REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MENTOURI CONSTANTINE
FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'ELECTRONIQUE

N° d'ordre :
Série :

Mémoire de Magistère

Présenté Par
Bounneche Meriem Dorsaf

Option
Contrôle

Thème

REDUCTION DE DONNEES POUR LE TRAITEMENT D'IMAGES

Président	Mme. N. Mansouri	Professeur	Université Mentouri Constantine
Rapporteur	Mme F. Hachouf	Maître de Conférences	Université Mentouri Constantine
Examinateur	M. Khamadja	Professeur	Université Mentouri Constantine
Examinateur	Mr M. Khamadja	Professeur	Université Mentouri Constantine

Année 2009

REMERCIEMENTS

Je tiens en premier lieu à remercier Mme F.Hachouf, Maître de Conférences à l'université Mentouri de Constantine, d'avoir proposée et encadrée ce sujet. Je lui exprime ma profonde gratitude pour m'avoir fait profiter de ses connaissances, mais aussi de ses méthodes de travail, et surtout de sa rigueur scientifique. Grâce à elle, j'ai découvert un domaine de recherche qui aujourd'hui me passionne.

Je remercie également Mme N.Mensouri, Professeur à l'université Mentouri de Constantine, d'avoir accepté la présidence de ce jury.

J'adresse mes plus vifs remerciements à Mr K.Belarbi, Professeur à l'université Mentouri de Constantine et à Mr M.Khamadja, Professeur à l'université Mentouri de Constantine, pour avoir acceptés d'être les rapporteurs de ce mémoire, et leur participation à ce jury.

Enfin, je souhaite témoigner toute mon amitié et mes remerciements à l'ensemble de mes collègues du laboratoire pour leur soutien technique, logistique, moral à cette thèse.

SOMMAIRE

L'évolution que connaissent l'acquisition et le stockage d'images de très haute qualité et donc à très grandes dimensions, a rendu leur exploitation complexe; ce qui redirige les recherches vers des moyens d'en extraire les informations utiles.

Plusieurs manières de réduction de dimensions existent dans la littérature. Elles visent à extraire et à utiliser uniquement la structure intrinsèque de ces données qui contient juste l'information utile pour les traitements postérieurs. Il s'agira, dans le cas présent, d'une classification.

Cette dernière nécessite la connaissance préalable du nombre de classes. nous nous proposons d'utiliser les méthodes issues de l'analyse discriminatoire pour réduire les dimensions des données et pour optimiser ce paramètre de classification.

Dans ce cadre nous nous proposons d'étudier différentes méthodes susceptibles d'être utilisées pour réduire et exploiter ces images avant de pouvoir les traiter. On s'intéressera principalement à l'analyse en composantes principales (PCA), l'analyse en composantes indépendantes (ICA), la décomposition en valeurs singulières (SVD) qui sont considérées comme des techniques non supervisées. Puis, nous exposerons quelques techniques supervisées comme l'analyse discriminante linéaire (LDA), l'algorithme de classification à erreur minimales (MCE) et les séparateurs à large marge (SVM).

Ce manuscrit est organisé en quatre chapitres :

Le premier chapitre va aborder le thème de la classification comme étape pertinente de l'extraction des connaissances contenues dans les données. Elle consiste à trouver celles qui ont des caractéristiques similaires et à les rassembler dans les mêmes clusters.

Le deuxième chapitre traitera des différentes méthodes de réductions de la masse de données, dans le cas non supervisé.

Dans le troisième nous présenterons les méthodes de réduction de données dans le cas supervisé.

Dans le chapitre quatre, nous présenterons les différentes combinaisons d'algorithmes implémentés. Nous illustrerons les méthodes étudiées pour la réduction,de données et l'optimisation du nombre de classes. Les différents résultats obtenus par l'application de ces algorithmes, sur plusieurs types d'images seront comparés.

Nous achèverons ce travail par une conclusion et des perspectives à notre travail.

Introduction

La qualité de l'interprétation d'une image couleur dépend fortement de celle de la segmentation, Parmi les méthodes de celle-ci, certaines s'intéressent à chercher les régions connexes homogènes dans l'image. C'est ce que l'on appelle un traitement de bas niveau qui sert, entre autres, de base à l'identification des classes présentes dans une image. Et par conséquent à la classification qui attribuera chaque pixel de l'image à l'une des classes identifiées.

Cette notion survient constamment dans la vie courante car il est souhaitable de réunir les éléments d'un ensemble hétérogène, en un nombre limité de classes les plus homogènes possibles suivant un critère connu (que ce soit la similarité entre les objet,les mesures probabiliste ou encore la densité des classes) . Son application résoudra plusieurs problèmes en reconnaissance des formes, imagerie, segmentation d'images couleur, data mining…

Le procédé de la classification se décrit en deux étapes, une première qui est celle de l'apprentissage : où il s'agira de créer des règles pronostiquant l'appartenance d'une donnée à une certaine classe ; la seconde sera celle de l'application de ces mêmes règles afin de designer la classe appropriée de chaque donnée.

Du moment où l'on ne dispose d'aucune information à priori sur les données à traiter, on parlera de classification non supervisée ; dans le cas contraire ça sera de la classification supervisée. Le travail élaboré dans cette étude s'inscrit dans le cadre de la classification non supervisée, adhérant aux recherches d'ensemble homogène au sein d'un mélange multidimensionnel où le nombre de groupes est inconnu.

