chaque présentation d'une observation X(t), on calcule pour chaque neurone m, la distance $d_m(t)$ entre le vecteur poids $W_m(t-1)$ et cette observation. Le neurone m* le plus proche est déclaré gagnant si :

$$m^* = \arg \min_{0 \le m < M} (d_m(t))$$

avec : $d_m(t) = (X(t)-W_m(t-1))^T(X(t)-W_m(t-1))$

La modélisation des interactions entre neurones consiste en une modification de la loi d'apprentissage compétitif en adaptant non seulement le neurone gagnant mais aussi ceux se trouvant dans son voisinage. Les neurones hors de son voisinage ne sont pas adaptés (cf. Figure III.10).

Ce mécanisme d'adaptation peut être schématisé sous la forme :

(i) $W_m(t)=W_m(t-1)+\alpha(t)(X(t)-W_m(t-1))$ si $m=m^*$

(ii) $W_m(t)=W_m(t-1)+\alpha(t)h_m(m^*,t)(X(t)-W_m(t-1))$ si $m \in V(m^*,r(t))$

(iii) $W_m(t)=W_m(t-1)$ si $m \notin V(m^*,r(t))$ et $m \neq m^*$

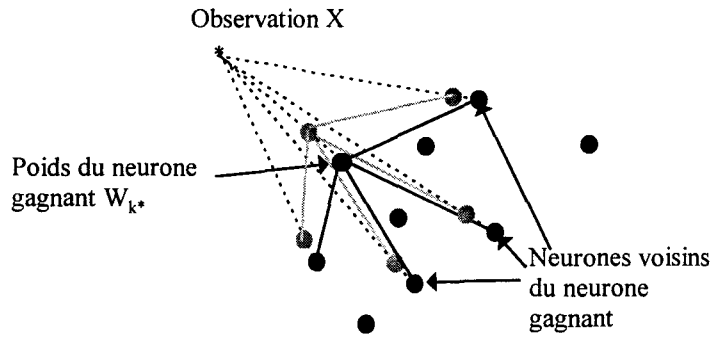


Figure III. 10: Mécanisme d'adaptation des poids dans l'algorithme d'apprentissage de Kohonen

ε est une constante positive vérifiant : $0<\varepsilon<<1$.

On pose, de plus, $r(0)=r_0$. En général, on fixe r_0 à la moitié de la taille maximale de la grille de neurones. Dans ce cas, r_0 se met sous la forme :

$$r_0 = \frac{1}{2} \text{Max}(I,J).$$

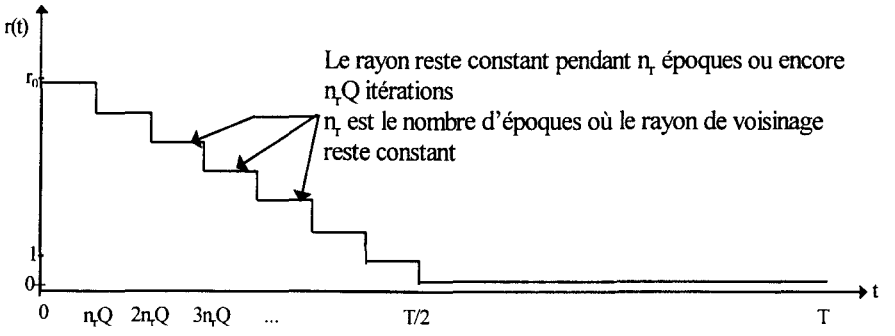


Figure III. 11 : Evolution de r en fonction de la variable d'itération t

On peut remarquer deux phases dans le processus d'apprentissage. Dans la première phase, les vecteurs poids subissent de nombreuses modifications. L'orientation, la direction et la norme d'un même vecteur peuvent être très différentes d'une itération à l'autre. C'est la phase d'auto-organisation. Dans la seconde phase, plus lente et plus fine que la phase précédente, les vecteurs poids des neurones convergent vers les barycentres des observations des zones d'influence des neurones.

L'expérimentation de ces réseaux a donné des résultats d'auto-organisation comparables à ceux observés sur les cellules du cerveau: deux neurones voisins sur la carte ont leurs vecteurs poids proches dans l'espace de représentation des observations [KOHO 90],[RITT 92],[DAYH 90].

Etant donnée la complexité de l'algorithme, la propriété d'auto-organisation ainsi que la convergence des poids suivant la densité de probabilité de l'échantillon n'ont été démontrées rigoureusement que dans le cas d'un échantillon de dimension 1 et pour un réseau d'ordre topologique 1 [COTT 87].

Rappelons que V(m*,r(t)) est l'ensemble des neurones voisins du neurone gagnant m*. En général, on choisit un grand rayon de voisinage au début de l'apprentissage et on le fait décroître au cours des époques.

On peut remarquer que les équations (i) et (iii) expriment une règle d'apprentissage compétitif classique. Kohonen a ajouté la règle (ii) qui permet aux neurones voisins du neurone gagnant d'être aussi adaptés. Ceci est exprimé par la fonction h_m, appelée loi d'interaction, qui dépend du rayon de voisinage r(t). Parmi les fonctions les plus utilisées, on distingue :

- la fonction cubique :

$$h_m(m^*,t) = \begin{cases} 1 & \text{si} \quad d^A(U_m, U_{m^*}) \le r(t) \\ 0 & \text{sinon} \end{cases}$$

- la fonction gaussienne :

$$h_m(m^*,t) = \exp\left(-\frac{d^A(U_m, U_{m^*})^2}{2r(t)^2}\right) .$$

Rappelons que $d^A(U_m,U_{m'})$ est la distance séparant les neurones m et m' sur la carte. r(t), le rayon de voisinage ou d'interaction, dépend du rang t de l'itération considérée . Une technique simple consiste à diminuer le rayon linéairement selon les époques. Dans ce cas, ce rayon diminue toutes les $n_r \times Q$ itérations, où n_r est le nombre d'époques à rayon constant et où Q est le nombre d'observations de l'échantillon (cf. Figure III.11). Le rayon r(t) s'exprime alors par :

$$r(t) = \begin{cases} r(t-1)-1 & \text{si} \quad t \bmod(n_r Q) = 0 \quad \text{et} \quad r(t) > 1 \\ \varepsilon & \text{si} \quad t \bmod(n_r Q) = 0 \quad \text{et} \quad r(t) \le 1 \\ r(t) & \text{sinon} \end{cases}$$

où :

Q est le nombre d'itérations par époques.

n_r est le nombre d'époques à rayon constant.

Algorithme d'apprentissage compétitif de Kohonen

1. Initialisation aléatoire des M vecteurs poids,

Initialisation du rayon d'interaction : $r_0 = \frac{1}{2} Max(I,J)$,

Rang d'itération t=1.

2. Tirage d'une observation X(t) dans l'échantillon $\mathcal{X}$.

3. Calcul pour chaque neurone de :

$d_m(t) = (X(t)-W_m(t-1))^T (X(t)-W_m(t-1))$

4. Activation du neurone m tel que :*

$$m^* = \underset{0 \le m < M}{Arg\ min}\left(d_m(t)\right)$$

$y_m = 1$ *si* $m = m^*$

$y_m = 0$ *si* $m \neq m^*$

5. Modification des vecteurs poids

Calcul de $\alpha(t)$, *de r(t) et des* $h_m(m^*,t)$

Adaptation des vecteurs poids :

$W_m(t) = W_m(t-1) + \alpha(t) \cdot (X(t)-W_m(t-1))$ *si* $m = m^*$

$W_m(t) = W_m(t-1) + \alpha(t).h_m(m^*,t).(X(t)-W_m(t-1))$ *si* $m \in V(m^*,r(t))$

$W_m(t) = W_m(t-1)$ *si* $m \notin V(m^*,r(t))$ *et* $m \neq m^*$

6. Test du critère d'arrêt,

Si la condition est vérifiée : fin de l'apprentissage.

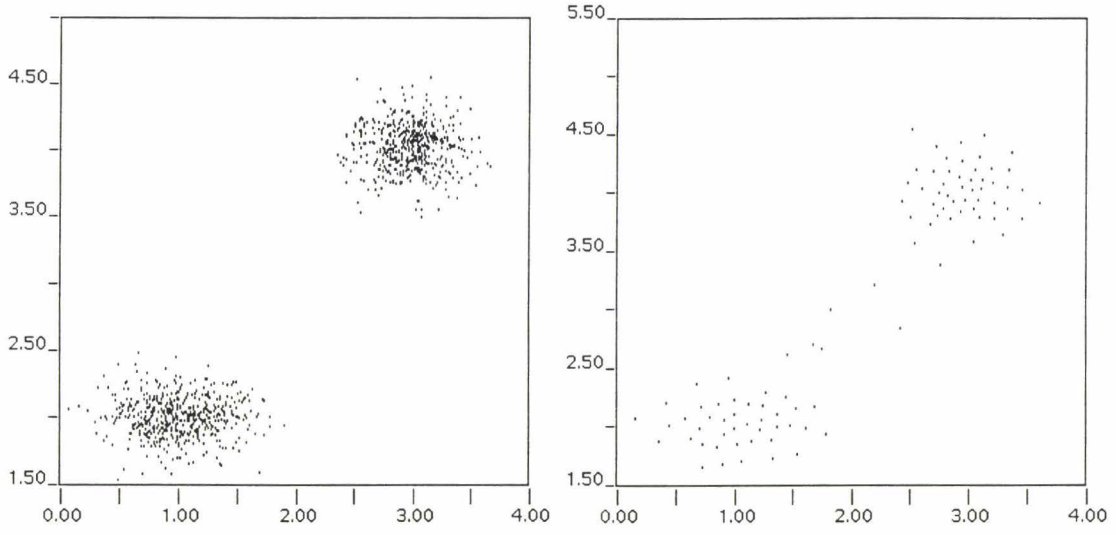
Sinon reprise du procédé à partir de l'étape 2 avec t=t+1.

III.2.7 Application des réseaux de Kohonen à la réduction de données

L'utilisation des relations de voisinage lors de la phase d'apprentissage des réseaux de Kohonen permet de réduire l'effet de l'initialisation des poids et de limiter le phénomène de sous-utilisation des neurones. Ainsi comme pour l'apprentissage compétitif sensible à la fréquence (cf §II.5.1), nous pouvons utiliser ces réseaux en classification et en réduction de données (cf. § II.6).

La figure III.12(a) représente l'échantillon de l'exemple I du chapitre précédent. Sur la figure III.12(b) sont représentés les vecteurs poids d'un réseau de 100 neurones après un

apprentissage compétitif de Kohonen de 200 époques. Le rayon est initialement fixé à 5 et décroît toutes les 20 époques. Au bout de 100 époques, le rayon est nul et l'apprentissage du réseau est poursuivi sur le mode compétitif classique. Nous pouvons remarquer sur la figure III.12(b) que certains vecteurs poids ont convergé entre les deux classes de l'échantillon alors qu'il n'y a aucune observation en cet endroit de l'espace : ce sont les vecteurs poids de neurones sous-utilisés. Nous pouvons constater que les résultats obtenus en réduisant l'échantillon par un apprentissage compétitif sensible à la fréquence (cf. Figure II.9) sont meilleurs que ceux obtenu par l'intermédiaire d'un apprentissage compétitif de Kohonen.



- (a) Echantillon d'origine - - (b) Echantillon réduit -

Figure III.12 : Réduction de données par l'apprentissage compétitif de Kohonen

Cependant, bien qu'il n'élimine pas en totalité la sous-utilisation des neurones, l' apprentissage compétitif de Kohonen donne de meilleurs résultats que ceux obtenus avec un apprentissage compétitif sensible à la fréquence, surtout lorsque le taux de réduction est peu élevé. La figure III.13(a) présente les vecteurs poids d'une carte de Kohonen de 20*20 neurones après un apprentissage compétitif de Kohonen de 200 époques. Le rayon est initialisé à 10 et décroît toutes les 10 époques. La figure III.13(b) représente les vecteurs poids après un apprentissage compétitif sensible à la fréquence de 100 époques suivi d'un apprentissage compétitif classique de 100 époques. Ces deux réseaux ont « appris » le même exemple I. Nous pouvons constater, en comparant les figures III.13(a) et III.13(b), que l'apprentissage compétitif de Kohonen, pour un même nombre d'itérations, génère moins de neurones sous-utilisés que l'apprentissage compétitif sensible à la fréquence. De plus, les

neurones sous-utilisés dans le réseau utilisant l'apprentissage compétitif de Kohonen se situent entre les zones denses en observations. La majorité des vecteurs poids des neurones sous-utilisés dans le réseau utilisant l'apprentissage compétitif sensible à la fréquence n'ont jamais été modifiés et restent donc aux positions fixées lors de l'initialisation de l'algorithme d'apprentissage.

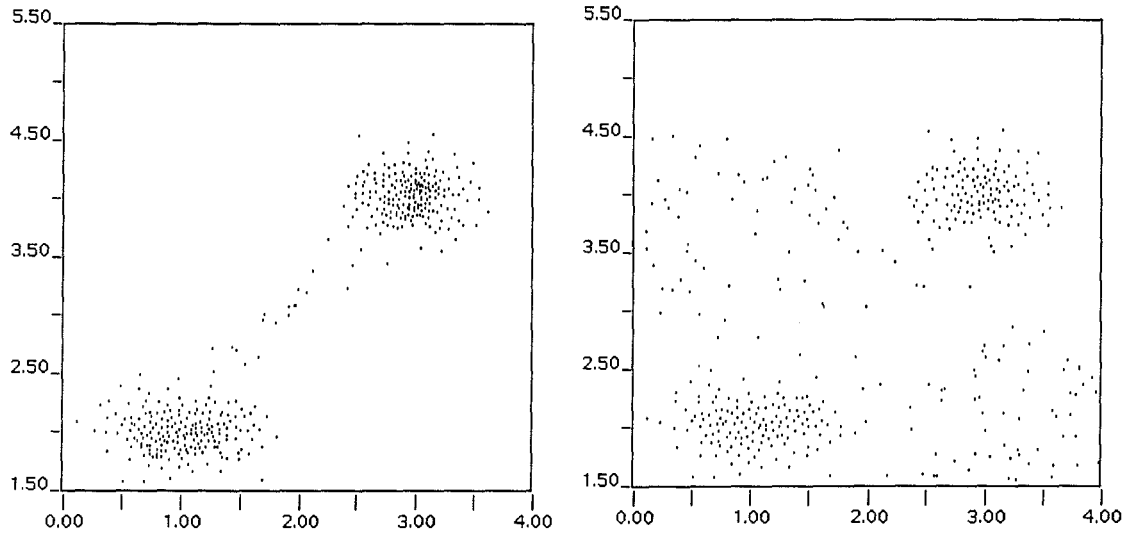


Figure III. 13 : Réduction de données par apprentissage compétitif de Kohonen (a) et par apprentissage compétitif sensible à la fréquence (b)

La combinaison des différents apprentissages permet d'améliorer la réduction de données. On peut ainsi successivement et dans cet ordre :

- Faire « tourner » l'apprentissage de Kohonen quelques époques pour faire converger rapidement et grossièrement les poids des neurones vers des zones à forte densité d'observations.

- Appliquer l'apprentissage sensible à la fréquence pour réduire, voire éliminer, la sous-utilisation des neurones « dead units » subsistant après l'apprentissage de Kohonen.

- Appliquer l'apprentissage compétitif pour assurer une convergence des poids vers les centres des observations se trouvant dans les zones réceptrices des neurones.

La figure III.14 représente la réduction de données en combinant les différents

algorithmes d'apprentissage. Ainsi, suite à l'apprentissage par apprentissage compétitif de Kohonen, dont les poids sont représentés sur la figure III.13(a), nous avons effectué un apprentissage par apprentissage compétitif sensible à la fréquence pendant 100 époques puis un apprentissage compétitif classique pendant 100 autres époques. Bien qu'à la suite de l'apprentissage compétitif de Kohonen, il reste des neurones sous-utilisés, les vecteurs poids de ces neurones se trouvent rapprochés des zones denses en observations. La convergence de ces vecteurs poids se trouve alors facilité avec l'utilisation d'un apprentissage compétitif sensible à la fréquence. Nous pouvons constater sur la figure III.14 qu'il n'existe plus d'observation entre les classes. Il n'y a plus de neurones sous-utilisés. Rappelons que, dans ce cas, l'échantillon réduit comporte 400 vecteurs poids et donc que le taux de réduction est moins important que dans l'exemple abordé dans le paragraphe II.6.

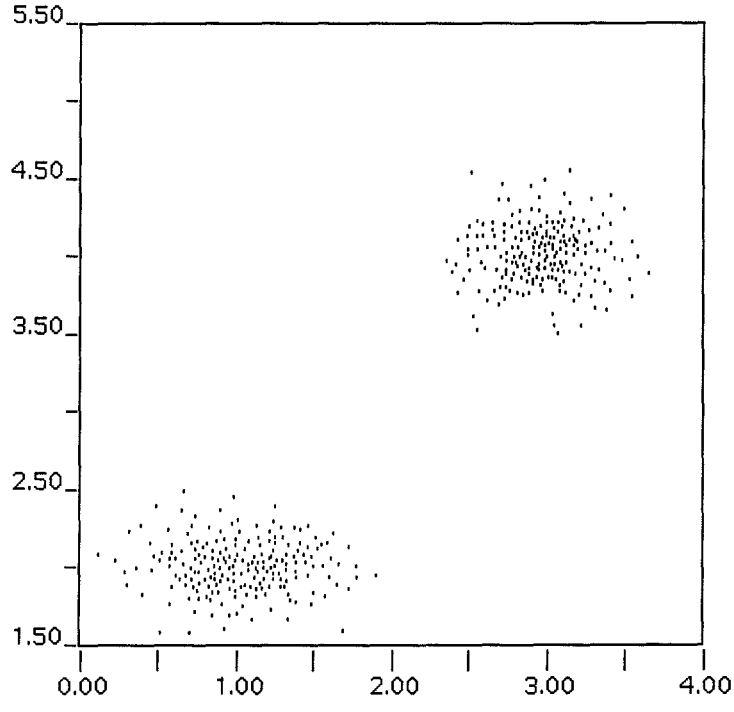


Figure III. 14 : Réduction de données en combinant les algorithmes d'apprentissage compétitif, de Kohonen, sensible à la fréquence et classique

III.2.8 Problèmes liés à la loi d'apprentissage de Kohonen

L'un des inconvénients de l'apprentissage de Kohonen est le nombre de paramètres supplémentaires à initialiser par rapport à l'apprentissage compétitif classique. On doit ainsi définir : *le type d'interaction, le type de distance* d^A *et la loi d'évolution du rayon d'interaction en fonction de la variable d'itération.* Ces paramètres de réglage sont laissés au

libre choix de l'analyste.

Le temps de calcul est aussi un inconvénient majeur quand les simulations sont réalisées sur une machine séquentielle. En effet, la taille du réseau et le nombre de voisins accroissent considérablement les temps de calcul. Le nombre de voisins dépend directement du type de distance d^A utilisé. Le tableau III.2 représente le nombre de neurones modifiés lors de la phase d'apprentissage pour un rayon d'interaction r.

Type de distance	Triangulaire	Sphérique	Carrée
Nombre de voisins	$2\times(r^2+r)$	$3\times(r^2+r)$	$4\times(r^2+r)$

Tableau III.2 : Nombre de voisins selon le type de distance

Bien entendu, ce problème disparaît dans le cas d'une implantation parallèle de ces réseaux.

Enfin, comme nous l'avons observé dans le paragraphe précédent, l'apprentissage de Kohonen ne résout que partiellement le problème de la sous-utilisation des neurones.

III.3 VISUALISATION D'UN ECHANTILLON ETIQUETE PAR CARTE DE KOHONEN

On définit la projection obtenue par le réseau comme une fonction non linéaire de l'espace de représentation des observations multidimensionnelles $\mathbb{R}^N$ sur l'espace discret $\mathbb{N}^2$:

$$\Phi_{CK} : \mathbb{R}^N \longrightarrow \mathbb{N}^2$$
$$X \longrightarrow U_{m^*}=(i^*,j^*)^T$$

où le neurone m* est le plus proche de X :

$$m^* = \arg \min_{0\le m<M}\left((X-W_m)^T(X-W_m)\right) .$$

A chaque observation, on associe le vecteur U_{m^*} déterminant la position du neurone dont le vecteur poids est le plus proche de l'observation. Autrement dit, U_{m^*} est la position du neurone gagnant lorsque l'on présente l'observation à l'entrée du réseau (cf. Figure III.15). Les composantes de U_{m^*} sont des entiers naturels.

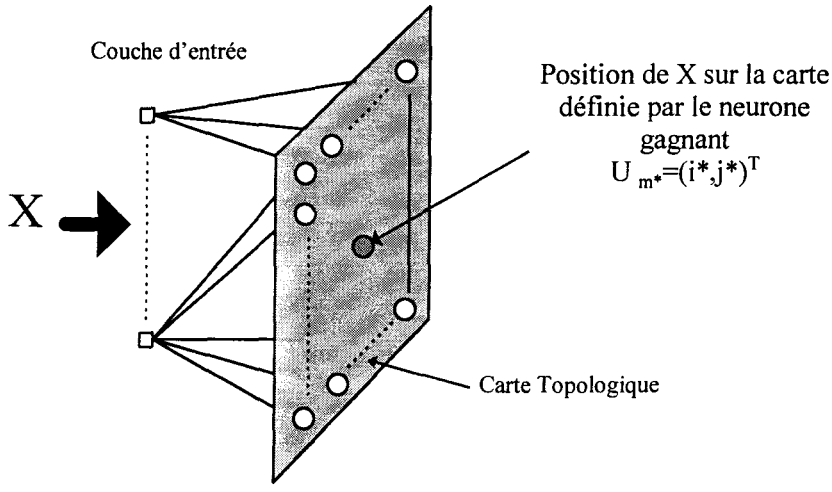


Figure III. 15 : Représentation d'une observation sur une carte

A la fin de l'apprentissage, les neurones sont organisés de telle sorte que deux neurones voisins sur la carte sont réceptifs à des observations voisines dans l'espace d'origine.

En phase d'exploitation, nous représentons chaque neurone par un pixel sur l'écran, la carte devient une image discrète de l'espace multidimensionnel.

Exemple 2 :

Pour illustrer ce type de projection, considérons un échantillon d'observations dans un espace à trois dimensions constitué de 4 classes gaussiennes générées artificiellement et dont les caractéristiques statistiques sont résumées dans le tableau III.3. La figure III.16 représente cet échantillon en perspective, chaque point représente une observation.

Rappelons qu'à chaque observation est associé un numéro de classe. Ce numéro est encore appelé étiquette. Nous appelons les échantillons, dont les observations possèdent une étiquette, échantillons étiquetés. Nous verrons, dans le chapitre suivant, que nous assignerons à des classes les observations des échantillons analysés dans un contexte non supervisé à l'aide d'algorithmes de classification. Les échantillons obtenus après classification sont appelés échantillons classés. Nous utilisons un échantillon étiqueté dans ce chapitre afin de tester la faculté d'auto-organisation des cartes de Kohonen. Dans le chapitre suivant, l'analyse non supervisé d'échantillons étiquetés va permettre de tester la qualité de la classification en calculant notamment le taux d'erreur de classement. L'exemple 2 est constitué de quatre classes. Chaque observation de cet échantillon a une étiquette comprise entre 0 et 3.

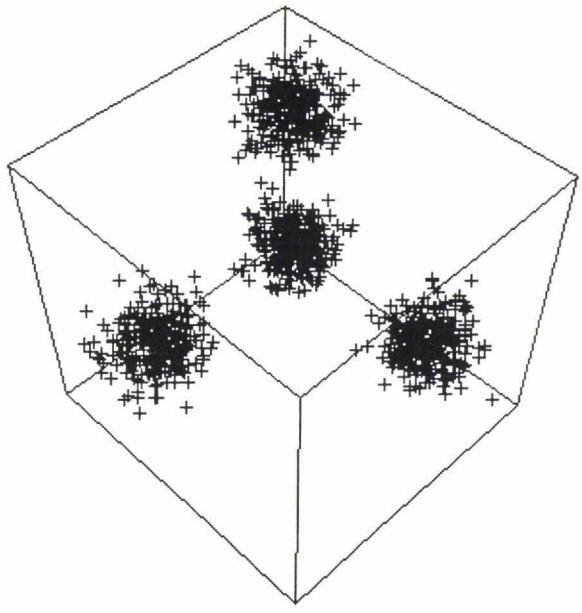


Figure III. 16: Vue de l'échantillon de l'exemple 2 en perspective

Classe	Nombre d'observations	Vecteur Moyenne	Matrice de covariance
Classe 0	300	$\begin{pmatrix} 0,0 \\ 0,02 \\ -0,26 \end{pmatrix}$	$\begin{pmatrix} 0,49 & 0 & 0,03 \\ 0 & 0,48 & 0 \\ 0,03 & 0 & 0,54 \end{pmatrix}$
Classe 1	300	$\begin{pmatrix} 6,02 \\ 0,0 \\ 0,0 \end{pmatrix}$	$\begin{pmatrix} 0,58 & -0,01 & -0,01 \\ -0,01 & 0,46 & -0,03 \\ 0,03 & -0,03 & 0,54 \end{pmatrix}$
Classe 2	300	$\begin{pmatrix} -0,04 \\ -0,05 \\ 5,95 \end{pmatrix}$	$\begin{pmatrix} 0,44 & 0,19 & 0 \\ 0,19 & 0,57 & 0,05 \\ 0 & 0,05 & 0,53 \end{pmatrix}$
Classe 3	300	$\begin{pmatrix} 0,05 \\ 5,96 \\ 0,02 \end{pmatrix}$	$\begin{pmatrix} 0,46 & -0,02 & -0,04 \\ -0,02 & 0,48 & 0,05 \\ -0,4 & 0,02 & 0,57 \end{pmatrix}$

Tableau III.3 : Paramètres statistiques de l'exemple 2

La projection d'un échantillon par l'intermédiaire du réseau CK suppose que l'on effectue au préalable un apprentissage de la carte avec cet échantillon. Pour cet exemple, nous utilisons un réseau de 50*50 neurones. Le rayon de voisinage est initialisé à 25 et décroît toutes les 20 époques. Nous utilisons donc des relations de voisinage durant les 500 premières époques. Ensuite, nous effectuons un apprentissage compétitif classique pendant 500 époques

ce qui porte le nombre total d'époques à 1000. Nous utilisons la fonction d'interaction cubique ainsi que la distance carrée pour déterminer les neurones voisins sur la carte. Le coefficient d'apprentissage est fixé à 0.5 et décroît linéairement vers 0 au cours de l'apprentissage.

La figure III.17 représente la projection de l'échantillon sur la carte.

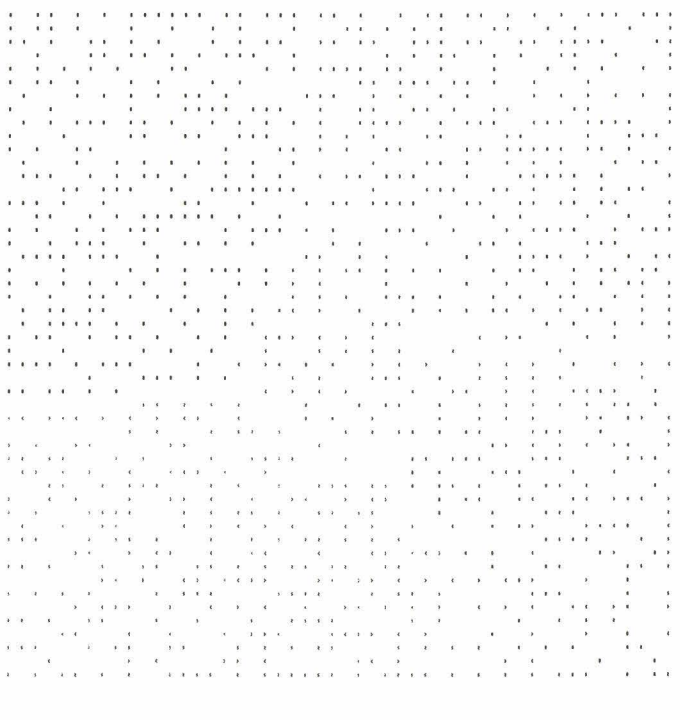


Figure III. 17 : Projection d'un échantillon par l'intermédiaire d'une CK

Remarquons qu'à chaque observation est associé un neurone unique. Par contre, certains neurones représentent une ou plusieurs observations du fait que leurs vecteurs poids se trouvent dans une zone dense de l'espace de départ, et que d'autres ne représentent aucune observation.

Pour mettre en évidence le processus d'auto-organisation des cartes de Kohonen, on étiquette chaque neurone par le numéro de la classe d'où provient l'observation l'activant. Ainsi, si une observation X appartient à la classe 0, on met un 0 à la position du neurone qui lui est réceptif.

Par ailleurs, il peut arriver que des observations appartenant à des classes différentes activent le même neurone. Ainsi, par exemple, considérons qu'un neurone soit activé par deux observations de la classe 0 et par trois observations de la classe 1. Dans ce cas, la classe la plus représentative est la classe 1. Ainsi nous étiquetons le neurone suivant le label de la

classe activant le plus souvent ce neurone, c'est à dire la classe 1. Lorsqu'il n'y a pas de classe majoritaire, nous ne représentons pas le neurone.

Nous rappelons que C_k est l'ensemble des observations appartenant à la classe k. Nous posons $C^E_{m^*}$ l'ensemble des observations de l'échantillon $\mathcal{X}$ activant le neurone m*, $C^E_{m^*}$ s'exprime par :

$$C^E_{m^*} = \left\{ X \in \mathcal{X} \,/\, m^* = \arg \max_{0 \le m < M} \left((X - W_m)^T (X - W_m) \right) \right\}$$

Nous obtenons une projection de l'échantillon étiqueté en affichant l'étiquette k* à la position U_{m^*} du neurone m* sur la carte, si k* vérifie :

$$k^* = \arg \max_{0 \le k < K} \left(\text{card}\left(C^E_{m^*} \cap C_k \right) \right)$$

Les neurones qui ne sont jamais activés par une observation ne sont pas représentés sur la carte.

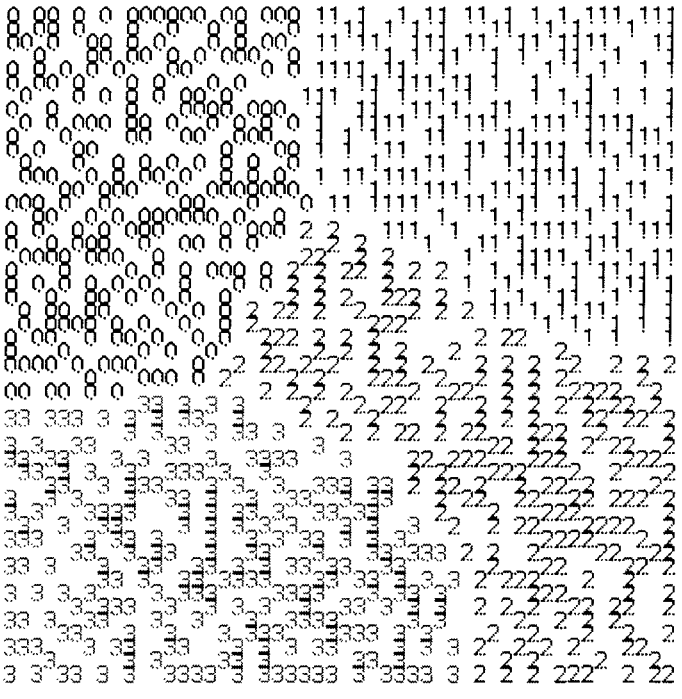


Figure III. 18 : Projection par CK d'un échantillon étiqueté

Sur la figure III.18, on peut remarquer que les observations issues de classes différentes sont projetées dans des régions différentes sur la carte. La carte est constituée de quatre régions dans lesquelles n'apparaissent que des projections d'observations d'une même classe. En général, deux neurones voisins sur la carte sont représentatifs d'une même classe, sauf pour ceux se situant aux frontières de séparation des classes.

Le réseau crée une structure entre les neurones qui reflète celle existant au sein des observations. Remarquons que bien que la structure soit respectée, les distances entre observations ne le sont pas. Ainsi, des classes relativement éloignées dans l'espace de représentation des observations peuvent se trouver projetées sur la carte côte à côte. Dans la suite, différentes techniques d'affichage seront détaillées dans le but de représenter les distances entre vecteurs poids de neurones voisins.

III.4 TECHNIQUE DE VISUALISATION DE ULTSCH

Cette méthode consiste à afficher entre deux neurones voisins la distance séparant leurs vecteurs poids [ULTS 90].

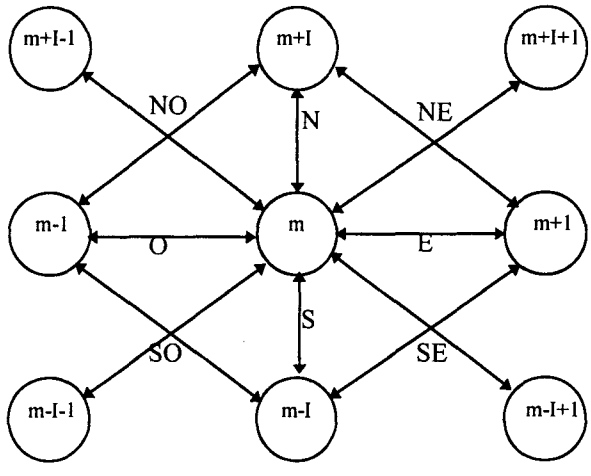


Figure III. 19 : Représentation des différentes distances de Ultsch autour d'un neurone

Soient N, S, E et O les distances séparant les vecteurs poids d'un neurone de ses voisins situés au Nord, Sud, Est et Ouest de celui-ci (cf Figure III.19). Ainsi pour le neurone m, on a :

$N(m)=d^E(W_m,W_{m+I})$

$S(m)=d^E(W_m,W_{m-I})$

$E(m)=d^E(W_m,W_{m+1})$

$O(m)=d^E(W_m,W_{m-1})$

où : $d^E(W_m,W_{m+I})=(W_m-W_{m+I})^T(W_m-W_{m+I})$

De même, soient NO, NE, SO, SE les distances séparant le vecteur poids d'un neurone de ses voisins se situant au Nord-Ouest, Nord-Est, Sud-Ouest et Sud-Est. Ultsch

approche ces distances par les expressions :

$$NO(m)=\frac{1}{2}\left(d^E(W_m, W_{m+I-1})+d^E(W_{m+I}, W_{m-1})\right)$$

$$NE(m)=\frac{1}{2}\left(d^E(W_m, W_{m+I+1})+d^E(W_{m+I}, W_{m+1})\right)$$

$$SO(m)=\frac{1}{2}\left(d^E(W_m, W_{m-I-1})+d^E(W_{m-I}, W_{m-1})\right)$$

$$SE(m)=\frac{1}{2}\left(d^E(W_m, W_{m-I+1})+d^E(W_{m+1}, W_{m-I})\right)$$

Ultsch représente ces distances par des barres en 3 dimensions positionnées entre les neurones (cf. Figure III.20). Grâce à cette représentation, les zones à haute densité d'observations sont représentées par des « bassins » séparés par des « murs » indiquant la présence de frontières entre les classes.

Représentation des distances de Ultsch pour un neurone m

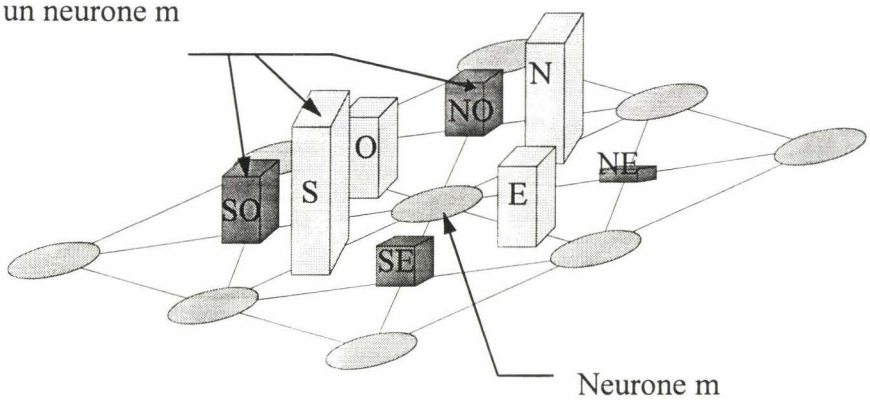


Figure III. 20 : Méthode de visualisation de Ultsch

Cette technique d'affichage permet de mettre en évidence les frontières entre les classes dans un contexte non supervisé. Cependant, bien qu'intéressante, cette technique ne semble pas logique. En effet, les distances N, S, E et O sont les distances réelles entre le neurone et ses voisins tandis que les distances NO, NE, SO, et SE sont des distances moyennes.

III.5 TECHNIQUE DE VISUALISATION DE KRAAIJVELD

Au lieu de tenter de représenter l'ensemble des distances entre le vecteur poids d'un neurone et ceux de ses voisins immédiats, Kraaijveld affiche, pour chaque neurone, la valeur maximale de ces distances [KRAA 92]. Nous appelons cette valeur le maximum des distances inter-neurones. Les distances inter-neurones sont celles séparant le vecteur poids d'un neurone m de ceux de ses voisins immédiats sur la grille qui appartiennent à l'ensemble V(m,1) (cf. §III.2.5).

Notons $z_{MAX}(m)$ le maximum des distances inter-neurones calculé dans le voisinage immédiat du neurone m. On a :

$$z_{MAX}(m) = \underset{m' \in V(m,1)}{Max} \left((W_m - W_{m'})^T (W_m - W_{m'}) \right)$$

avec $V(m,1) = \left\{ m' \in [0, M[, m' \neq m / d^A(U_m, U_{m'}) = |i - i'| + |j - j'| \leq 1 \right\}$

Kraaijveld représente chaque neurone m par un carré dont le niveau de gris dépend de la valeur de $z_{MAX}(m)$. Pour des raisons de clarté des représentations, nous associons un gris foncé une valeur importante de z_{MAX} et, inversement, un gris clair à valeur faible (cf. Figure III.21).

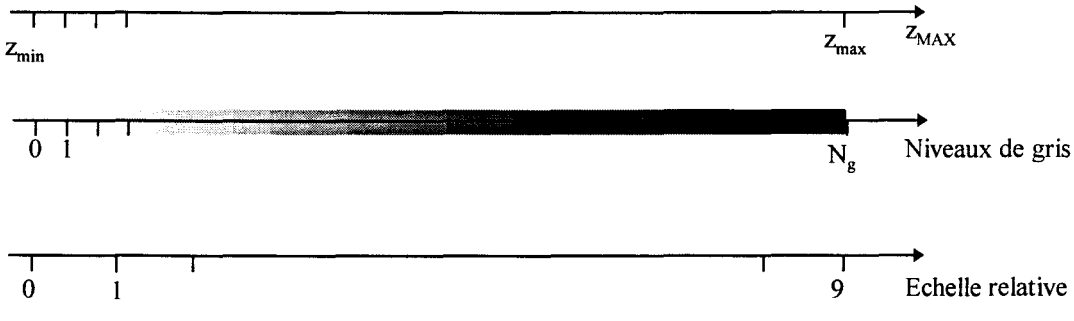


Figure III. 21 : Echelle de niveaux de gris

La figure III.22 illustre la représentation de Kraaijveld pour l'exemple 2. Nous avons placé sur la droite de la carte une échelle de niveau de gris graduée selon les valeurs relatives des maxima des distances inter-neurones. Ainsi les niveaux élevés de cette échelle, proches de 9, représentent des maxima relativement élevés. On leurs associe des niveaux de gris sombres. Inversement, les niveaux de gris clairs sont associés à des valeurs relativement faibles des maxima des distances inter-neurones. Nous utiliserons souvent, dans la suite de ce travail,

cette technique de représentation. Cela nous amènera, dans certains cas et uniquement pour améliorer la qualité de représentation des cartes, à effectuer une inversion de niveau de gris. Aussi, nous associons pour chacune de ces représentations, l'échelle de niveaux de gris avec l'échelle relative définie entre 0 et 9.