Les résultats de classification obtenus dépendent fortement du nombre de classes fixé. Il est donc impératif de choisir le nombre exact de classes pour prétendre à une bonne qualité de classification.

Plusieurs algorithmes ont été proposés dans ce sens [1, 2, 3, 4, 5]. Certains nécessitent un seuillage de l'histogramme ou le réglage de certains paramètres, d'autres sont limités aux cas où les différentes classes correspondent à des nuages séparables dans l'espace de mesure, tandis que d'autres ne prennent pas en compte les relations géométriques des pixels dans l'image.

1. La classification non supervisée

La classification non supervisée ou classification automatique - *clustering* - est une étape importante de l'analyse de données ; elle consiste en l'identification des groupes d'objets ou d'individus similaires – *clusters* – à partir d'un ensemble de données sans en connaître au préalable la structure. Elle n'est pas à confondre avec la classification supervisée qui consiste à déterminer les règles qui permettent de séparer un ensemble d'individus en classes connues à priori [6]. Ce chapitre survolera rapidement quelques concepts et les notions de base de ce domaine.

Nous commençons par rappeler quelques concepts et définitions avant de présenter quelques approches utilisées en classification automatique. La classification sous contrainte est ensuite présentée comme un moyen d'introduire des connaissances à priori aux algorithmes de classification automatique.

1.1. Concepts et définitions

1.1.1. Définition de la classification

Le concept de classification et la notion de partition d'un ensemble fini sont étroitement liés. Plusieurs techniques ont été définies se distinguant ou par le type de résultats obtenus ou alors par la méthode de regroupement qui constitue les classes (celles-ci se basent sur le calcul des distances). Dans le premier cas selon qu'il y ait chevauchement entre les classes ou pas, on parlera de *classification dure, douce ou floue* ; selon qu'il y ait objet classé ou non on parlera de *classification partielle*. Ces mêmes résultats peuvent par ailleurs être représentés sous forme de structure plate ou d'une hiérarchie de classes.

Définition-1- Partition d'un ensemble fini: *Étant donné un ensemble fini d'objets* Ω*, on appelle partition de* Ω *toute famille de parties non vides de* Ω *disjointes dont l'union forme l'ensemble. Ainsi, si C est une partition de* Ω*, alors :*

$$C = \left\{ C_i \in P(\Omega) \,/\, \bigcap_{i=1}^{K} C_i = \{\Phi\}, \bigcup_{i=1}^{K} C_i = \Omega \right\} \qquad (1.1)$$

Cette première définition correspond à l'algorithme de *classification dure,* c'est la manière la plus simple de représenter les résultats d'une classification ou chaque objet doit appartenir à une classe unique, c'est ce que l'on appelle *méthodes de partitionnement*. La Figure 1.1. représente un exemple de cette méthode.

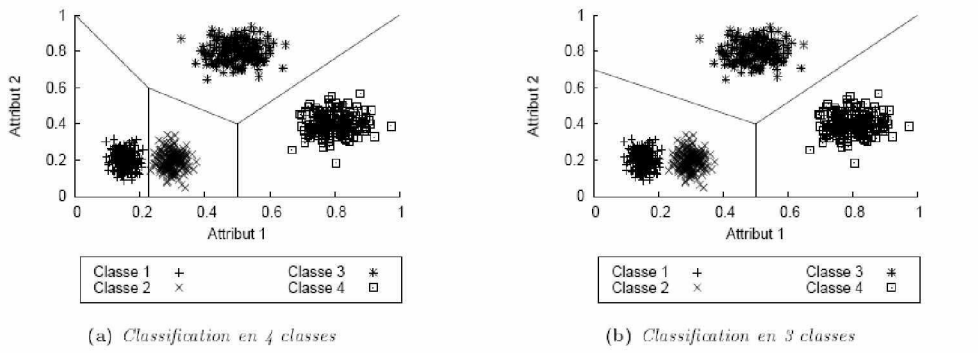


Fig1.1. Classification par partitionnement

Certes ce type de classification a l'avantage d'être aisément interprétable. Mais il est parfois inadéquat quand on a besoin de donner plus de flexibilité à la notion de classe afin de ne pas altérer la qualité de l'information. Dans le cas où des objets sont trop différents des autres, il est préférable de ne pas les classer. Ainsi il s'agira de *classification dure partielle*.

Si la contrainte de disjonction est relâchée, un objet peut appartenir à différentes classes, on parlera de *classification douce*. Mais aussi de *classification douce partielle* s'il existe des objets qui n'appartiennent à aucune classe.

Enfin, dans le cas où chaque objet appartient à toutes les classes avec un degré d'appartenance, on va vers la notion de *classification floue*. Elle aboutit souvent vers une classification dure après une defuzzification des résultats par maximum d'appartenance par exemple, ou encore vers une classification douce ou les classes sont ordonnées par ordre décroissant du degré d'appartenance. Chaque objet est alors affecté aux classes dont il vérifie les conditions.

1.1.2. Groupe d'objets similaires

La notion de similarité est un élément essentiel de la classification automatique et le biais introduit par la mesure de similarité permet de former des groupes. Ce concept est dual à celui de la dissimilarité où deux individus sont autant plus similaires qu'ils sont proches au sens d'une mesure de dissimilarité.

Définition -2- Mesure de dissimilarité: *On appelle indice ou mesure de dissimilarité sur un ensemble* Ω *, une application* $d : \Omega \times \Omega \rightarrow R^+$ *qui vérifie les propriétés suivantes pour tout couple* (x, y) $\in \Omega x \Omega$:

$$\begin{aligned} d(x,y) &= d(y,x) && \textit{(symétrie)} \\ d(x,y) = 0 \quad &\Leftrightarrow \quad x = y && \textit{(séparabilité)} \end{aligned} \tag{1.2}$$

Définition -3- Métrique: *On appelle métrique sur un ensemble* Ω *, une application*

$d: \Omega x \Omega \rightarrow R^+$ *qui vérifie les propriétés suivantes pour tout* $(x, y, z) \in \Omega \times \Omega \times \Omega$*:*

$$\begin{aligned} d(x,y) &= d(y,x) && \textit{(symétrie)} \\ d(x,y) = 0 \quad &\Leftrightarrow \quad x = y && \textit{(séparabilité)} \\ d(x,y) &\leq d(x,z) + d(z,y) && \textit{(inégalité triangulaire)} \end{aligned} \tag{1. 3}$$

Définition -4- Ultramétrique: *On appelle ultramétrique sur un ensemble, une application*

$d : \Omega \times \Omega \rightarrow \boldsymbol{R}^+$ *qui vérifie les propriétés suivantes pour tout couple* $x, y, z) \in \Omega \times \Omega \times \Omega$*:*

$$\begin{aligned} d(x,y) &= d(y,x) && \textit{(symétrie)} \\ d(x,y) = 0 \quad &\Leftrightarrow \quad x = y && \textit{(séparabilité)} \\ d(x,y) &\leq \max\{d(x,z), d(z,y)\} && \textit{(inégalité ultramértique)} \end{aligned} \tag{1. 4}$$

L'homogénéité des individus regroupés au sein d'un groupe est souvent évaluée à l'aide d'un critère statistique appelée **variance** dont la définition est rappelée ci-dessous.

Définition -5- Variance: *On définit la variance V (Ci) d'un groupe d'objets Ci ainsi :*

$$V(C_i) = \frac{1}{N_i} \sum_{x_j \in C_i} d^2(x_j - \mu_i) \quad (1.5)$$

où Ni et μi sont respectivement le nombre d'objets et le centroïde du groupe Ci.

Dans la classification automatique, on peut également distinguer la variance intra-classe V_{intra}, que l'on souhaite minimiser, de la variance inter-classe V_{inter}, que l'on cherche à maximiser :

$$V_{intra} = \frac{1}{N} \sum_{C_i \in C} N_i \times V(C_i) \quad (1.6)$$

$$V_{inter} = \frac{1}{N} \sum_{C_i \in C} N_i \times (\mu_i - \mu)^2 \quad (1.7)$$

où Ni et μi sont respectivement le nombre d'objets et le centroïde du groupe Ci, et de manière analogue, N et μ désignent respectivement le nombre d'objets et le centroïde de Ω. La première évalue l'homogénéïté moyenne des groupes d'une partition et la seconde permet de quantifier la différence entre les groupes. Le théorème de König-Huyghens permet de relier la variance intra-classe et inter-classe à la variance totale $Vtotale = V(\Omega)$:

$$V_{totale} = V_{intra} + V_{inter} \quad (1.8)$$

1.2. Les méthodes de classification classiques

1.2.1. Méthodes hiérarchiques

Généralement, les résultats d'une classification sont sous la forme de structure plate, mais elle peuvent aussi être sous forme de hiérarchie de classes : où une classe peut être divisée en sous-classes, l'ensemble alors forme une hiérarchie représentée par un arbre. On définit une classification hiérarchique comme une suite ordonnée de partitions emboîtées dont le premier terme est la partition la plus fine qui ne contient que des singletons, et le dernier terme est la partition la plus grossière qui ne comporte qu'une seule partie. Un objet appartient donc a une seule classe et à sa classe mère ; on distingue ici deux types d'approche :

a. Méthodes descendantes *-divisive-*

Elles considèrent l'ensemble des observations et procèdent par division successive jusqu'à obtenir une partition formée de singletons. Nous ne détaillerons pas d'avantage ces méthodes qui sont trop coûteuses pour être utilisées sur les volumes de données manipulés aujourd'hui. En effet, la division d'une partie à N éléments nécessite l'évaluation des $(2^{N-1}- 1)$ divisions possibles.

b. Méthodes ascendantes *-agglomerative-*

Partant d'un grand nombre de classes, contrairement à la première, elles fusionnent progressivement les classe similaires entres elles jusqu'à obtention de la partition la plus grossière. On obtient ainsi un arbre binaire dont la racine correspond à la partition ne comportant qu'une seule partie et dont les feuilles s'identifient aux différents singletons. Les différents noeuds intermédiaires correspondent à la fusion de deux parties.

La figure1.2. illustre une Classification Ascendante Hiérarchique (CAH) des données représentées dans la figure. 1.1.La racine de l'arbre correspond à l'ensemble des données à classifier. Les figure1.3. et 1.4. illustrent le résultat d'une CAH est fortement conditionné par le choix du critère d'agrégation.