On remarque la présence de 4 régions claires représentant des valeurs relativement

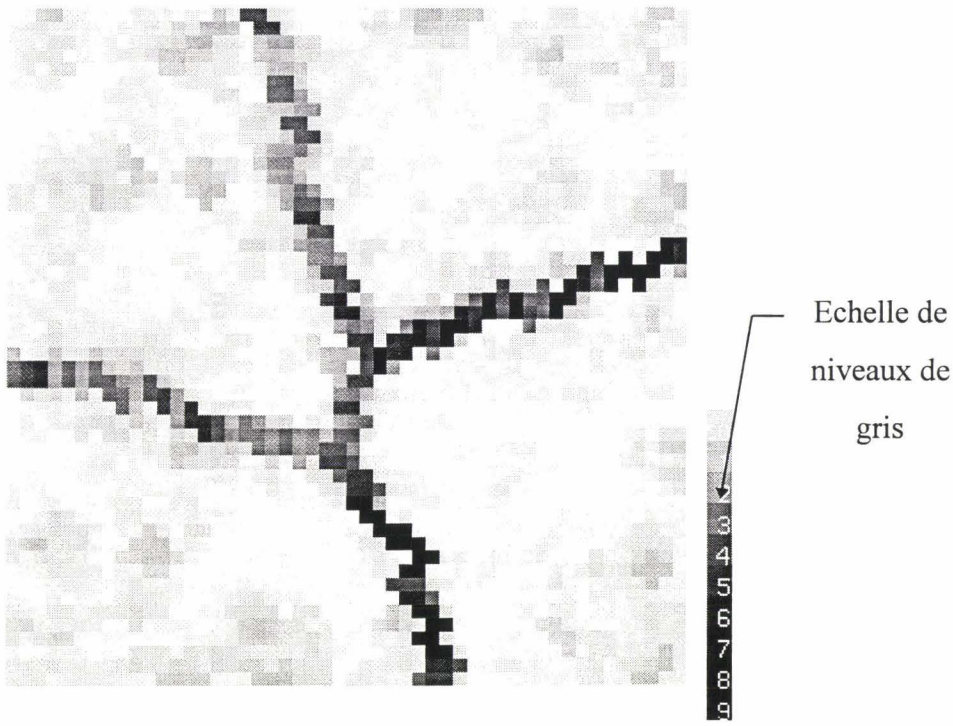


Figure III. 22 : Visualisation de z_{MAX} pour l'exemple 2

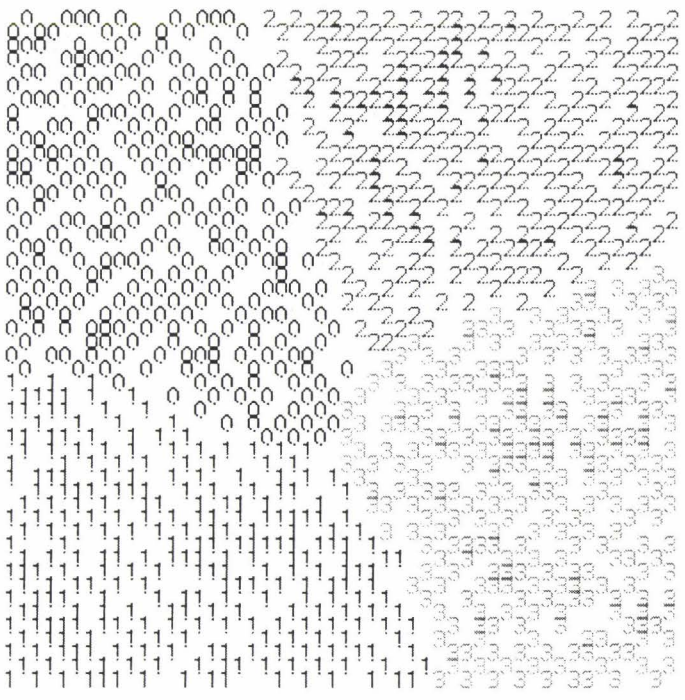


Figure III. 23 : Projection de l'échantillon étiqueté de l'exemple 2.

faibles des maxima des distances inter-neurones z_{MAX}. La projection de l'échantillon étiqueté (cf. Figure III.23) nous montre la concordance de ces zones avec la localisation des différentes classes. Les régions sombres de la représentation de Kraaijveld correspondent aux frontières entre les classes.

III.6 GENERALISATION DE LA TECHNIQUE DE VISUALISATION DE KRAAIJVELD

La technique de visualisation proposée par Kraaijveld permet, en calculant le maximum des distances inter-neurones z_{MAX}, de déceler les classes dans un contexte non supervisé. Nous pouvons étendre facilement la définition de ce maximum des distances inter-neurones à tous types de distances. Cependant, nous calculons ce maximum en utilisant la distance d^A utilisée lors de la phase d'apprentissage. Ainsi, par exemple, si nous utilisons la distance sphérique lors de la phase d'apprentissage, nous employons cette distance pour déterminer les voisins immédiats lors du calcul de z_{MAX}. Le calcul de z_{MAX} basé sur cette même distance permet d'évaluer l'organisation des vecteurs poids du réseau. Ainsi une valeur faible du maximum des distances inter-neurones pour un neurone m indique que celui-ci, ainsi

que ses voisins, ont des vecteurs poids similaires. Par contre, une valeur relativement élevée indique qu'il existe un ou plusieurs neurones voisins dont les vecteurs poids ne sont pas similaires au vecteur poids du neurone m.

Nous généralisons la notion de maximum des distances inter-neurones en posant :

$$z_{MAX}(m) = \underset{m' \in V(m,1)}{Max} \left(d^{E}(W_m, W_{m'}) \right)$$

avec : $V(m,1) = \left\{ m' \in [0, M[, m' \neq m / d^{A}(U_m, U_{m'}) \leq 1 \right\}$

Le calcul de z_{MAX} peut s'effectuer à partir de n'importe quelle distance d^{A}. Ainsi, la figure III.24 représente le maximum des distances inter-neurones z_{MAX} calculées à partir de la sup-distance, pour l'exemple 2. Ce réseau a préalablement subi un apprentissage utilisant la sup-distance afin de déterminer le voisinage des neurones gagnants. On constate que l'utilisation de la sup-distance fait apparaître les frontières plus épaisses que celles apparaissant sur les représentations basées sur une distance triangulaire.

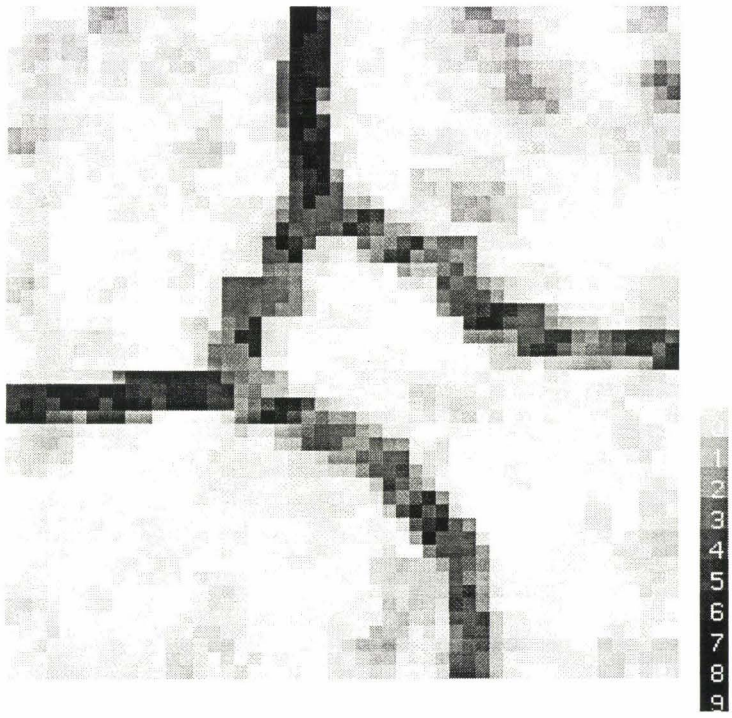


Figure III. 24: Visualisation de z_{MAX} calculé avec la Sup-distance

La définition de z_{MAX} s'applique directement aux cartes à maillage hexagonal. La figure III.25 représente une carte dont les neurones sont répartis suivant un tel maillage. On calcule z_{MAX} en utilisant la distance Euclidienne. Chaque neurone est représenté par un

hexagone teinté de gris. Les frontières entre les classes obtenues en utilisant le maillage hexagonal apparaissent plus régulières que celles apparaissant sur les représentations de z_{MAX} calculés avec les autres types de distances (cf. Figures III.24 et III.22).

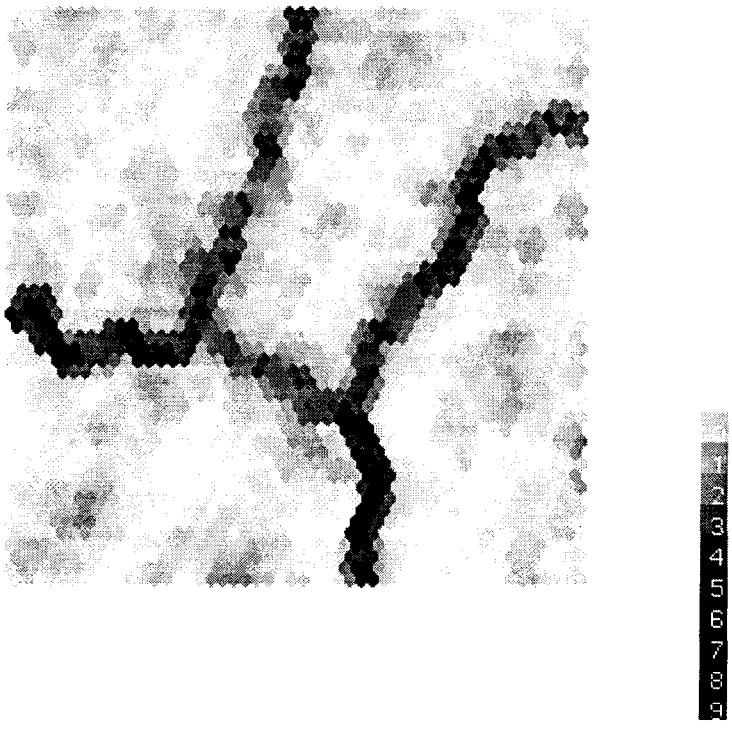


Figure III. 25 : Visualisation de z_{MAX} calculé avec la distance Euclidienne sur un réseau à maillage hexagonal

Nous avons représenté le maximum des distances inter-neurones. Cependant nous pouvons utiliser d'autres mesures pour caractériser l'ensemble des distances inter-neurones.

La moyenne des distances inter-neurones

On peut ainsi visualiser la moyenne des distances entre le vecteur poids du neurone considéré et les vecteurs poids de ses plus proches voisins en calculant :

$$z_{MOY}(m) = \frac{\sum_{m' \in V(m,1)} d^E(W_m, W_{m'})}{card(V(m,1))}$$

avec : $V(m,1) = \{m' \in [0, M[, m' \neq m / d^A(U_m, U_{m'}) \leq 1\}$.

et : $d^E(W_m, W_{m'}) = (W_m - W_{m'})^T (W_m - W_{m'})$

La valeur médiane des distances inter-neurones

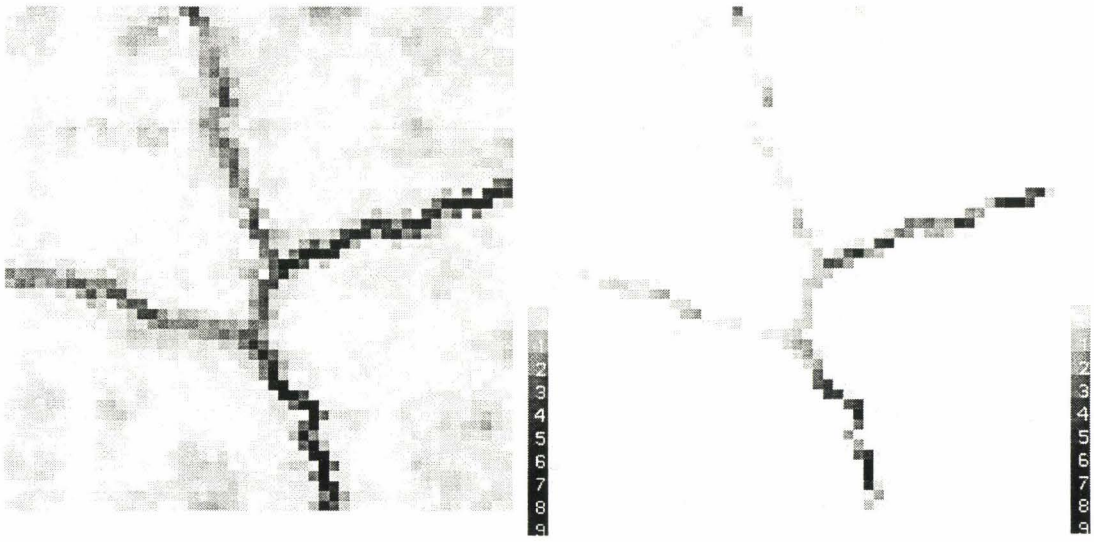
Nous pouvons caractériser aussi l'ensemble de ces distances inter-neurones par leur valeur médiane. La recherche de cette caractéristique consiste à classer les distances par ordre croissant et à retenir leur valeur médiane. Ainsi, dans le cas où un neurone a quatre voisins, on ordonne les distances entre le vecteur poids de ce neurone et les vecteurs poids de ses voisins. Si :

$$d_1 \leq d_2 \leq d_3 \leq d_4,$$

la valeur médiane est la moyenne de d_2 et d_3 :

$$z_{MED}(m) = \frac{d_2 + d_3}{2}.$$

Comme le montre la figure III.26, les représentations de ces deux dernières caractéristiques permettent de distinguer, comme avec les différentes représentations de z_{MAX}, quatre classes. Bien entendu, z_{MOY} et z_{MED} peuvent être calculées à partir d'autre types de distances. De plus, on peut représenter ces caractéristiques sur des cartes à maillage hexagonal. Les tests effectués sur l'exemple 2 avec d'autres types de distances font apparaître, de manière analogue à la figure III.26, 4 régions claires.



- (a) Représentation de z_{MOY} - - (b) Représentation de z_{MED} -

Figure III. 26 : Représentation de z_{MOY} et z_{MED} calculées avec la distance triangulaire

Sur la figure III.26(a), la représentation de z_{MOY} n'apporte pas de grandes différences par rapport à celle de z_{MAX}. La représentation de z_{MED} (cf. Figure III.26(b)) fait apparaître des frontières plus fines. De plus, les régions claires sont plus uniformes.

Nous avons représenté, par l'intermédiaire de cartes, des caractéristiques locales qui reflètent les distances entre les vecteurs poids de neurones voisins. Cependant, nous pouvons visualiser, avec les mêmes outils, toute fonction z(.) définie dans l'espace de représentation des observations $\mathbb{R}^N$ selon la même technique que celle utilisée pour visualiser les caractéristiques de distance à partir des cartes de Kohonen et rappelée ci-après.

Technique de visualisation par carte de Kohonen

1. Effectuer un apprentissage par carte de Kohonen de l'échantillon d'observations.

2. Calculer la valeur de $z(W_m)$ pour chaque neurone m, $m \in [0,M[$, au point de l'espace de représentation des observations défini par les composantes du vecteur poids W_m.

3. Représenter sur la carte, pour chaque neurone m, la valeur de z(Wm) en niveau de gris.

Une application directe et très intéressante de cette technique de visualisation consiste à représenter la fonction de densité de probabilité sous-jacente à l'échantillon d'observations analysé.

III.7 VISUALISATION DE LA FONCTION DENSITE DE PROBABILITE SOUS-JACENTE SUR UNE CARTE DE KOHONEN

Les techniques de représentation proposées sont basées sur le calcul des distances entre vecteurs poids. La visualisation des distances sur une carte de Kohonen ne fait apparaître les classes que lorsque celles-ci sont relativement éloignées les unes des autres. Ainsi, on ne peut percevoir à l'aide de ces représentations que les classes ayant un degré de chevauchement nul. Lorsque deux classes sont très proches, les distances calculées aux frontières de ces classes sont relativement faibles et deviennent indécelables. Dans ce type de situation, la représentation de la fonction de densité de probabilité permet, d'une part d'obtenir une information relative à l'échantillon d'observations et, d'autre part, de distinguer la présence de classes, même en cas de chevauchements importants.

Pour visualiser la fonction densité de probabilité sous-jacente, les estimations sont calculées aux points de l'espace $\mathbb{R}^N$ définis par les composantes des vecteurs poids W_m. Nous utilisons l'estimateur non paramétrique de Parzen (cf. Annexe 2). Cette estimation s'exprime, en W_m, par :

$$\hat{p}(W_m) = \frac{1}{Q}\sum_{q=1}^{Q} \frac{1}{V[D(W_m)]}\Omega\left(\frac{W_m - X_q}{h_Q}\right) \quad .$$

Pour chaque neurone m, on représente en niveau de gris la valeur relative de la fonction densité de probabilité estimée $\hat{p}(W_m)$. La figure III.27 représente une estimation de la fonction densité de probabilité sous-jacente à l'échantillon d'observations de l'exemple 2. On distingue quatre régions sombres indiquant la présence de 4 zones de forte densité d'observations dans l'espace de représentation des données. Les zones de faible densité coïncident avec les frontières décelées dans la représentation de z_{MAX} effectuée à partir de la même carte (cf. Figure III.24).

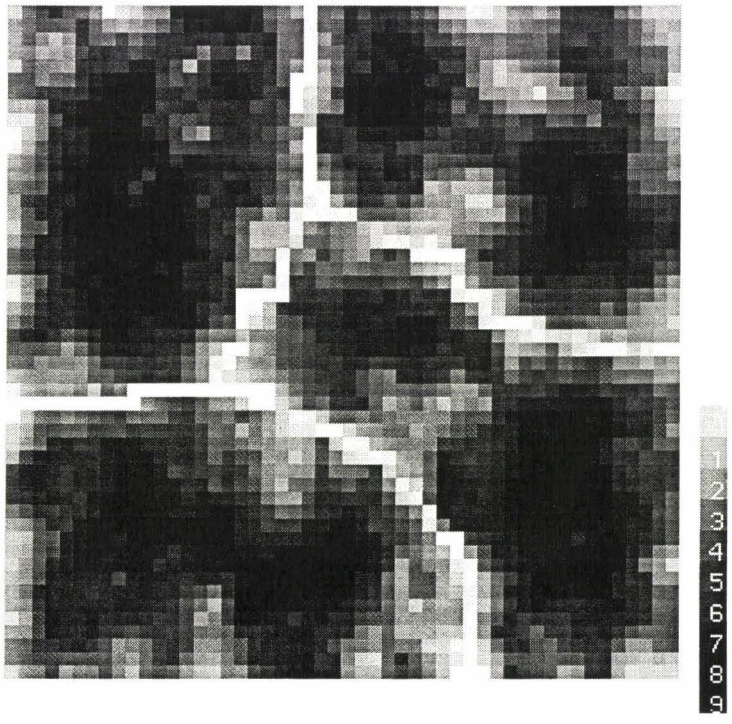


Figure III. 27 :Visualisation de l'estimation par la méthode du noyau de la fonction densité de probabilité sous-jacente à l'échantillon d'observations de l'exemple 2

On peut représenter, par cette même technique, la fonction densité de probabilité sous-jacente à l'échantillon des vecteurs poids. Comme précédemment, les points

d'estimation sont les vecteurs poids. Par contre, l'estimation de la fonction densité de probabilité en ces points s'effectue à partir de l'échantillon des vecteurs poids.

$$\hat{p}(W_m) = \frac{1}{M} \sum_{m'=1}^{M} \frac{1}{V[D(W_m)]} \Omega\left(\frac{W_m - W_{m'}}{h_Q} \right)$$

La figure III.28 représente l'estimation de la fonction densité de probabilité sous-jacente à l'échantillon des vecteurs poids pour l'exemple 2.

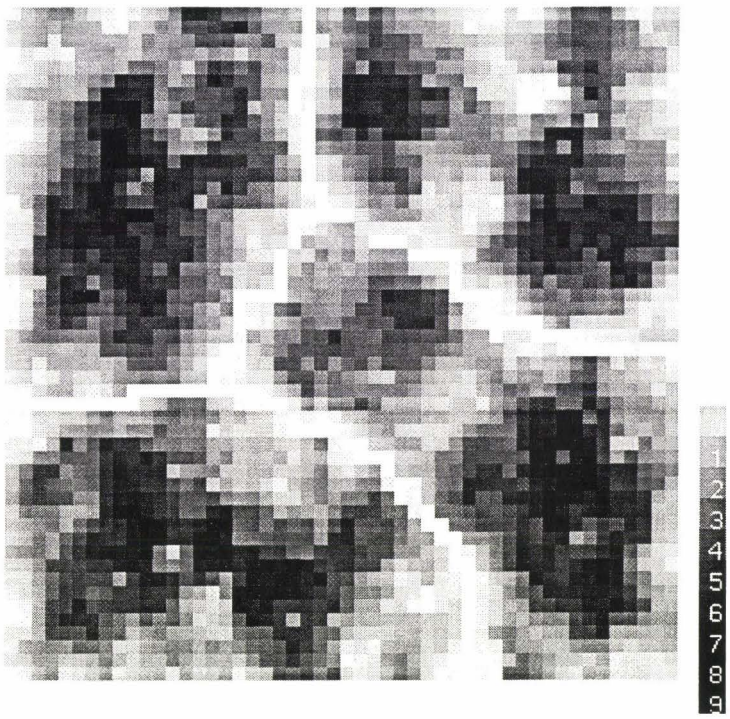


Figure III. 28 : Visualisation de la fonction densité de probabilité sous-jacente à l'échantillon des vecteurs poids pour l'exemple 2

La comparaison des deux représentations permet de vérifier si la convergence des poids s'est effectuée de telle sorte la fonction densité de probabilité de l'échantillon des poids reflète celle de l'échantillon d'apprentissage. Dans ce cas, les régions de densités importantes doivent être localisées aux mêmes endroits sur les deux cartes. Nous pouvons constater, en comparant les figures III.27 et III.28, que les maxima de ces estimations de la fonction densité de probabilité sont localisés aux mêmes endroits sur les cartes indiquant ainsi que les vecteurs poids ont convergé vers les zones de l'espace $\mathbb{R}^N$ denses en observations.

III.8 ETUDE EXPERIMENTALE DES REPRESENTATIONS PAR CARTES DE KOHONEN

Dans ce paragraphe, on se propose de mettre en évidence les propriétés de conservation de la structure locale des données des représentations par cartes de Kohonen. Pour ce faire, nous présentons deux exemples. Le premier est constitué de deux classes. Les observations de la première classe sont réparties selon une distribution gaussienne en trois dimensions. Les observations de la seconde classe sont réparties à la périphérie d'une sphère. Ces deux classes ont le même centre. Le second exemple est composé de deux classes en forme de tores imbriquées l'une dans l'autre.

III.8.1 Exemple 3

On génère un exemple artificiel constitué d'une classe gaussienne incluse dans une autre dans un espace à trois dimensions. Les paramètres statistiques de la première classe, notée classe 0, sont indiqués dans le tableau III.4 :

Nombre d'observations	Vecteur Moyenne	Matrice de Covariance
200	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0,28 & 0 & 0 \\ 0 & 0,35 & 0 \\ 0 & 0 & 0,36 \end{pmatrix}$

Tableau III.4: Paramètres statistiques de la classe 0

La seconde classe, notée classe 1, regroupe 800 observations formant un nuage de forme sphérique. Nous avons généré cette classe en utilisant les équations :

$$x_1 = R \times \cos\theta \times \cos\varphi$$
$$x_2 = R \times \sin\theta \times \cos\varphi$$
$$x_3 = R \times \sin\varphi$$

θ et φ sont des variables générées aléatoirement suivant des distributions uniformes entre 0 et 2π pour θ et entre $-\pi/2$ et $\pi/2$ pour φ. R est généré suivant une loi normale de moyenne μ et de variance σ. Les valeurs de ces paramètres figurent dans le tableau III.5.

Nombre d'observations	θ	φ	R
800	$[0;2\pi]$	$]-\pi/2;\pi/2[$	$\mu = 8$ $\sigma = 1$

Tableau III.5 : Paramètres statistiques de la classe 1

L'échantillon tridimensionnel est représenté sur la figure III.29 où l'on constate un certain chevauchement de la classe 0 avec la classe 1 dû à la dispersion du rayon R.

Nous avons effectué un apprentissage de l'échantillon sur un réseau de 50*50 neurones répartis suivant un maillage hexagonal. On initialise le coefficient d'apprentissage à la valeur 0,5. Ce paramètre décroît linéairement vers zéro en fonction des époques. Le voisinage initial, de rayon 25, décroît toutes les 20 époques. Au bout de 500 époques, ce rayon devient nul. L'apprentissage est alors conduit selon le mode compétitif classique. La loi d'interaction est de type sphérique.

La carte de la figure III.30(a) représente les maxima des distances inter-neurones z_{MAX}. On y distingue deux régions connexes claires, nettement séparées par une frontière sombre, bien que les classes aient le même centre. La carte de la figure III.30(b) représente z_{MED}. La représentation du médian des distances inter-neurones n'apporte pas d'indication supplémentaire par rapport à la représentation de z_{MAX}. On peut noter que la frontière entre les classes est plus fine que celle aperçue sur la carte représentant z_{MAX}. La figure III.31 représente la projection de l'échantillon étiqueté où les observations de la classe 0 sont représentées par des 0, alors que les observations de la classe 1 sont représentées par des 1. Nous pouvons constater que la localisation des classes correspond à la localisation des régions délimitées par la frontière décelée par les représentations de z_{MAX} et de z_{MED}.

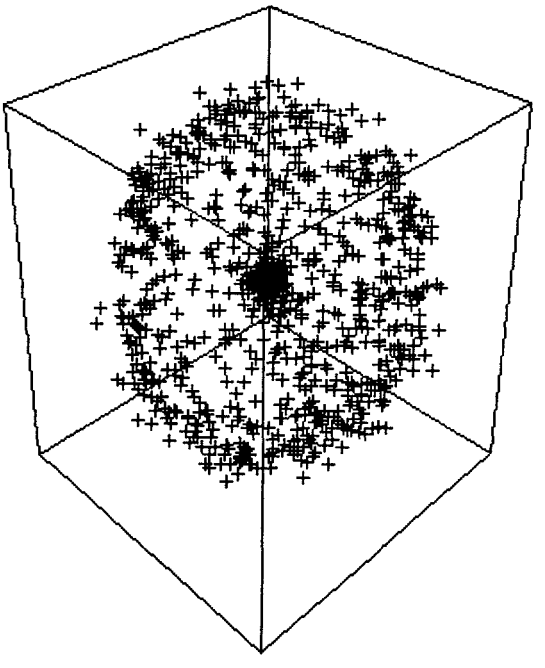
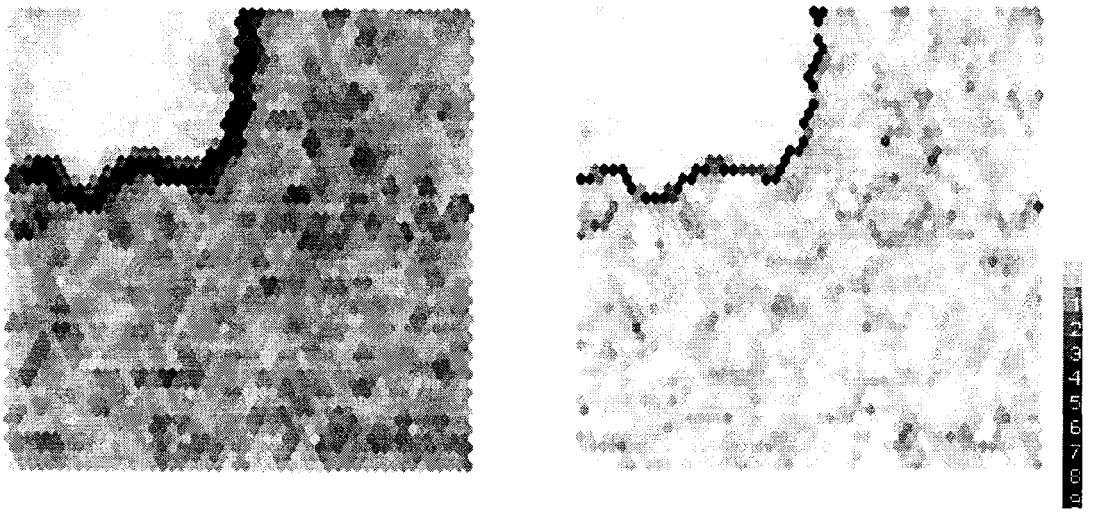


Figure III. 29 : Représentation en trois dimensions de l'échantillon de l'exemple 3



- (a) z_{MAX} -

- (b) z_{MED} -

Figure III. 30 : Représentation de z_{MAX} et z_{MED} pour l'exemple 3

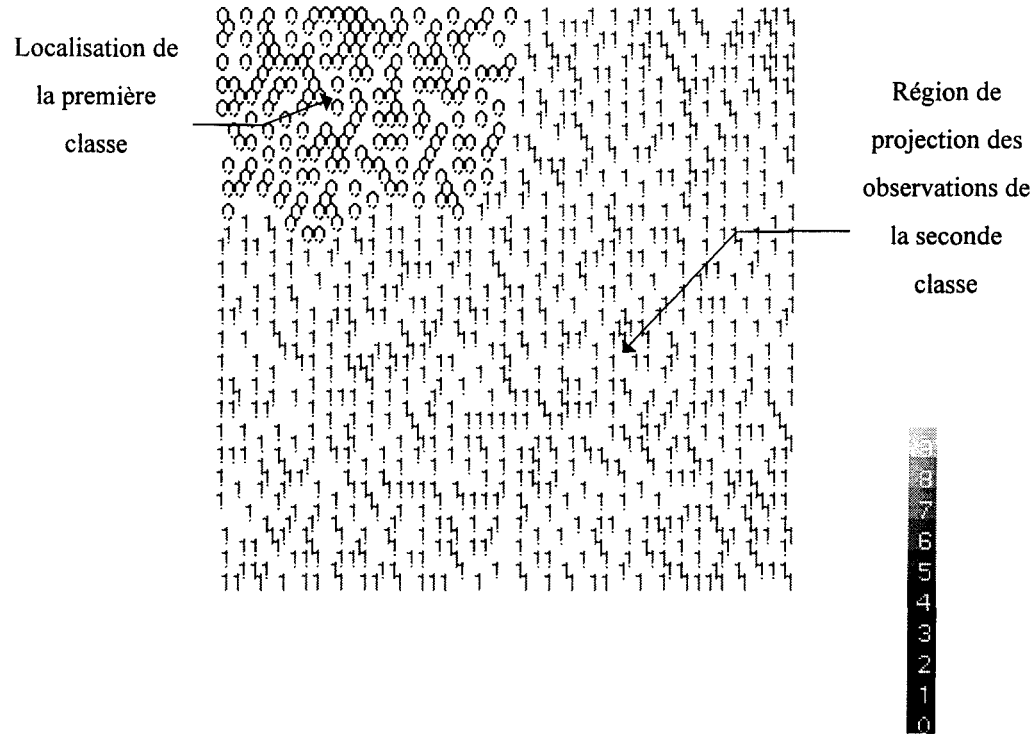


Figure III. 31 : Projection de l'échantillon étiqueté de l'exemple 3 sur la carte

Nous avons représenté sur la figure III.32 une estimation non paramétrique avec un noyau gaussien de la fonction densité de probabilité de l'échantillon des observations. La représentation de cette estimation fait apparaître nettement la classe gaussienne qui se manifeste par des valeurs relativement élevées de la fonction densité de probabilité dans la région où elle est projetée sur la carte. La seconde classe, qui occupe tout le reste de la carte, est nettement moins dense que la première.

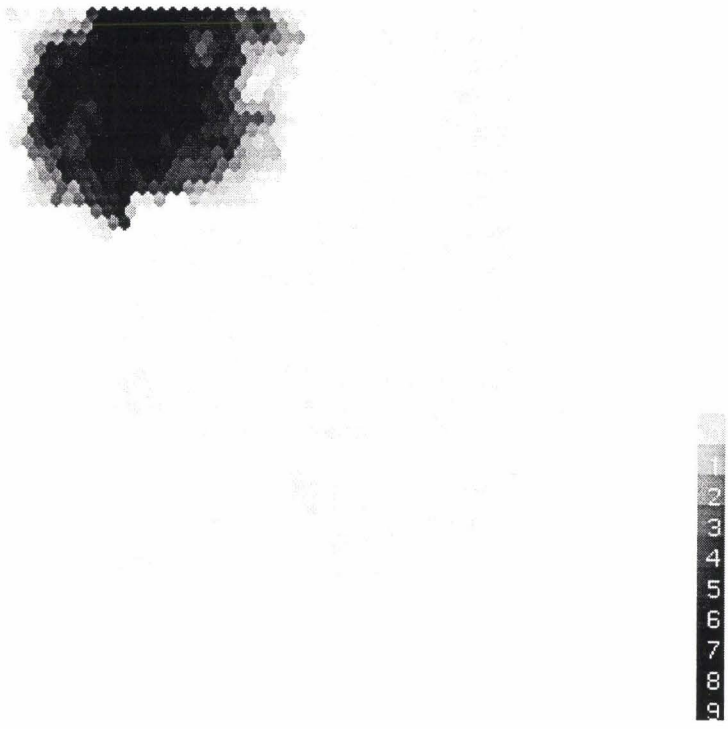


Figure III. 32 : Représentation de la fonction densité de probabilité sous-jacente à l'échantillon d'observations de l'exemple 3

Les différentes représentations font nettement apparaître deux classes. Cependant, il est à noter que sur ces représentations la classe 0 se situe dans un coin de la carte. Or celle ci se situe au centre de la classe 1 dans l'espace de représentation des observations. Il semblerait normal que sur les représentations par carte de Kohonen, la classe 0 apparaisse au centre de la classe 1. Nous avons effectué, lors de nos simulations, différents apprentissages de CK. Les différentes représentations obtenues ont toutes fait apparaître la classe 0 dans un coin de la carte. Ainsi cet exemple met en évidence des problèmes de conservation de la topologie des classes sur les représentations par CK. Ceci implique, pour cet exemple, que si l'on suppose que l'on ne dispose d'aucune information sur l'échantillon, alors les représentations par CK, bien qu'elles révèlent la présence de deux classes, ne permettent pas de déduire que les deux classes ont le même centre et que l'une d'entre elles entoure l'autre.

III.8.2 Exemple 4

Nous utilisons un quatrième exemple pour montrer la capacité des représentations des cartes de Kohonen à présenter de manière distincte des classes très enchevêtrées. L'échantillon de cet exemple est constitué de deux classes en formes de tores entrelacés dans un espace à trois dimensions (cf. Figure III.33). Chacune de ces classes regroupe 500 observations.

Les observations de la première classe, notée classe 0, sont générées à l'aide des équations suivantes :

$$x_1 = A_1\cos\rho + A_2\cos\theta$$

$$x_2 = A_1\cos\rho + A_2\sin\theta + B_1$$

$$x_3 = A_1\sin\rho$$

où θ et ρ sont des variables aléatoires normales, de moyennes et de variances respectives m, m_1, s et s_1.

A_1 est une variable aléatoire normale de moyenne μ_1 et de variance σ_1.

A_2 et B_1 sont des constantes.

Les valeurs de ces paramètres sont détaillées dans le tableau III.6.

Classe	θ	ρ	A_1	A_2	B_1
0	$m = 0$ $s = \pi$	$m_1 = 0$ $s_1 = \pi$	$\mu_1=0$ $\sigma_1=5$	50	50

Tableau III.6 : Paramètres statistiques de la classe 0

Les observations de la seconde classe, notée classe 1, sont obtenues à l'aide des équations suivantes :

$$x_1 = A_1'\cos\rho' + A_2'\cos\theta'$$

$$x_2 = A_1'\sin\rho'$$

$$x_3 = A_1'\cos\rho' + A_2'\sin\theta'$$

où θ' et ρ' sont des variables aléatoires normales, de moyennes et de variances respectives m', m_1', s' et s_1'.

A_1' est une variable aléatoire normale de moyenne μ_1' et de variance σ_1'.

A_2' est une constante.

Les valeurs de ces paramètres sont indiquées dans le tableau III.7.

Nombre d'observations	θ'	ρ'	A_1'	A_2'
500	$m' = 0$ $s' = \pi$	$m_1' = 0$ $s_1' = \pi$	$\mu_1'=0$ $\sigma_1'=2$	50

Tableau III.7 : Paramètres statistiques de la classe 1

Cet échantillon a été appris par un réseau de 50*50 neurones dans les mêmes conditions que celles de l'exemple 3. La figure III.34(a) représente z_{MAX}. Cette carte se décompose en deux régions claires. La représentation de z_{MED} est similaire à celle de z_{MAX} (cf. Figure III.34(b)). Les frontières apparaissent plus fines dans la représentation de z_{MED} que dans celle de z_{MAX}.

On peut constater qu'il y a correspondance entre ces régions et la localisation des classes sur la carte (cf. Figure III.35), à l'exception d'une observation de la classe 1 projetée au sein de la classe 0. Nous pouvons observer, de plus, que cette observation isolée est localisée dans une région frontières dans les représentations de z_{MAX} et z_{MED}.

La représentation de la fonction densité de probabilité (cf. Figure III.36), permet de distinguer aussi les classes, car l'une d'elles est plus dense que l'autre et est représentée par la région la plus foncée. Cette région sombre correspond à la région où sont projetées les observations de la classe 1. Notons aussi que la région sombre apparaissant au milieu de la carte dans les représentations des distances n'est pas visible dans la représentation de la fonction densité de probabilité.