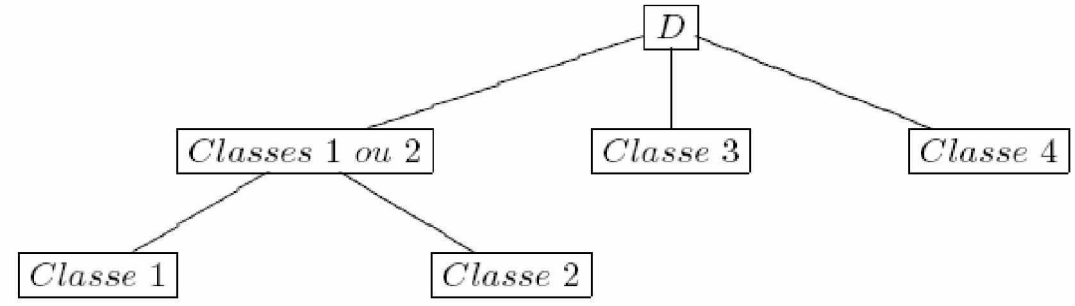


Fig. 1.2- Classification ascendante hiérarchique

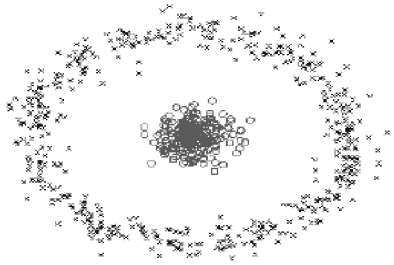


Fig.1.3. Cas d'une couronne

Dans le cas d'une couronne, une CAH utilisant l'indice du saut minimum identifiera parfaitement les deux groupes, en revanche l'utilisation de la distance entre les centroïdes conduira à une classification sans réel intérêt.

L'indice du saut minimum est défini comme la distance minimale qui sépare deux éléments issus de groupes différents.

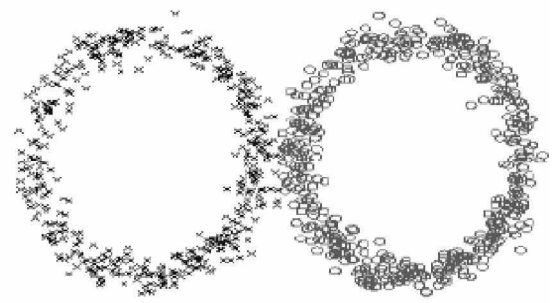


Fig.1.4. Cas de deux anneaux

Lorsque les groupes ne sont pas suffisamment séparés, l'utilisation de l'indice du saut minimum est à proscrire car elle conduirait à ce qu'on appelle "effet de chaîne" : les groupes sont fusionnés de proche en proche et la CAH se révèle incapable de mettre en exergue les deux anneaux. La distance entre les centroïdes ou le critère de Ward (qui revient faire fusionner successivement deux classe en minimisant la distance de Ward entre elles) conduisent dans ce cas à des classifications plus pertinentes.

Distance de Ward est $D(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} d^2(g_i, g_j)$ où n_i et g_i sont respectivement l'effectif et le centre de la classe C_i.

1.2.2. Nuées dynamiques

La représentation des distinctes classes obtenues par les méthodes hiérarchiques dont on parlait précédemment est très claire, contrairement à celle des technique dites *nuées dynamique* qui est un prototype que l'on appelle centre. L'affectation de chaque objet à une classe se fait par rapport au centre dont il est le plus proche (Fig.1.5.). La partition obtenue est alors représentée implicitement. Nous commençons par introduire l'algorithme des K-moyennes avant d'en présenter une extension aux classifications floues.

Fig.1.5. Exemple de classification par l'algorithme des K-moyennes

chaque groupe est représenté par un prototype, encore appelé centre, et chaque objet est affecté au groupe dont il est le plus proche.

a. K-moyennes

La technique des K-moyenne [7] cherche un partitionnement défini en fonction des centres de classes en minimisant l'erreur quadratique entre les centre de classe et les objet de celle-ci.

Définition -6- L'erreur quadratique d'une classification non supervisée est définie par :

$$EQ_{km}(c,C) = \sum_{1\le k\le K}\sum_{o\in C_k} d^2(o,c_k) \qquad (1.9)$$

avec « c » l'ensemble des centres de classes et « C » est l'ensemble des classes.

L'algorithme des K-moyennes choisit arbitrairement des centres initiaux et améliore la classification obtenue de manière itérative en alternant les deux étapes suivantes jusqu'à stabilisation. Chacune des étapes consiste à fixer l'un des paramètres de la fonction d'évaluation puis à estimer la valeur optimale de l'autre :

- *Redéfinition des centres :* le centre c_k est défini comme l'isobarycentres des objet de la k-iéme classe ;
- *Affectation des objet aux classes:* un objet est affecté à la classe dont il est le plus proche.

Le critère optimisé par cet algorithme est défini par l'équation (1.9).

Malgré le fait qu'il soit beaucoup plus rapide que la CAH, cet algorithme est très instable et tend à converger vers des minima locaux. Le choix de la meilleure solution se fait généralement après multiples exécutions de l'algorithme sans toutefois avoir garanti l'optimalité globale de la partition retenue. Cependant, des variantes de cet algorithme initial ont été proposées pour essayer de palier à

ces problèmes. L'exemple de l'algorithme des K-moyennes adaptatifs [7] qui consiste à présenter les données une à une, plusieurs fois dans un ordre aléatoire. Le centre le plus proche de la donnée présentée est déplacé vers celle-ci de la manière suivante :

$$c'_{k,i} = c_{k,i} + \eta(o_i - c_{k,i}) \tag{1.10}$$

où $\eta \in [0;1]$ est le pas d'apprentissage qui peut être constant ou varier au cour de l'apprentissage.