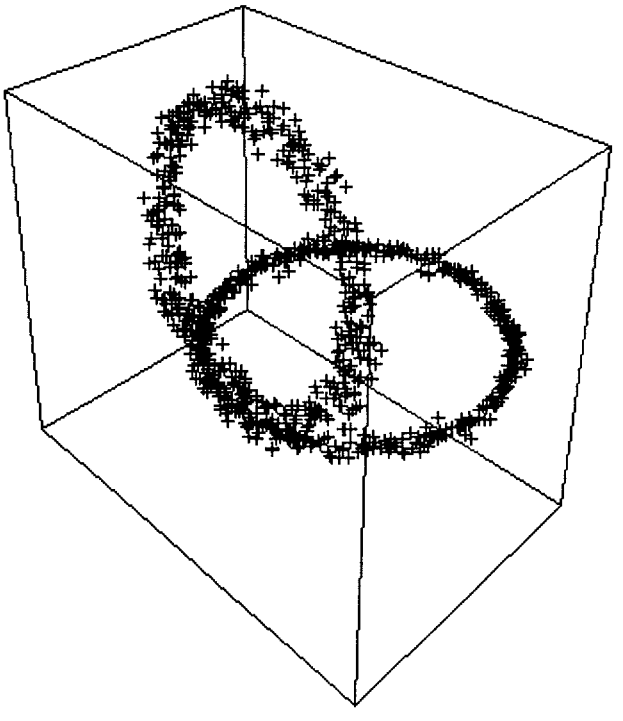
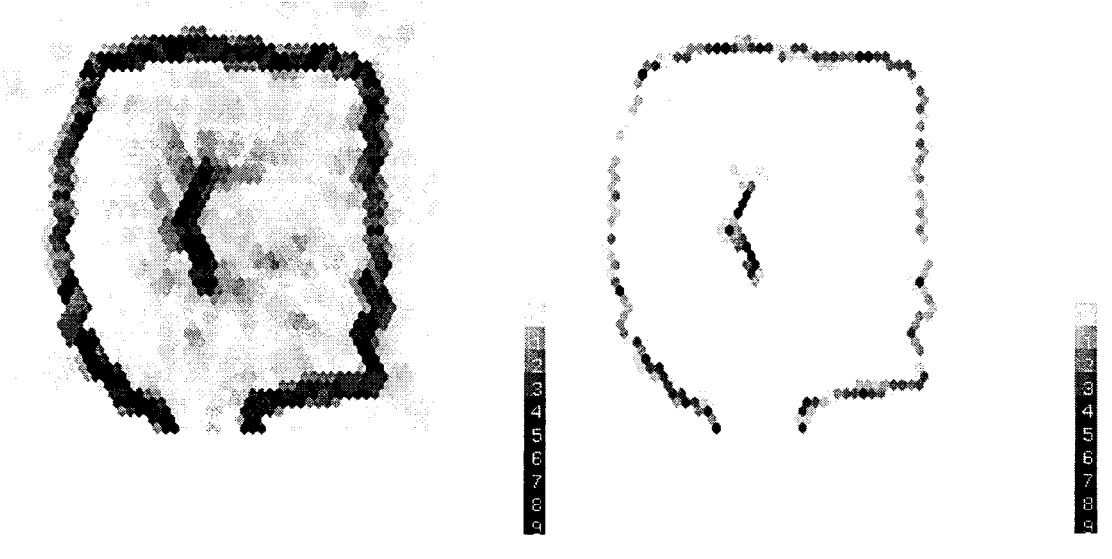


Figure III. 33 : Représentation en trois dimensions de l'échantillon de l'exemple 4



- (a) z_{MAX} -

- (b) z_{MED} -

Figure III. 34 : Représentation de z_{MAX} et de z_{MED} pour l'exemple 4

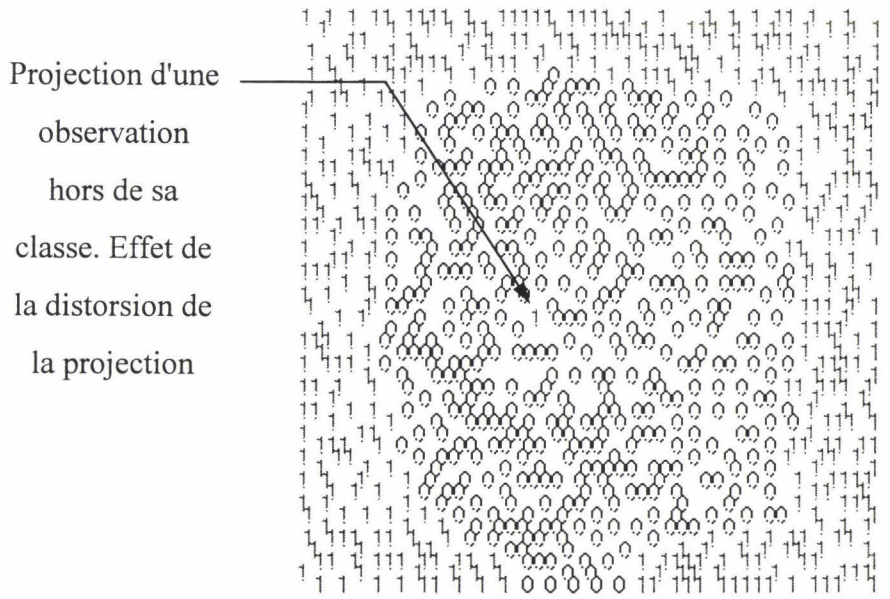


Figure III. 35 : Projection de l'échantillon étiqueté de l'exemple 4 sur la carte

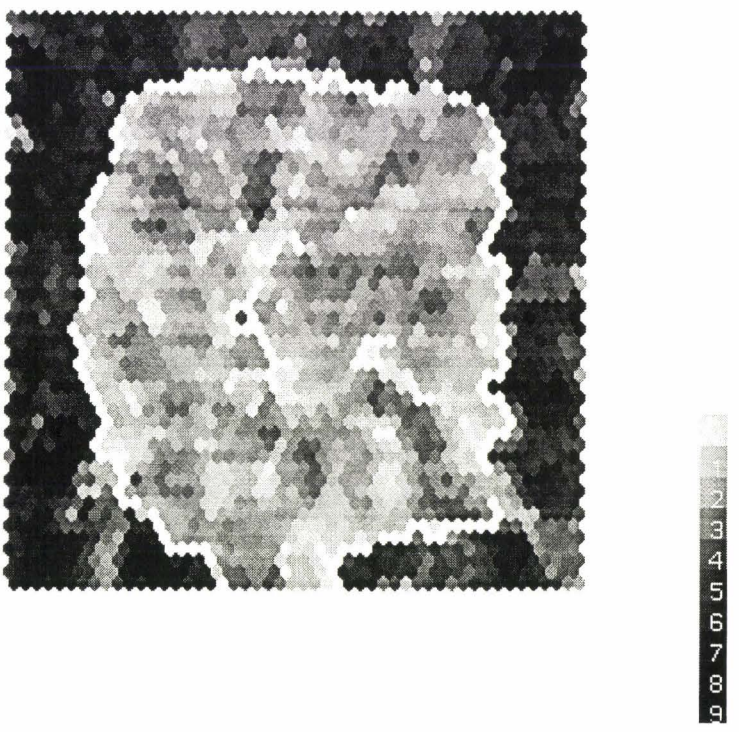


Figure III. 36 : Représentation de la fonction densité de probabilité sous-jacente à l'échantillon d'observations de l'exemple 4

III.9 CONCLUSIONS

L'introduction des phénomènes d'interaction entre neurones dans la phase d'apprentissage permet d'organiser les neurones en fonction de leurs positions sur la carte. Ainsi les relations de voisinage forcent d'une part un grand nombre de vecteurs poids à converger vers les zones denses en observations. Ce type d'apprentissage limite ainsi le nombre de neurones sous-utilisés ce qui rend l'apprentissage compétitif de Kohonen intéressant dans le cadre de la réduction de données. D'autre part, la convergence des vecteurs poids ne se fait pas de manière totalement désordonnée. En effet, seuls les vecteurs poids de neurones voisins sur la carte convergent vers une même zone de l'espace de représentation. A la fin de l'apprentissage, deux neurones voisins ont leurs vecteurs poids proches dans l'espace de représentation des observations et, de plus, la majorité de ces vecteurs poids sont situés dans des zones denses en observations.

Dans un contexte non supervisé, la visualisation en niveaux de gris des distances relatives entre le vecteur poids d'un neurone et les vecteurs poids de ses plus proches neurones sur la carte permet d'obtenir une image discrète des données multidimensionnelles.

Nous avons généralisé la technique de visualisation de Kraaijveld en l'étendant aux réseaux à maillage hexagonal. Nous avons également représenté d'autres caractéristiques qui reflètent les distances entre les vecteurs poids de neurones voisins, telle que la moyenne ou la valeur médiane de ces distances.

Cette technique de visualisation permet de représenter n'importe quelle fonction définie dans l'espace de représentation des observations. Nous avons appliqué cette technique à la représentation de la fonction densité de probabilité sous-jacente à l'échantillon d'observations, ainsi qu'à celle sous-jacente à l'ensemble des vecteurs poids. La représentation de la fonction densité de probabilité sous-jacente permet de distinguer les classes les plus denses. De plus, l'analyse de cette représentation peut mettre en évidence des classes ayant un degré de chevauchement important. Cependant, la représentation de la fonction densité de probabilité ne permet pas de mettre en évidence, comme le font les représentations basées sur les distances inter-neurones, les distorsions de l'ordre des distances entre vecteurs poids.

Toutes ces représentations permettent de découvrir la structure d'échantillons multidimensionnels. Nous allons intégrer, dans le chapitre suivant, ces techniques de

représentation ainsi que les techniques de classification et de réduction de données présentées dans le chapitre III dans un processus de classification interactive. Expérimentalement, aucune des caractéristiques présentées dans ce chapitre ne s'est révélée être vraiment supérieure aux autres. Aussi, nous nous proposons, dans le cadre d'un processus de classification interactif, d'offrir à l'analyste un choix étendu de représentations. Pour cela, nous décrirons dans le chapitre suivant, différents outils informatiques susceptibles d'aider l'analyste à utiliser ces différentes représentations simultanément.

CHAPITRE IV :

APPROCHE METHODOLOGIQUE POUR LA CLASSIFICATION INTERACTIVE PAR RESEAUX DE NEURONES

IV.1 PRESENTATION DE LA METHODOLOGIE

La classification interactive a pour objet d'impliquer l'opérateur humain dans les prises de décision. Elle consiste à lui fournir une projection plane des données multidimensionnelles et un ensemble d'outils d'aide graphique pour découvrir des groupements et repérer des points isolés au sein des données.

Nous pouvons représenter les différentes étapes d'un processus de classification interactif selon le schéma de la figure IV.1. Nous considérons que l'on dispose d'un échantillon d'observations prêt à être analysé. Cela suppose que l'on ait effectué l'acquisition des informations, les prétraitements et défini les différents attributs. Dès lors, on effectue les opérations suivantes :

- Projections :

Nous utilisons à ce stade les représentations par cartes de Kohonen auxquelles on applique un apprentissage préalable à leur exploitation. Afin d'éviter de baser l'analyse ultérieure sur de mauvaises cartes, nous effectuons sur le même échantillon plusieurs

apprentissages avec différentes initialisations. Nous pouvons également employer des projections plus classiques telles que celles obtenues par l'algorithme de Sammon ou par la méthode d'analyse en composantes principales afin de comparer les différents résultats.

- Analyse visuelle de l'échantillon :

Cette étape exploite les capacités visuelles de l'opérateur qui analyse les différentes projections sur un écran graphique. A ce stade, l'analyste découvre la structure des données et émet des hypothèses quant au nombre de classes en présence dans l'échantillon. Il peut, de plus, initialiser certains paramètres des algorithmes de classification employés dans la phase suivante. Ceci équivaut, pour les réseaux de neurones à apprentissage compétitif, à prendre un nombre de neurones égal au nombre de classes et à initialiser les vecteurs poids de ces neurones aux centres des classes. Nous serons amenés, à ce stade, à décrire les outils informatiques spécialement conçus afin d'aider l'opérateur dans sa tâche d'analyse.

- Classification :

Cette étape correspond à l'application des algorithmes de classification. Nous pouvons utiliser des techniques classiques comme les algorithmes des K-means ou du plus proche voisin ou comme ceux faisant appel aux réseaux de neurones à apprentissage compétitif traités dans le chapitre II.

- Vérification par projection de l'échantillon classé :

Dans l'étape précédente, les algorithmes de classification assignent chaque observation de l'échantillon à l'une des classes décelées. A ce stade du processus de classification interactif, nous projetons l'échantillon classé afin de vérifier que les classes apparaissant sur la carte se confondent avec les régions décelées lors de l'analyse visuelle de l'échantillon. Dans le cas où les régions préalablement décelées se confondent avec les classes, on considère que les résultats de la classification sont cohérents avec l'analyse visuelle de l'échantillon. Cette étape permet de mettre en évidence les mauvaises cartes, mais aussi de révéler l'utilisation inappropriée de certains algorithmes de classification.

- Analyse locale des classes :

Cette étape consiste à analyser séparément les classes mises en évidence afin de détecter la présence éventuelle d'autres classes. En effectuant des projections séparées des classes, on peut obtenir une meilleure représentation de celles-ci et ainsi distinguer d'éventuelles classes indécelables lors de l'analyse globale de l'échantillon.

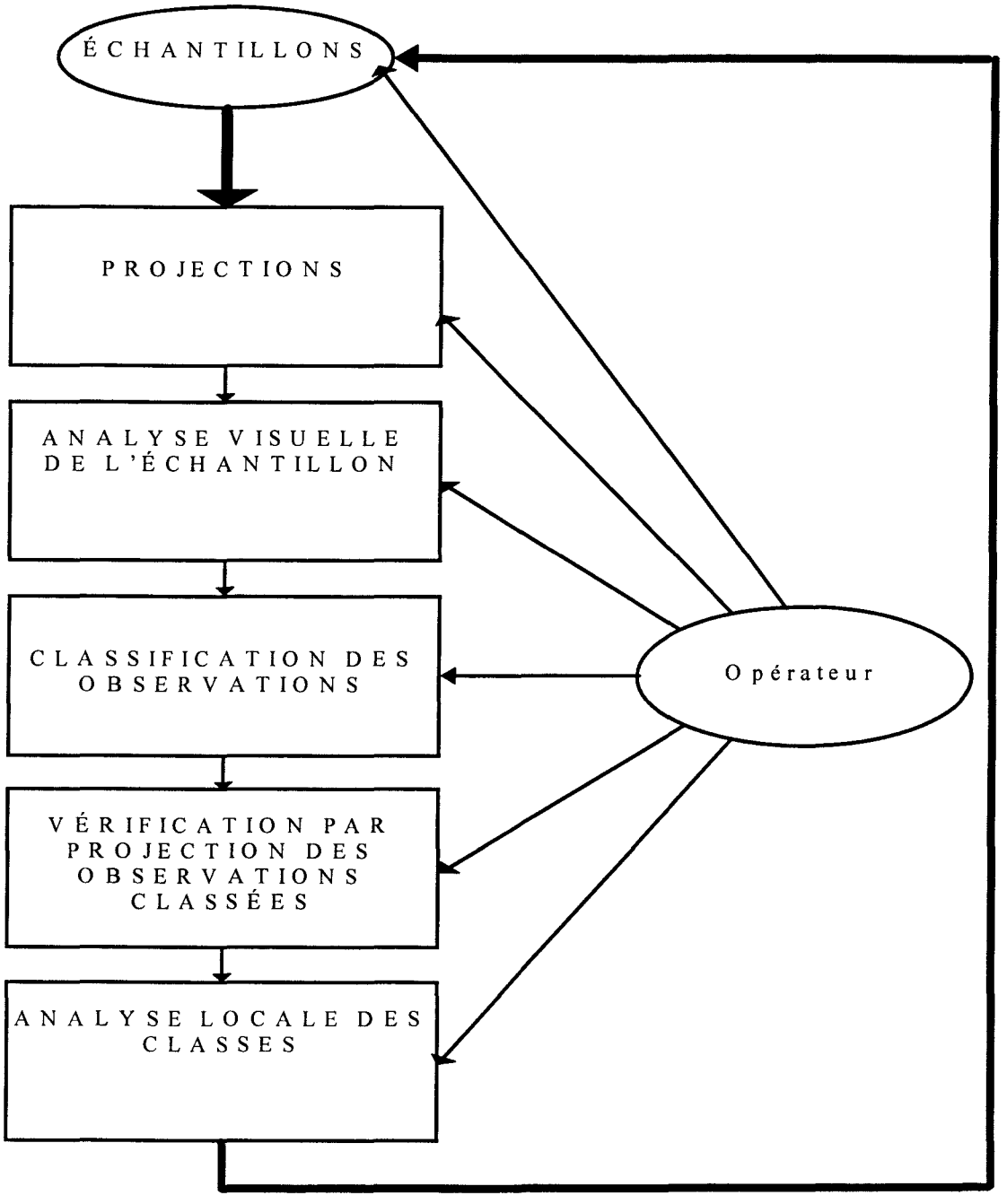


Figure IV. 1 : Descriptif du processus de classification interactive

IV.2 ANALYSE VISUELLE DE L'ECHANTILLON

Pour illustrer cette démarche, nous utilisons l'échantillon de l'exemple 2 du chapitre III (cf. Figure IV.2), constitué de 4 classes dont les paramètres statistiques sont consignés dans le tableau IV.1. Ces paramètres sont calculés après la génération de l'échantillon.

Classe	Nombre d'observations	Vecteur moyenne	Matrice de covariance
Classe 0	300	$\begin{pmatrix} 0 \\ 0,02 \\ -0,26 \end{pmatrix}$	$\begin{pmatrix} 0,49 & 0 & 0,03 \\ 0 & 0,48 & 0 \\ 0,03 & 0 & 0,54 \end{pmatrix}$
Classe 1	300	$\begin{pmatrix} 6,02 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0,58 & -0,01 & 0,03 \\ -0,01 & 0,46 & -0,03 \\ 0,03 & -0,03 & 0,54 \end{pmatrix}$
Classe 2	300	$\begin{pmatrix} -0,04 \\ -0,05 \\ 5,95 \end{pmatrix}$	$\begin{pmatrix} 0,44 & 0,19 & 0 \\ 0,19 & 0,57 & 0,05 \\ 0 & 0,05 & 0,53 \end{pmatrix}$
Classe 3	300	$\begin{pmatrix} 0,05 \\ 5,96 \\ 0,02 \end{pmatrix}$	$\begin{pmatrix} 0,46 & -0,02 & -0,04 \\ -0,02 & 0,48 & 0,02 \\ -0,04 & 0,02 & 0,57 \end{pmatrix}$

Tableau IV. 1 :Paramètres statistiques des différentes classes de l'échantillon de l'exemple 2

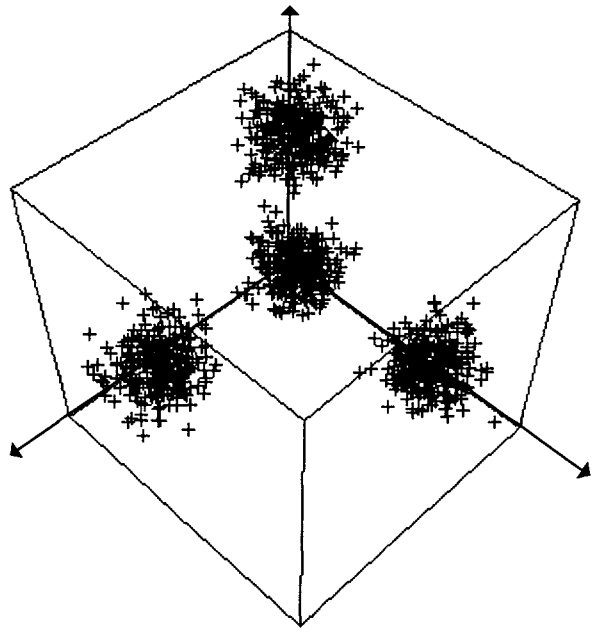


Figure IV. 2 : Vue en perspective de l'échantillon de l'exemple 2

La difficulté majeure dans un contexte non supervisé est de déterminer le nombre de classes constituant un échantillon soumis à l'analyse. Une interface graphique conviviale doit être élaborée pour aider l'analyste dans cette tâche importante. Nous avons développé un

certain nombre d'outils informatiques pour faciliter l'analyse.

IV.2.1. Interface multifenêtrage

Avant d'entreprendre la classification de l'échantillon, il est intéressant de proposer à l'opérateur différentes représentations issues de différentes méthodes de projection. Afin de faciliter cette tâche et d'éviter des erreurs de manipulation, nous avons conçu un logiciel permettant de visualiser simultanément différentes représentations sur un même écran que l'on peut diviser en fenêtres à volonté. On peut ainsi comparer les différentes cartes et autres techniques de projection en limitant le nombre de manipulations informatiques. L'interface graphique apparaît sous la forme de fenêtres munies de curseurs sur les côtés (cf. Figure IV.3). Ces curseurs servent à déplacer les différentes représentations à l'intérieur des fenêtres. Pour scinder une fenêtre de l'interface graphique, l'opérateur utilise le menu contextuel des curseurs, accessible à l'aide de la souris. Il peut aussi, avec ce même menu, fusionner deux fenêtres adjacentes. Chaque fenêtre dispose de son propre menu contextuel permettant différents traitements indépendants, tels l'apprentissage des réseaux, la classification, ou l'estimation de la fonction densité de probabilité. Un historique des apprentissages et des divers traitements est aussi accessible par ce menu contextuel. L'analyste peut ainsi, à son gré, scinder ou fusionner les fenêtres au cours de son analyse et faire apparaître différentes représentations. Nous avons effectué différents apprentissages à partir de l'échantillon de l'exemple 2 sur des cartes composées de 50*50 neurones. Toutes les caractéristiques permettant d'évaluer les distances inter-neurones présentées dans le chapitre précédent, ainsi que les estimations de la fonction densité de probabilité, peuvent être visualisées à partir des cartes à l'intérieur de chacune des fenêtres créées par l'opérateur. Ainsi, sur l'écran visualisé sur la figure IV.3, on voit apparaître les représentations de z_{MAX} dans les fenêtres du haut calculées avec la distance triangulaire (carte de gauche), sphérique (carte du centre) et carrée (carte de droite). Les fenêtres inférieures représentent z_{MED} calculées avec la distance triangulaire (carte de gauche), sphérique (carte du milieu) et carrée (carte de droite).

On distingue sur chacune des cartes de la figure IV.3, quatre régions. Les frontières apparaissant entre ces régions sont plus épaisses dans les représentations de z_{MAX} que dans celles de z_{MED}.

Les cartes issues des différents apprentissages font apparaître des régions qui ne sont

pas localisées aux mêmes endroits sur les différentes cartes. Sur les représentations de z_{MAX} et z_{MED} calculés avec les distance sphérique et carrée, on distingue nettement les quatre régions. Avec les représentations de z_{MAX} et z_{MED} associées à la distance triangulaire, ces quatre régions apparaissent moins distinctement et l'opérateur peut hésiter entre quatre et cinq classes. En visualisant les six projections, on opte plus volontiers vers la présence de quatre classes.

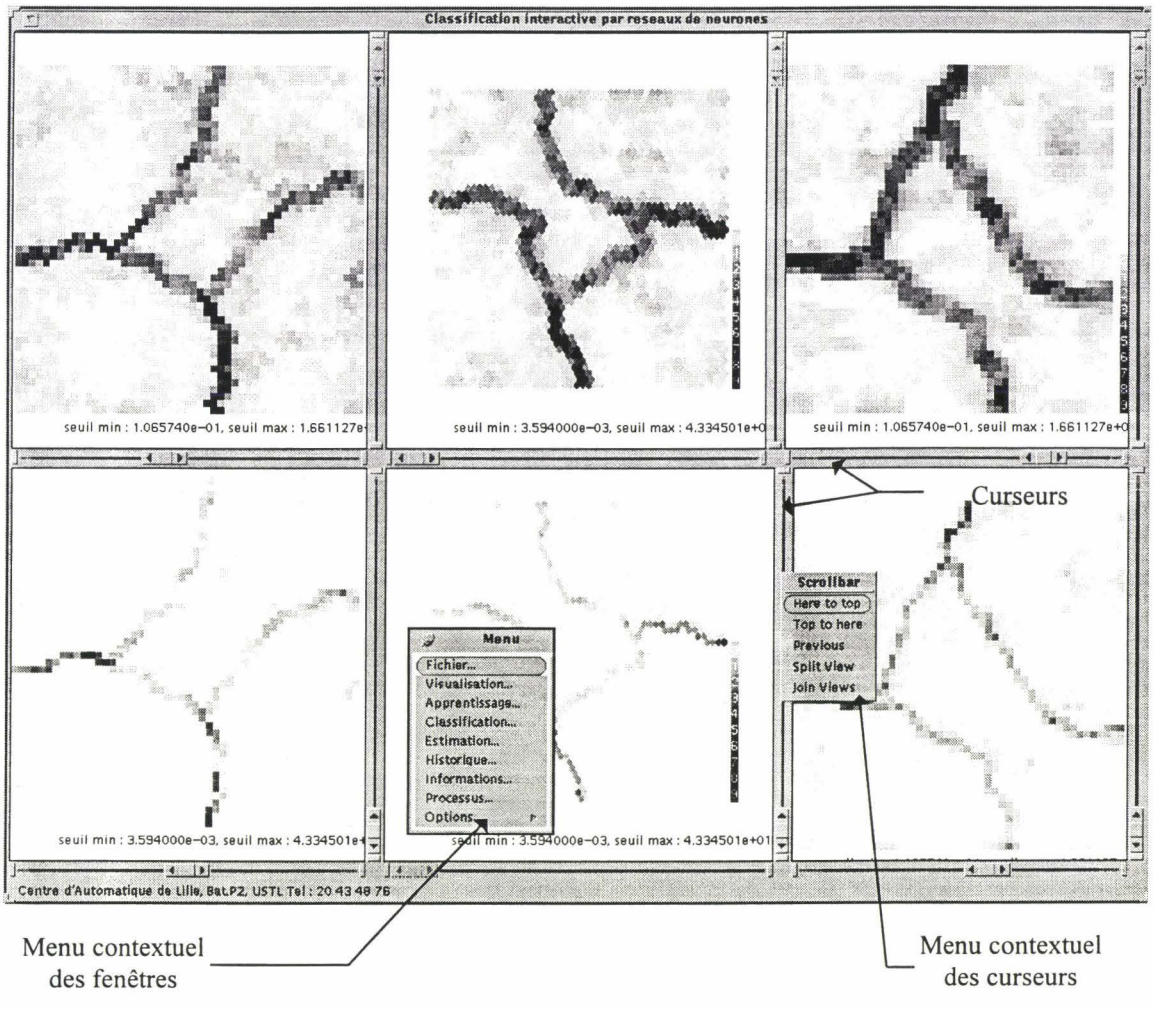


Figure IV. 3 : Interface de visualisation multifenêtrage de données multidimensionnelles

IV.2.2 Outils de seuillage

Les techniques de visualisation font apparaître l'échantillon multidimensionnel sous la forme d'images numériques en niveaux de gris, ou en pseudo-couleurs, ce qui permet de visualiser différents types d'informations. Ainsi, par l'intermédiaire des cartes, on peut représenter des informations concernant les distances inter-neurones, des estimations de la fonction densité de probabilité ou encore les observations classées. Afin d'aider à l'analyse

visuelle de ces représentations, nous proposons des outils de seuillage pour mettre les classes en évidence. Ces outils apparaissent sous forme d'un menu à l'intérieur des fenêtres (cf. Figure IV.4).

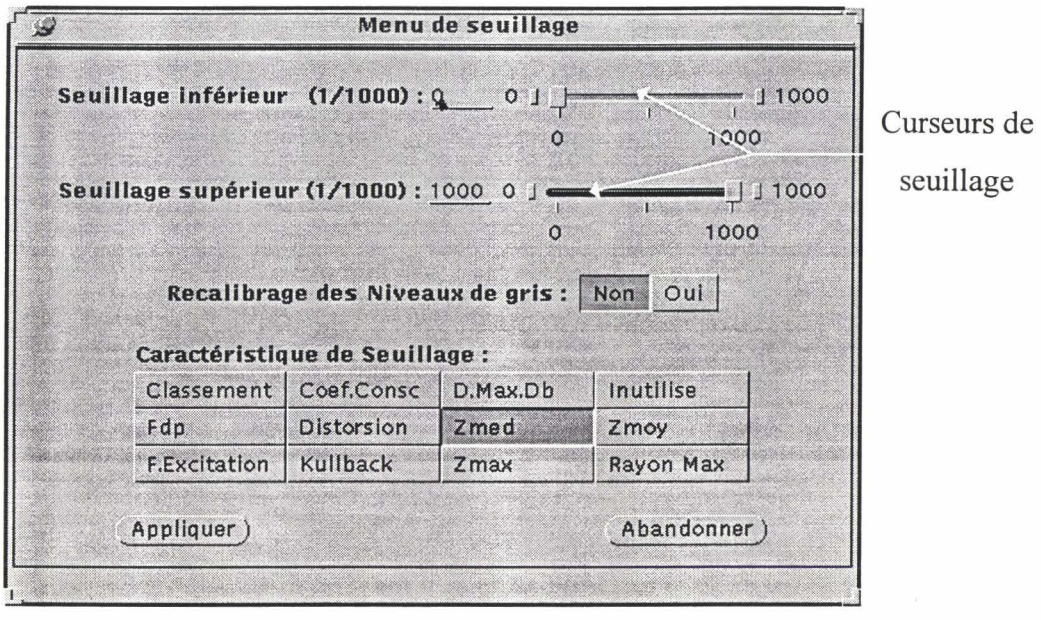


Figure IV. 4 : Menu de seuillage

Sur la figure IV.4 apparaissent deux curseurs. Celui situé en haut du menu permet de fixer la valeur d'un seuil appelé seuil bas. Lorsque l'opérateur fixe ce seuil, les neurones dont la valeur relative de la caractéristique est inférieure au seuil bas sont éliminés de la carte. Sur l'écran, ces neurones sont mis en blanc et se confondent avec le fond de la fenêtre. Cette technique de seuillage par valeurs inférieures permet d'extraire les frontières. Ainsi la figure IV.5 présente la carte représentant z_{MAX} (cf. fenêtre en haut et au centre de la figure IV.3) sur laquelle on a appliqué cette technique de seuillage. Les régions sont en blanc et les frontières apparaissent distinctement.

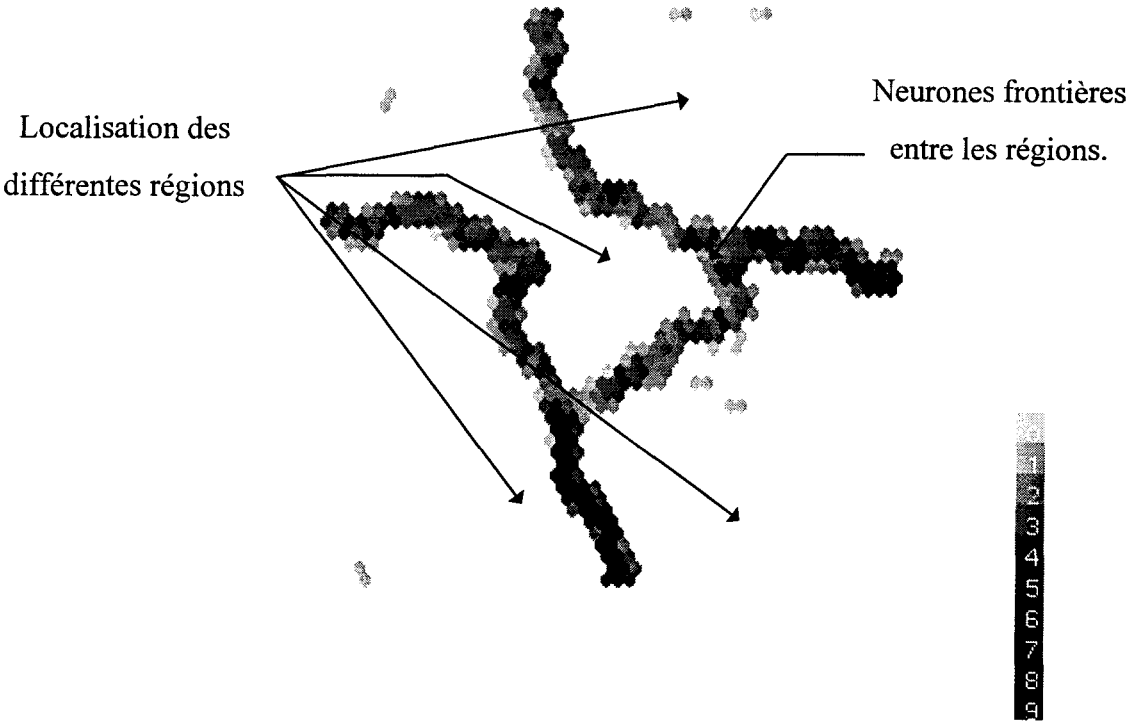


Figure IV. 5 : Seuillage par valeurs inférieures

Le second curseur de la figure IV.4 permet de fixer la valeur d'un seuil appelé seuil haut. Lorsque l'opérateur fixe ce seuil, les neurones dont la valeur de la caractéristique est supérieure au seuil haut sont éliminés de la carte. Cette technique de seuillage par valeurs supérieures permet d'extraire les régions. La figure IV.6 présente la carte représentant z_{MAX} (cf. fenêtre en haut et au centre de la figure IV.3) sur laquelle on a appliqué cette technique de seuillage. Les frontières sont en blanc et les régions apparaissent distinctement.

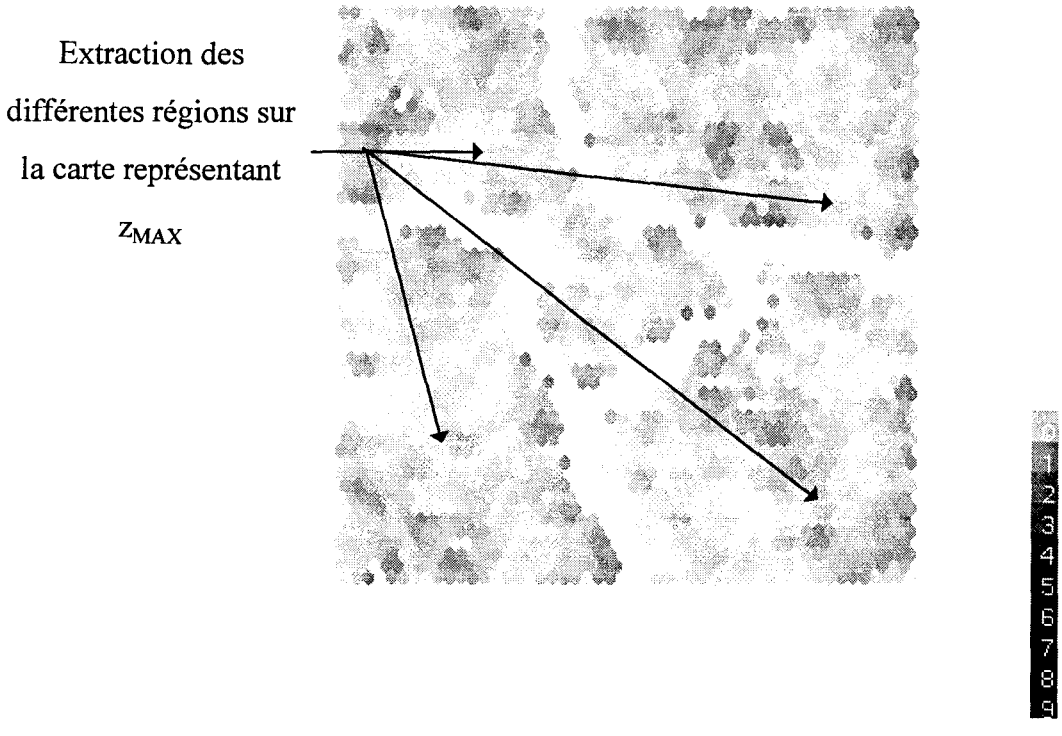
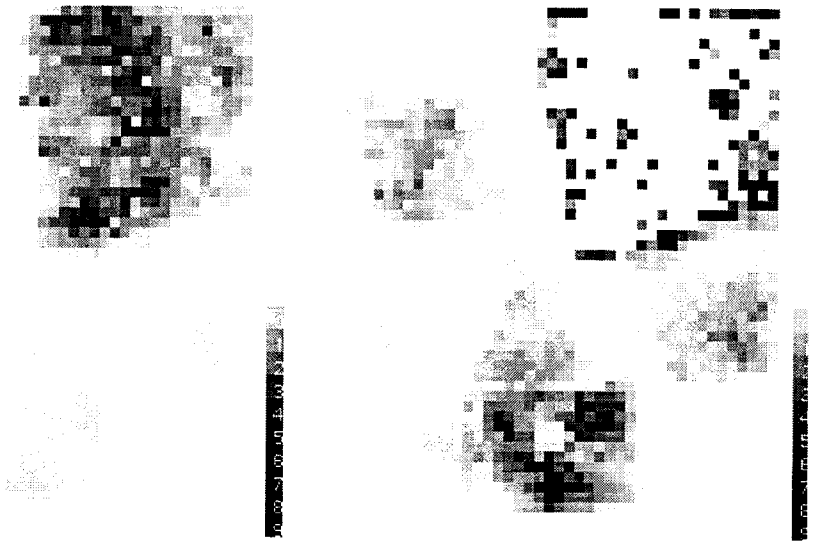


Figure IV. 6 : Seuillage par valeurs supérieures

En dessous des curseurs de la figure IV.4, se situent deux boutons permettant de valider ou d'invalider le recalibrage des niveaux de gris. On peut ainsi exploiter toute la dynamique des niveaux de gris pour améliorer les représentations. Le recalibrage des niveaux de gris s'effectue en fonction des valeurs relatives des seuils bas et haut. Les neurones dont la valeur relative de la caractéristique est supérieure au seuil haut ou inférieure au seuil bas sont éliminés. Le niveau de gris des neurones dont la valeur est comprise entre le seuil bas et le seuil haut est recalé pour utiliser toute la dynamique de l'image. Pour illustrer l'utilité d'un tel recalage, considérons l'exemple représenté sur la figure IV.7(a). Il s'agit d'une représentation d'une estimation de la fonction densité de probabilité sous-jacente à un échantillon composé de quatre classes non équiprobables dans un espace à 6 dimensions. Sur la carte de gauche, on distingue une région sombre indiquant la présence d'un mode de la fonction densité de probabilité. En utilisant la technique de seuillage avec recalage des niveaux de gris, on élimine les neurones dont la valeur relative de la fonction densité de probabilité estimée est supérieure à la valeur du seuil haut fixé par l'opérateur. Ensuite, on recale la dynamique des niveaux de gris en fonction des seuils bas et haut. La carte de droite de la figure IV.7(b) illustre la représentation de la fonction densité de probabilité avec l'emploi de cette technique.

On distingue maintenant trois autres régions sombres indiquant la présence de trois modes de la fonction densité de probabilité.



- (a) Représentation de la fonction densité de probabilité sans recalage des niveaux de gris -

- (b) Représentation de la fonction densité de probabilité avec recalage des niveaux de gris -

Figure IV. 7 : Exemple de seuillage avec recalage des niveaux de gris

Enfin, dans le menu de seuillage apparaît un sous menu de sélection de la caractéristique de seuillage (cf. Figure IV.4). Les caractéristiques présentées dans le chapitre III apparaissent sous forme de boutons. Certaines de ces caractéristiques n'ont pas de rapport direct avec la classification interactive et ne sont donc pas détaillées dans ce mémoire. Nous pouvons constater aussi qu'il est possible d'ajuster les seuils en fonction des numéros de classe. Cela permet de faire apparaître séparément les classes sur la carte. Il est à noter, de plus, qu'il est possible de représenter une caractéristique et d'utiliser les techniques de seuillage à partir d'une autre caractéristique cela afin de comparer visuellement l'information apportée par les caractéristiques représentées et seuillées. Nous détaillerons ce type de seuillage et son utilité dans le paragraphe IV.4.

Les outils de seuillage présentés dans ce paragraphe sont simples. Il est possible d'utiliser des techniques de seuillage plus élaborées. Daoudi a exploité, dans le cadre de la classification interactive des techniques de seuillage basées sur la morphologie mathématique [DAOU 93].

IV.2.3 Analyse des cartes et initialisation des algorithmes de classification

Avec l'aide des outils présentés dans le paragraphe précèdent, l'analyste découvre le nombre de classes et pour initialiser les vecteurs poids du réseau de neurones à apprentissage compétitif pour la classification.

L'opérateur commence par déterminer le nombre de classes en présence en relevant le nombre de régions sur les cartes. L'examen visuel des différentes cartes apparaissant dans la figure IV.3 nous indique la présence de quatre classes. En plus des caractéristiques représentant les distances inter-neurones, il est possible d'étudier la fonction densité de probabilité. Dans ce cas, la présence d'une classe va coïncider avec la présence d'un maximum de cette fonction sur la carte. Ainsi le nombre de maxima locaux significatifs de cette fonction sur la carte va correspondre au nombre de modes de l'échantillon, car on peut supposer qu'un mode est représentatif d'une classe [ASSE 89], [POST 82b]. L'analyse visuelle des représentations de la fonction densité de probabilité est surtout à envisager lorsque que l'on ne peut plus discerner, à l'aide des représentations des caractéristiques de distances inter-neurones, la présence de classes. Dans ce cas cette analyse permet de déceler d'éventuelles classes présentant un degré de chevauchement important. Dans la méthodologie que nous proposons, nous classons d'abord les classes distinctes c'est à dire celles apparaissant sur les représentations des caractéristiques de distances inter-neurones. Ces classes ont un degré de chevauchement très faible. Puis lors de l'analyse locale des classes et s'il n'y a plus d'éventuelles classes distinctes, nous étudions la fonction densité de probabilité relative à la distribution des observations de chaque classe. En procédant de cette façon, nous obtenons de meilleures représentations de la fonction densité de probabilité. Nous aborderons un cas de classes présentant un degré de chevauchement important dans le chapitre V.

Après avoir déterminé le nombre de classes, il convient de déterminer leurs centres apparents afin d'initialiser le réseau de neurones classifieur.

Pour cela, l'opérateur, à l'aide de la souris, clique sur une carte, que nous appelons carte d'initialisation, aux endroits qu'il estime être représentatifs des centres des classes. On attribue un numéro de classe à chaque centre sélectionné. Ainsi, lors de la classification, on assigne à chaque observation le numéro représentatif de cette classe.

La figure IV.8 représente un exemple d'initialisation. Notre initialisation s'effectue à partir d'une représentation de la fonction densité de probabilité. Sur cette représentation on

distingue les différents numéros de classes aux endroits où l'opérateur a cliqué avec la souris. Ces endroits se situent dans des régions sombres de la carte représentant en l'occurrence des régions où les valeurs de la fonction densité de probabilité sont élevées. On peut ainsi considérer que les vecteurs poids des neurones sélectionnés se situent près des centres des classes. On initialise ainsi le réseau de neurones classifieur avec les vecteurs poids des neurones se situant dans des régions où les valeurs de la fonction densité de probabilité sont importantes.

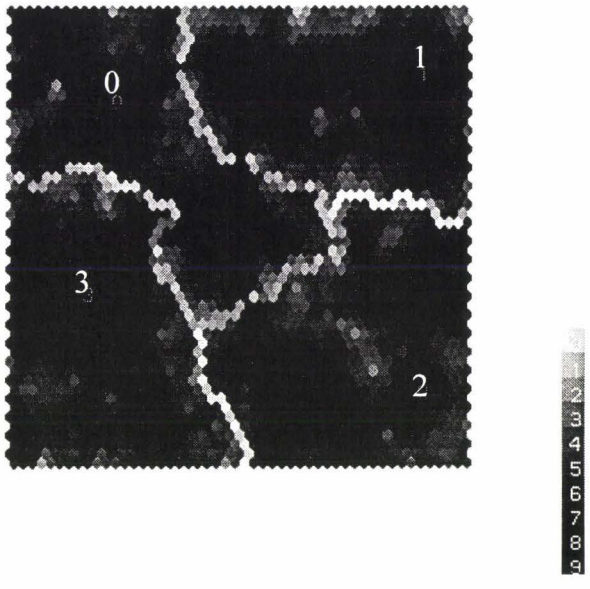


Figure IV. 8 : Initialisation des vecteurs poids pour la classification

Il est possible d'effectuer l'initialisation à partir de représentations caractérisant les distances inter-neurones. Dans ce cas, on sélectionnera de préférence les neurones se situant au centre des régions. Cependant, dans ce cas, il se peut que les vecteurs poids des neurones sélectionnés soient relativement éloignés du centre réel de la classe.

IV.3 CLASSIFICATION

A ce stade du traitement, nous disposons de K neurones que l'on estime être représentatifs des K classes de l'échantillon. L'utilisation des techniques de représentation plane permet de classer l'échantillon d'observations en effectuant soit une classification directement sur la carte d'initialisation, donc dans un espace à deux dimensions, soit dans l'espace de représentation des observations et donc dans un espace à N dimensions.

IV.3.1 Classification sur la carte

L'analyste, en utilisant les techniques de seuillage, dispose d'une carte où apparaissent les frontières délimitant des régions distinctes indiquant la présence des classes. Une solution séduisante consiste à considérer que les neurones d'une même région appartiennent à la même classe. L'assignation des neurones sur la carte est alors immédiate.

La classification des observations de l'échantillon s'effectue ensuite en assignant les observations à la classe du neurone classé le plus proche. On peut résumer cette technique de classification selon le schéma suivant :

Algorithme de classification sur la carte :

- Délimiter par l'intermédiaire des outils de seuillage les régions représentant les classes.

- Assigner tous les neurones constitutifs d'une même région à la classe associée à cette région.

- Attribuer les observations de la zone d'influence de chaque neurone à la classe de ce neurone. Ainsi :

Soit $X \in \mathcal{X}$, *et* m^* *vérifiant :*

$$m^* = \underset{m \in [0,M[}{Arg\,min}\,(d^E(X, W_m))$$

Si $W_{m^*} \in C_k$, *alors on assigne* X *à la classe* C_k, $X \in C_k$

Cette technique de classification est très rapide. Cependant elle présente certains inconvénients :

- Tout d'abord, elle n'assigne pas les neurones frontières éliminés lors du seuillage. Les observations de l'échantillon activant ces neurones, ne sont pas classées. Il faut, dans ce cas, recourir à une autre technique de classification pour classer ces observations.

- La technique peut aussi engendrer des erreurs de classification. Ainsi, il arrive que des neurones frontières soient activés par des observations provenant de classes différentes. Si l'on utilise cette technique de classification à partir de la carte, alors toutes ces observations sont attribuées automatiquement à la même classe. Cela entraîne des erreurs uniquement dues à cette technique de classification.

- Enfin, cette méthode suppose que les cartes utilisées résultent d'une convergence correcte des procédures. Autrement dit, elles doivent constituer une image aussi fidèle que possible de l'échantillon, ce qui n'est pas toujours assuré. Bien évidemment une classification sur une mauvaise carte conduit à une classification erronée.

Une autre méthode, permettant d'éviter ces inconvénients, consiste à utiliser les vecteurs poids des neurones sélectionnés par l'opérateur afin d'initialiser des algorithmes de classification pour attribuer les observations aux différentes classes dans leur espace de représentation d'origine.

IV.3.2 Classification dans l'espace d'origine

Contrairement à la technique précédente, la classification peut être effectuée directement sur l'échantillon dans l'espace d'origine. Il s'agit, dans ce cas, de s'aider des informations apportées par la carte afin d'initialiser au mieux les méthodes de classification neuronales ou classiques. Pour cela, l'analyste sélectionne, à l'aide de la souris, les différents neurones qu'il considère représenter les centres des classes. Ainsi, les vecteurs poids associés aux neurones sélectionnés lors de la phase d'initialisation sont les vecteurs poids initiaux dans les réseaux à apprentissage compétitif ou les vecteurs centres initiaux de l'algorithme des K-means.

Algorithme de classification dans l'espace de représentation d'origine :

- Sélectionner les différents neurones représentant les centres des classes sur la carte.

- Initialiser l'algorithme de classification avec les vecteurs poids des neurones sélectionnés par l'analyste.

- Attribuer les observations de l'échantillon $\mathcal{X}$ aux différentes classes mises en évidence.

La figure IV.9 représente la projection de l'échantillon classé par cette technique. Nous pouvons constater que les régions où sont localisées les classes coïncident avec les régions apparaissant dans la représentation de la fonction densité de probabilité (cf. Figure

IV.8). La comparaison de l'échantillon classé avec l'échantillon étiqueté révèle un taux d'erreur de classement nul, ce qui indique, par ailleurs, qu'il n'y a aucun chevauchement entre les classes.

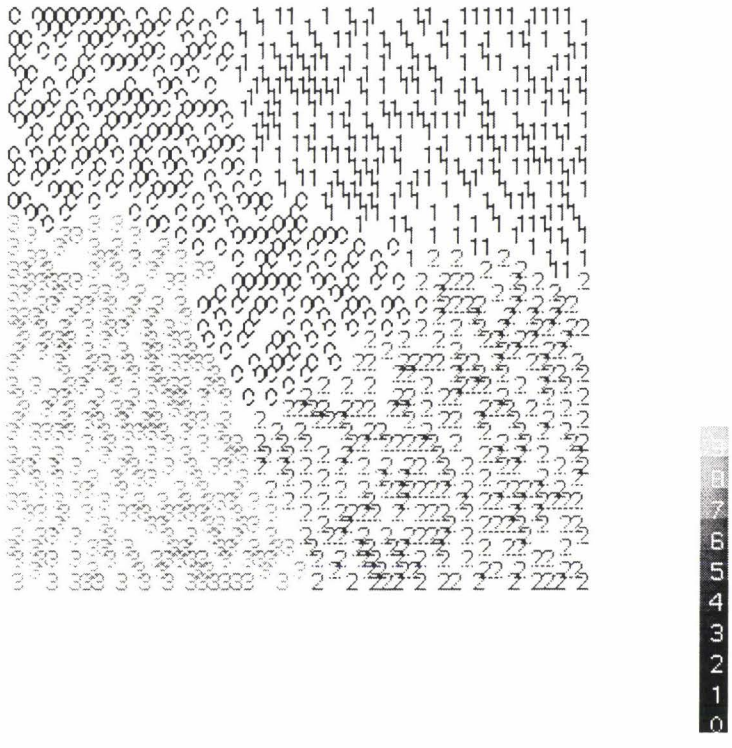


Figure IV. 9 : Projection de l'échantillon classé

Remarque :

L'objectif est bien entendu de classer l'échantillon d'observations. Néanmoins nous pouvons aussi classer l'échantillon des vecteurs poids. Cela peut être utile si l'on utilise la carte en tant que classifieur. Dans ce cas, une observation inconnue sera attribuée à la classe du vecteur poids le plus proche. D'autre part, la visualisation de la projection des poids classés permet de distinguer nettement les frontières entre classes. La figure IV.10 représente la projection de l'échantillon des vecteurs poids classé par un réseau de neurone à apprentissage compétitif dans l'espace d'origine. Nous utilisons les vecteurs poids sélectionnés par l'opérateur pour initialiser l'algorithme de classification par réseaux de neurones à apprentissage compétitif. Tous les vecteurs poids des neurones sont assignés à l'une des 4 classes décelées. Notons que la classification des vecteurs poids dans l'espace d'origine permet de classer les neurones frontières. En effet, il n'est pas besoin d'utiliser les techniques de seuillage qui éliminent ces neurones. Ainsi, la classification s'effectue sur l'ensemble des

vecteurs poids. La projection de l'échantillon des vecteurs poids classés par cette technique permet alors d'obtenir une frontière nette entre les classes. Par ailleurs, comme tous les neurones sont classés, il est possible de classer une observation inconnue en lui assignant le numéro de classe du neurone activé par cette observation.

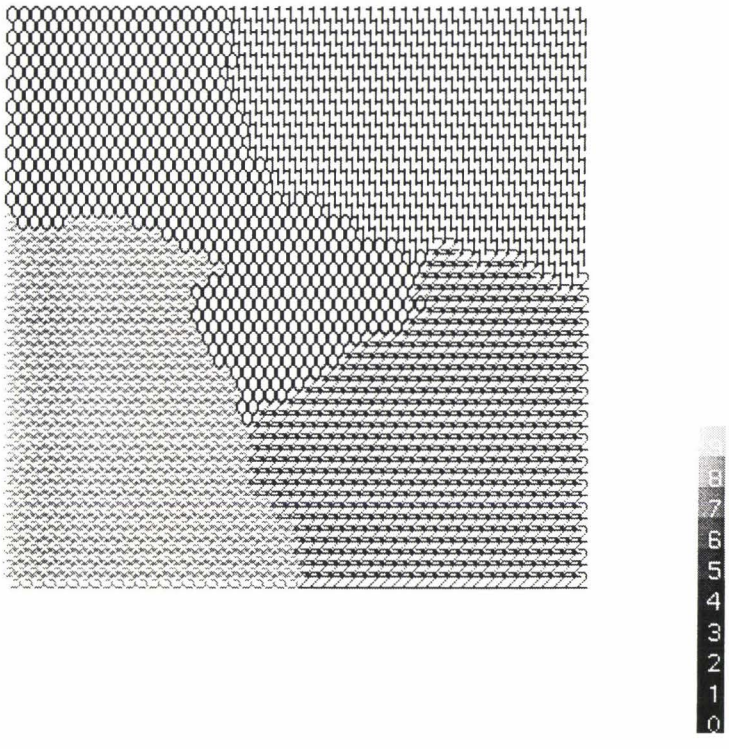


Figure IV. 10 : Projection de l'échantillon des vecteurs poids classés

IV.4 VERIFICATION PAR PROJECTION DE L'ECHANTILLON CLASSE

IV.4.1 Technique de vérification

A ce stade de l'analyse, nous avons classé l'échantillon d'observations. Pour cela nous nous sommes basés sur les représentations par cartes de Kohonen qui nous permettent de déterminer le nombre de classes sous-jacentes et d'initialiser les algorithmes de classification. Comme nous classons l'échantillon dans l'espace de représentation des observations, il convient de projeter l'échantillon classé sur les différentes cartes afin de s'assurer que les régions de projection des classes correspondent aux régions décelées sur les représentations des différentes caractéristiques locales. Pour illustrer cette technique de vérification, nous

utilisons la représentation de z_{MAX} sur une carte à maillage hexagonal (cf. Figure IV.11(a)), mais on peut, bien entendu, utiliser d'autres caractéristiques et d'autres cartes. Sur la projection de l'échantillon classé de la figure IV.11(b), on discerne les régions où sont concentrées les observations d'une même classe. En examinant la représentation de z_{MAX} et cette projection de l'échantillon classé, on constate que les régions apparaissant sur les deux cartes se confondent. Cependant, on peut également utiliser les techniques de seuillage afin de vérifier visuellement que les régions apparaissant sur les deux cartes se correspondent. Ainsi la carte de la figure IV.11(c), qui représente l'échantillon classé, a été seuillée par valeurs supérieures, de telle sorte que les neurones en blanc ont des valeurs relatives de z_{MAX} supérieures au seuil haut. Les neurones éliminés de cette carte sont les neurones frontières. Nous pouvons constater que les régions de la figure IV.11(c) sont toutes composées de projections d'observations provenant de mêmes classes. Dans ce cas, on considère que les régions de ces deux représentations coïncident et que les résultats de la classification sont cohérents. La vérification sur les autres cartes fait apparaître de la même manière la cohérence des résultats de la classification.

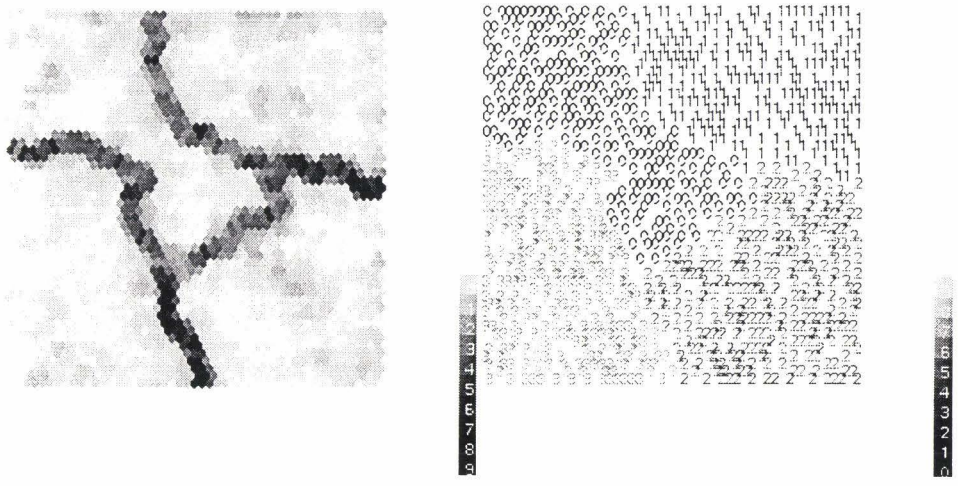
Dans le prochain chapitre, nous vérifions aussi la cohérence de la classification au moyen d'autres types de projections, tels que la projection par la méthode de Sammon ou la méthode d'analyse en composantes principales.

Vérification de la cohérence des résultats de la classification à l'aide des cartes :

- Projeter l'échantillon classé sur les différentes cartes

- Seuiller les cartes à partir soit d'une caractéristique évaluant les distances inter-neurones, soit d'une fonction densité de probabilité jusqu'à ce que les régions apparaissent séparées les unes des autres.

- Les résultats de la classification concordent si les régions isolées sont toutes constituées de projections d'observations provenant d'une même classe.



- a : Représentation de z_{MAX} - - b : Projection de l'échantillon classé -

-c : Représentation de l'échantillon classé seuillée selon z_{MAX} -

Figure IV. 11 : Vérification de la cohérence des résultats de la classification

Cette technique de vérification fait apparaître deux cas de figure selon qu'il existe un degré de chevauchement important ou non entre les classes.

Lorsque les classes ont un faible degré de chevauchement celles-ci apparaissent distinctement lors de l'analyse des représentations des caractéristiques de distances inter-neurones. La vérification des résultats de classification s'effectue alors à partir de ces représentations car elles permettent d'éliminer les neurones frontières et, de ce fait, la vérification est plus précise que celle obtenue à partir d'une représentation de la fonction densité de probabilité. Cependant, elle ne permet de juger les résultats de la classification pour les neurones frontières. De plus, il se peut qu'il existe un nombre restreint d'observations

isolées au sein d'autres classes. Dans ce cas, c'est l'appréciation de l'opérateur qui décide si la classification de l'échantillon d'observations est cohérente.

Dans le cas de l'examen de classes présentant un chevauchement important, seules les représentations de la fonction densité de probabilité sont susceptibles de faire apparaître les modes des classes. On considère dans ce cas que les résultats de la classification sont cohérents lorsque les régions définissant les modes sont toutes constituées d'observations provenant d'une même classe. Il est évident dans ce cas que la technique de vérification est limitée aux modes de la fonction densité de probabilité.

Nous allons maintenant aborder le cas où les régions décelées sur la carte d'initialisation ne coïncident pas avec les régions de projection des observations des différentes classes.

IV.4.2 Cas d'incohérence dans les résultats de la classification

Il peut arriver que les projections des observations classées appartenant à une même classe se répartissent entre plusieurs régions décelées. Ces incohérences peuvent provenir, dans l'ordre de traitement du processus de classification interactif, soit de la mauvaise convergence de la carte lors de sa phase d'apprentissage, soit d'une mauvaise initialisation, soit de l'utilisation inadaptée de l'algorithme de classification. Ainsi, en cas d'incohérence dans les résultats de la classification, le premier maillon du processus à mettre en cause est la qualité de représentation de la carte d'initialisation.

IV.4.2.1 Cas d'une mauvaise carte

Lors de la phase d'apprentissage, le réseau peut ne pas converger correctement en introduisant notamment des distorsions importantes au niveau de la préservation de la structure de l'échantillon. Dans ce cas les observations d'une classe peuvent être présentes dans plusieurs régions distinctes de la carte. Il convient alors d'utiliser d'autres cartes d'initialisation issues d'autres apprentissages. Si, sur ces autres cartes, les résultats de la classification sont cohérents, alors l'hypothèse de la mauvaise carte est vérifiée. Si par contre, les résultats des classifications issus de l'analyse de ces autres cartes se révèlent également incohérents, il convient de mettre en cause les autres maillons du traitement.

IV.4.2.2 Cas d'une mauvaise initialisation

En général, une mauvaise initialisation est due à une mauvaise qualité de la carte. Cependant on peut étudier la stabilité des résultats obtenus en réalisant des classifications avec des initialisations différentes basées sur d'autres cartes. Si les résultats obtenus sont toujours incohérents, il convient de mettre en cause le maillon suivant du processus de classification interactif : les algorithmes de classification.

IV.4.2.3 Algorithme de classification non adapté

Il peut arriver que l'algorithme de classification ne soit pas adapté à l'échantillon. Ainsi, on sait par expérience que le réseau à apprentissage compétitif et l'algorithme des K-means ne sont pas adaptés aux classes de formes non globulaires. Dans ce cas, les projections des observations classées obtenues sur toutes les cartes seront incohérentes. La solution consiste à utiliser une autre méthode de classification, comme par exemple l'algorithme du plus proche [DUDA 73]. Si l'incohérence des résultats de la classification est due à l'algorithme de classification, alors l'utilisation d'autres algorithmes de classification doit fournir des résultats cohérents.

IV.4.2.4 Conclusions

Nous avons montré comment se manifeste une incohérence dans les résultats d'une classification et exploré, à l'intérieur de ce paragraphe, les raisons possibles de ces incohérences. Nous distinguons essentiellement deux cas d'incohérences. Ainsi, il arrive que toutes les projections de l'échantillon classés sont incohérentes. Dans ce cas, il convient de tester d'autres techniques de classification. Le second cas se produit lorsque certaines projections de l'échantillon classé se révèlent incohérentes. Dans ce cas, on peut mettre en cause la mauvaise convergence des réseaux lors de la phase d'apprentissage entraînant ainsi des distorsions. Le recours à d'autres cartes et d'autres techniques de projection permet de vérifier cette hypothèse.

IV.5 ANALYSE LOCALE DES CLASSES

A ce stade, il convient d'analyser localement les classes en raison principalement de la qualité de représentation des cartes de Kohonen qui est conditionnée par deux paramètres indépendants de la phase d'apprentissage : l'échelle de niveaux de gris et le nombre I×J de neurones composant la carte. Nous avons vu précédemment que nous pouvons résoudre les problèmes d'affichage par niveaux de gris en recalant la dynamique des niveaux de gris entre deux valeurs de seuils pour apercevoir les classes moins distinctes (cf. § IV.2.2).

Une carte de Kohonen ayant un nombre de neurones insuffisant peut aussi rendre les frontières entre classes indécelables lors de l'examen visuel. En réexaminant chacune des classes isolées lors de l'analyse globale, nous affinons leurs représentations.

Cette méthodologie d'analyse de données à plusieurs échelles est analogue à l'analyse multi-résolution, utilisant un modèle pyramidal, employée en traitement d'images. Les images de faible résolution permettent de traiter les principales régions constituant l'image tandis que les images de haute résolution permettent un traitement plus affiné de ces régions [COQU 95]. Cette structure pyramidale permet de faciliter l'extraction d'informations et constitue une étape vers une structuration de plus haut niveau. La construction de la structure pyramidale est laissée à l'initiative de l'opérateur qui peut affiner son analyse selon les connaissances qu'il dispose des données.

Par ailleurs, différents algorithmes de classification peuvent être employés en fonction de la classe analysée. La classification complète de l'échantillon peut être obtenue en combinant ces algorithmes de classification à différents niveaux de l'analyse. La figure IV.12 illustre un exemple d'analyse où l'opérateur classe son échantillon à différents niveaux. D'après cette figure, l'analyste découvre six classes qui se situent aux branches terminales de l'arborescence.

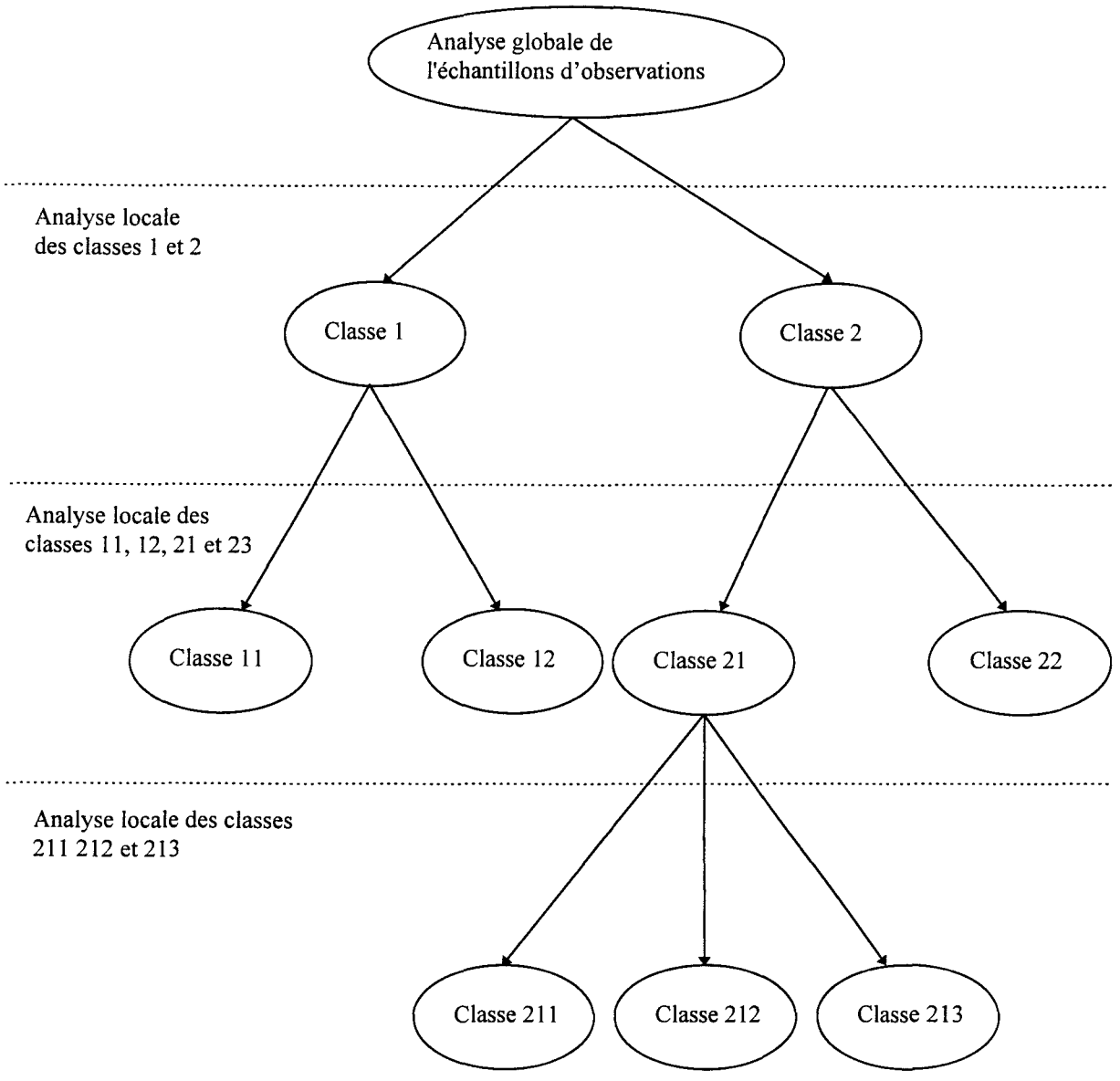


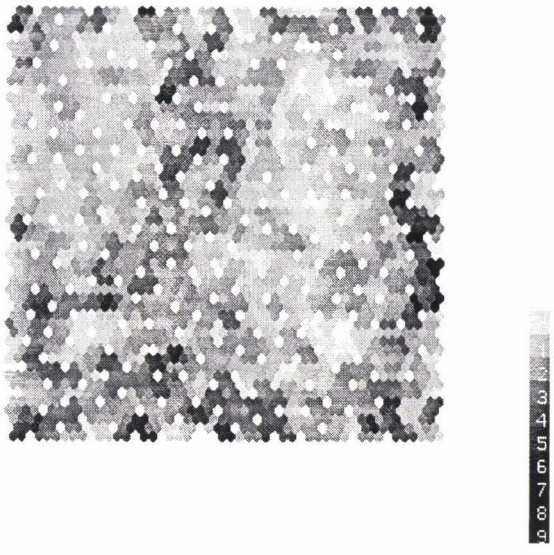
Figure IV. 12 : Exemple d'analyse d'un échantillon

Nous avons classé l'échantillon de l'exemple 2 dans lequel nous avons découvert les quatre classes. L'erreur de classement obtenu est nulle puisqu'il n'y pas de chevauchement entre les classes. Comme nous nous plaçons dans un cadre non supervisé, il s'agit de vérifier s'il n'existe pas de sous-classes dans les classes obtenues. Aussi, il convient d'analyser séparément chaque classe afin de déceler la présence éventuelle d'autres classes. Pour chacune de ces classes, on effectue de nouveaux apprentissages de cartes.

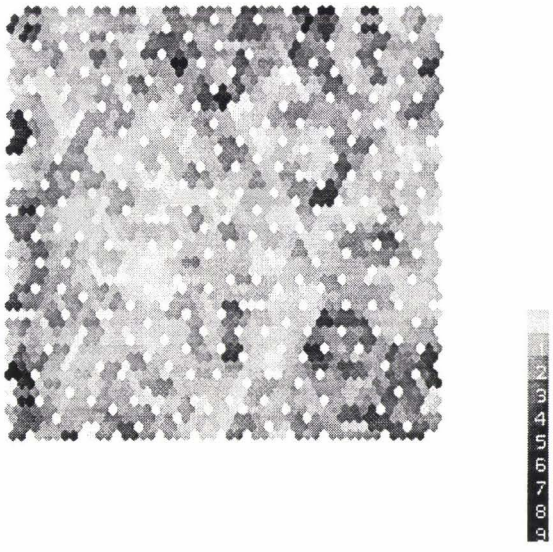
L'interface graphique permet, après une première de décomposition globale de l'échantillon en classes, d'effectuer les différents traitements pour chaque classe. La figure

IV.12 montre la représentation de z_{MAX} pour chaque classe traitée indépendamment. Pour obtenir une représentation d'une classe, nous effectuons un apprentissage avec les observations provenant cette classe. Ainsi sur la figure IV.12, nous avons quatre cartes représentant chacune une des classes décelée lors de l'analyse de l'échantillon complet. Ainsi chacune de ces cartes est spécifique à une classe. On constate sur ces cartes qu'on ne peut discerner la présence de sous-classes. Afin d'examiner le cas éventuel de l'existence de classes présentant un chevauchement important, il convient d'étudier la fonction densité de probabilité sous-jacente aux quatre classes. Pour cela, nous représentons pour chaque carte spécifique à une classe, la fonction densité de probabilité sous-jacente à cette classe. La représentation des quatre fonction densité de probabilité (cf. Figure IV.13) ne permet pas non plus de distinguer la présence d'autres classes.

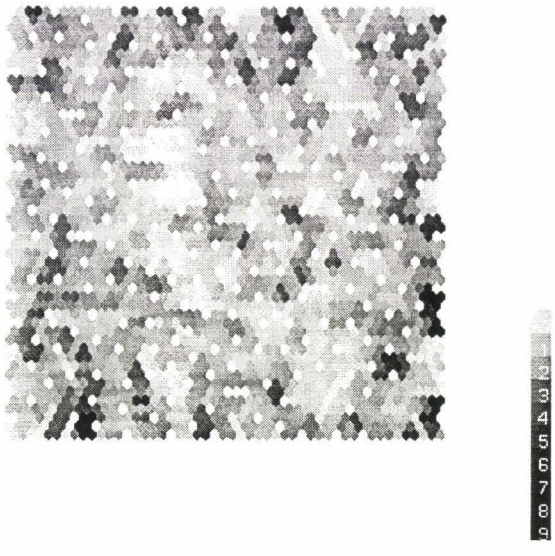
L'analyse locale des classes ne révèle pas la présence de classes supplémentaires. Nous en déduisons que l'échantillon est effectivement composé de quatre classes et les résultats obtenus sont satisfaisant puisque nous avons décelé les quatre classes générées artificiellement. L'analyse de l'échantillon est terminé.



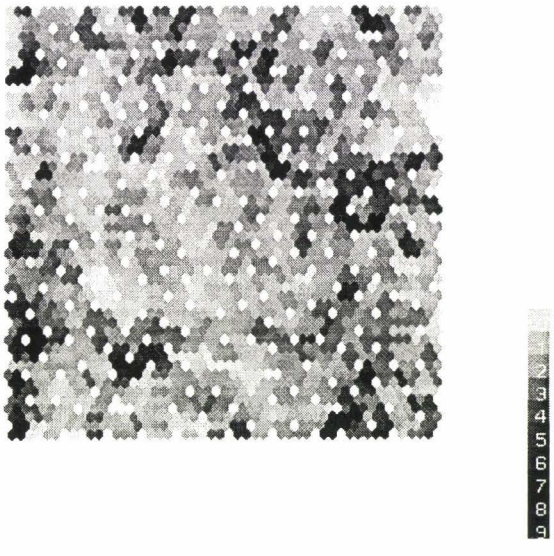
- classe 0-



- classe 1-

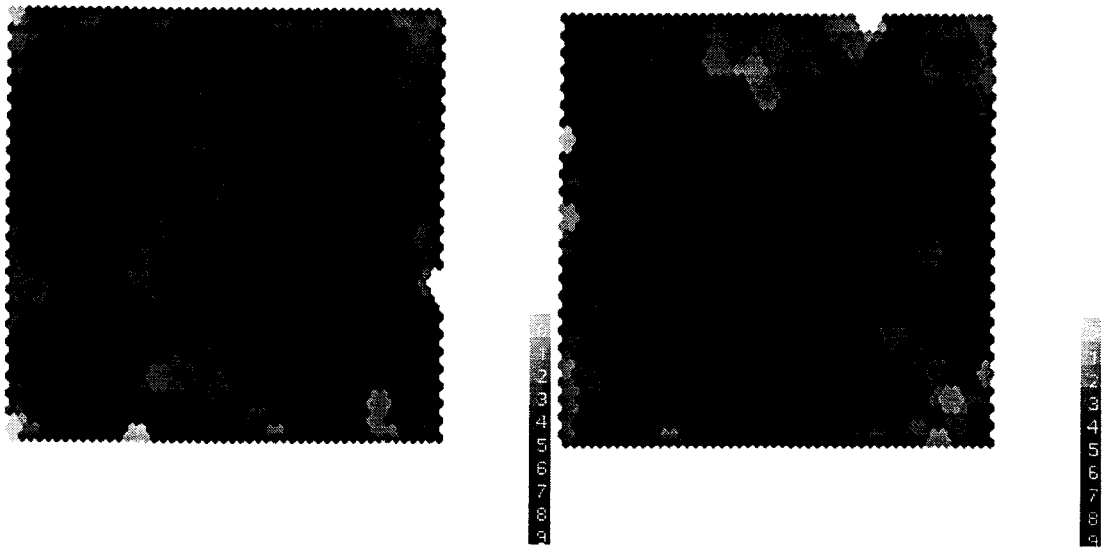


- classe2 -



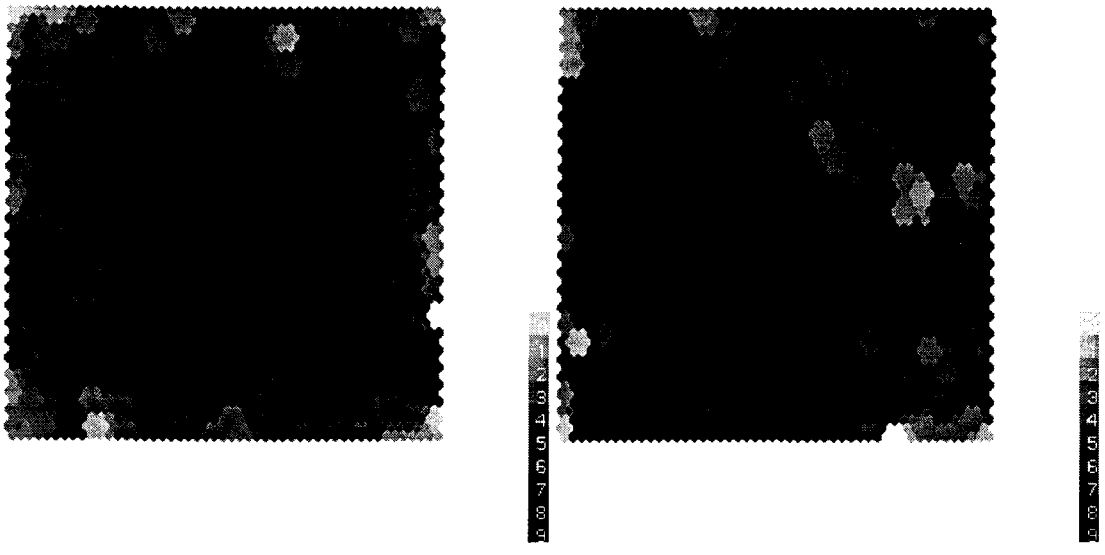
- classe 3 -

Figure IV. 13 : Représentation de z_{MAX} pour chaque classe



- classe 0-

- classe 1-



- classe 2 -

- classe 3 -

Figure IV. 14 : Représentation de la fonction densité de probabilité sous-jacente à chaque classe

IV.6 CONCLUSIONS

Nous avons décrit, au cours de ce chapitre, la méthodologie employée pour l'analyse d'un échantillon d'observations multidimensionnelles dans un contexte non supervisé. Elle consiste à visualiser, au moyen de la carte de Kohonen, certaines caractéristiques représentant

l'ensemble des distances inter-neurones ainsi que des estimations de la fonction densité de probabilité sous-jacente à l'échantillon d'observations. Une interface graphique a été développée afin que l'analyste puisse visualiser sur le même écran ces différentes représentations. De plus, des techniques de seuillage lui facilitent l'examen visuel de ces cartes. Après l'examen visuel des différentes cartes, l'analyste initialise, à l'aide de la souris, les algorithmes de classification. Après la classification de l'échantillon, il effectue une vérification en projetant l'échantillon classé sur les différentes cartes et compare les régions initialement repérées avec les régions apparaissant sur la projection de l'échantillon classé. Lorsque les régions se correspondent, les résultats de la classification sont cohérents. Dans le cas contraire, l'analyste, en mettant en cause les différents maillons de traitement, peut découvrir les raisons de ces incohérences. Enfin, une étude locale des classes permet de découvrir les classes invisibles lors de l'analyse de l'échantillon complet.

L'échantillon de l'exemple 2 nous a permis d'illustrer les différentes étapes de la méthodologie proposée. Cependant, cet exemple ne nous a pas permis d'explorer toutes les situations qu'un analyste peut rencontrer. Aussi, dans le chapitre suivant, nous nous proposons de tester les techniques exposées dans ce travail sur des exemples plus délicats à classer.

CHAPITRE V :

RESULTATS EXPERIMENTAUX

Comme nous l'avons vu dans les chapitres précédents, les réseaux de neurones à apprentissage compétitif permettent, en utilisant les algorithmes d'apprentissage adéquats, d'être utilisés en tant qu'algorithmes de classification, de réduction de données et de représentation de données multidimensionnelles. Les différentes possibilités qu'offrent ces réseaux nous ont amenés à construire une interface graphique permettant d'effectuer facilement les multiples opérations d'un processus de classification interactif. Le logiciel élaboré comporte des outils interactifs particulièrement adaptés aux représentations par cartes de Kohonen. L'objet de ce chapitre consiste à tester l'ensemble des outils décrits précédemment sur des exemples pour montrer leur intérêt et mettre leurs performances et leurs limites en évidence. L'analyse en composantes principales (cf. Annexe 3), l'algorithme de Sammon (cf. Annexe 4) et une technique de projection par réseau de neurones multicouches en mode auto-associatif (cf. Annexe 5) seront utilisés pour représenter, en deux dimensions, les différents échantillons. Ainsi, nous pourrons, d'une part, comparer ces projections avec les techniques de représentation par cartes de Kohonen et, d'autre part, offrir à l'analyste un moyen supplémentaire d'obtenir une représentation de l'échantillon d'observations multidimensionnelles sur l'écran.

Le premier des exemples présentés dans ce chapitre concerne la biométrie des abeilles. On dispose d'un échantillon d'abeilles prélevé sur différentes ruches des îles de la

Guadeloupe. Il s'agit d'étudier l'adaptation des différentes races aux écosystèmes de ces îles. L'analyse de cet exemple justifiera l'étude séparée des classes.

Le second exemple vise à montrer la puissance de l'approche interactive. Cet exemple a été utilisé lors du chapitre III afin de mettre en évidence les capacités de représentation des cartes (cf. §III.8.1). L'échantillon est constitué de deux classes ayant le même vecteur moyenne. Cette analyse va mettre en évidence un cas important d'incohérence dû à l'algorithme de classification par réseaux de neurones à apprentissage compétitif. Malgré cela, en utilisant l'algorithme du plus proche voisin que nous décrirons succinctement dans ce chapitre, nous obtiendrons un taux d'erreur de classification de 0%. Nous présenterons, de plus, une autre technique de classification exploitant les informations de distances des cartes. Avec cette nouvelle technique, nous obtiendrons aussi un taux d'erreur de classement nul. L'avantage de cette dernière technique est son coût réduit en temps de calcul. En effet, la classification d'un échantillon par cette technique est quasi immédiate par rapport à celle du plus proche voisin.