Cet algorithme peut être implémenté comme un réseau de neurones. On considérera le réseau constitué d'une couche d'entrée où le nombre de neurone d'entrée « n » est celui des attributs (une neurone pour chaque attribut) ainsi qu'une couche de sortie dont le nombre de neurones « k» est celui du nombre de classes (un par classe). Chaque neurone de la couche d'entrée est connecté à tout ceux de la couche de sortie, chacune de ces connexions est pondérée par un poids. L'activation du neurone de sortie est la somme des activations des neurones d'entrée, pondérée par les poids des connexions. Le neurone le plus activé est celui qui correspond à la classe de l'objet.

L'ensemble des données est présenté plusieurs fois au réseau de neurone. La mise à jour des poids est faite après chaque présentation d'un objet après une première initialisation arbitraire. Les poids des connexions des neurones les plus activés par un objet sont modifiés comme suit :

$$w'_{v,j} = \frac{w''_{v,j}}{\left\| w''_v \right\|}, \text{avec } w''_{v,j} = w_{v,j} + \eta(o_j - w_{v,j}) \tag{1.11}$$

Chaque neurone de la couche de sortie représente un centre et les poids représentent les attributs de ce même centre. Le degré d'activation par un objet d'un neurone de sortie est une mesure de similarité entre l'objet et le centre de sa classe.

Ces approches nécessitent de connaître à priori le nombre de centres qu'on ignore souvent en pratique. Il est donc nécessaire d'exécuter l'algorithme pour différentes valeurs de ce paramètre.

Malgré les multiples exécutions requises par l'utilisation de la méthode des K-moyennes, cette approche conserve l'avantage sur la CAH lorsque le nombre de centres K reste faible devant le nombre d'objets.

b. K-moyennes floues

Definition-7- L'erreur quadratique d'une classification non supervisée floue est définie par :

$$EQ_{K-moyenne-floue} = \sum_{\alpha \in \Omega} \sum_{i=1}^{K} (\mu_i(x))^f \left\| x - c_i \right\|^2 \qquad (1.12)$$

Où *K, μi(x)* et c_i sont respectivement le nombre de centres, le degré d'appartenance de l'objet x au groupe Ci, le centre du groupe *C*i. Le paramètre f > 1 permet d'ajuster le niveau d'importance accordé aux degrés d'appartenance.

Une version de l'algorithme k-means ; nommée *Fuzzy-C-means*, a été définie [8]et[9].la fonction à minimiser est alors la version floue de l'erreur définie au dessus.

Comme pour k-means chaque itération de l'algorithme comporte deux étapes :

- *Redéfinition des centres :* le centre c_k est défini comme le barycentre de tous les objet pondéré par leur degré d'appartenance à la $k^{ème}$ classe.
- *Calcul du degré d'appartenance des objets aux classes :* il est recalculé, en fonction du paramètre f, par :

$$\mu_k(o) = \frac{1}{\sum_{k'=1}^{K} \left(\frac{d(o,c_k)}{d(o,c_{k'})} \right)^{\frac{2}{f-1}}} \qquad (1.13)$$

L'algorithme des K-moyennes présenté ci-dessus conduit à une partition dure et Dunn en a proposé une extension qui conduit à une partition floue. Celle-ci minimise la fonction de coût de l'équation (1.12).

1.2.3. Modèles de mélange

a. Principe

Vu que la densité de probabilité *f*(*x*) en un point *x* est inconnue, le principe d'un modèle de mélange est de décomposer cette densité en une somme de *c* composantes $f(\bullet|\Theta_k)$ correspondant aux *c* classes dont on va estimer les paramètres $\Theta_k (k = 1,...,c)$ à partir d'un échantillon χ.

χ est un ensemble de *n* réalisations x_i d'un vecteur aléatoire *X* de dimension *p* et s'écrit :

$$\chi = \{x_1,...,x_n\}$$

Ces densités de probabilités $f(\bullet|\Theta_k)$ peuvent aller du modèle le plus simple aux distributions multimodales les plus complexes. Les proportions π_k entre les différentes composantes représentent les probabilités a priori des différentes classes. En général, ces proportions sont également inconnues et l'on doit les estimer sous les contraintes :

$$\pi_k \in]0\ ;1[et \sum_{k=1}^{c} \pi_k = 1 \qquad (1.14)$$

Ainsi, on notera $\Theta = [\pi_1,...,\pi_c,\Theta_1,...,\Theta_c]^T$ le vecteur de paramètres à estimer. La densité de probabilité *f*(*x*) en un point *x* est ainsi approximée conditionnellement aux paramètres Θ par $f(x|\Theta)$:

$$f(x|\Theta) = \sum_{k=1}^{c} \pi_k f(x|\Theta_k) \qquad (1.15)$$

b. Algorithme EM

L'algorithme EM [10] [11] est un algorithme de classification probabiliste, il cherche les paramètres $\Phi = (\pi_1,...,\pi_k,\theta_1,...,\theta_k)$ d'une loi de mélange. L'algorithme EM est une technique itérative qui maximise la loi de la log-vraisemblance en présence de données incomplètes:

Définition -8- La log-vraisemblance

La log-vraisemblance d'un modèle de mélange de paramètre Φ qui a une fonction de densité $P(o,\Phi)$ est définie par :

$$L = \sum_{o\in\Omega} \log(P(o,\Phi)) \qquad (1.16)$$

Chaque itération de l'algorithme se fait en deux phases, la première est *la phase* E pour *expectation* dites espérance ou prévision qui consiste à évaluer la densité de probabilité $P(o,\Phi)$ pour chaque objet « o » en fonction des paramètres du modèle, ainsi la probabilité d'appartenance (assimilable au degré d'appartenance) de « o » à chacune des classes est définie par :

$$t_k(o) = \pi_k \times P(o,\theta_k) \Big/ \sum_{k'=1}^{K} \pi_{k'} \times P(o,\theta_{k'}) \quad (1.17)$$

La seconde, *la phase* M pour *Maximisation* estime, en fonction des probabilités d'appartenance des objets aux classes, les paramètres du modèle maximisant la log- vraisemblance des objets. La définition des paramètres θ_k dépendra de la distribution de la classe, alors que les paramètres π_k seront définis par :

$$\pi_k = \frac{1}{N} \sum_o t_k(o) \quad (1.18)$$

L'algorithme EM alterne successivement dans l'algorithme les deux phases décrites plus haut. L'une des propriétés de cet algorithme est d'améliorer la vraisemblance $L(\Theta)$ des paramètres après chaque itération ce qui permet de démontrer sa convergence.

1.3. Approche neuromimétique : les cartes auto-organisées de Kohonen

1.3.1. Sources historiques et principes

La méthode des *cartes auto-organisatrices*, ou *cartes topologiques* - *Self-Organizing Maps (SOM)* - [12] est un algorithme de classification non supervisé basé sur un réseau de neurones artificiels. Les neurones de la couche de sortie sont reliés entre eux pour former un réseau qui reste fixe durant l'apprentissage. Chaque neurone possède des coordonnées dans l'espace des données et représente un prototype d'une classe. Deux neurones reliés s'influencent l'un l'autre au cours de l'apprentissage. L'algorithme « SOM » permet de découvrir la topologie des données. Les neurones proches les uns des autre dans cette topologie (souvent rectangulaire) peuvent correspondre à la même classe.

Cet algorithme de classification a des propriétés assez intéressantes, la première est son apprentissage qui a l'avantage d'être non supervisé. Il est considéré comme une méthode de quantification vectorielle, regroupant les informations en leurs classes respectives tout en tenant compte de leur topographie dans l'espace des observations et ceux en :

- Définissant a priori une notion de voisinage entre classes.

- Faisant appartenir des observations voisines dans l'espace des données à la même classe ou à des classes voisines, après classement.

- Compressant de données multidimensionnelles tout en préservant leurs caractéristiques.

Les variantes de l'algorithme « SOM » sont multiples et leurs utilisations le sont autant, ils sont assez proches en classification de l'algorithme K-means et peuvent être utilisées pour trouver automatiquement le nombre de classes sans aucune supervision [13] [14]. Les cartes de Kohonen peuvent aussi être utilisées pour la détection de contours dans le traitement d'image. Le réseau prend, comme données d'entrée, les vecteurs de position des arêtes préalablement déterminées à l'aide d'un opérateur de détection d'arêtes et fournit en sortie une image contours. Bélanger [15] remplace une information rigide que sont les arêtes par une information plus riche que sont les gradients de l'image.

2.3.2. Description

Le procédé d'auto-organisation proposé par Kohonen cherche à transformer une information de départ de dimension généralement grande, en une information à une ou deux dimensions. Les corrélations qui sont présentes dans les données présentées à l'entrée sont reproduites en sortie du réseau. D'une manière générale, les cartes auto-organisatrices vont projeter les données initiales sur un espace discret et régulier de faible dimension (en général 1 ou 2). Les espaces utilisés sont des treillis réguliers dont chacun des noeuds est occupé par un neurone formel. La notion de voisinage entre neurones dépend directement de la structure et définit une topologie de la carte.

La conservation de la topologie et des relations de voisinage qui lient les données initiales, surtout que celles-ci sont inconnues avant l'apprentissage, donne un grand avantage à ce procédé d'auto-organisation, vue qu'il conserve ces caractéristiques dans les réponses du réseau. Ces informations utiles vont permettre l'interprétation des données initiales. En particulier, ils vont définir la notion de formes proches dans l'espace initial.

Les réseaux SOM sont constitués de deux couches (Fig. 1.6.) :

- la couche d'entrée où les données à classer sont présentées. Les états de tous les neurones de cette couche sont forcés aux valeurs des caractéristiques décrivant les formes d'entrées. Tout

individu à classer est représenté par un vecteur multidimensionnel (le vecteur d'entrée). A chaque individu est affecté un neurone qui représente le centre de la classe.

- la couche (topologique) d'adaptation ou dite encore couche de compétition est composée du treillis de neurones selon une géométrie prédéfinie. Les neurones de cette couche entrent en compétition, seuls les meilleurs gagnent.

Chaque neurone i de la couche topologique est totalement connecté aux neurones de la couche d'entrée.