Le troisième exemple a aussi été abordé dans le chapitre III (cf. § III.8.2). Il est constitué de deux classes en forme de tores dans un espace à trois dimensions. Ces deux classes sont imbriquées l'une dans l'autre. Comme pour l'exemple précèdent, l'analyse de cet échantillon va faire apparaître des incohérences dues à la classification par l'algorithme des réseaux de neurones à apprentissage compétitif mais, en plus, certaines cartes vont présenter des distorsions au niveau de la conservation de la structure de l'échantillon. L'analyse de cet exemple va faire apparaître des incohérences de différentes origines. Cet exemple sera traité aussi avec succès puisque nous obtiendrons un taux d'erreur de classement nul.

V.1 EXEMPLE 5: BIOMETRIE DE L'ABEILLE

V.1.1 Présentation

L'échantillon utilisé est issu d'une population de 1336 abeilles sur lesquelles on a mesuré la pilosité du cinquième tergite, la coloration du second tergite abdominal, la largeur du tomentum, la longueur de la langue ainsi que les longueurs des nervures A et B de l'index cubital (cf. Figure IV.1). Les composantes des observations sont les valeurs relevées sur chacune des abeilles. L'échantillon est ainsi composé de 1336 observations dans un espace de représentation à six dimensions.

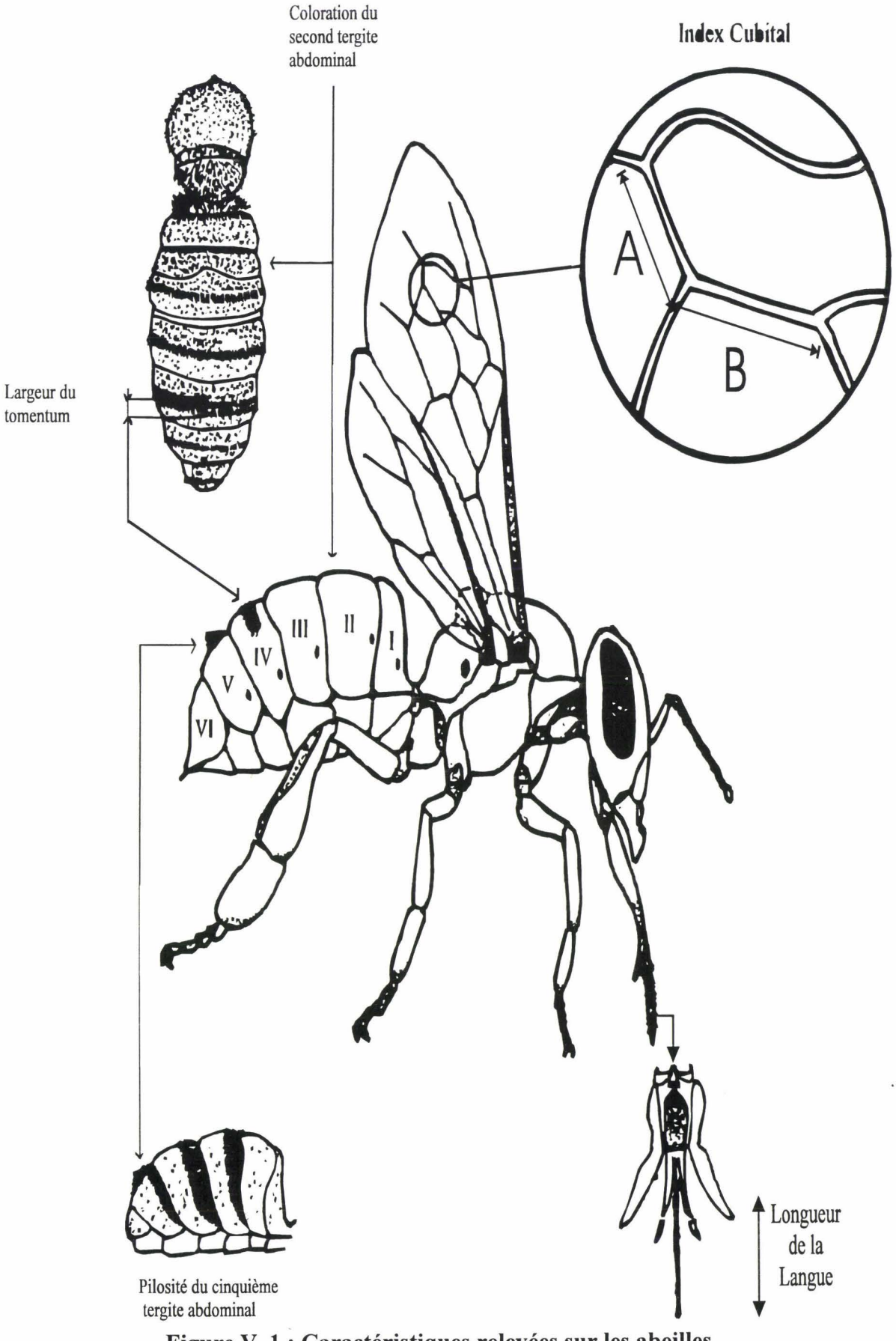


Figure V. 1 : Caractéristiques relevées sur les abeilles

V.1.2 Projections

V.1.2.1 Projection par analyse en composantes principales

Nous avons projeté l'échantillon suivant ses deux axes principaux. Les valeurs propres sont consignées dans le tableau V.1. Le pourcentage d'information du plan principal de projection est de 46,43%.

La figure V.2 représente cette projection sur le plan principal.

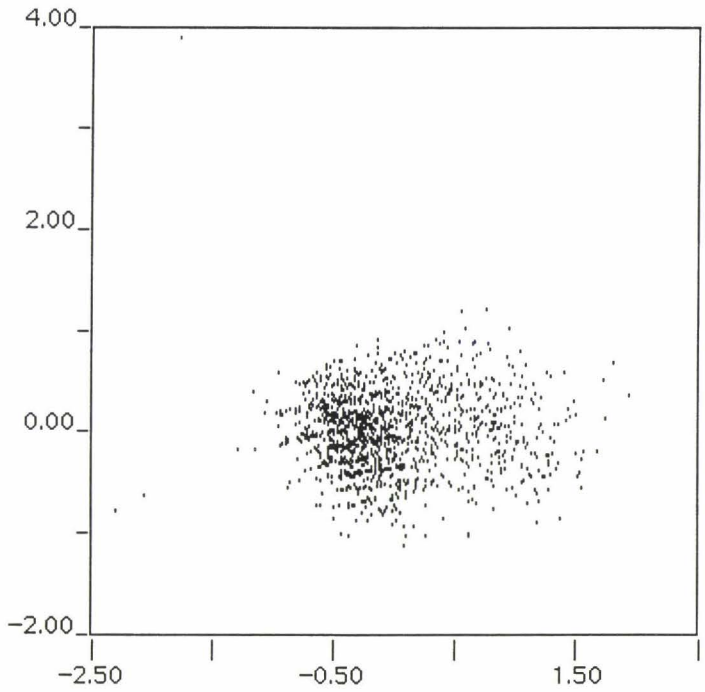


Figure V. 2 : Projection par analyse en composantes principales

Sur cette projection on distingue un seul groupe d'observations. L'examen de cette seule projection laisse à penser que l'échantillon n'est constitué que d'une seule classe.

Valeurs propres	Pourcentage d'information par axes
1,77	29,51%
0.9	15%
0.63	10,51%
0.71	11,77%
0.98	16,27%
1.02	16,92%

Tableau V. 1: Valeurs propres et pourcentage d'information par axes

V.1.2.2 Projection par l'algorithme de Sammon

La projection par l'algorithme de Sammon a été obtenue au bout de 500 itérations. Nous avons auparavant initialisé l'algorithme avec l'échantillon issu de la projection par analyse en composantes principales. La projection de l'échantillon par l'algorithme de Sammon est représentée sur la figure V.3. On y distingue 4 groupes d'observations révélant la présence de 4 classes distinctes.

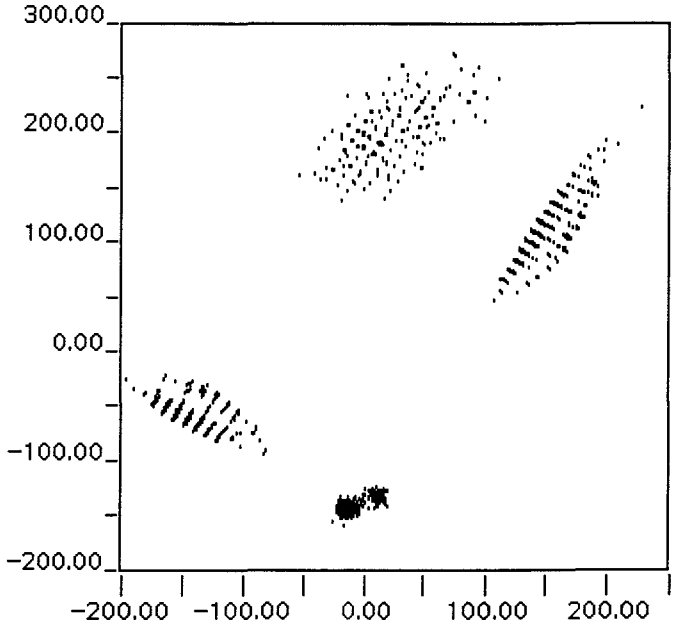


Figure V. 3 : Projection par l'algorithme de Sammon

V.1.2.3 Projection par perceptron multicouches

La projection par perceptron multicouches, représentée sur la figure V.4, a été obtenue en fixant le nombre de neurones des couches cachées 1 et 3 à 30 neurones. L'apprentissage a été effectué durant 1000 époques. Le coefficient d'apprentissage et le momentum ont été fixés respectivement à 0,9 et 0,5.

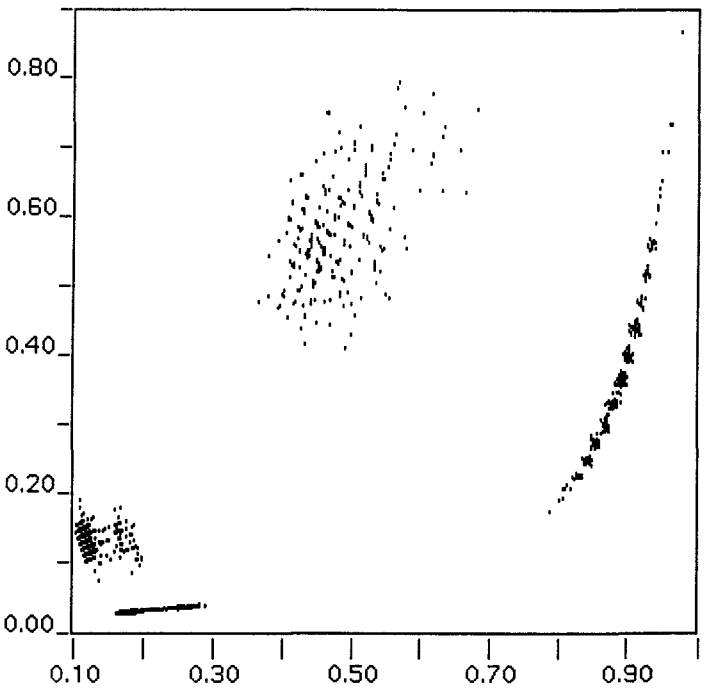
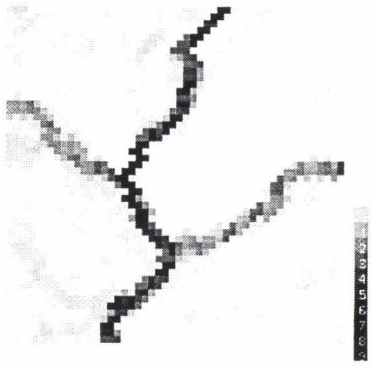


Figure V. 4 : Projection par perceptron multicouches

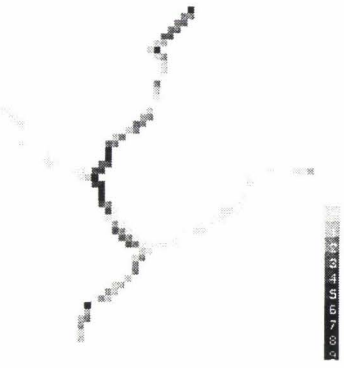
Sur cette projection par perceptron multicouches, on distingue 4 groupes d'observations indiquant également la présence de 4 classes distinctes.

V.1.2.4 Visualisations par cartes de Kohonen

Pour représenter cet échantillon à l'aide des cartes de Kohonen, nous utilisons un réseau de 50*50 neurones. L'apprentissage est effectué durant 1000 époques. Nous avons utilisé les distances triangulaire, sphérique et carrée avec des initialisations des vecteurs poids différentes. Le rayon d'interaction initial a été fixé à 25 et nous l'avons fait décroître toutes les 20 époques. Le coefficient d'apprentissage initial a été fixé à 0,5. Il décroît linéairement vers 0,001 en fonction des époques. Nous avons effectué au total 12 apprentissages. La figure V.5 représente les distances z_{MAX} et z_{MED} pour les trois distances utilisées. Ces cartes présentent toutes quatre régions indiquant ainsi la présence de quatre classes. Nous ne faisons apparaître sur la figure V.5 que trois de ces cartes représentant z_{MAX} (cf. Figure V.5, cartes (1a), (1b) et (1c)) et z_{MED} (cf. Figure V.5, cartes (2a), (2b) et (2c)). Comme les différentes représentations issues des 9 autres cartes n'offrent pas d'informations supplémentaire, nous ne les avons pas fait figurer dans ce mémoire.

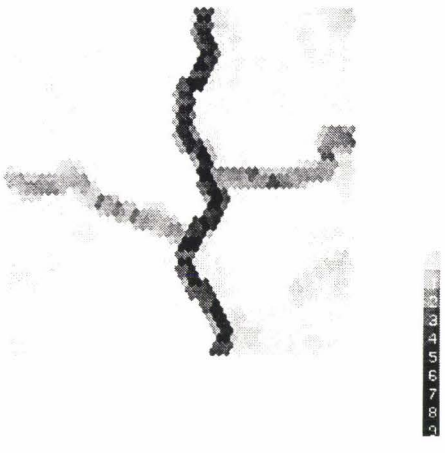


(1a)

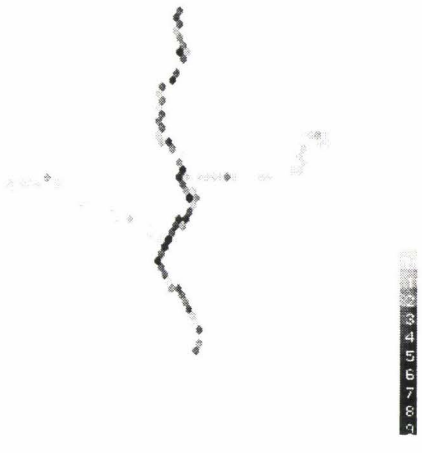


(2a)

-Distance triangulaire -

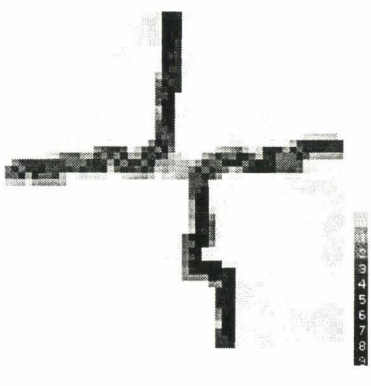


(1b)

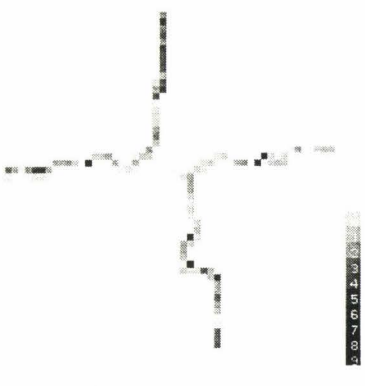


(2b)

- Distance sphérique -



(1c)



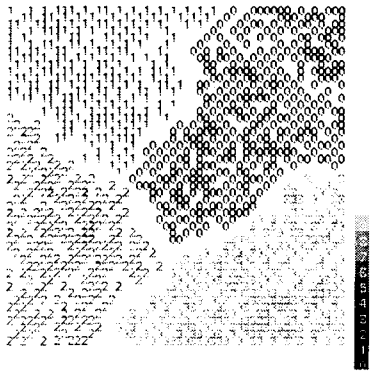
(2c)

- Distance carrée -

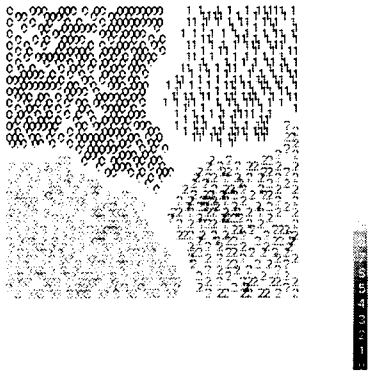
Figure V. 5 : Représentation de z_{MAX} et z_{MED}

V.1.3 Classification et vérification par projections de l'échantillon classé

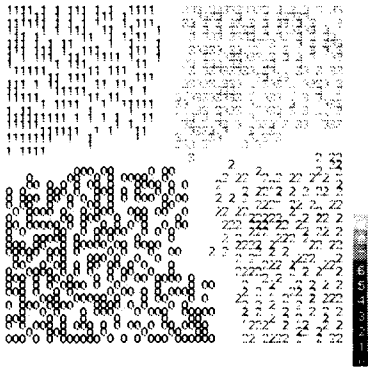
A part la projection par analyse en composantes principales qui ne fait apparaître qu'une seule classe, les autres techniques de représentation indiquent la présence de quatre classes. Nous effectuons l'initialisation de l'algorithme de classification par réseaux de neurones à apprentissage compétitif à partir de la représentation de z_{MAX} sur une carte à maillage hexagonal. Bien entendu, d'autres représentations basées sur d'autres cartes conviennent parfaitement à l'initialisation de l'algorithme de classification. Les projections de l'échantillon classé par l'algorithme de classification par réseaux de neurones à apprentissage compétitif sur les différentes cartes sont présentées sur la figure V.6. Sur ces différentes cartes, nous avons éliminé les neurones frontières apparaissant sur les représentations de z_{MAX} à l'aide des outils de seuillage (cf. §IV.3.1). Les régions isolées par cette technique sont toutes constituées d'observations provenant de mêmes classes. Il n'y a donc pas d'incohérence dans les résultats de la classification.



- (a) Distance triangulaire -



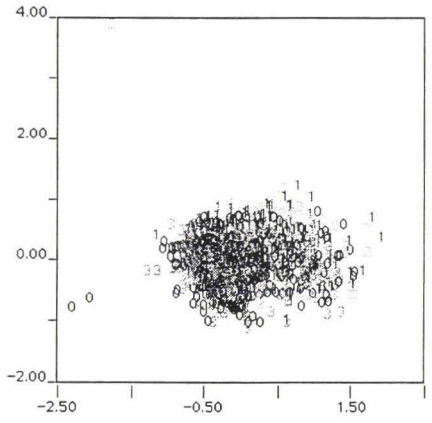
- (b) Distance sphérique -



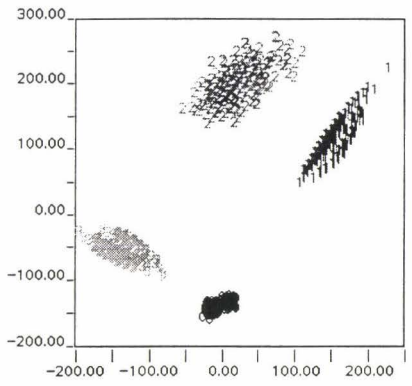
- (c) Distance carrée -

Figure V. 6 : Projections de l'échantillon classé sur les cartes visualisant z_{MAX}

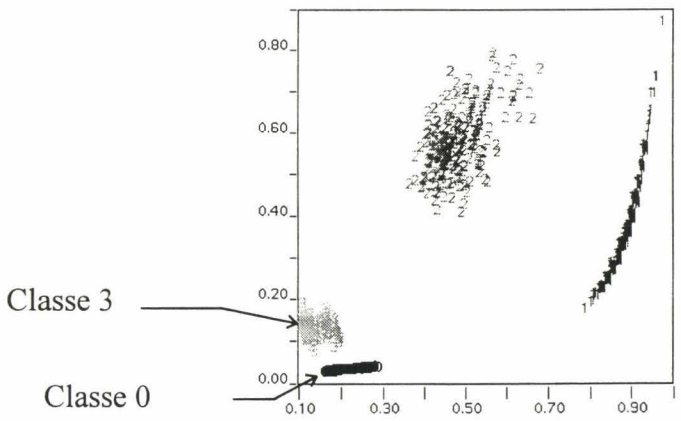
Nous pouvons par ailleurs vérifier si les groupes d'observations apparaissant sur les autres techniques de projection sont constitués d'observations provenant de même classe. Ainsi la figure V.7 représente les projections de l'échantillon classé par l'analyse en composantes principales, la projection de Sammon et par la projection par perceptron multicouches. Bien entendu, la projection par l'analyse en composante principale ne faisant apparaître qu'une seule classe, les projections des observations classées se confondent. Sur les projections par l'algorithme de Sammon et par le perceptron multicouches, les groupes d'observations précédemment décelés sont tous constitués d'observations de même classe.



- (a) Projection par analyse en composantes principales -



- (b) Projection par Sammon -



- (c) Projection par perceptron multicouches -

Figure V. 7 : Projections de l'échantillon classé

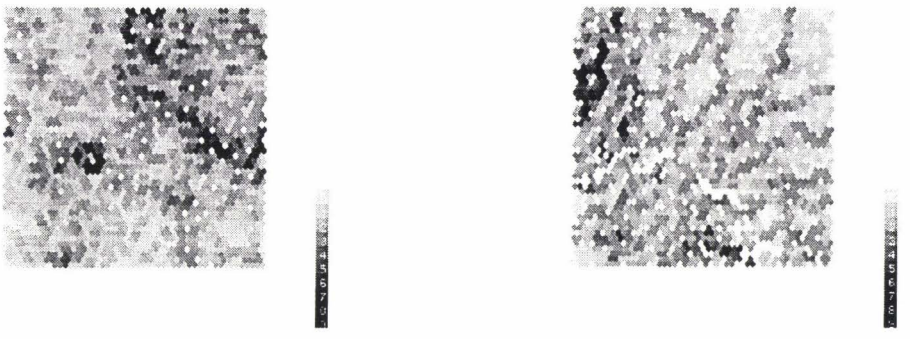
V.1.4 Analyse locale des classes

L'analyse locale des classes doit être effectuée avec précaution. En effet, il est toujours possible de décomposer une classe en plusieurs sous-classes. Il convient d'intégrer dans cette phase la connaissance d'un spécialiste. Les représentations de z_{MAX} sur les différentes cartes apprises à partir des 4 classes apparaissent sur la figure V.8. On ne peut pas distinguer sur ces différentes représentations la présence d'autres classes. Comme les représentations des caractéristiques de distances inter-neurones ne font apparaître que les classes ayant un degré de chevauchement très faible, il nous reste à vérifier l'existence de classes présentant un degré important de chevauchement en représentant des estimations de la fonction densité de probabilité sous-jacentes à chacune des classes. La figure V.9 illustre la représentation sur les cartes des estimations de la fonction densité de probabilité sous-jacente à chacune des classes décelées lors de l'étape précédente. Ainsi sur la carte de la figure V.9(a) dont l'apprentissage a été effectué à partir d'observations provenant de la classe 0, nous avons représenté l'estimation de la fonction densité de probabilité sous-jacente à cette même classe. Nous avons procédé de manière similaire pour les autres classes.

La carte de la figure V.9(a) fait apparaître deux régions différentes, suggérant la présence de deux sous-classes. La région sombre indique une sous-classe de densité importante. Nous avons initialisé, à partir cette carte, l'algorithme de classification par réseaux de neurones à apprentissage compétitif. Nous avons représenté sur la figure V.10 des projections des nouvelles classes obtenues à partir de la classe 0. Les deux nouvelles classes sont représentées sur les projections par des 0 et des 1.

On peut représenter ces deux nouvelles classes dans l'échantillon complet. Sur la figure V.11, nous avons représenté l'échantillon avec l'ensemble des cinq classes. Sur ces projections les observations de la classe 1 de la figure V.10 sont représentées par un 4 dans la figure V.11.

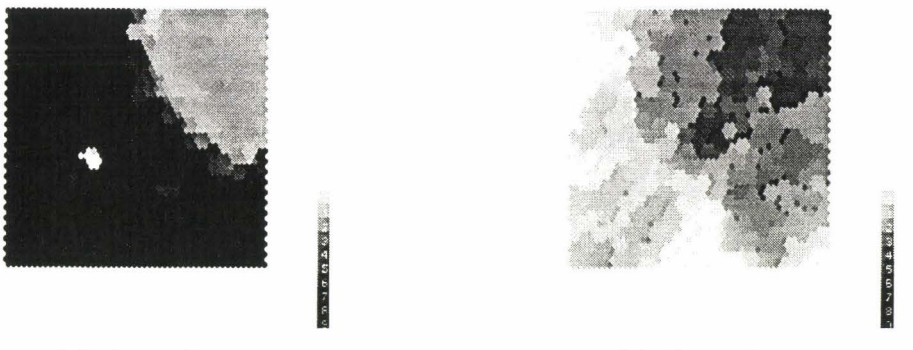
L'analyse globale de l'échantillon d'abeilles a révélé la présence de 4 classes distinctes. L'une de ces classes a été décomposée en deux classes présentant un degré de chevauchement important. On peut éventuellement décomposer les classes 1, 2 et 3. Rappelons que l'analyse locale des classes distinctes doit être effectuée avec précaution et en intégrant l'expérience d'un spécialiste. Il se peut, par exemple, que l'effet des imprécisions dans le relevé des mesures fassent apparaître des classes. Dans ce cas, l'apport d'un spécialiste



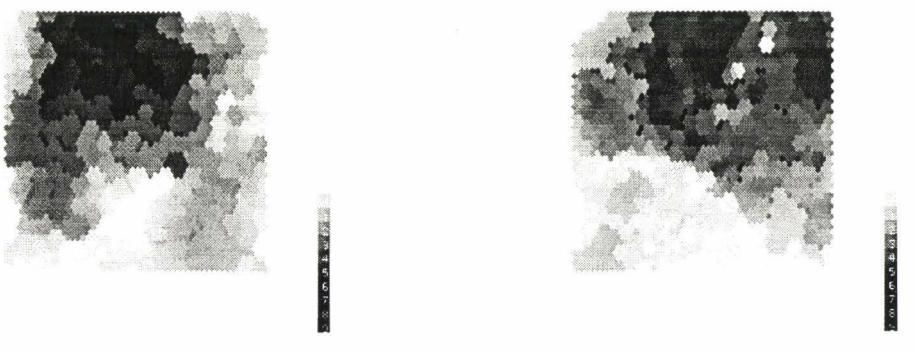
- Classe 0 - - Classe 1 -

- Classe 2 - - Classe 3 -

Figure V. 8 : Représentation de z_{MAX} pour les différentes classes

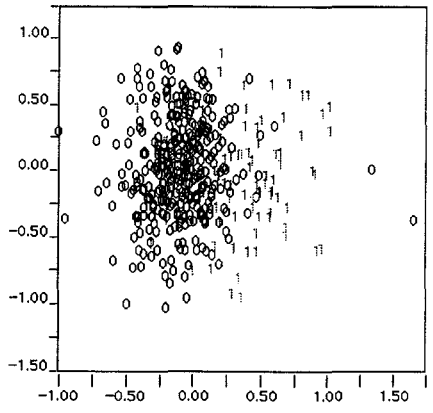


- (a) classe 0 - - (b) classe 1 -

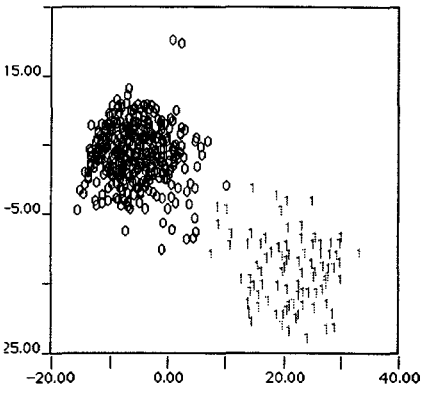


- (c) classe 2 - - (c) classe 3 -

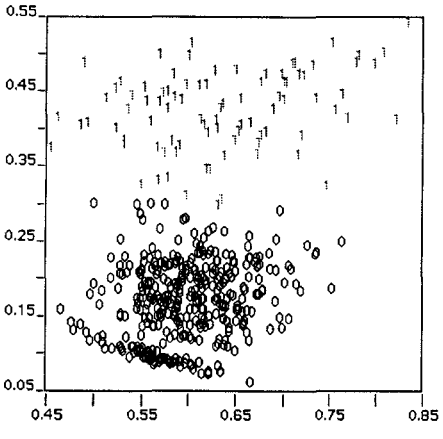
Figure V. 9 : Représentation de la fonction densité de probabilité pour les différentes classes



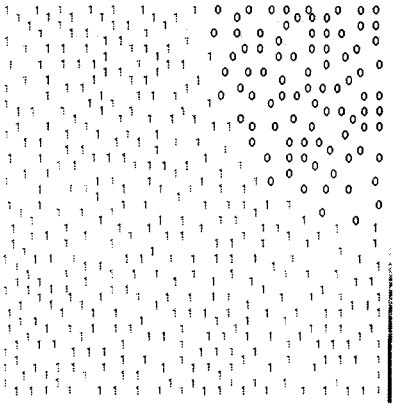
- (a) Projection par analyse en composantes principales -



- (b) Projection par Sammon -

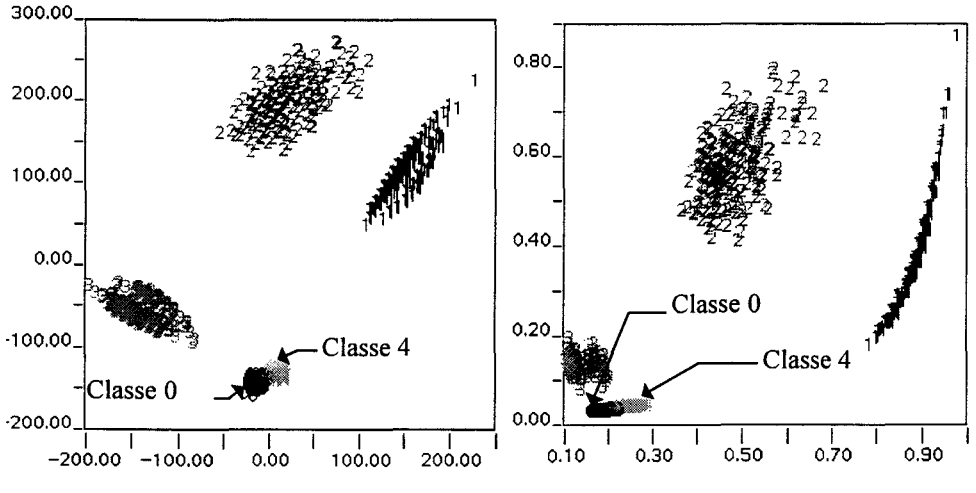


- (c) Projection par perceptron multicouches -



- (d) Projection sur la carte de Kohonen -

Figure V. 10 : Projections des deux classes issues de l'analyse séparée de la classe 0



(a) (b)

Figure V. 11 : Projection de l'ensemble des classes par l'algorithme de Sammon (a) et par le perceptron multicouches(b)

permet de lever les éventuelles incertitudes.

Les représentations de la fonction densité de probabilité sous-jacente aux classes 1, 2 et 3 apparaissant moins contrastées que la représentation de la fonction densité de probabilité sous-jacente à la classe 0 et compte-tenu des incertitudes dans le relevé des caractéristiques sur les abeilles, nous préférons arrêter l'analyse à ce stade.

V.2 ANALYSE DE L'EXEMPLE 3 :

V.2.1 Présentation

L'échantillon de l'exemple 3 a été traité brièvement dans le chapitre III. Rappelons qu'il s'agit d'un échantillon dans un espace à trois dimensions comprenant une classe gaussienne contenant 200 observations et une classe contenant 800 observations réparties à la périphérie d'une sphère. Cet exemple a été généré artificiellement. Nous considérons, au cours de ce paragraphe, que l'on ne dispose comme information sur l'échantillon que du nombre d'observations et de la dimension de l'espace de représentation des observations. Le lecteur pourra, s'il le désire, se référer au paragraphe III.8.1 pour de plus amples informations au sujet des paramètres statistiques de chacune des classes générées.

V.2.2 Projections

V.2.2.1 Projection par analyse en composantes principales

Les valeurs propres de la matrice de covariance de l'échantillon sont représentées dans le tableau V.2. La projection suivant le plan principal d'inertie est représentée dans la figure V.12. Le pourcentage d'information apporté par le plan principal de projection est de 68%.

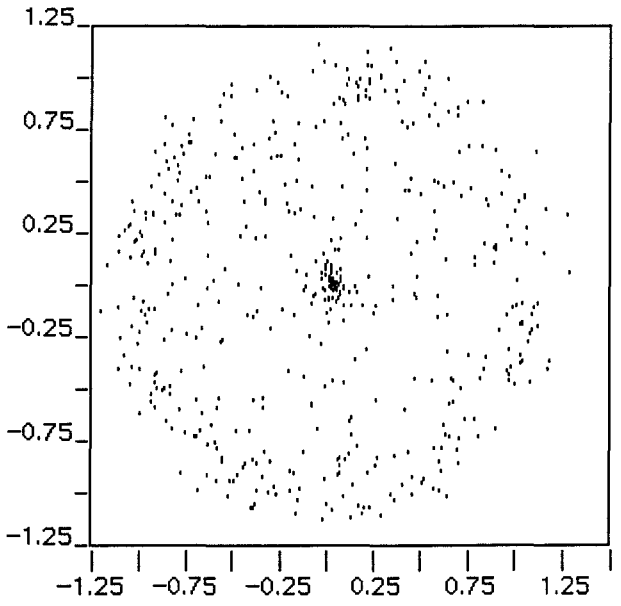


Figure V. 12: Projection par analyse en composantes principales de l'exemple 3

Valeurs propres	Pourcentage d'information par axe
1,04	34,53%
1,0	33,49%
0,96	31,97%

Tableau V. 2: Valeurs propres et pourcentage d'information par axe de l'exemple 3

L'examen de la figure V.12 est délicat. Ainsi, on peut soit considérer qu'il y a une seule classe, soit qu'il existe deux classes ayant un degré de chevauchement important. Dans ce second cas, la classe au centre de la figure V.12 apparaît être la plus dense.

V.2.2.2 Projection par l'algorithme de Sammon

L'algorithme de projection de Sammon a été initialisé avec les observations projetées issues de l'analyse en composantes principales. Le nombre d'itérations a été fixé à 500.

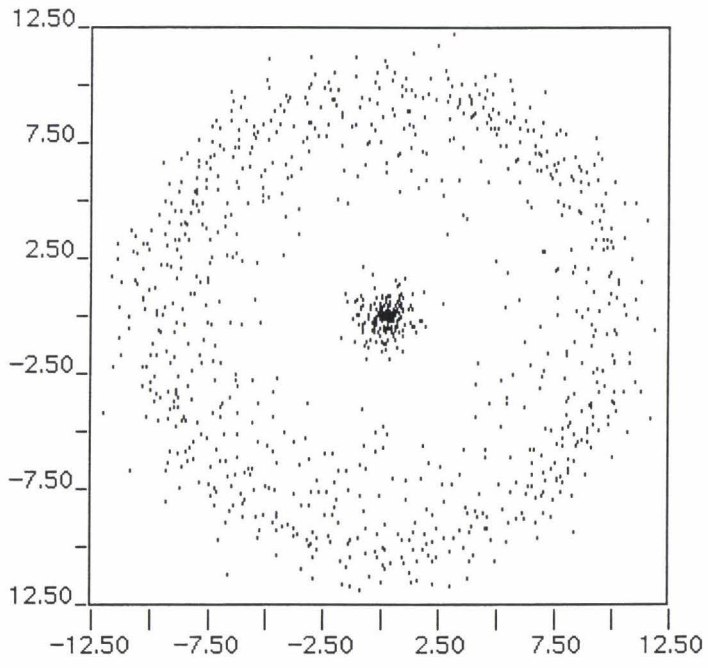


Figure V. 13: Projection par l'algorithme de Sammon de l'exemple 3

L'incertitude sur le nombre de classes apparaissant lors de l'examen visuel de la projection par analyse en composantes principales disparaît lors de l'analyse de la projection par l'algorithme de Sammon (cf. Figure V.13). En effet, sur cette projection, on distingue nettement deux groupements d'observations indiquant la présence de deux classes distinctes.

V.2.2.3 Projection par le perceptron multicouches

Nous avons fixé pour cet exemple le nombre de neurones des couches cachées 1 et 3 à 15 neurones. L'apprentissage a nécessité 2000 époques. Le coefficient d'apprentissage et le momentum ont été fixés respectivement à 0,9 et 0,5.

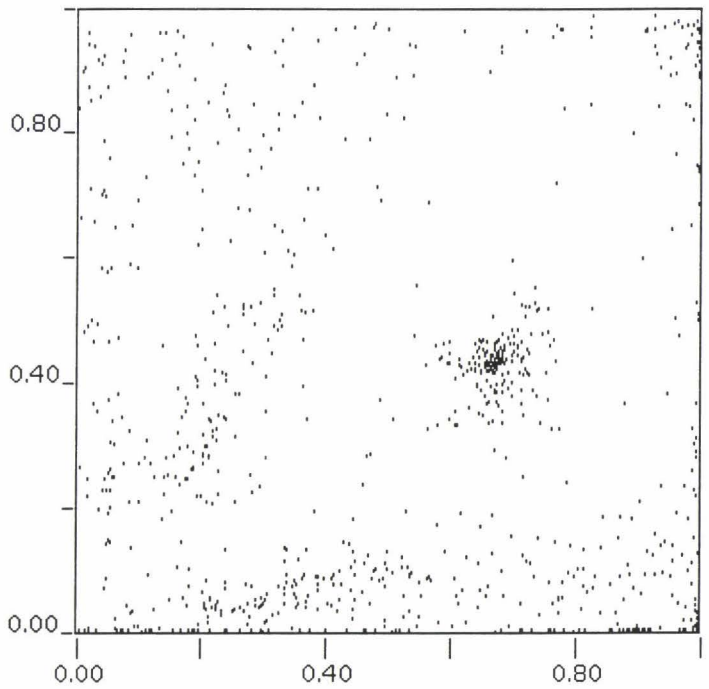


Figure V. 14 : Projection par perceptron multicouches de l'exemple 3

L'examen de la projection par perceptron multicouches confirme l'hypothèse de la présence de deux classes dans l'échantillon de l'exemple 3 (cf. Figure V.14).

V.2.3.4 Visualisations par cartes de Kohonen

Les cartes ont été apprises en respectant les mêmes paramètres que ceux de l'exemple précédent. Ainsi l'apprentissage a nécessité 1000 époques. Le rayon d'interaction initial à été fixé à 25 et nous l'avons fait décroître toutes les 20 époques. Le coefficient d'apprentissage initial a été fixé à 0,5.

Nous ne représentons sur la figure V.14 qu'une seule carte. On distingue sur cette carte deux régions distinctes indiquant la présence de deux classes.

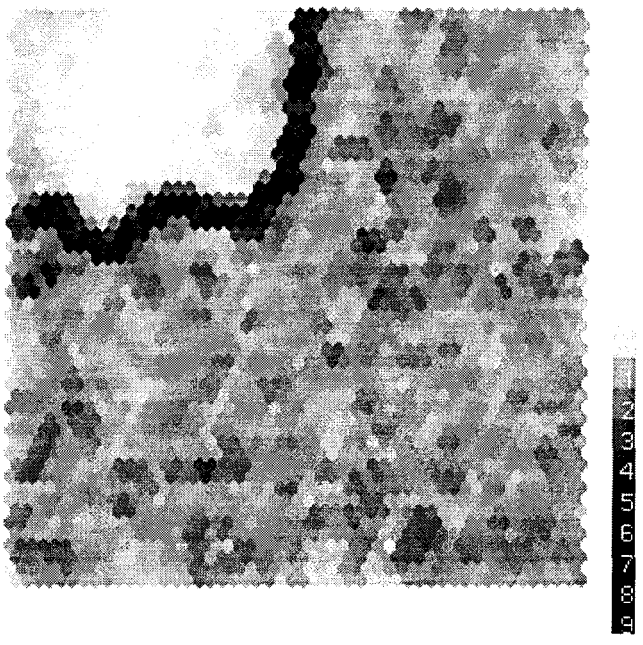


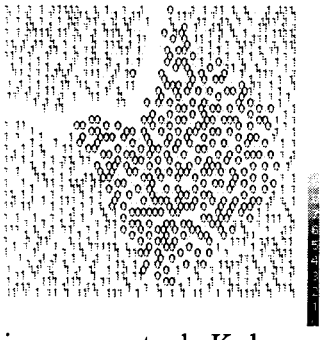
Figure V. 15 : Représentation de z_{MAX} calculé à partir d'une distance sphérique

V.2.3 Classification et vérification par projection de l'échantillon classé

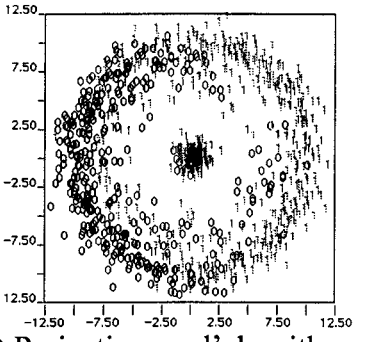
Toutes les représentations, à part la projection par analyse en composantes principales qui peut prêter à confusion, font apparaître deux classes. Bien entendu, la classe 1 n'étant pas de forme globulaire, la classification par réseau à apprentissage compétitif provoque une incohérence dans la projection de l'échantillon classé (cf. Figure V.16). La classe 1 sur la carte de la figure V.16(a) est projetée sur deux régions associées à deux classes différentes. Sur les autres projections, les classes se confondent alors qu'elles devraient normalement apparaître séparées. Il semble évident que ces incohérences sont dues à l'utilisation inadaptées de la classification par réseaux de neurones à apprentissage compétitif.

V.2.3.1 Classification de l'échantillon par l'algorithme du plus proche voisin

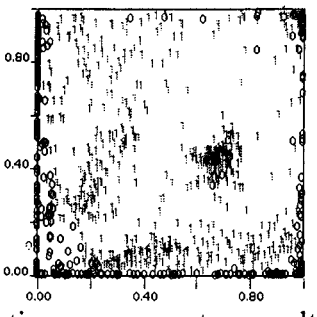
Afin de vérifier l'hypothèse de l'utilisation inappropriée de la classification par réseaux de neurones à apprentissage compétitif, nous faisons appel à un autre algorithme: l'algorithme du plus proche voisin. Le principe de cet algorithme considère qu'une observation appartient à la même classe que l'observation la plus proche de celle-ci [COVE 67][DUDA 73]. Il existe différents algorithmes utilisant cette méthode de classification. Pour classer correctement l'échantillon de l'exemple 3, nous utilisons l'algorithme le plus coûteux en temps de calcul. Cet algorithme assigne l'observation non classée la plus proche d'une



- (a) Projection par carte de Kohonen-

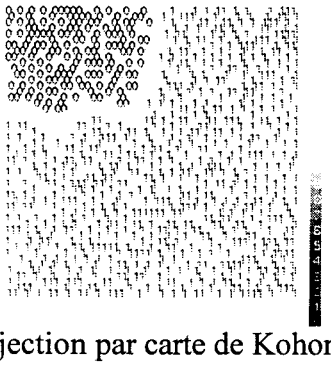


- (b) Projection par l'algorithme de Sammon -

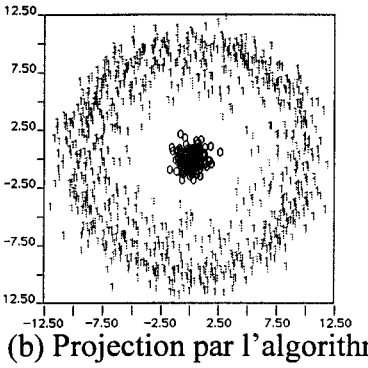


- (c) Projection par perceptron multicouches -

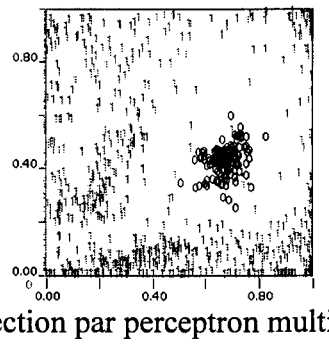
Figure V. 16 : Projections de l'échantillon classé par réseaux de neurones à apprentissage compétitif



- (a) Projection par carte de Kohonen-



- (b) Projection par l'algorithme de Sammon -



- (c) Projection par perceptron multicouches -

Figure V. 17 : Projections de l'échantillon classé par plus proche voisin

observation déjà classée. Cela suppose dans un premier temps de calculer toutes les distances entre les observations non classées et les observations déjà classées, puis d'assigner l'observation la plus proche d'une observation déjà classée, à cette même classe. Il convient de disposer au début de l'algorithme d'observations déjà classées. L'opérateur désigne ces observations par l'intermédiaire de la carte d'initialisation. Ainsi, il sélectionne, à l'aide de la souris, les neurones sur la carte qu'il estime être représentatifs des classes. Nous initialisons l'algorithme du plus proche voisin en assignant les observations les plus proches des vecteurs poids des neurones sélectionnés par l'opérateur.

La classification par l'algorithme du plus proche voisin donne des résultats cohérents pour la projection par cartes et pour celle obtenue par l'algorithme de Sammon (Figure V.17). Ainsi, la projection de l'échantillon classé par l'algorithme du plus proche voisin fait apparaître les classes dans les régions claires de la représentation de z_{MAX}. De même, sur la projection par l'algorithme de Sammon, les deux classes apparaissent distinctement. La projection par perceptron multicouches présente quelques points litigieux. En effet certaines observations de la classe 1 se confondent avec la classe 0. Cependant, nous pouvons considérer que les résultats sont satisfaisants et que les points litigieux sont dus à une mauvaise convergence du réseau au cours de la phase d'apprentissage. On peut vérifier cette hypothèse en effectuant d'autres apprentissages du réseau perceptron multicouches. La classification par l'algorithme du plus proche voisin permet de classer correctement cet échantillon. Cependant cette technique est coûteuse en temps de calcul.

V.2.3.2 Classification par l'algorithme d'assignation avec rayon

Une solution permettant d'obtenir des résultats analogues à ceux obtenus à l'aide de l'algorithme plus proche voisin tout en réduisant les coûts de calcul consiste à utiliser les représentations de z_{MAX} afin d'obtenir une estimation de la distance minimale interclasse R. Nous définissons la distance interclasse comme la distance minimale séparant deux observations de classes différentes. L'algorithme d'assignation avec rayon (AR) utilise le minimum de ces distances afin de classer les observations. Cet algorithme de classification considère que si une observation non classée se situe à une distance inférieure à la distance minimale interclasse R d'une observation déjà classée alors cette observation appartient à la même classe que l'observation déjà classée. Bien entendu, la difficulté consiste à obtenir une

estimation de cette distance. Pour cela, considérons la figure V.18 sur laquelle sont représentées deux classes dans un espace à deux dimensions. Sur cette figure, les points noirs correspondent aux vecteurs poids d'une carte comprenant 5*5 neurones. Nous avons reliés par des segments de droite les vecteurs poids de neurones voisins sur la carte. Ainsi la valeur de z_{MAX} calculé pour un neurone correspond à la longueur du plus grand segment reliant ce neurone. Pour les neurones se situant à la frontière des deux classes, z_{MAX} correspond à la distance séparant deux vecteurs poids appartenant à différentes classes. Le minimum de z_{MAX} calculés pour les neurones frontières peut être considéré comme une grossière estimation de la moitié de la distance interclasse séparant les deux classes.

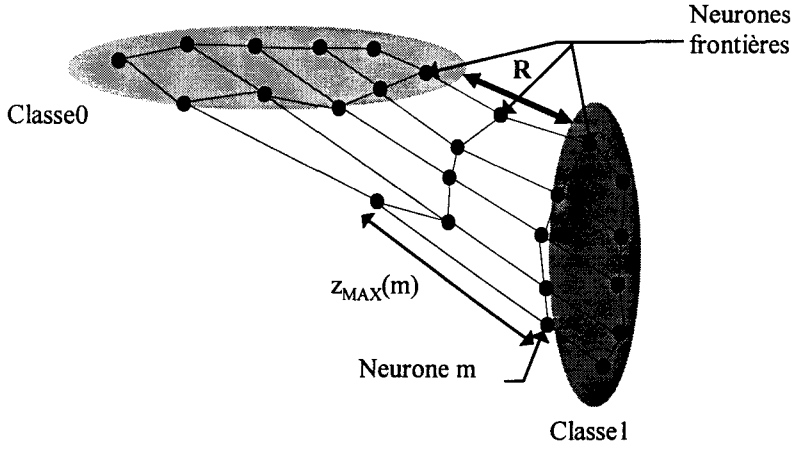


Figure V. 18 : Distance interclasse R

Pour obtenir une approximation de la distance minimale interclasse R, nous utilisons l'outil de seuillage par valeurs supérieures en l'appliquant à la représentation de z_{MAX}. L'opérateur utilise cette technique de seuillage afin d'éliminer les neurones frontières. Il détermine par essais successifs la valeur du seuil haut c_h permettant d'éliminer tous les neurones frontières. Lorsque toutes les régions apparaissent séparées sur la carte, la valeur de seuil haut c_h permet d'obtenir une grossière estimation de la distance minimale interclasse $\hat{R}$. Cette estimation s'exprime par la formule :

$$\frac{\hat{R}}{2} = c_h \times \frac{\left(z_{sup} - z_{inf}\right)}{1000} + z_{inf}$$

où : $\begin{cases} z_{sup} = \max\limits_{0 \le m < M} z_{MAX}(m) \\ z_{inf} = \min\limits_{0 \le m < M} z_{MAX}(m) \end{cases}$

Compte-tenu des imprécisions de $\hat{R}$, une valeur de cette approximation supérieure à la distance minimale interclasse réelle R a pour effet de chaîner des observations de classes différentes. Aussi, afin de pallier ce problème, nous utilisons la moitié de la distance minimale interclasse pour assigner les observations. La règle d'assignation s'effectue alors selon le schémas :

Soit X une observation non classée et X' une observation assignée à la classe $C_{k'}$

Si $d^E(X,X') < \frac{\hat{R}}{2}$ alors l'observation X est assignée à la classe $C_{k'}$.

Cette règle d'assignation suppose qu'il existe des observations déjà classées. Comme pour l'algorithme du plus proche voisin, l'opérateur sélectionne à l'aide d'une carte les neurones qu'il estime les plus représentatifs des classes. L'initialisation de l'algorithme consiste alors à assigner les observations les plus proches des vecteurs poids des neurones sélectionnés. Il doit en plus à l'aide de la technique de seuillage par valeurs supérieures trouver une approximation de la distance interclasse.

Algorithme d'assignation avec rayon

1. Initialisation des K observations représentatives des K classes.

Initialisation de la distance minimale interclasse $\hat{R}$.

2. Sélection aléatoire d'une observation X non classée.

3. Sélection aléatoire d'une observation classée X' pour laquelle la distance $d^E(X,X')$ n'a pas été calculée.

4. Calcul de la distance $d^E(X,X')$.

5. Règle d'assignation,

Si $d^E(X,X') \leq \frac{\hat{R}}{2}$ alors on assigne l'observation X à la classe de l'observation X'.

Si $d^E(X,X') > \frac{\hat{R}}{2}$ alors :

Si il existe des observations classées X' pour lesquelles la distance $d^E(X,X')$ n'a pas encore été calculée alors on reprend le procédé à partir de l'étape 3.

Si toutes les observations classées ont été comparées avec

l'observation X, alors on passe à l'étape suivante.

6. Test d'arrêt,

Si toutes les observations non classées ont été traitées sans qu'il n'y ait aucune nouvelle assignation, alors fin de l'algorithme.

Sinon on reprend le procédé à partir de l'étape 2.

Nous avons appliqué cette technique de classification à l'échantillon de l'exemple 3. L'approximation du rayon obtenue par la technique de seuillage est de 7,8. La distance minimale interclasse réelle que nous avons calculée à partir de l'échantillon classé est de 6,21. Nous avons obtenu exactement les mêmes résultats que pour la classification par l'algorithme du plus proche voisin. Le taux d'erreur de classement est de 0%. Cependant les temps de calcul des deux algorithmes sont très différents. Ainsi pour l'exemple 3, le temps de calcul pour classé l'échantillon par l'algorithme d'agrégation avec rayon est de 1 seconde. alors que la classification par l'algorithme du plus proche voisin a nécessité 10 minutes. De plus, l'algorithme d'agrégation avec rayon ne classe pas les observations isolées. Cette caractéristique est très intéressante, surtout lorsque l'échantillon analysé est bruité. Dans ce cas, il existe de nombreuses observations isolées résultant d'erreurs de mesures. Ces observations sont alors considérées comme ne faisant partie d'aucune classe. Bien entendu, nous pouvons classer ces observations isolées en appliquant un autre algorithme de classification ou en utilisant l'algorithme d'agrégation avec rayon avec une valeur de $\frac{\hat{R}}{2}$ plus élevée. Il est à noter que cet algorithme n'est applicable que lorsque l'on traite des classes ayant un degré de chevauchement nul.

L'analyse locale n'a pas permis de déceler d'autres classes. Nous avons ainsi déceler deux classes et classer l'échantillon d'observations. La comparaison de l'échantillon classé avec l'échantillon étiqueté d'origine a permis de calculer un taux d'erreur de classement nul.

V.3 EXEMPLE 4

V.3.1 Présentation

L'échantillon de l'exemple 4 a été aussi traité brièvement dans le chapitre III. Rappelons qu'il s'agit d'un échantillon généré artificiellement, constitué de deux classes en forme de tores entrelacés dans un espace à trois dimensions. Le lecteur pourra, s'il le désire, se référer au paragraphe III.8.2 pour de plus amples informations au sujet des paramètres statistiques de chacune des classes générées.

V.3.2 Projections

V.3.2.1 Projection par analyse en composante principale

Les valeurs propres de la matrice de covariance de l'échantillon sont représentées dans le tableau V.3. La projection suivant le plan principal d'inertie est représentée figure V.19. Le pourcentage d'information apporté par le plan principal de projection est de 68%.

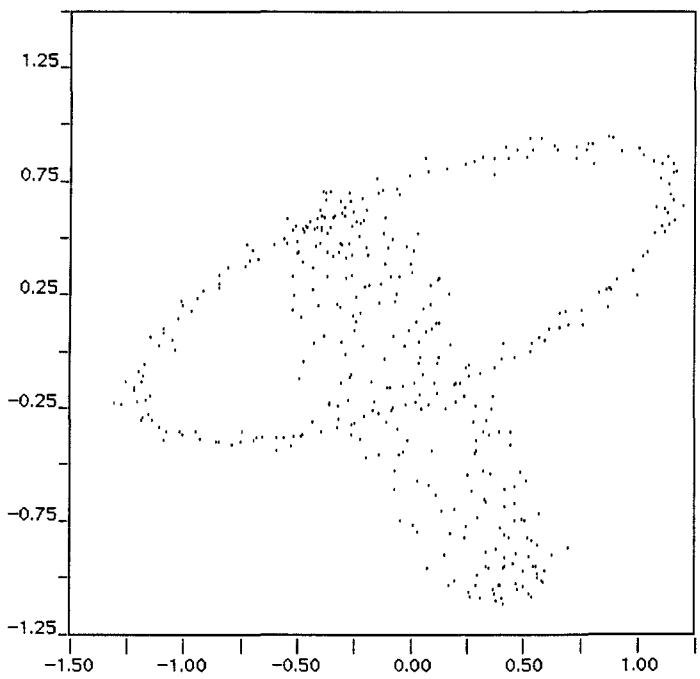


Figure V. 19: Projection par analyse en composantes principales de l'exemple 4

Sur la figure V.19 nous pouvons distinguer une classe en forme d'anneau et une autre

en forme d'ellipse. Ces deux classes apparaissent avoir un degré de chevauchement non nul.

Valeurs propres	Pourcentage d'information par axe
1,04	34,53%
1,0	33,49%
0,96	31,97%

Tableau V. 3: Valeurs propres et pourcentage d'information par axe de l'exemple 4

V.3.2.2 Projection par l'algorithme de Sammon

L'échantillon de l'exemple 4 comporte 1000 observations. Bien que nous pouvons projeter l'échantillon complet, nous avons préféré projeter par l'algorithme de Sammon un échantillon réduit. Nous avons bien entendu réduit l'échantillon d'observations à l'aide de la technique neuronale exposée dans le paragraphe III.2.7. Ainsi, la figure V.20 représente la projection par l'algorithme de Sammon des vecteurs poids d'un réseau comportant 400 neurones et ayant subi successivement un apprentissage compétitif de Kohonen, puis un apprentissage compétitif sensible à la fréquence, et enfin un apprentissage compétitif. Le nombre d'époques pour chaque apprentissage a été fixé à 100. L'algorithme de projection de Sammon a été initialisé avec les vecteurs projetés issus de l'analyse en composantes principales. Le nombre d'itérations de l'algorithme de Sammon a été fixé à 500.

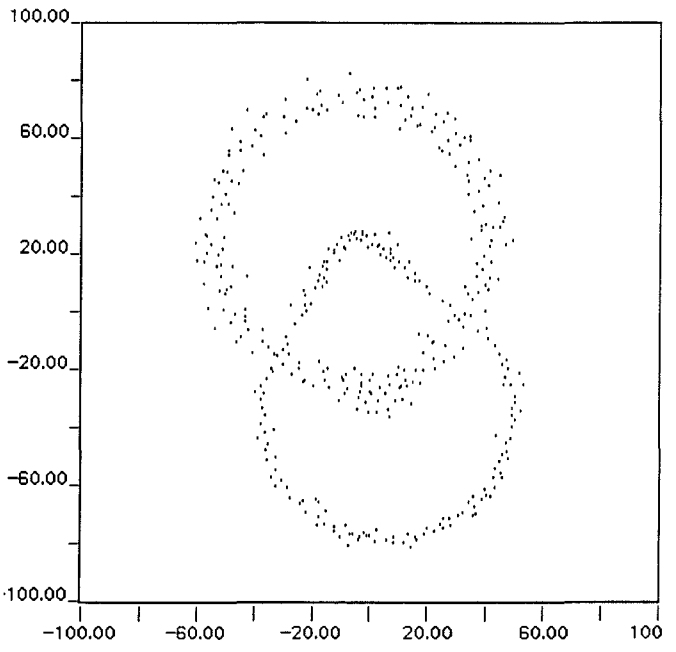


Figure V. 20: Projection par l'algorithme de Sammon d'un échantillon réduit de l'échantillon de l'exemple 4

Comme pour la projection par l'analyse en composantes principales, la projection obtenue avec l'algorithme de Sammon fait apparaître deux classes en forme d'anneau (cf. Figure V.20). Cependant ces classes apparaissent moins confondues sur la projection de Sammon que sur la projection par analyse en composantes principales. D'autre part, il apparaît sur la projection de Sammon que l'une des classes est plus dense que l'autre.

V.3.2.3 Projection par perceptron multicouches

La projection par perceptron multicouches représentée sur la figure V.21 est celle de l'échantillon réduit utilisé pour la projection par l'algorithme de Sammon. Nous avons fixé pour cet exemple le nombre de neurones des couches cachées 1 et 3 à 15 neurones. Le coefficient d'apprentissage et le momentum ont été fixés respectivement à 0,9 et 0,5. Les apprentissages effectués se sont révélés inexploitables. La figure V.21 illustre la projection obtenue au bout de 10000 époques.

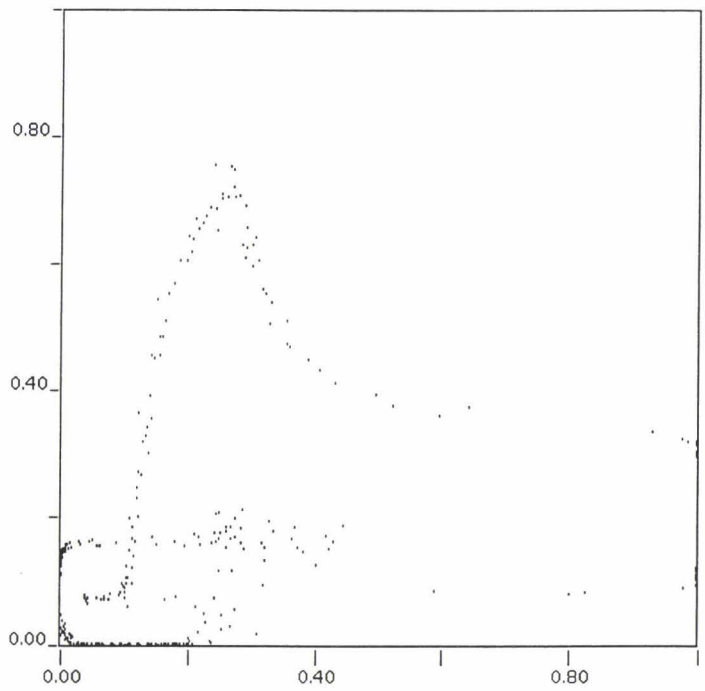


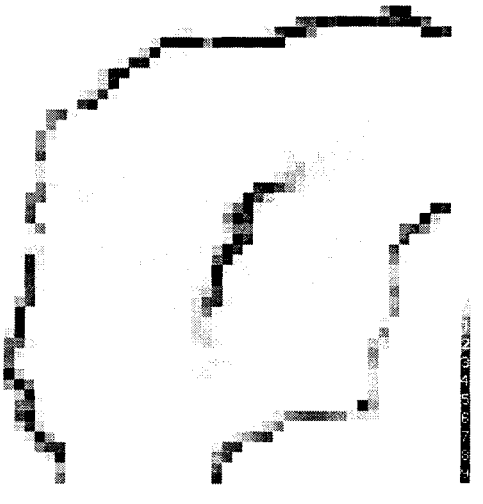
Figure V. 21 : Projection par perceptron multicouches de l'exemple 4

V.3.3.4 Visualisations par cartes de Kohonen

Les cartes ont été apprises en respectant les mêmes paramètres que ceux des exemples précédents. Sur la figure V.22, nous avons représenté z_{MAX} à partir de trois cartes.

- (a) distance triangulaire -

- (b) distance sphérique -



- (c) distance carrée -

Figure V. 22 : Représentation de z_{MAX}

L'examen des différentes cartes de la figure V.22 est délicat. En effet sur les cartes V.22(a) et V.22(c) on distingue trois régions tandis que la carte V.22(c) fait apparaître 2 régions. Il semble que certaines cartes aient mal convergé lors de la phase d'apprentissage.

V.3.3 Classification de l'échantillon

L'examen visuel des cartes et des différentes projections ne permet pas de déterminer avec certitude le nombre de classes de l'échantillon d'observations. L'analyste peut opter pour la présence de 2 ou de 3 classes dans l'échantillon. Pour classer l'échantillon nous allons considérer les deux cas. Ainsi nous allons supposer dans le premier cas que l'échantillon est composé de trois classes. Puis dans le second cas, nous allons considérer qu'il composé de deux classes.

V.3.3.1 cas 1 :hypothèse de trois classes

Si l'on considère que l'échantillon est composé de trois classes, nous utilisons comme carte d'initialisation une carte faisant apparaître 3 régions. Nous choisissons ainsi la carte de la figure V.22(c) afin d'initialiser l'algorithme de classification par réseaux de neurones à apprentissage compétitif mais aussi l'algorithme de classification d'agrégation avec rayon. Nous pouvons bien entendu utiliser sous l'hypothèse des trois classes la carte de la figure V.22(a). Sur la carte de la figure V.22(c), nous sélectionnons à l'aide de la souris un neurone pour chaque classe.

V.3.3.2 cas 2 : hypothèse de deux classes

Sous cette hypothèse nous utilisons la carte de la figure V.22(b) comme carte d'initialisation. Sur cette carte, nous sélectionnons, à l'aide de la souris, un neurone pour chaque classe.

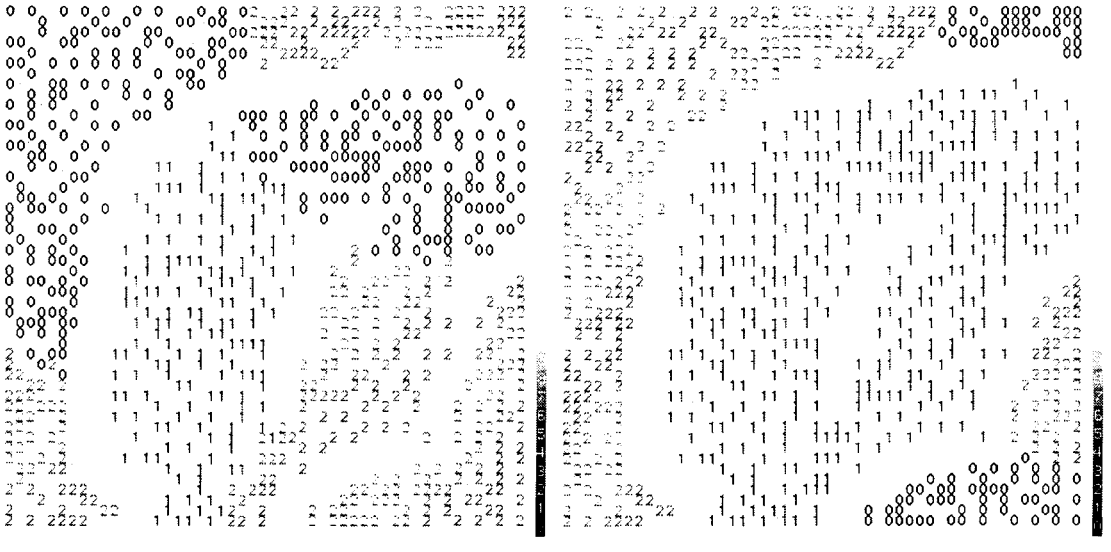
V.3.4 Vérification par projection de l'échantillon classé

Nous allons vérifier la cohérence des résultats de la classification en projetant les échantillons classés dans les 2 cas.

V.3.4.1 cas 1 : hypothèse de trois classes

La projection de l'échantillon classé par l'algorithme de classification par réseaux de neurones à apprentissage compétitif sur la carte d'initialisation apparaît incohérente (cf. Figure V.23(a)). Ainsi des observations provenant d'une même classe apparaissent dans différentes régions de la carte. Nous pouvons supposer que ces incohérences résultent de

l'utilisation inadaptée de la classification par réseaux de neurones à apprentissage compétitif. Or la projection de l'échantillon classé par l'algorithme d'agrégation avec rayon apparaît aussi incohérente (cf. Figure V.23(b)). Notons que sur la figure V.23(b), les observations de la classe 1 sont toutes projetées à l'intérieur d'une même région. De plus, comme les observations de la classe 0 et de la classe 2 sont projetées au sein des mêmes régions, il semble que ces observations fassent parties d'une même classe ce qui confirmerait ainsi que l'échantillon n'est composé que de deux classes. Nous allons maintenant vérifier l'hypothèse des deux classes.



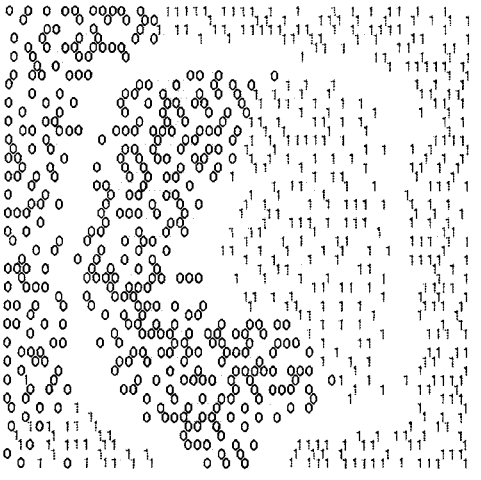
- (a) échantillon classé par réseaux de neurones à apprentissage compétitif -

- (b) échantillon classé par l'algorithme d'agrégation avec rayon -

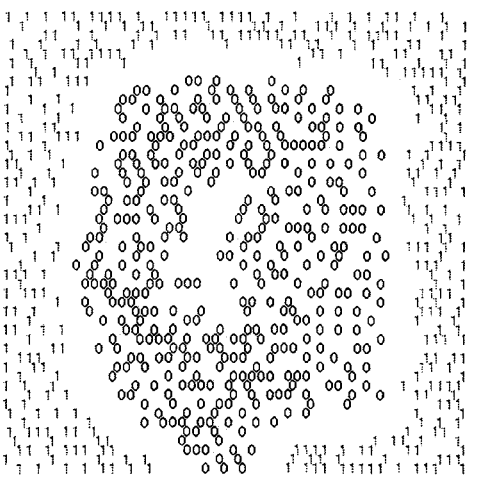
Figure V. 23 : projection par carte de l'échantillon classé

V.3.3.2 cas 2 : hypothèse de deux classes

La projection de l'échantillon classé par réseaux de neurones à apprentissage compétitif sur la carte d'initialisation fait apparaître les classes dans des régions différentes (cf. Figure V.24(a)). Nous vérifions qu'il s'agit d'une incohérence due à l'algorithme de classification en projetant l'échantillon classé par l'algorithme d'agrégation avec rayon (cf. Figure V.24(b)). Cette fois-ci, la projection se révèle cohérente. Sur la figure V.25 sont représenté les projections de l'échantillon réduit classé par l'algorithme de Sammon et par

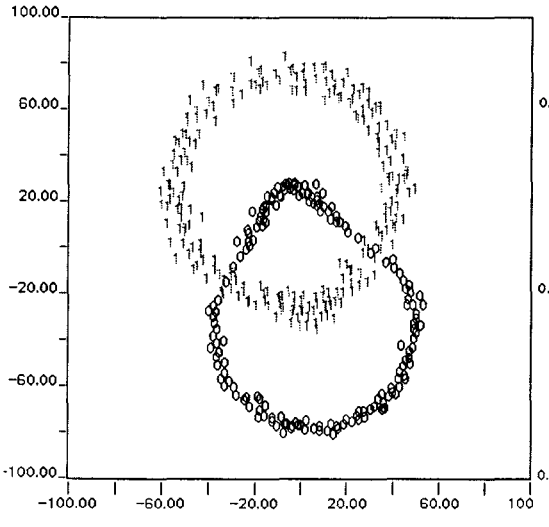


- (a) échantillon classé par réseaux de neurones à apprentissage compétitif -

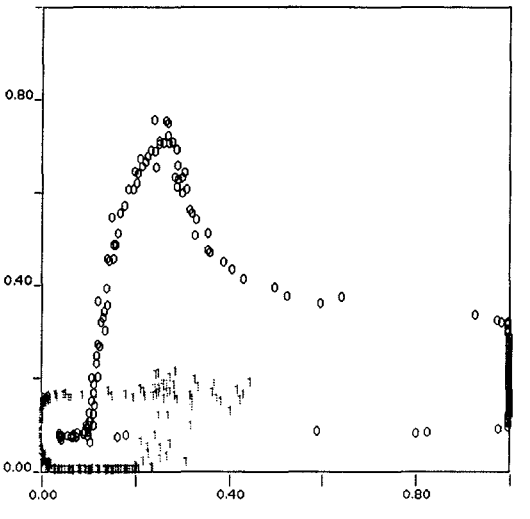


- (b) échantillon classé par l'algorithme d'agrégation avec rayon -

Figure V. 24 :Projections par carte de Kohonen de l'échantillon classé



- (a) projection par l'algorithme de Sammon -



- (b) projection par l'algorithme perceptron multicouches -

Figure V. 25 : Projections de l'échantillon classé par l'algorithme d'assignation avec rayon

perceptron multicouches. Nous pouvons constater que les classes correspondent à celles décelées lors de l'examen visuel initial.

L'analyse locale n'a pas permis de découvrir d'autres classes. L'analyse de l'échantillon est terminée. La comparaison de l'échantillon classé avec l'échantillon étiqueté a permis de calculer un taux d'erreur de classement nul.

V.4 DISCUSSIONS

Nous avons expérimenté dans ce chapitre les réseaux à apprentissage compétitif dans un contexte de classification interactive. Sous une même architecture, mais en utilisant différents algorithmes d'apprentissage, ces réseaux peuvent être utilisés en tant qu'algorithmes de classification, de réduction de données et outil de représentation de données multidimensionnelles. Les expérimentations présentées dans ce chapitre nous amènent à effectuer quelques remarques générales.

Nous avons utilisé, pour nos représentations, des cartes de Kohonen comprenant 50*50 neurones. Cette taille nous a paru satisfaisante en termes de définition des cartes et de temps d'apprentissage. Rappelons en effet que ces essais ont été réalisés sur une machine séquentielle et que le temps d'apprentissage des cartes sur ces machines évolue fonction du carré du nombre de neurones. On peut utiliser des cartes comportant moins de neurones, ce qui rend l'apprentissage plus rapide. Des essais ont été menés avec satisfaction sur des réseaux de 20*20 neurones pour l'ensemble des exemples traités. Par ailleurs, l'analyse locale des classes permet, dans ce cas, de déceler des classes rendues invisibles par la diminution de la définition des cartes.

Nous avons appris les cartes avec différentes initialisations des vecteurs poids. Nous n'avons pas remarqué d'incidence des différentes initialisations des poids sur la qualité des représentations obtenues et ce, quelque soit la distance utilisée.

Bien qu'elle n'ait pas d'influence notable sur les résultats de la classification interactive, le type de distance permettant de définir les relations de voisinage sur la carte influe directement sur les temps d'apprentissage lorsque les simulations sont effectuées sur une machine séquentielle. Nous avons relevé les temps d'apprentissage sur une machine séquentielle à partir de l'exemple 4. Ainsi, un apprentissage utilisant une distance triangulaire nécessite en moyenne 6600 secondes, un apprentissage utilisant une distance sphérique 9600 secondes et un apprentissage utilisant une distance carrée 14000 secondes. Rappelons qu'une implantation parallèle de ce type de réseau élimine ce problème. Nous avons cependant préféré baser nos analyses sur des cartes à maillage hexagonal en utilisant une distance Euclidienne, ceci en raison de la représentation plus isotropique due à ce type de maillage.

Parmi les caractéristiques de distances inter-neurones exposées au cours de ce travail, aucune n'offre en général des qualités de représentation supérieures aux autres. Aussi pour

analyser les échantillons les plus divers, nous proposons d'utiliser plusieurs de ces caractéristiques.

En ce qui concerne les techniques de projection plus classiques, l'analyse en composantes principales, basée sur des moments d'ordre 1 et 2, s'avère incapable de révéler de façon satisfaisante les classes des exemples de ce chapitre.

La technique de projection de Sammon a permis de discerner distinctement les classes des différents exemples de ce chapitre. Par rapport à la technique de représentation par carte de Kohonen, l'algorithme de Sammon offre certains avantages. Ainsi l'erreur de Sammon fournit une indication sur la qualité de la projection. De plus cette technique nécessite peu de coefficients à régler. Cependant, lorsque la taille de l'échantillon augmente, les temps de calcul deviennent excessifs. Dans ce cas une réduction de données à l'aide de techniques neuronales détaillées dans le second chapitre permet de résoudre ce problème. Par ailleurs, la technique de projection par l'algorithme de Sammon ne permet pas de discerner les modes de classes ayant un degré important de chevauchement.

La projection par perceptron multicouches offre des résultats de projection divers. Ainsi l'exemple 5 a nécessité un apprentissage de 1000 époques tout en offrant une projection facilement exploitable. L'apprentissage de l'exemple 3 a nécessité 2000 époques pour que la projection soit exploitable. Ce nombre d'époques a considérablement augmenté pour l'apprentissage de l'exemple 4. Ainsi il a été fixé à 10000 époques. Cette technique de projection s'est avérée être la plus coûteuse en temps de calcul lors de nos essais. Par ailleurs, le nombre de neurones des couches cachées 1 et 3 est délicat à fixer.

Nous nous sommes servi, pour classer les différents échantillons, des techniques de classification par réseaux de neurones à apprentissage compétitif. Lors de cette étape, nous avons utilisé l'algorithme d'apprentissage compétitif classique. Les algorithmes d'apprentissage compétitif sensible à la fréquence, avec pénalisation du rival, généralisé et de Kohonen ne sont pas nécessaires dans ce cadre spécifique puisque l'opérateur initialise, par l'intermédiaire des cartes, les vecteurs poids initiaux au centre des classes. Ainsi, il n'existe plus de neurones sous-utilisés. Il est cependant possible d'utiliser ces techniques en parallèle avec la classification interactive dans le but d'apporter une information supplémentaire à l'analyste.

L'algorithme d'assignation avec rayon permet d'exploiter l'information de distances fournie par les cartes. Cette technique a été employée avec succès sur différents échantillons.

L'approximation de la distance minimale interclasse obtenue par l'intermédiaire des cartes est de l'ordre de la moitié de la valeur réelle de R, ce qui permet de classer correctement les classes ayant un degré de chevauchement nul.

Il nous apparaît évident qu'en intégrant d'autres techniques de projection et d'autres techniques de classification, nous pourrons obtenir un outil de classification interactif plus puissant capable d'analyser une grande diversité d'échantillons. C'est dans cette optique que nous avons élaboré, d'une part différents types de représentation par carte de Kohonen et ,d'autre part, une interface multifenêtrage permettant d'obtenir simultanément différentes représentations d'un échantillon d'observations multidimensionnelles.

CONCLUSION GENERALE

Dans ce mémoire, nous avons exploré les possibilités offertes par les réseaux de neurones à apprentissage compétitif et ses dérivés dans le cadre de la classification non supervisée. Il nous est apparu qu'avec ce type de réseau et en utilisant des techniques d'apprentissage adéquates, nous pouvons traiter différents problèmes tels que la classification automatique, la réduction de données et la réduction de dimension à des fins de classification interactive par projection des données sur un plan.

Le premier chapitre nous a permis d'exposer de manière succincte des techniques de classification ainsi que les techniques neuronales. Nous avons, de plus, présenté les applications existantes des réseaux de neurones dans le domaine de la classification.

Nous avons détaillé, dans le second chapitre, les réseaux de neurones à apprentissage compétitif, faisant apparaître certaines analogies entre les réseaux à apprentissage compétitif et l'algorithme de classification des K-means. Nous avons vu que l'ajout de mécanismes de « compétition », « conscience », « voisinage » et de « rivalité » dans les techniques d'apprentissage de ces réseaux permettent de limiter les effets de l'initialisation des vecteurs moyennes initiaux sur le résultat de la classification par rapport au comportement de l'algorithme des K-means. Ces réseaux ont, de plus, montré leurs capacités à réduire les données lorsque l'on introduit, dans la phase d'apprentissage, des coefficients de conscience ou des relations de voisinage. Le couplage de différentes techniques d'apprentissage permet d'obtenir d'excellents résultats dans ce domaine.

Le troisième chapitre a été consacré à la représentation de données multidimensionnelles par l'intermédiaire des réseaux de Kohonen. Ces réseaux constituent

une généralisation des réseaux à apprentissage compétitif. On prend en compte, dans ce type de réseau, la position des neurones dans un espace à deux dimensions. Ces neurones sont disposés de manière régulière dans cet espace suivant un maillage carré ou hexagonal formant ainsi une carte. L'introduction de relations de voisinage dans la phase d'apprentissage auto-organise le réseau de telle façon que deux neurones voisins sur la carte représentent deux observations ou groupes d'observations voisins dans l'espace de représentation des observations. Après la phase d'apprentissage, nous utilisons ces cartes afin de représenter des caractéristiques des distances inter-neurones ou des estimations de la fonction densité de probabilité en niveaux de gris. La représentation de ces différentes caractéristiques permet de révéler la structure des échantillons analysés.

Un processus de classification interactive intégrant ces nouveaux outils est détaillé dans le quatrième chapitre. Un logiciel graphique a été spécialement développé pour permettre une utilisation aisée de toutes ces techniques.