Le vecteur poids $\omega_i = (\omega_{1i}, \ldots, \omega_{ni})$ de ces connexions forme le *référent* ou le prototype associé au neurone. il est de la même dimension que les formes d'entrée.

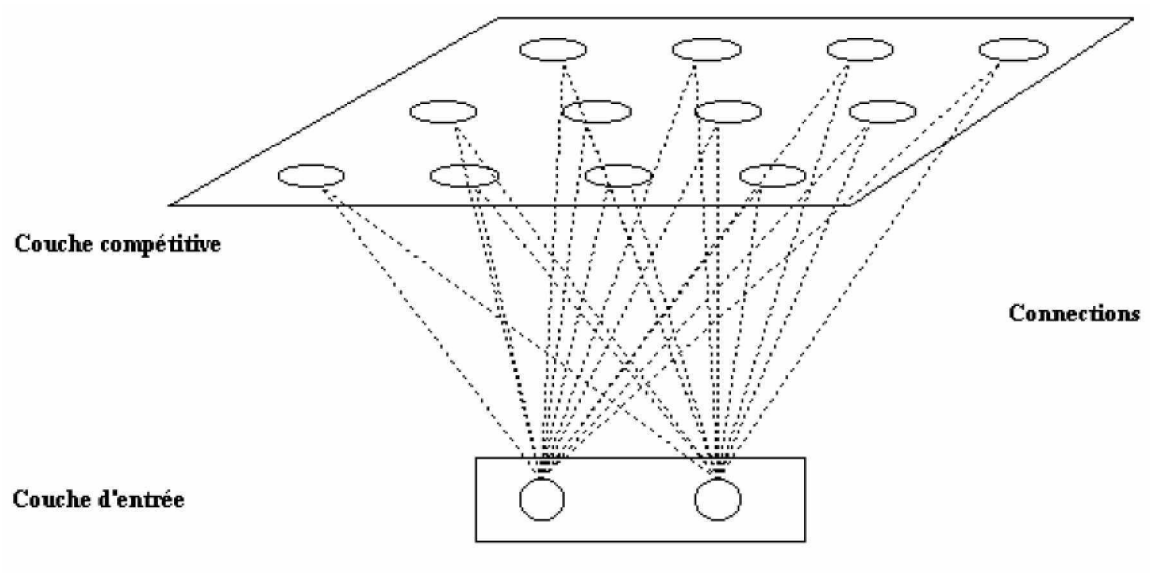


Fig. 1.6. Architecture du réseau pour l'algorithme des cartes topologiques.

Comme pour tout réseau de neurones, son utilisation n'est possible qu'après l'avoir paramétrer en lui présentant les données : c'est la phase d'apprentissage. Dans le cas des cartes de Kohonen, l'objectif de cette phase est de *déplacer* certains prototypes vers les données d'apprentissage. Pour ce faire, on présente successivement ces données à tous les neurones qui composent la carte auto-organisatrice. Celui dont le prototype est le plus proche (au sens de la distance euclidienne par exemple) de la donnée présentée est nommé *neurone gagnant* (et son prototype, le *prototype*

gagnant). À partir de ce neurone gagnant, on sélectionne un ensemble de neurones faisant partie de son *voisinage*. On rapproche linéairement les prototypes des neurones de ce voisinage ainsi que le prototype gagnant vers la donnée d'apprentissage. C'est cette notion de voisinage qui introduit les contraintes topologiques dans la géométrie définitive de la carte. Finalement, on réitère ces actions jusqu'à ce que les déplacements des prototypes soient *faibles*.

2.3.3. Algorithme d'apprentissage

L'apprentissage met en correspondance l'espace des entrées et la carte d'adaptation des poids *W* de telle manière que des exemples proches dans l'espace d'entrée soient associés au même neurone ou à des neurones proches dans la carte. La mise à jour des poids pour chaque neurone se fait par évaluation du voisinage du neurone gagnant dans la carte. L'adaptation est d'autant plus forte que les neurones sont voisins. Il est à noter que la taille du voisinage du prototype gagnant (qui détermine les prototypes candidats au rapprochement) est fonction de l'itération (plus on avance dans la phase d'apprentissage plus la taille du voisinage se réduit). De la même manière, la vitesse de rapprochement des prototypes du voisinage et du neurone gagnant est fonction de l'itération (plus on avance dans l'itération moins la vitesse est grande) et du prototype gagnant (plus le neurone gagnant est éloigné du prototype plus la vitesse est faible).

L'algorithme se déroulera comme suit :

- Initialisation aléatoire des poids *W*

à chaque itération t faire :

- Présentation d'un exemple d'apprentissage *X(t),* choisi au hasard, à l'entrée de la carte Comparaison de l'exemple à tous les vecteurs poids, le neurone gagnant *j** est celui dont le vecteur poids $W_{j^*}(t)$ est le plus proche de l'entrée *X(t) (phase de compétition :)*

$$d_N(X(t),W_{j^*}(t)) = \min_j d_N(X(t),W_j(t)) \tag{1.19}$$

d_N : distance dans l'espace d'entrée

- Évaluation du voisinage du neurone gagnant dans la carte :

$$h_{j^*}(j,t) = h(d(j,j^*),t) \tag{1.20}$$

d : distance dans la carte.

*h :*fonction voisinage.