Enfin nous avons testé dans le dernier chapitre les outils de classification neuronaux sur des données réelles relatives à une étude biométrique des abeilles, puis sur deux exemples générés artificiellement. Ces deux derniers exemples nous ont donnés l'occasion de présenter une technique de classification non neuronale, exploitant l'information fournie par les représentations des caractéristiques de distances inter-neurones. Cette nouvelle technique permet de traiter ces deux exemples beaucoup plus rapidement que ne le fait l'algorithme de classification du plus proche voisin.

D'une manière générale les réseaux à apprentissage compétitif héritent des avantages bien connus des techniques neuronales. Ils sont massivement parallèles, permettant ainsi de traiter des échantillons d'apprentissage volumineux. Ils sont robustes et sont susceptibles de fournir des résultats très intéressants dans un environnement non stationnaire. Sous une même architecture et en ne modifiant dans la phase d'apprentissage que la loi d'activation ou la règle d'apprentissage, nous pouvons utiliser ces réseaux de neurones comme techniques de classification, de réduction de données ou comme techniques de représentation de données multidimensionnelles.

Les techniques de représentation par carte de Kohonen se distinguent des autres techniques de projection plus traditionnelles par les points suivants :

- Le temps d'apprentissage sur une machine séquentielle dépend linéairement de la taille de l'échantillon d'apprentissage.

- L'apprentissage de la carte sépare la notion d'ordre de celle de distance entre observations comme c'est le cas de l'algorithme de Sammon. Ainsi, la position des neurones sur la carte reflète l'ordre des distances entre observations.

- Les positions relatives des vecteurs poids des neurones sont affichées en niveaux de gris sur la carte. Lorsque la densité de probabilité est connue ou estimée, il est aussi possible de la représenter en niveaux de gris sur la carte.

- La représentation des caractéristiques des distances inter-neurones nous donne une indication concernant les distances interclasses.

- Les cartes de Kohonen peuvent être aussi utilisées pour la réduction de données et la classification.

Bien entendu, les cartes de Kohonen présentent quelques inconvénients :

- Ainsi la forme géométrique des classes n'est pas respectée.

- Les vecteurs poids sont sensés représenter les observations et leurs convergences conditionnent les résultats de la projection : la position des neurones sur la carte reflète l'ordre des distances entre vecteurs poids dans l'espace.

- L'apprentissage du réseau nécessite un grand nombre de coefficients à régler.

- Les cartes de Kohonen expriment indirectement les relations d'ordre de distance existant au sein des données. En effet, elle est basée sur la position des vecteurs dans l'espace et non pas sur les données elles-mêmes.

- Les critères permettant de quantifier la qualité de représentation des cartes sont heuristiques et sont de plus très coûteux en temps de calcul.

- Enfin, la convergence des poids et la propriété d'auto-organisation des neurones ne sont pas démontrées pour des réseaux d'ordre topologique supérieur à 1.

Cependant ce type de représentation offre de nouvelles perspectives:

- Différents critères mesurant l'ordre des distances entre neurones sur la carte par rapport à l'ordre des distances entre vecteurs poids dans l'espace ont été élaborés [BAUER 92], [BEZD 95], [VILL 96]. Ces critères fournissent une information a posteriori sur la capacité des cartes à conserver l'ordre des distances entre neurones et n'interviennent pas dans l'apprentissage du réseau. Malgré l'effort pour élaborer un critère de comparaison entre les différentes méthodes de projection, aucun ne nous paraît satisfaisant. Une synthèse

intéressante et prometteuse a été effectuée par Pal et Bezdek [BEZD 95].

- Les techniques d'affichage des caractéristiques des distances inter-neurones (maximum, médian) sur la grille définissent des frontières entre les classes. Ces frontières dépendent de la « bonne » convergence des vecteurs poids qui n'est pas contrôlée.

- Les cartes de Kohonen utilisent un apprentissage dérivé de l'apprentissage compétitif et semblent par conséquent bien adaptées à un environnement non stationnaire. Rappelons que, dans ce cas, l'échantillon évolue au cours du temps. Une recherche dans cette voie nous semble prometteuse.

- La carte de Kohonen, basée sur une imitation de l'organisation du système nerveux, produit une projection non linéaire discrète respectant l'ordre au sein des données. Une étude plus poussée mérite d'être effectuée.

L'interface graphique développée permet aisément de comparer les différentes cartes et projections. Les couplages entre les projections et les algorithmes de classification sont ainsi facilités. Cependant, afin d'obtenir un outil plus puissant, il serait intéressant d'intégrer toutes les techniques classiques de projections et de classification.

Une très grande attention a été apportée à la facilité d'utilisation cependant ce système peut être amélioré en y apportant des outils de filtrage , zoom, etc... Malgré ces efforts, l'utilisation de toutes ces techniques reste très délicate pour un non initié et surtout lors de l'interprétation des résultats. Il serait très intéressant de munir cette interface d'un système d'aide à la décision.

ANNEXE 1:

CRITERE QUANTITATIF DE DISTORSION DE KULLBACK

La communication présentée ci-après a été exposée à la conférence IEEE SMC, Le Touquet, Vol. 4, pp. 496-500, Octobre 1993.

Competitive Learning Neural Network Applied to Multivariate Data Set Reduction

Stephane Delsert, Denis Hamad, Mohamed Daoudi and Jack-Gérard Postaire

Centre d'Automatique de Lille
Université des Sciences et Technologies de Lille
F-59655 Villeneuve d'Ascq Cedex, France

Abstract - **This paper presents three competitive learning neural networks applied to the multivariate data set reduction problem. The synaptic vectors of a neural network are used as prototypes of the data set. The quality of the results are compared, using an example, by means of an informational criterion. This criterion evaluates the quality of the matching between the density function estimated from the whole data set and that determined from the reduced set.**

I. INTRODUCTION

In a wide range of applications of cluster analysis, including image processing, speech processing or fault diagnosis, it is necessary to reduce the size of the data set in order to reduce storage and computation time [1]. The problem is to find a small number of representative prototypes of the whole data set, so that the resulting reduced set contains as much information as possible about the original data. A classical approach to sample size reduction consists in finding a small number of prototypes (or codevectors) by means of the vector quantization technique [2]. The set of prototypes (or codevectors) is called a codebook. The vector quantization technique consists to mapping the n-dimensional Euclidean space into a finite subset of codevectors such that the average distance between the samples and their codevectors is minimised [2], [3]. However this distance function is generally not convex and may contain multiple local minima. Therefore the vector quantization approach produces non optimal codebooks [3].

Another solution is to select the prototypes so that the probability density estimate with these prototypes remains as close as possible to the density estimated with all the available data set. The probability density estimation is given by means of the nearest neighbour density estimate [4] or by the Parzen density estimate [5]. This approach splits the data samples into two subsets. The first one called STORE and the second called TEST. The procedure is iterative and each iteration exchanges one sample between the two subsets in order to minimise an entropy criterion used as a similarity distance. At each iteration, the procedure requires the estimation of the probability density functions of the whole data set and of the reduced set.

In this paper, we propose an alternative approach to reduce the data information involving both neural networks and a performance criterion. This criterion is used to evaluate the quality of the matching between the density function estimated from the entire data set and that determined from the reduced set. To be more specific, the data reduction problem is solved by means of competitive learning neural networks. Note that, unlike vector quantization which is a batch algorithm, the competitive learning is an on-line algorithm.

The neural networks structure consists in two layers of neurones feedforward connected by means of weight vectors. They are characterised by their competitive, unsupervised, or self-organising learning paradigm. Three solutions for learning are proposed : competitive learning, Kohonen self- organising feature map and frequency sensitive competitive learning. For each of them, unlabelled observations are sequentially presented to the network during the learning stage. At the end of the learning, the weight vectors of the network provide prototypes of these observations (sectionII).

We have compared the three training algorithms by means an informational criterion to measure the quality of the data reduction given by each of the learning procedures. This criterion measures the "similarity distance" between the probability density function (p.d.f.) of the selected prototypes and the p.d.f. underlying the distribution of the whole data set (section III).

Related simulations using artificially generated multi-dimensional data sets are reported in section IV.

II. NEURAL NETWORK TRAINING ALGORITHMS FOR DATA COMPRESSION

neighbourhood is decreased as the training goes on, until the learning of the network becomes a competitive learning procedure after a sufficient amount of training.

The weight vector of this winning unit, noted i^*, and its neighbours i are modified according to the equation (5) :

$$W_i(t+1) = W_i(t) + a_i(t)\Phi_i(i^*, t)y_i(t)[X(t) - W_i(t)] \quad (5)$$

where i^* is the winning unit defined by :

$$i^* = \text{Arg} \min_i [d(X(t), W_i(t))] \quad (6)$$

$a_i(t)$ is the learning coefficient rate at time instant t defined in the precedent section.
$\Phi_i(i^*, t)$ is a neighbourhood function that is used at time t to alter the step size of the i^{th} weight vector. It is a function of the physical distance between its associated node on the lattice and the winner i^*. Typically, it is non zero for nodes close to the i^{th} unit in the lattice, and is zero outside this neighbourhood.

At the end of the training step, the Kohonen self-organising scheme, as the competitive learning procedure, partitions the input space into regions. Each region is assigned to the neurone which is the most sensitive to the sample belonging to that region. Furthermore, the network may avoid a local minima thanks to the neighbourhood notion.

One drawback of the self-organising feature map is that additional computation arises from both the calculation of the neighbourhood of the winning unit and from the updating of all the neighbourhood members.

C. *Frequency sensitive competitive learning*

In this last approach, each neural unit incorporates a count of the number of times it has been the winner during the learning stage [8], [9]. To be more specific, the distance given by equation (2) is modified as follows :

$$d(X(t), W_i(t)) = f_i(t)\|X(t) - W_i(t)\| \quad (7)$$

where :

$$f_i(t) = f_i(t-1) + y_i(t) \quad (8)$$

is a frequency sensitive coefficient which is the number of times the i^{th} unit wins the competition during the learning stage.

If a given neural unit wins the competition frequently, its count and, consequently, its distance to the observation vector increases. This reduces the likelihood that the unit will be the winner in the next steps. Then, the other units, with lower count values, have more chance to win the competition.

III. AN INFORMATION CRITERION FOR PERFORMANCE EVALUATION

Given the Q samples $\{X_1, ..., X_q, ..., X_Q\}$ drawn from a density function $f(X)$ of a random vector X, we wish to select M prototypes ($M < Q$), such that the k-nearest neighbour density estimates for the N samples and the M prototypes are as close as possible.

A. *Non parametric probability density estimation*

Let $\{X_1, ..., X_q, ..., X_Q\}$, where $X_q = (x_{q1}, ...x_{qi}, ..., x_{qn})^T$ the set of unlabelled observations and $\{W_1, ..., W_i, ..., W_M\}$ where $W_i = (w_{i1}, ..., w_{ij}, ..., w_{in})^T$ the set of prototypes of these observations given by the one of the three learning algorithms shown above.

In order to evaluate the performance of these data reduction algorithms, we proceed by means of a partition of the data space into a set of H exclusive hypercubes whose centres define a lattice. The p.d.f. is estimated at each hypercube centre using the k-nearest neighbours procedure which consists in determining the smallest domain containing the k-nearest neighbours for that centre. This estimation procedure is applied to the whole date set and to each prototype set given by means of one of the three learning procedures.

Let $\{Z_1, ..., Z_h, ..., Z_H\}$ the set of hypercube centres, $D(Z_h)$ the smallest domain surrounding the point Z_h and $V[D(Z_h)]$ the volume of this domain. In what follows, we use the Euclidean distance. Thus, the domain $D(Z_h)$ is an hypersphere centred at Z_h with radius $r_Q(Z_h)$. $r_Q(Z_h)$ is the distance between Z_h and its k^{th} nearest neighbour in the data space. The volume of this hypersphere is given by :

$$V[D(Z_h)] = c\, r_Q^n(Z_h) \quad (9)$$

where c is a constant.

Our notation for the k-nearest neighbour density estimate at Z_h, given Q samples and M prototypes is :

$$\hat{f}_Q(Z_h) = k / [Q\, c\, r_Q^n(Z_h)] \quad (10)$$

$$\hat{f}_M(Z_h) = k / [M\, c\, r_M^n(Z_h)] \quad (11)$$

where $r_M(Z_h)$ is the distance from Z_h to its k-nearest neighbour among the M prototypes.

We estimate the non parametric density based on the original data set and on the prototypes by means of this k-nearest neighbour procedure. An informational criterion is

The competitive learning network structure consists of two layers of neurones (Fig. 1) [7], [10]. The first one, called the input layer, receives the input signals while the second layer elaborates prototypes of the data. The neural units of the first layer are feedforward connected to the units of the second layer. Each interconnection from an input unit j to an output unit i has a weight w_{ij}. That means that each output unit i has a corresponding weight vector W_i. The neural units of the second layer are interconnected to elaborate the winning neural units by inhibiting the other units. The number n of units in the input layer is equal to the dimension of the data space while the number of units in the output layer is determined by the topological dispersion of the classes present in input patterns. Let M be the set of outputs units.

The training of the network involves the repeated presentation of the set of unlabelled input samples $\{X_1, ..., X_q, ..., X_Q\}$, where $X_q = (x_{q1}, ..., x_{qi}, ... , x_{qn})^T$. At the end of the learning, the synaptic vectors $\{W_1 , ..., W_i , ..., W_M\}$ where $W_i = (w_{i1}, ..., w_{ij}, ... ,w_{in})^T$ are used as the prototypes of theses observations since they converge towards the centroides of the clusters.

Three learning procedures are considered and evaluated : competitive learning, self organising feature map and frequency sensitive competitive learning.

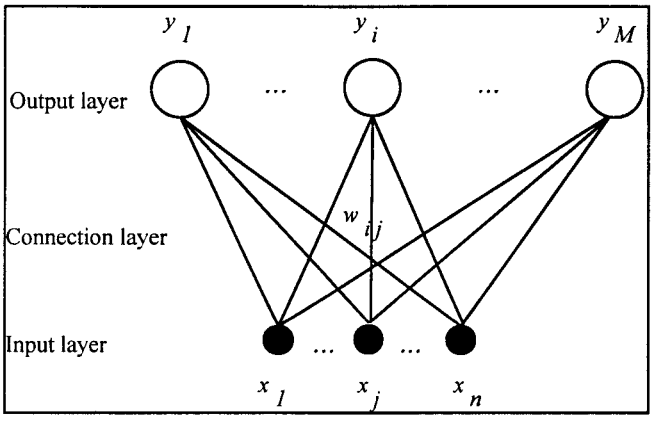


Fig. 1. Competitive learning network structure

A. Competitive learning

Competitive learning encodes statistically data vectors in order to compress the data [7]. Competitive learning is an adaptive process, in which the units of the neural network are tuned to specific features of inputs. The basic principle underlying competitive learning is as follows.

Data samples are sequentially presented to the network. At each presentation of a sample $X(t)$ of $\{X_1, ...,X_q, ..., X_Q\}$, a competition is held among the neurones of the competitive layer. The winner is the one whose weight vector is the closest to this sample. The output of the winner is then equal to 1 while all other output units are equal to 0.

$$y_i(t) = \begin{cases} 1 \text{ if } d(X(t), W_i(t)) \leq d(X(t), W_{i'}(t)) \; i' \neq i \\ 0 \text{ else} \end{cases} \quad (1)$$

where :

$$d(X(t), W_{i'}(t)) = \|X(t) - W_{i'}(t)\| \quad (2)$$

is an Euclidean distance between the observation vector $X(t)$ and the weight vector $W_{i'}(t)$ of the unit i' of the output layer.

The network updates the winning weight vector according to the difference between the observation vector and the corresponding weight vector. The updating rule for the winner can be described as follows :

$$W_i(t+1) = W_i(t) + a_i(t) y_i(t) [X(t) - W_i(t)] \quad (3)$$

- where t is the current learning time.
- $W_i(t)$ is the weight vector, at time t, of the group of links connecting the input layer units to the i^{th} unit of the output layer.
- $X(t)$ is the observation vector presented to the input layer at time t.
- $a_i(t)$ is the learning rate, at time t, which satisfies to the constraints of stochastic approximations :

$$\sum_t a_i(t) = \infty \text{ and } \sum_t a_i^2(t) < \infty \quad (4)$$

Note that the learning rate can be an hyperbolic, exponential or linear function of time t. In what fallows, it is assumed to be a linear function, i.e. it decays to zero as t increases.

Competitive learning presents the drawback that the resulting synaptic vectors depend on the initiation phase such that only the neurones which have their weight vectors near a cluster of observations win the competition during the leaning procedure. The self-organising feature map uses the neighbourhood notion to solve the initiation problem.

B. Self-organising feature map

In the self-organising feature map of Kohonen [10], the neurones are arranged in a lattice structure, in which each neurone is assigned a specific position. Furthermore, the neurones also organise themselves in such a way that the structural relationships between regions in the data space are captured by the lattice structure.

During the training process, the winning neurone and its neighbouring neural units are updated. The size of the

then used as an efficient measure of the quality of the data reduction obtained with each learning procedure.

B. Informational criterion

In order to evaluate the similarity of the two estimated p.d.f., we use the concept of entropy [6]. We introduce the Kullback-Leiber number in which the expectation appearing in the expression is approximated by a sample mean over the whole data set. The criterion is of the form :

$$J = \int \delta\left(\hat{f}_Q(Z), \hat{f}_M(Z)\right) dZ \qquad (12)$$

where $\delta(.,.)$ is the measure of the "distance" between $\hat{f}_Q(z)$ and $\hat{f}_M(z)$, which is related to the log-likelihood ratio :

$$\delta\left(\hat{f}_Q(Z), \hat{f}_M(Z)\right) = \left| Ln\left[\hat{f}_Q(Z) / \hat{f}_M(Z)\right] \right| \qquad (13)$$

Equation (12) can be approximated by :

$$J = (1/H) \sum_{h=1}^{H} \left| Ln\left[\hat{f}_Q(Z_h) / \hat{f}_M(Z_h)\right] \right| \qquad (14)$$

which can be written using (10) and (11) as :

$$J = (1/H) \sum_{h=1}^{H} nLn\left[r_Q(Z_h)\right] - nLn\left[r_M(Z_h)\right] + Ln(M) - Ln(Q) \qquad (15)$$

J is used as a quality measure of the data reduction. The experimental results are reported in the following section.

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we present an example of data sets artificially generated and we apply the informational criterion (15) to evaluate the performance of the three learning algorithms.

Example Data set has been artificially generated by combining two clusters composed of 400 samples each. The first cluster is generated as follows :

$$\begin{aligned} x_1 &= A\cos\theta + n_1 \\ x_2 &= A\sin\theta + n_2 \end{aligned} \qquad (16)$$

where $A = 20.0$, θ is a normal random variable with mean π and standard deviation $\pi/5$. n_1 and n_2 are independant identically distributed normal random variables with zero means and unit variances. The second cluster is also generated as :

$$\begin{aligned} x_1 &= A\cos\theta + n_1 \\ x_2 &= A\sin\theta + n_2 - A \end{aligned} \qquad (17)$$

where $A = 20.0$, θ is a random variable with mean π and standard deviation $\pi/5$. n_1 and n_2 are random independant identically distributed normal variables with unit variances. Fig. 2 shows the original data set.

The number of units in the input layer of the neural network is equal to the dimension of the input space, i.e. two, while the number of units in the output layer is fixed to 400.

For the three learning procedures, the number of iterations is fixed to 5000 and the weight vectors are randomly generated. The learning rate $a(t)$ is initialized at 0.7 and decays to 0 during the learning time. The Kohonen network is organised as a 20x20 lattice. The number of neighbours is initialized at 10 and is decreased of one unit every 400 iterations.

Fig 3 shows the results of the data reduction for the competitive learning (Fig. 3.a), the Kohonen self-organising feature map (Fig. 3.b) and frequency sensitive competitive learning (Fig. 3.c).

The quality of the data reduction for each learning procedure of the neural network is compared by means of the proposed informational criterion (15). Fig. 4 shows the information criterion in terms of the learning time for the three learning procedures. The data reduction obtained with the frequency sensitive competitive learning and the self-organising feature map are better than those given by the competitive learning scheme. Furthermore, it appears that the self organising feature map algorithm is very much time consuming, so that this comparative study demonstrates that the best solution is to use the frequency sensitive competitive learning algorithm.

REFERENCES

[1] R. O. Duda and P.E Hart, *Pattern Classification and Scene Analysis*, J. Wiley Editions, New-York, 1973.

[2] Y. Linde, A. Buzo, and R. Gray "An Algorithm for Vector Quantizer Design," *IEEE Trans. on communications*, Vol. 28, N°. 1, pp. 84-95, 1980.

[3] E. Yair, K. Zeger, and A. Gersho "Competitive Learning and Soft Competition for Vector Quantizer Design," *IEEE Trans. on Signal Processing*, Vol. 40, N0. 2, 1992.

[4] K. Fukunaga and J. M. Mantock, "Nonparametric Data Reduction," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 6, N° 1, pp.115-118, January, 1984.

[5] K. Fukunaga and R. Hayes, "The Reduced Parzen Classifier", *IEEE Tans. on Pattern Analysis and Machine Intelligence*, Vol. 11, N° 4, pp. 423-425, April, 1989.

[6] S. Kullback, *Information Theory and Statistics*, J. Wiley Editions, New-York, 1959.

[7] B. Kosko, *Neural Networks and Fuzzy Systems, a Dynamical Systems Approach to Machine Intelligence,* Printice Hall International, 1992.

[8] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton, "Competitive Learning Algorithm for Vector Quantization," *Neural Networks*, Vol. 3, pp. 277-290, 1990.

[9] W. C. Fang, B. J. Sheu, O. T.-C. Chen, and J. Choi, "A VLSI Neural Processor for Image Data Compression Using Self-Organisation Networks," *IEEE Trans. on Neural Networks*, Vol. 3, N° 3, pp. 506-518, May, 1992.

[10] Kohonen T, *Self-Organisation and Associative Memory,* 2nd Edition New York, Springer-Verlag,1984.

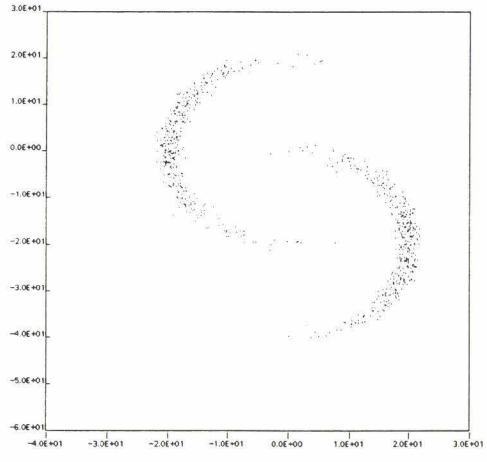
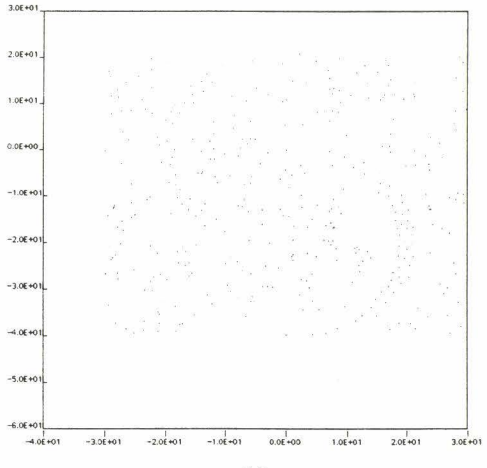
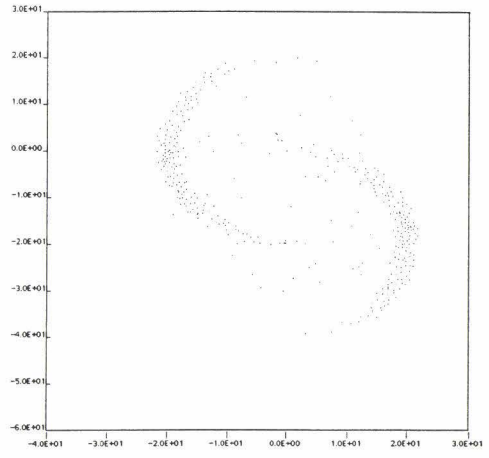


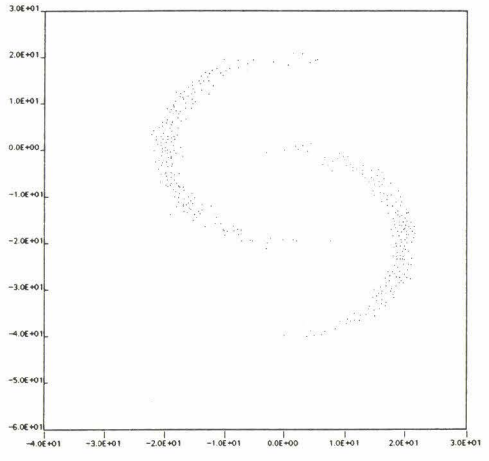
Fig. 2. Two-dimensional diagram of the data set.



(a)



(b)



(c)

Fig. 3. Data set reduction. (a) Competitive learning. (b) Self organising feature map. (c) Frequency sensitive competitive learning.

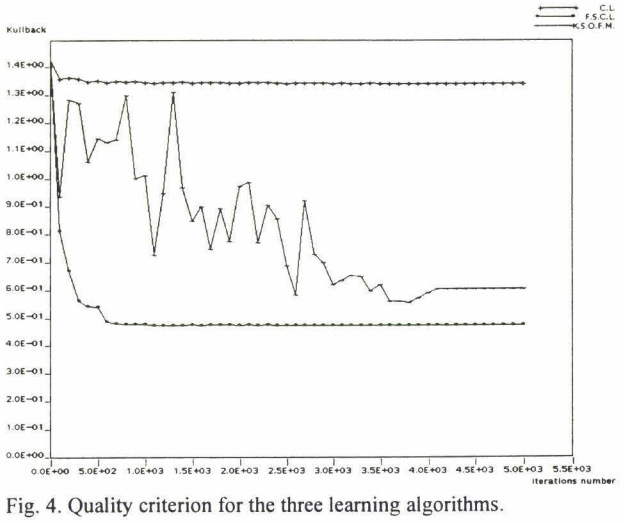


Fig. 4. Quality criterion for the three learning algorithms.

ANNEXE 2:

ESTIMATEUR NON PARAMETRIQUE DE PARZEN

Cette méthode d'estimation fût proposée initialement par Parzen et Rosenblatt [ROSE 56][PARZ 62], puis étendue au cas multidimensionnel par Cacoullos et Murthy [CACO 62][MURT 66]. L'estimateur de la fonction densité de probabilité au point d'estimation X prend la forme suivante :

$$\hat{p}(X) = \frac{1}{Q}\sum_{q=1}^{Q} \frac{1}{V[D(X)]}\Omega\left(\frac{X - X_q}{h_Q}\right)$$

où :

D(X) est le domaine d'estimation. Dans le cas, où le domaine correspond à une hypersphère de rayon h_Q centrée au point d'estimation X, le volume V[D(X)] est donné par :

$$V[D(X)] = \frac{\pi^{N/2}}{\Gamma\left(\frac{N}{2}+1\right)} {h_Q}^N$$

où Γ est la fonction gamma qui s'exprime, pour une dimension entière, par :

$$\Gamma\left(\frac{N}{2}+1\right) = \left(\frac{N}{2}\right)! \qquad \text{pour N pair}$$

$$\Gamma\left(\frac{N}{2}+1\right) = \frac{(N+1)!\sqrt{\pi}}{2^{(N+1)}\left(\frac{N+1}{2}\right)!} \qquad \text{pour N impair.}$$

Ω est appelée la fonction noyau. Pour assurer la convergence de l'estimateur, cette fonction doit satisfaire les conditions de convergence suivantes :

- $\forall X, \exists B \in IR / 0 \leq \Omega(X) \leq B$
- $\lim_{\|X\| \to \infty} \Omega(X)\|X\|^N = 0$
- $\lim_{Q \to \infty} V[D(X)] = 0$
- $\lim_{Q \to \infty} QV[D(X)] = \infty$

Plusieurs types de fonctions satisfont ces conditions. Parmi celles-ci, on peut citer :

- Le noyau cubique :

$$\Omega(X) = \begin{cases} 1 \text{ si } \|X\| \leq 1 \\ 0 \text{ si } \|X\| > 1 \end{cases}$$

- Le noyau triangulaire

$$\Omega(X) = \begin{cases} 1 - \|X\| \text{ si } \|X\| \leq 1 \\ 0 \text{ si } \|X\| > 1 \end{cases}$$

- Le noyau de Cauchy

$$\Omega(X) = \frac{1}{\pi}\left(1 - X^T X\right)^{-1}$$

- Le noyau Gaussien

$$\Omega(X) = \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{1}{2}X^T X\right)}$$

Cet estimateur suppose que l'on définisse une valeur pour le rayon h_Q. Afin que l'estimateur satisfasse les conditions de convergence, la valeur de ce rayon doit être reliée au nombre d'observations de l'échantillon. En général, on définit h_Q suivant l'une des relations suivantes :

$$h_Q = h_0 \sqrt{Q} \quad \text{ou} \quad h_Q = h_0 \log Q .$$

Le paramètre h_0 a une grande influence sur la qualité de l'estimation. S'il est trop grand, les petits maxima de la densité seront indécelables. Inversement, si h_Q est trop petit, on obtient un estimateur erratique avec beaucoup de maxima parasites.

ANNEXE 3 : METHODE D'ANALYSE EN COMPOSANTES PRINCIPALES

L'analyse en composantes principales est une méthode de projection linéaire. Elle cherche à déterminer le sous-espace de dimension R, R<N, pour lequel l'erreur e_{ACP} définie par :

$$e_{ACP} = \frac{1}{2} \sum_{0 \le q < Q} (Y_q - \overline{Y})^T (Y_q - \overline{Y}) ,$$

avec : $Y_q = PX_q - O$

et $$\overline{Y} = \frac{1}{Q} \sum_{0 \le q < Q} Y_q$$

est minimale [SAPO 90][LAGA 83].

Y_q est le vecteur projeté de dimension R de l'observation X_q de dimension N,

P est la matrice de passage de dimension R×N,

et O est le vecteur origine de la transformation.

On montre que l'erreur est minimale lorsque le vecteur O est le vecteur projeté du vecteur moyenne de l'échantillon $\mathcal{X}$, et lorsque la matrice de passage P est constituée des R vecteurs propres P_r, r=1, ..., r, ..., R, associés aux R plus grandes valeurs propres λ_r, r=1, ..., r, ..., R de la matrice de covariance S de l'échantillon $\mathcal{X}$.

Si l'on pose :

$\underline{X}_C = [X_1 - \overline{X}, ..., X_q - \overline{X}, ..., X_Q - \overline{X}]$,

où $\overline{X}$ est le vecteur moyenne de l'échantillon $\mathcal{X}$, la matrice de covariance S s'exprime par :

$S=\underline{X_C}\,\underline{X_C}^T$.

De plus, si l'on indice les r plus grandes valeurs propres de la matrice de covariance S, suivant la relation :

$\lambda_1 > ... > \lambda_r > ... > \lambda_R$,

la matrice de passage P s'exprime par :

$P=[P_1...P_r...P_R]^T$,

où le vecteur P_r^T est le vecteur propre associé à la valeur propre λ_r.

Comme nous effectuons une projection sur un plan, la dimension R est fixée à 2. La matrice de passage P est alors composée des deux vecteurs propres associés aux deux plus grandes valeurs propres de la matrice de covariance S. Le plan de projection est appelé plan principal ou plan factoriel.

Algorithme de projection par analyse en composantes principales :

- *Calcul de la matrice de covariance S.*

- *Détermination des valeurs propres de S.*

- *Calcul des deux vecteurs propres, P_1 et P_2, associés aux deux plus grandes valeurs propres de S.*

- *Construction de la matrice de passage $P=[P_1P_2]^T$.*

- *Projection Y des observations X selon la relation :*

$$Y_q = P(X_q - \overline{X}).$$

On définit l'information apportée par chacun des axes de projection par le rapport suivant :

$$\frac{\lambda_r}{\sum_{0\le n<N}\lambda_n}.$$

On peut exprimer l'information apportée par le plan principal en calculant la formule suivante :

$$\frac{\sum_{n=0}^{1}\lambda_n}{\sum_{0\le n<N}\lambda_n}.$$

L'avantage de l'analyse en composantes principales est qu'elle est basée sur des techniques de calcul de l'algèbre linéaire et qu'elle est bien éprouvée.

ANNEXE 4 : ALGORITHME DE PROJECTION DE SAMMON

L'algorithme de Sammon fait partie des techniques appelées « Multidimensionnal Scaling » [SIED 88a][SAMM 69]. Ces techniques minimisent un critère d'erreur basé sur les distances entre les observations dans l'espace d'origine et les distances entre les observations projetées. Pour l'algorithme de Sammon cette erreur s'exprime par :

$$e_{SAM} = \frac{1}{\sum_{0\le q<Q}\sum_{q'<q} d^E(X_q, X_{q'})} \sum_{0\le q<Q}\sum_{q'<q} \frac{\left(d^E(X_q, X_{q'}) - d^A(Y_q, Y_{q'})\right)^2}{d^E(X_q, X_{q'})},$$

où : $d^E(X_q,X_{q'})$ est la distance Euclidienne dans l'espace de représentation des observations E.

et : $d^A(Y_q,Y_{q'})$ est la distance Euclidienne dans l'espace de projection A.

L'algorithme de Sammon se propose d'adapter de manière itérative un échantillon de points projetés $\mathcal{Y}$ initial de façon à minimiser cette erreur. Cet échantillon est composé de Q vecteurs à deux dimensions. Chacun de ces vecteurs correspond à la projection d'une observation. On note Y_q la projection de l'observation X_q. L'échantillon de points projetés $\mathcal{Y}$ peut être initialisé aléatoirement. On peut aussi utiliser l'échantillon de points projetés par l'analyse en composantes principales (cf. Annexe 3). Dans ce cas la projection initiale Y_q de l'observation X_q s'exprime selon la formule :

$$Y_q = P(X_q - \overline{X})$$

où P est la matrice de passage et $\overline{X}$ est le vecteur moyenne de l'échantillon $\mathcal{X}$.

Pour minimiser l'erreur e_{SAM}, l'algorithme utilise une technique de descente du gradient. Ainsi, pour chaque observation X_q, q = 1, 2, ..., Q, on modifie Y_q selon le schéma suivant :

$$Y_q(t+1) = Y_q(t) - \alpha \frac{\frac{\partial e_{SAM}}{\partial Y_q}}{\left|\frac{\partial^2 e_{SAM}}{\partial^2 Y_q}\right|},$$

où α est une constante que Sammon appelle « magic factor » et suggère de la fixer à 0,2.

Si l'on pose :

$$d^E(X_q, X_{q'}) = d_{qq'},$$

et : $$d^A(Y_q, Y_{q'}) = d^*_{qq'},$$

les dérivées s'expriment sous la forme :

$$\frac{\partial e_{SAM}}{\partial Y_q} = \frac{-2}{c} \sum_{\substack{q'<Q \\ q' \neq q}} \frac{d_{qq'} - d^*_{qq'}}{d_{qq'}} \left(Y_q - Y_{q'}\right)$$

et $$\frac{\partial^2 e_{SAM}}{\partial^2 Y_q} = \frac{-2}{c} \sum_{\substack{q'<Q \\ q' \neq q}} \frac{1}{d_{qq'} \times d^*_{qq'}} \left(\left(d_{qq'} - d^*_{qq'}\right) - \left\|Y_q - Y_{q'}\right\| \frac{d_{qq'}}{d^*_{qq'}} \right),$$

avec : $$c = \sum_{q'<q} d_{qq'} \text{ et } 0 \leq q < Q.$$

Remarque :

Avant d'utiliser l'algorithme, il convient d'éliminer les observations identiques car, dans ce cas, le dénominateur de l'erreur de Sammon est nul. On note Q' le nombre d'observations non identiques.

L'adaptation des observations projetées s'effectue de manière itérative. Ainsi, à l'itération de rang t, on calcule les gradients pour l'ensemble des observations. Puis on adapte l'ensemble des observations projetées Y_q. On répète ces opérations jusqu'à ce que e_{SAM} soit inférieure à un seuil prédéfini, ou jusqu'à ce qu'un nombre maximal d'itérations prédéfini soit atteint.

Algorithme de projection de Sammon :

- Eliminer dans l'échantillon les observations identiques. Soit Q' le nombre d'observations restantes.

- Initialiser aléatoirement les Q' observations projetées de l'échantillon $\mathcal{X}$.

- Pour toutes les observations, calculer $\frac{\partial e_{SAM}}{\partial Y_q}$ *et* $\frac{\partial^2 e_{SAM}}{\partial^2 Y_q}$.

- Mettre à jour tous les vecteurs Y_q *selon le schéma :*

$$Y_q(t+1) = Y_q(t) - \alpha \frac{\frac{\partial e_{SAM}}{\partial Y_q}}{\left| \frac{\partial^2 e_{SAM}}{\partial^2 Y_q} \right|}.$$

- Réitérer les deux dernières étapes jusqu'à ce que le critère d'arrêt soit vérifié.

- A la fin de l'algorithme une observation X_q *est représentée dans l'espace de projection par le vecteur* Y_q *de même indice.*

ANNEXE 5 : PROJECTION PAR PERCEPTRON MULTICOUCHES

5.1 PRESENTATION

La projection par perceptron multicouche est relativement récente [KRAM 91]. Pour projeter une observation sur un plan par cette technique, le perceptron multicouches doit être composé d'au moins une couche cachée, que nous appelons couche de projection, comprenant seulement deux neurones. Les composantes d'une observation projetée par cette technique sont les valeurs de sortie des deux neurones de la couche de projection. Comme pour les réseaux AC, la projection par cette technique s'effectue après une phase d'apprentissage. Cet apprentissage s'effectue en mode auto-associatif. Ainsi la sortie désirée du réseau doit correspondre à l'observation présentée à l'entrée du réseau. Cela implique que les couches d'entrée et de sortie ont le même nombre de neurones. L'algorithme d'apprentissage du perceptron multicouches est basé sur la technique de rétropropagation du gradient d'un terme d'erreur [PARK 85][RUME 85b][LECU 87]. La figure V.1 montre l'architecture d'un réseau à cinq couches.

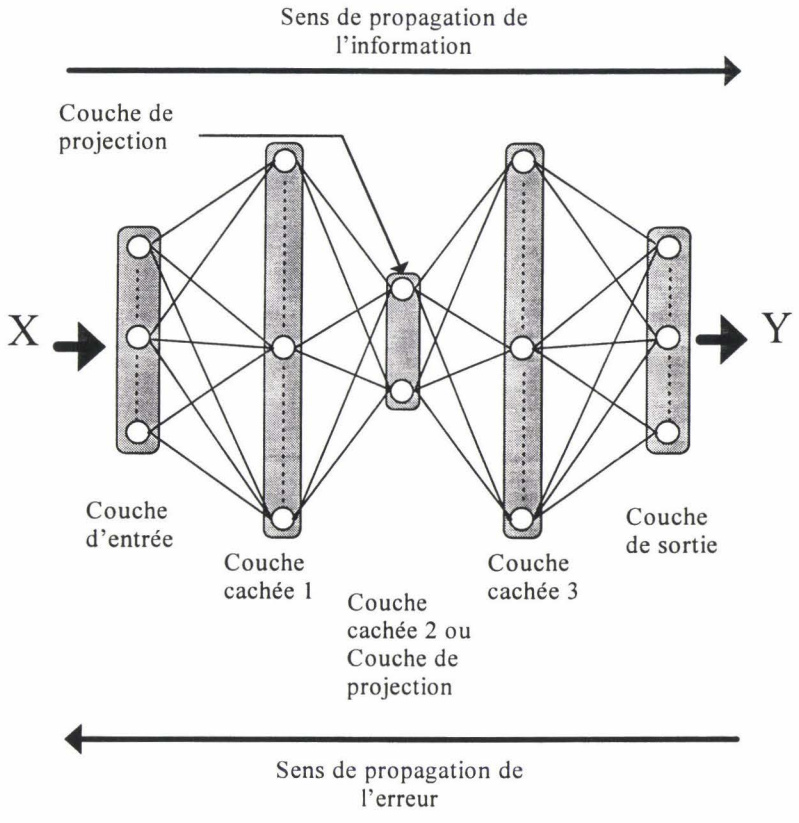


Figure 5. 1: Architecture du réseau

La première couche est, comme pour les réseaux à AC, composée de neurones servant à transmettre les signaux aux entrées des neurones de la couche suivante. Elle est composée de N neurones identités. La couche de sortie comprend, elle aussi, N neurones. Le nombre de neurones des couches 1 et 3 est fixé arbitrairement. Le nombre de neurones de la couche de projection est toujours fixé à 2 pour obtenir une projection plane.

Les neurones n'appartenant pas à la couche d'entrée sont souvent schématisés comme sur la figure 5.2.

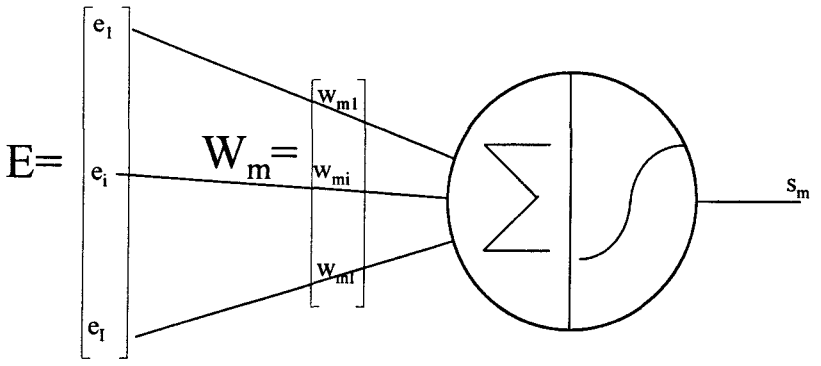


Figure 5. 2 : Représentation des neurones

E est le vecteur d'entrée du neurone m

I est le nombre de neurones de la couche précédente et correspond ainsi à la dimension du vecteur E et du vecteur W_m.

W_m est le vecteur poids du neurone m.

La sortie s_m d'un neurone m est déterminée par les équations :

$v_m=E^T.W_m$

et $s_m=f(v_m)$.

Nous appelons v_m l'entrée totale du neurone m et s_m sa sortie.

La fonction f(.) est la fonction d'activation. On utilise en général une fonction sigmoïde (cf. Figure 5.3) :

$$f(v_m)=\frac{1}{1+e^{-v_m}}.$$

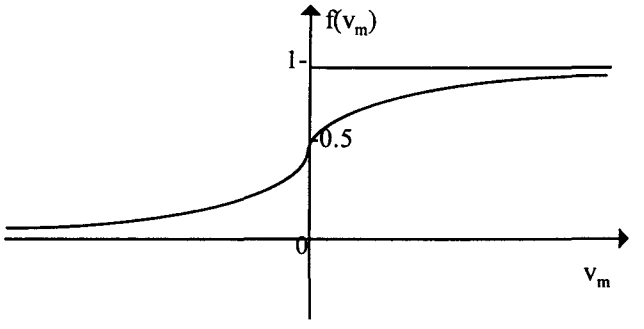


Figure 5. 3 : Fonction sigmoïde

Lors de la présentation d'une observation X à l'entrée du réseau, le signal se propage de la couche d'entrée vers la couche de sortie. En mode auto-associatif, la sortie Y du réseau, représentant la reconstitution de l'observation X présentée à l'entrée du réseau, a ses composantes comprises entre 0 et 1. Aussi, il est nécessaire dans ce cas, de normaliser l'échantillon d'observations afin que toutes les observations aient ses composantes comprises

entre 0 et 1, puisque les composantes de Y ne peuvent excéder ces valeurs. On effectue ainsi une normalisation de l'échantillon d'observations afin que celles-ci soient incluses dans un hypercube de côté 1.

5.2 APPRENTISSAGE DU RESEAU

L'apprentissage des perceptrons multicouches est basé sur la minimisation d'un critère d'erreur quadratique. Il existe deux critères d'erreurs conduisant à deux techniques d'apprentissage des perceptrons multicouches basées sur l'algorithme de rétropropagation du gradient.

La première technique est basée sur la minimisation de l'erreur totale qui, dans ce cas, est de la forme :

$$e_{PMCT} = \frac{1}{2}\sum_{q<Q}(X_q - Y_q)^T(X_q - Y_q),$$

où Y_q est le vecteur de sortie du réseau lorsque l'on présente l'observation X_q à son entrée.

La seconde technique est basée sur la minimisation de l'erreur partielle ou stochastique e_{PMCP} :

$$e_{PMCP}(X(t)) = \frac{1}{2}(X(t) - Y(t))^T(X(t) - Y(t)).$$

Nous utilisons cette deuxième technique pour nos projections. Ainsi, pendant la phase d'apprentissage du perceptron multicouches, on modifie lors de la présentation, à l'itération de rang t, d'une observation X(t), les vecteurs poids de chaque neurone selon l'équation :

$$W_m(t) = W_m(t-1) - \alpha\Delta W_m ,$$

avec : $$\Delta W_m = \frac{\partial}{\partial W_m} e_{PMCP}(X(t)) ,$$

où α est le pas du gradient qui vérifie l'inégalité: $0<\alpha<1$,

et où ΔW_m est le gradient de l'erreur partielle.

Calcul du gradient de l'erreur partielle pour un neurone de la couche de sortie

Soit $\Delta W_n=[\Delta w_{n1}, ..., \Delta w_{ni}, ..., \Delta w_{nI}]^T$ le gradient de l'erreur partielle pour un neurone n de sortie. Les composantes de ce vecteur sont définies suivant la formule :

$$\Delta w_{ni} = \frac{\partial}{\partial w_{ni}} e_{MPCP}(X(t)).$$

Ce terme se décompose en dérivées partielles selon l'expression suivante :

$$\Delta w_{ni} = \frac{\partial e_{PMCP}(X(t))}{\partial s_n} \frac{\partial s_n}{\partial v_n} \frac{\partial v_n}{\partial w_{ni}}.$$

Il nous faut déterminer ces trois dérivées partielles. Posons :

$$\Delta s_n = \frac{\partial e_{PMCP}(X(t))}{\partial s_n},$$

pour simplifier l'écriture de la première dérivée partielle. Pour un neurone de la couche de sortie, si l'on note $X(t)=[x_1, ..., x_n, ... x_N]^T$ et $S(t)=[s_1, ..., s_n, ... s_N]^T$, e_{PMCP} se met sous la forme :

$$e_{PMCP}(X(t)) = \frac{1}{2} \sum_{0<n\leq N} (x_n - s_n)^2,$$

et ainsi Δs_n s'exprime selon :

$$\Delta s_n = s_n - x_n.$$

La seconde dérivée partielle, $\frac{\partial s_n}{\partial v_n}$, s'exprime en fonction de Δs_n par la formule :

$$\frac{\partial s_n}{\partial v_n} = s_n(1-s_n).$$

Comme v_n s'exprime suivant la relation :

$$v_n = \sum_{0<i\leq I} w_{ni} s_i,$$

la dernière dérivée partielle, $\frac{\partial v_n}{\partial w_{ni}}$, nous donne la formule suivante :

$$\frac{\partial v_n}{\partial w_{ni}} = s_i.$$

En remplaçant toutes ces dérivées partielles dans l'expression définissant Δw_{ni}, on obtient la formule suivante :

$$\Delta w_{ni}(t) = (s_n - x_n) s_n (1-s_n) s_i .$$

Calcul du gradient de l'erreur partielle pour un neurone d'une couche cachée:

Pour exprimer le gradient de l'erreur partielle d'un neurone m appartenant à une couche cachée nous devons tout d'abord considérer la figure V.4. Ainsi, nous définissons la couche amont du neurone m comme la couche précédent celle à laquelle appartient le neurone. D'une manière similaire nous définissons la couche aval du neurone m comme la couche suivant celle à laquelle appartient le neurone. Les couches précédente et suivante d'une couche sont définies selon le sens de propagation de l'information, c'est à dire de la couche d'entrée vers la couche de sortie. De plus, nous notons I et J les nombre de neurones des couches aval et amont du neurone m.

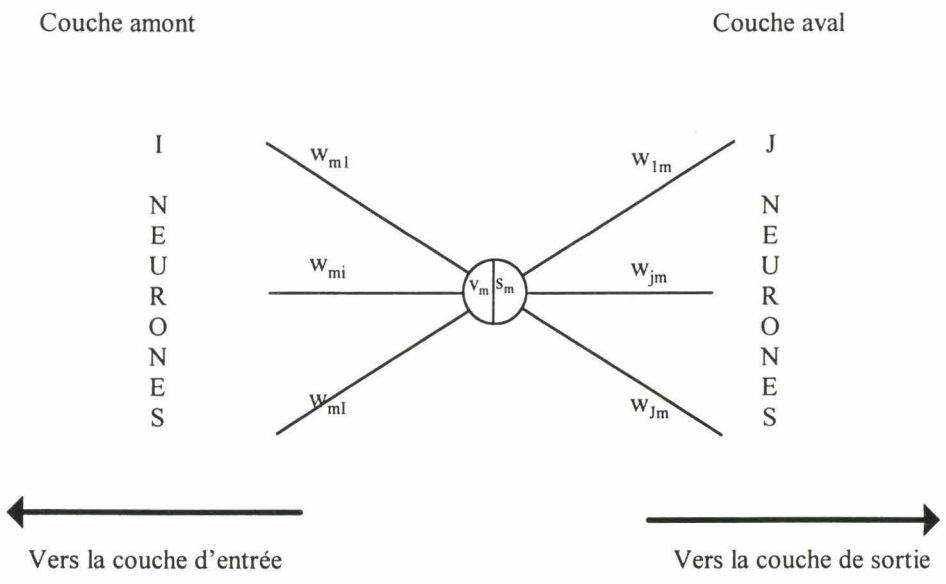


Figure 5. 4 : Neurone d'une couche cachée

La variation des poids du neurone m modifie la sortie de ce neurone et, par conséquent, les entrées de tous les neurones des couches se situant en aval de celui-ci. La sensibilité de l'erreur partielle par rapport à une modification de la sortie d'un neurone s'exprime par :

$$\frac{\partial e_{PMCP}(X(t))}{\partial s_m} = \sum_{j<J} \frac{\partial e_{PMCP}(X(t))}{\partial v_j} \frac{\partial v_j}{\partial s_m}.$$

Or, comme la dérivée partielle $\frac{\partial v_j}{\partial s_m}$ est égale à w_{jm}, la sensibilité de l'erreur partielle par rapport à la sortie d'un neurone s'exprime par :

$$\frac{\partial e_{PMCP}(X(t))}{\partial s_m} = \sum_{j<J} \frac{\partial e_{PMCP}(X(t))}{\partial v_j} w_{jm} .$$

Si l'on pose :

$$\Delta s_m = \frac{\partial e_{PMCP}(X(t))}{\partial s_m} \quad \text{et} \quad \Delta v_j = \frac{\partial e_{PMCP}(X(t))}{\partial v_j},$$

la sensibilité de l'erreur partielle Δs_m se met sous la forme :

$$\Delta s_m = \sum_{j<J} \Delta v_j w_{jm} ,$$

où Δv_j est la sensibilité de l'erreur par rapport à l'entrée totale. La sensibilité par rapport à un poids w_{mi} s'exprime alors par :

$$\Delta w_{mi} = \frac{\partial e_{PMCP}(X(t))}{\partial w_{mi}} .$$

Cette sensibilité peut aussi s'écrire sous la forme :

$$\Delta w_{mi} = \frac{\partial e_{PMCP}(X(t))}{\partial s_m} \frac{\partial s_m}{\partial v_m} \frac{\partial v_m}{\partial w_{mi}} .$$

La dérivée partielle de l'entrée totale du neurone m par rapport à W_{mi} prend la forme suivante :

$$\frac{\partial v_m}{\partial w_{mi}} = s_i .$$

Si la fonction d'activation des neurones est une fonction sigmoïde, la dérivée partielle de s_m par rapport à v_m est donnée par la formule :

$$\frac{\partial s_m}{\partial v_m} = s_m (1 - s_m) ,$$

On obtient alors l'expression de Δw_{mi} sous la forme :

$$\Delta w_{mi} = \Delta s_m s_m (1 - s_m) s_i .$$

Pour que la sensibilité par rapport aux poids puisse être déterminée, il convient de définir Δv_m :

$$\Delta v_m = \frac{\partial e_{PMCP}(X(t))}{\partial v_m} = \frac{\partial e_{PMCP}(X(t))}{\partial s_m} \frac{\partial s_m}{\partial v_m} = \Delta s_m s_m (1 - s_m) .$$

Ainsi pour déterminer la dérivée partielle Δw_{mi}, il faut d'abord calculer les sensibilités suivantes :

$$\Delta s_m = \sum_{j<J} \Delta v_j w_{jm} \qquad \text{et} \qquad \Delta v_m = s_m (1 - s_m) .$$

La sensibilité de l'erreur partielle par rapport à W_{mi} s'obtient alors en calculant l'expression :

$$\Delta w_{mi} = \Delta s_m \Delta v_m s_i .$$

Ainsi on peut formuler d'une manière générale la sensibilité de l'erreur partielle par rapport à W_{mi} selon le schéma :

$$\Delta w_{mi} = \Delta s_m \Delta v_m s_i ,$$

avec :

$$\Delta v_m = s_m (1 - s_m),$$

$$\Delta s_m = \sum_{j<J} \Delta v_j w_{jm}$$ si m appartient à une couche cachée

et $\Delta s_n = s_n - x_n$ si m appartient à la couche de sortie.

Algorithme d'apprentissage par rétropropagation de l'erreur partielle :

1. Initialisation aléatoire des poids du réseau, variable d'itération t=0.

2. Normalisation de l'échantillon d'observation $\mathcal{X}$ dans un hypercube de côté 1.

3. Tirage d'une observation X(t) dans l'échantillon $\mathcal{X}$.

4. Calcul des sorties de tous les neurones de la première couche d'entrée jusqu'à la couche de sortie. La sortie d'un neurone s'exprime par :

$$v_m = \sum_{i<I} s_i w_{mi} ,$$

avec : s_i *la sortie du neurone i appartenant à la couche située en amont,*

w_{mi} *le poids de l'arc reliant le neurone i au neurone m.*

et s_m *la sortie du neurone m, s'exprimant par :*

$$s_m = f(v_m) = \frac{1}{1 + e^{-v_m}} .$$

5. Calcul des sensibilités Δv_m *et* Δs_m *en commençant par les neurones de la couche de sortie et en remontant vers la première couche cachée. Ces sensibilités s'expriment par :*

$$\Delta v_m = \Delta s_m s_m (1 - s_m),$$

$\Delta s_m = s_m - x_m$ *si m appartient à la couche de sortie,*

$\Delta s_m = \sum_{j<J} \Delta v_j w_{jm}$ *si m appartient à une couche cachée.*

6. Adaptation des poids du réseau selon le schéma :

$$w_{mi}(t) = w_{mi}(t-1) + \alpha \Delta w_{mi},$$

avec :

$$\Delta w_{mi} = \Delta s_m s_m (1 - s_m) s_i .$$

7. Test du critère d'arrêt,

Si la condition est vérifiée : fin de l'apprentissage.

Sinon reprise du procédé à partir de l'étape 3 avec t=t+1.

Remarque :

Pour remédier aux fluctuations qui risquent d'apparaître au cours de la phase d'apprentissage, Rumelhart [RUME 85a] a introduit dans la règle de mise à jour des poids un terme momentum η tel que :

$$\Delta w_{mi}(t) = \alpha \Delta s_m s_m (1 - s_m) s_i + \eta \Delta w_{mi}(t-1)$$

avec η vérifiant l'inégalité :

$$0 \leq \eta \leq 1 .$$

Pour les perceptrons multicouches comprenant plus d'une couche cachée, on doit fixer arbitrairement le nombre de neurones des couches cachées supplémentaires. Ce nombre est laissé au libre choix de l'opérateur. Lors de nos simulations nous utilisons un perceptron multicouches comprenant cinq couches (cf. Figure 5.1). En effet, Kramer a constaté expérimentalement qu'en adoptant un réseau à cinq couches, on obtient de meilleurs résultats que ceux obtenus avec des réseaux comprenant moins de couches [KRAM 91]. Daoudi a par ailleurs utilisé avec succès ces réseaux en classification interactive [DAOU 93]. Le nombre de neurones des couches cachées 1 et 3 étant laissé au libre choix de l'opérateur, nous précisons ces nombres pour chaque exemple.

REFERENCES BIBLIOGRAPHIQUES

[ACKL 85] Ackley, D.H., Hinton, G.E. et Sejnowski, T.J. **A learning algorithm for Boltzmann machine.** *Cognitive Science*, Vol. 9, pp. 147-160, 1985.

[AHAL 90] Ahalt, S. C. Krishnamurthy, A.K. et Chen, Prakoon **Competitive learning algorithms for vector quantization.** *Neural Networks*, Vol. 3, pp 277-290, 1990.

[AHME 75] Ahmed, N. et Rao, K. R. **Orthogonal Transforms for digital signal processing.** *Springer*, Berlin, 1975.

[ANDE 73] Anderberg, M. R. **Cluster Analysis for Applications.** *Academic Press*, Inc. , New York, 1973.

[ASSE 89] Asselin de Beauville, J. P. **Panorama de l'utilisation du mode en classification automatique.** *RAIRO-APII*, AFCET, N° 23, pp. 113-137, 1989.

[BAUE 92] Bauer H.-U. et Pawelzik, K. R. **Quantifying the Neighborhood Preservation of Self-Organizing Feature Maps.** *IEEE Trans. on Neural Networks*, Vol. 3, N° 4, pp. 570-579, July 1992.

[BAYN 80] Bayne, C. K. Beauchamp, J. J. Begovitch, C. L. et Kane, V. E. **Monte Carlo comparison of selected clustering procedures.** *Pattern recognition*, Vol. 12, pp. 51-62, 1980.

[BEZD 95] Bezdek, J. C. et Pal, N. R. **An Index of Topological Preservation For Feature Extraction.** *On Pattern Recognition*, Vol. 28, N° 3, pp. 381-391, 1995.

[BISW 81] Biswas, G. Jain, A. K. et Dubes, R. C. **Mapping algorithms in ISPAHAN.** *IEEE Trans. Pattern Anal. Mach. Intell., Vol.* PAMI-3, pp. 701-708, 1981.

[BOCK 79] Bock, H. **Clustering by density estimation.** Analyse de Données et Informatique, INRIA, pp. 173-186, 1979.

[BREI 84] Breiman, L., Friedman, J. H., Olshen, R. A. et Stone, C. J. **Classification and regression tress.** Wadsworth Int. Group, Belmont, CA, 1984.

[CACO 62] Cacoullos, T. **Estimation of multivariate density.** *Ann. Inst. Stat. Math.*, Vol. 18, pp. 179-189, 1962.

[CARP 87] Carpenter, G. A. et Grossberg, S. **ART2 : Self-organisatin of stability category recognition machine codes for analog input pattern.** *Applied Optics*, Vol. 26, N°4, pp. 919-930, 1987.

[CHER 73] Chernoff, H. **The use of faces to represent points in k-dimensionnal space graphically.** *Journal of American Statistical Association*, Vol. 58, N° 342, pp. 361-368, June 1973.

[CHIE 76] Chien, Y. T. **Interactive pattern recognition : Techniques and Systems.** *IEEE Computers*, Vol. 9, N° 5, pp. 11-25, May 1976.

[CHIE 78] Chien, Y. T. **Interactive Pattern Recongnition.** Marcel Dekker Inc., New York 1978.

[COOL 71] Cooley, W. W. et Lohnes, P. **Multivariate data analysis.** *Wiley*, New York, 1971.

[COTT 87] Cottrell, M. et Fort, J.-C. **Etude d'un processus d'auto-organisation**, *Ann. Inst. Henri Poincaré*, Vol. 23, N°1, pp. 1-20, 1987.

[COVE 67] Cover, J. M. **Nearest neighbour pattern classification** *IEEE Trans. Inf. Theory*, Vol. IT-13, N° 1, pp. 21-22, 1967.

[DAOU 93] Daoudi, Mohamed **Classification interactive multidimensionnelle par les réseaux neuronaux et la morphologie mathématique**. *Thèse de doctorat, Université des Sciences et Technologies de Lille*, Novembre 1993.

[DAVA 89] Davalo, E. et Naïm, P. **Des réseaux de neurones**. Editions Eyrolles, 1989.

[DAYH 90] Dayhoff, J. E. **Neural Network architectures, an introduction**. Van Nostrand Reinhold, New York, 1990.

[DELS 93] Delsert; S., Hamad, D., Daoudi, M., Postaire, J.G. **Application of Neural Networks to Gradient Search Techniques in Cluster Analysis.** *Artificial Neural Nets and Genetic Algorithms, Proc. of Int. Conference In Innsbrück, Austria* R.F., pp 154-160, Albrecht,C.R. Reeves et N.C. Steele (Eds) Springer-Verlag, 1993.

[DESI 88] DeSieno, D. **Adding a conscience to competitive learning**. *Proceedings to the IEEE International Conference on Neural Networks*, pp. 117-124. San Diego, CA. 1988.

[DIDA 71] Diday, E. **Une nouvelle méthode en classification automatique et reconnaissance des formes : la méthode des nuées dynamiques.** *Rev. Stat. Appl.*, Vol. 19, N° 2, pp. 20-33, 1971.

[DIDA 79] Diday, E. **Optimisation en classification automatique.** Tomes 1 et 2, INRIA, 1979.

[DOOZ 95] Dooze, D. **Réseaux de neurones à apprentissage compétitif pour l'analyse de données multidimensionnelles.** Rapport de DEA, Université des Sciences et Technologies de Lille, Juin 1995.

[DUDA 73] Duda, R. O. et Hart, P. E. **Pattern classification and scene analysis**. *Editions J. Wiley*, New York, 1973.

[EVER 74] Everitt, B. S. **Cluster Analysis**. *John Wiley and Sons*, Inc., New York, 1974.

[FANG 92] Fang, W.-C. all. **A VLSI Neural Processor For Image Data Compression Using Self-Organisation Networks** *IEEE Trans. on Neural Networks*, Vol. 3, pp. 506-518, May 1992.

[FEHL 78] Fehlauer, J. et Eisenstein **A declustering criterion for feature extraction in pattern recognition.** *IEEE Trans. Comput.*, Vol. C-27, pp. 881-266, 1978.

[FIRM 95] Firmin, C. et Hamad, D. **Gaussian basis neural networks applied to cluster analysis problem.** From Data To Knowledge : Theorical and Practical Aspects of Classification, Data Analysis and Knowledge Organisation, pp. 159-166., W. Gaul, D. Pfeifer Eds., Springer Berlin, 1995.

[FISH 36] Fisher, **The use of multiple measurements in taxonomic problems.** *Ann. Eugenics*, Vol. 7, pp. 178-188, 1936.

[FRIE 67] Friedman, H. P. et Rubin, J. **On some invariant criteria for grouping data**. *J. American Statistical ASSn*, Vol. 8, pp 107-114, 1967

[FRIE 74] Friedman, J. H. 1 Tuckey, J. W. **A projection pursuit algorithm for exploratory data analysis**. *IEEE Trans. Comput.*, Vol. C-23, pp. 881-890, 1974.

[FUKU 70] Fukunaga, K. et Koontz, W. **Application of Karhunen-Loeve expansion to feature selection and ordering**, *IEEE Trans. Comput.*, Vol. C-19, pp. 881-890, 1974.

[FUKU 75] Fukunaga, K. et Hostetler, L. D. **The estimation of the gradient of a density function, with applications in pattern recognition**. *IEEE Trans.; on Information Theory*, Vol. IT-21, N°. 1, January 1975.

[GORD 87] Gordon, A. D. **Parsimonious trees**. Journal of Classification, pp. 85-101, 1987.

[GOWE 66] Gower, J. C. **Some distance properties of latent root and vector methods in multivariate analysis**. *Biometrica*, Vol. 53, pp. 325-338, 1966.

[GROS 82] Grossberg, S. **Studies of Mind and Brain.** Eds Reidel, 1982.

[GROS 87] Grossberg, S. **Competitive Learning : From Interactive Activation to Adaptative Resonance.** Cognitive Science, Vol. 11, pp. 23-63, 1987.

[HAYK 94] Haykin, S. **Neural Networks A comprehensive Foundation** Macmillan College Publishing, New York, 1994.

[HEBB 49] Hebb, D. O. **The Organisation of Behaviour**, J. Wiley Eds, New York, 1949.

[HERT 91] Hertz, J., Krogh, A. et Palmer, R. G. **Introduction to the theory of neural computation.** Addison-Wesley, Redwood City, 1991.

[HORN 92] Hornik, K. et Kuan, C. -M. **Convergence analysis of local feature extraction algorithm.** *Neural Networks*, 5, pp. 229-240, 1992.

[JAIN 92] Jain, A. K. et Mao, J. **Artificial neural network for nonlinear projection of multivariate data.** *In Proc. IEEE Intl. Joint. Conf. On Neural Networks,* Vol. 3, pp. 335-340, Baltimore, Maryland , June 1992.

[JAMB 78] Jambu, M. **Classification automatique pour l'analyse des données.** I - *Méthodes et algorithmes*. Dunod, Paris, 1978.

[JONE 68] Jones, K. L. **Problems of grouping individuals and the method of modality.** *Behavorial Science* , Vol. 13, pp. 496-511, 1968.

[KOHO 88a] Kohonen, T. **Self-organisation and associative memory**. 2d ed. New York, Springer-Verlag. 1988.

[KOHO 88b] Kohonen T. **Learning vector quantization.** *Neural Networks*, N°1, 1988.

[KOHO 90] Kohonen, T. **The self-organizing map.** *Proc. IEEE*, Vol. 78, N° 9, pp. 1464-1480, Sept 1990.

[KOHO 93] Kohonen, T. **Things you haven't heard about the self organizing map.** *on Proc. IEEE ICNN*, Vol. 2, pp. 1147-1156, San Francisco, CA., March 1993.

[KOON 76] Koontz, W. L. G., Narendra, P. M. Et Kokunaga, K. **A graph theoretic approach to nonparametric cluster analysis.** *IEEE Trans. Comp.*, Vol. C-25, N° 9, pp. 936-944, 1976.

[KOSK 92] Kosko B. **Neural Networks and Fuzzy Systems, a Dynamical Systems Approach to Machine Intelligence.** Prentice Hall International, 1992.

[KRAA 92] Kraaijveld, M. A., Mao, J., Jain, A. K. **A Non-Linear Projection Method Based on Kohonen's Topology Preserving Maps** *In Proc. of Int. Conf. on Pattern Recognition*, Vol. 2, pp. 41-45, The Hagues, Netherlands 1992.

[KRAM 91] Kramer, M. A. **Nonlinear Principal Component Analysis Using Autoassociative Neural Networks.** *AICHE Journal*, Vol. 37, N° 2, pp. 233-243, February 1991.

[KRUS 77] Krusal, J. B. **Multidimensionnal scaling and other methods for discovering structures**. *Statistical methods for Digital Computers*, Wiley , New York, Vol.3, pp. 296-339, 1977.

[KULL 59] Kullback S. **Information Theory and Statistics**, J. Wiley Eds., New York, 1959.

[LAGA 83] Lagarde, J. **Initiation à l'analyse de données.** Dunod, 1983.

[LANE 97] Lane, G. N. et Williams, W. T. **A general theory of classificatory sorting strategies**. *Hierarchical Systems Comuter J.*, Vol. 9, pp. 973-980, 1967.

[LECU 87] Le Cun, Y. **Modèles connexionistes de l'apprentissage.** Thèse de doctorat, Paris, 1987.

[LECU 89] Le Cun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. et Jackel, L. D. **Backpropagation applied to handwritten zip code recognition.** *Neural computation*, Vol. 1, N° 4, pp. 541-551, 1989.

[LERM 91] Lerman, I. C. et Ghazzali **What do we retain from a classification tree? An experiment in image coding**. *Symbolic-Numeric Data Analysis 1 Learning*, INRIA, pp. 27-42, 1991.

[LIPP 87] Lippman, R. P. **An introduction to computing with neural nets**. *IEEE ASSP Magazine*, Vol. 4, pp. 4-22, April 1987.

[LIPP 89] Lippman, R. P. **Pattern classification using neural networks.** *In IEEE Comm. Magazine*,

Vol.11, pp. 47-64, November 1989.

[LOFT 65] Loftsgaarden, D. O., et Quesenberry, C. P., **A non parametric estimate of a multivariate density function**, *Ann. Math. Stat.*, Vol. 36, pp 1049-1051, 1965.

[LUKA 79] Lukasova, A. **Hierarchical agglomerative procedure**. *Pattern Recognition*, Vol. 11, pp. 365-381, 1979.

[MAKO 77] Makov, U. E. et Smith, F. M. **A quasi-Bayes unsupervised learning procedure for priors**. *IEEE. Trans. Info. Theory*, Vol. IT-24, N° 26, pp. 761-764, 1977.

[MAO 93] Mao, J. et Jain, A. K. **Discriminant analysis neural networks.** *In Proc. of IEEE Intl. Conf. on Neural Networks*, Vol. 1, pp. 300-305, San Francisco, CA, March 1993.

[MAO 94] Mao, J. **Design and Analysis of Neural Networks for Pattern Recognition.** Ph D of Michigan State University, 1994.

[MCCL 86] Rumelhart, D. E., et McClelland, J. L., **Parallel Distributed processing**, Vol. I, Boston : MIT press, 1986.

[MCCU 43] McCulloch, W. W., et Pitts, W. **a logical calculus of the ideas imminent in nervous activity.** *Bulletin of mathematical biophysics*. Vol. 5, pp.115-133, 1943.

[MELT 92] Melton, M. S. et all. **The Tin MANN VLSI chip.** *IEEE Trans. on Neural Networks*, Vol. 3, N°. 3, pp. 375-384, May 1992

[MILL 90] Miller, W. T. Glanz, F. H. 1 Kraft, L. G. **CMAC: an associative neural networkalternative to backpropagation.** *In Proc. of IEEE*, Vol. 78, N° 10, pp. 1561-1567, October 1990.

[MIZO 75] Mizoguchi, R. et Shiruma, M. **An approach to Unsupervised Learning Classification** *IEEE. Trans. Comput.*, Vol. C-24, N° 10, pp. 979-983, 1975.

[MOOD 89] Moody, J. et Darken, C. J. **Fast Learning in Networks of Locally-Tuned Processing Units.** Neural Computation 1. pp. 281-294, 1989.

[MURT 66] Murty, V. K. **Non parametric estimation of multivariate densities with application.** Multivariate Analysis, Academic Press, New york, pp. 43-56, 1966.

[PAL 93] Pal, N. R., Bezdek, J. et Tsao, E. C.-K. **Generalized Clustering Networks and Kohonen's Self-Organizing Scheme.** *IEEE Trans. on Neural Networks*, Vol. 4, N° 4, pp. 549-557, July 1993.

[PARK 85] Parker, D. **Learning Logic.** Technical Report T-R 87, Center for Computational Research in Economics and Management Science, MIT, Cambridge, 1985.

[PARZ 62] Parzen, E. **On Estimation of a Probability Density Function and Mode.** *Ann. Math. Stat.* , Vol. 33, pp. 1065-1076, 1962.

[POST 81] Postaire, J.-G. et Vasseur, C. **An approximate Solution to Normal Mixture Identification with application to Unsupervised Pattern Classification.** *IEEE. Trans. Anal. Machine Intell.*, Vol. PAMI-3, N°2, pp. 163-179, 1981.

[POST 82a] Postaire, J.-G. et Vasseur, C. **A Fast Algorithm for Non parametric Probability Density Function.** *IEEE. Trans. Pattern Anal. Machine Intell.*, Vol. PAMI-4, N°6, pp. 663-666, 1982.

[POST 82b] Postaire, J.-G. **Optimisation du Processus de Classification Automatique par Analyse de Convexité des Fonctions de Densité.** *Thèse d'Etat*, Université de Lille, 1981.

[POST 82c] Postaire, J.-G. **Fonctions convexes et Optimisation du processus de Classification Automatique : Identification des mélanges gaussiens par estimation de la convexité des fonctions de densité multivariables.** *RAIRO-automatique*, AFCET, Vol. 16, N°4, pp. 357-379, 1982.

[POST 87] Postaire, J. G. **De l'image à la décision. Analyse des images numériques et théorie de la décision**. *Editions Dunod informatique*, 1987.

[POST 93] Postaire, J.-G., Zhang R. D. et Botte-Lecocq, C. **Cluster analysis by binary morphology.** *IEEE. Trans. Pattern Anal. Machine. Intell.*, Vol. PAMI-15, N°2, pp. 170-180, 1993.

[RITT 92] Ritter, H., Martinez, T. et Schulten, K. **Neural Computation and self-organising maps, an introduction.** Comput. and Neural system series, Addison Wesley Company, 1992.

[ROSE 56] Rosenblatt, A. **Remarks on some non parametric estimates of density function** *Ann. Math. Stat.*, Vol. 27, pp. 232-237, 1956.

[RUHM 85a] Ruhmelhart, D. E. Hinton, G. E. Et Williams, R. J. **Learnig internal representation by error propagation parallel distributed processing.** *Explorations in the micro structures of cognition*, MIT Press, Cambridge, Mass, Vol. 1, pp. 318-362, 1985.

[RUHM 85b] Ruhmelhart, D. E. et Zipser, D. **Feature discovery by competitive learning.** *Cognitive science* Vol. 9, pp. 75-112. 1985.

[SAMM 69] Sammon, J.W. J. **A nonlinear mapping for data structure analysis.** *IEEE Trans. Comput., Vol.* C-18, pp. 401-409, 1969.

[SAPO 90] Saporta, G. **Probabilités, Analyse des données et Statique.** Eds. Technip, 1990.

[SBIH 94] Sbihi, A. **Extraction des modes des fonctions de densité de probabilité multivariables par analyse statistique et morphologique. Application a la classification automatique des données multidimensionnelles.** Thèse d'Etat, Univ. Ibn Tofail, Kenitra, Maroc, 1995.

[SCHR 76] Schroeder, A. **Analyse d'un mélange de distributions de probabilité de même type.** *Rev. Statist. App.*, Vol. 24, N°1, pp. 39-62, 1976.

[SEBE 84] Seber, G. A. F. **Multivariate Observations.** *Wiley*, New York, 1971.

[SIED 88a] Siedlecki, W., Siedlecka K. et Slansky J. **An overview of mapping techniques for exploratory analysis.** *Pattern Recognition*, Vol. 21, N° 5, pp. 411-429, 1988.

[SIED 88b] Siedlecki, W., Siedlecka K. et Slansky J. **Mapping techniques for exploratory pattern analysis.** *Pattern Recognition and Artificial Intelligence*, E. S. Geselma and L. N. Kanal Eds. , 1988.

[SNYD 91] Snyder, W., Nissman, D., Van den Bout, D. et Bilbro, G. **Kohonen Networks and Clustering : Comparative Perfomance in Color Clustering**, *In Advances in Neural Information Processing Systems 3*, Mogan Kauffman Publishers, 1991.

[SPEC 90] Specht, D. F. **Probabilistic neural networks and the polynomial adaline as complementary techniques for classification.** *IEEE Trans. Neural Networks*, Vol., N°1, pp. 111-121, March 1990.

[TOU 74] Tou, J. T et Gonzalez, R. G. **Pattern Recognition Principles** , Addison-Wesley Publishing Company , 1974.

[TOUZ 89] Touzani, A. et Postaire, J.-G. **Clustering by Mode Boundary Detection.** *Pattern Recognition Letters*, Vol. 9, pp. 1-12, 1989.

[ULTS 90] Ultsch, A. Siemon, H. P. **Kohonen's Self Organizing Feature Maps For Exploratory Data Analysis**, *Proc. Int. Neural Networks*, Kluwer Academic Press, pp. 305-308, Paris 1990.

[ULTS 92] Ultsch, A. **Self Organizing Neural Networks for Knowledge Acquisition**, *Proc ECAI*, pp. 208-210, Wien 1992.

[VASS 80] Vasseur, C., et Postaire, J.G. **A Convexity Testing Method for Cluster Analysis.** *IEEE. Trans Sys. Man. et Cybern.*, Vol. SMC-10, N°3, pp. 145-149, 1980

[VILL 94] Villmann, T. Der, R. Herrmann, M. et Martinetz, T. **Topology Preservation in Self-Organizing Feature Maps : Exact Definition and Measurement.** *in Proc. IEEE Iint. Conf. on Neural Networks*, Vol. 2, Orlando, Floride, 1996.

[WAI 92] Wai-Chi, F., Sheu, B. J., Chen, O. T.-C. et Choi, J. **A VLSI neural processor for image data compression using self-organisation networks.** *IEEE Trans. on Neural Networks*, Vol. 3, N°. 3, pp. 506-518, May 1992.

[WASS 89] Wasserman, P. D. **Neural computing theory and practice.** *Editions Van Nostrand Reinhold*, New York 1989.

[WATA 67] Watanabe, S. **Evaluation and selection of variables in pattern recognition.** *Computer and Information Sciences*, Vol. II, J. T. Tou eds., pp. 91-122, Academic Press, New York, 1967.

[WEBB 90] Webb, A. R. et Lowe, D. **The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis.** *Neural Networks*, Vol. 3, pp. 367-375, 1990.

[WERB 91] Werbos, P. J. **Links between artificial neural networks (ann) and statistical pattern recognition.** *Artificial Neural Networks and Pattern recognition : Old and New Connections*, I. Sethi and A. K. Jain, editors, pp. 11-31. Elsevier, Amsterdam, 1991.

[XU 93] Xu, L., Krzyzak, A. et Oja, E. **Rival Penalized Competitive Learning for Clustering Analysis, RBF Net, and Curve Detection**, *IEEE Trans. on Neural Networks*, Vol.4, N° 4, pp. 636-649, July 1993.

[YAIR 92] Yair, E., Zeger, K. et Gersho, A. **Competitive learning and soft competition for vector quantizer design.** *IEEE trans SP*, Vol 40, N°2, pp. 294-309, 1992.

REFERENCES LIEES AU TRAVAIL

1- S. DELSERT, D. HAMAD, M. DAOUDI and J.-G. POSTAIRE

"Application of Neural Networks To Gradient Search Techniques in Cluster Analysis".

Inter. Conf. on Neural Networks and Genetic Algorithms, Proceeding Vol. I, p.p. 154-159, Innsbruck, AUSTRIA, April 13-16, 1993.

2- S. DELSERT, D. HAMAD, M. DAOUDI and J.-G. POSTAIRE

"Competitive Learning Neural Networks Applied to Multivariate Data Set Reduction".

IEEE Inter. Conf. on Systems, Man and Cybernetics (SMC), Proceeding Vol. IV, p.p. 496-500, Le Touquet, FRANCE, October 17-20, 1993.

3- S. DELSERT, D. HAMAD

"Application des Cartes Topologiques de Kohonen à la Classification Interactive Non Supervisée".

Secondes Rencontres de la Société Francophone de Classification, Tours, 12-13 Sept. 1994.

4- S. DELSERT, D. HAMAD

"Nonlinear Mapping Procedures for Unsupervised Pattern Classification".

International Conference on Engineering Applications of Neural Networks EANN '95, Proceeding Vol 1, p.p. 457-461, Otaniemi, Espoo, Finlande, Août 1995.

5- M. BETROUNI, S. DELSERT, D. HAMAD

"Interactive Pattern Classification by Means of Artificial Neural Networks".

IEEE Inter. Conf. on Systems, Man and Cybernetics (SMC), pp. 3275-3279, Vancouver, British Columbia, Canada, October 22-25 1995.

6- D. HAMAD, S. DELSERT

"Exploratory Data Analysis by Means of Artificial Neural Networks".

IEEE Inter. Conf. on Systems, Man and Cybernetics (SMC), CESA 96 Lille, France, Juillet 1996 (Accepté).