- Mise à jour des poids pour tous les neurones de la carte, l'adaptation est d'autant plus forte que les neurones sont voisins de *j* (phase de coopération)* :

$$W_j(t+1) = W_j(t) + \Delta W_j(t)$$
$$\Delta W_j(t) = \varepsilon(t).h_{j^*}(j,t).(X(t) - W_j(t)) \quad (1.21)$$

ε :pas d'apprentissage

Les résultats de l'algorithme varieront selon le choix des paramètres d'apprentissage

- *La fonction de voisinage* dans la carte de kohonen est une fonction continue de forme gaussienne, La décroissance de la taille du voisinage s'obtient par diminution de l'écart type σ .

Plus σ est grand, plus il y aura des neurones qui se rapprocheront de *X(t),*s'il est petit, l'adaptation restera très localisée

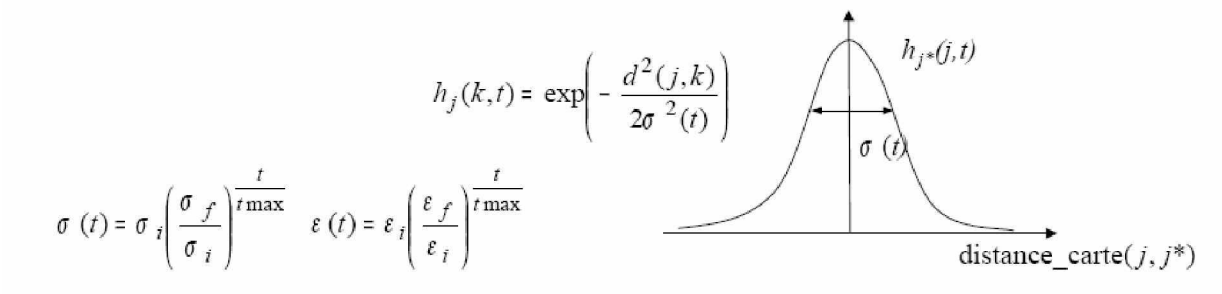


Fig.1.7. Exemple d'une fonction de voisinage.

- *Le pas de l'apprentissage* contrôle la vitesse d'apprentissage. Si ε est trop petit, le modèle ne s'adaptera pas assez aux données, si ε est trop grand, il y aura risque d'instabilité.
- *Condition de convergence*

  Choisir les valeurs σ et ε assez grandes au départ et les réduire graduellement (on choisira des paramètres à décroissance exponentielle).

  Ordonnancement rapide au départ qui s'affine ensuite

Une version *batch* de cet algorithme a été proposée: les vecteurs poids ne sont mis à jour qu'après la présentation de toutes les formes d'entrées et on remplace alors le prototype des neurones par le barycentre pondéré à l'aide de la fonction de voisinage des formes d'entrées qui les ont activés.

Introduction

Les capteurs d'images actuels fournissent des images dont les dimensions sont astronomiques et leur interprétation requiert une segmentation -dans le but d'une meilleure exploitation- qui classifiera les pixels en fonction de leurs critères pertinents. Ces données nécessitent donc une étape de prétraitement en vue de réduire leur dimensionnalité. Elles peuvent être représentées dans un espace vectoriel de dimension égale à leurs dimensions intrinsèques, suffisante à l'extraction de l'information utile, et surtout inférieure à celle de l'espace vectoriel fournit par le capteur.

L'étape de réduction de dimensionnalité fait partie intégrante du processus global de traitement des images, Plusieurs des méthodes qui le permettent ont été développées et appliquées sur les images: Analyse en composantes principales (PCA : Principal Components Analysis), la décomposition en valeurs singulières (SVD : Singular value decomposition), Analyse en composantes indépendantes (ICA : Independent Component Analysis), Poursuite de projection (PP : Projection Poursuit), Analyse discriminante linéaire (LDA: Linear Discriminant Analysis).

Ces méthodes de prétraitement à la classification peuvent être supervisées ou non supervisées, selon que les données aient déjà subi une classification antérieure et là il s'agira d'améliorer son résultat ou que celles-ci ne furent pas traitées. Nous consacrerons ce chapitre à l'introduction de quelques unes des techniques de prétraitement non supervisées.

1. Les méthodes non supervisées

Dans les cas des méthodes non supervisées, nous exploitons les données sans connaissances préalables du modèle. Nous exposerons quelques unes des ces techniques de prétraitement de données précédant l'analyse postérieure de celles-ci : l'analyse en composantes principales, la décomposition en valeur singulière ainsi que l'analyse en composantes indépendantes.

Les deux différences majeures entre la sélection des caractéristiques, et les méthodes du prétraitement non supervisé destinées à projeter les données dans un nouvel espace de dimension inférieur (et donc dans un but de réduction) sont:

- Au lieu de choisir un sous espace de caractéristiques, la projection de données crée de nouvelles dimensions définies par une fonction de toutes les autres caractéristiques.

- Puis cette même réduction ne considère pas le label des classes, mais plutôt les données point par point.

1.1. L'analyse en composantes principales

L'analyse en composantes principales fait partie des méthodes appelées décompositions orthogonales propres introduites par Lumley [16] après des investigations indépendantes de Kosambi (1943), Loève (1945), Karhunen (1946), Pougachev (1953) ainsi que Obukhov (1954).

PCA est une technique statistique largement utilisée dans la réduction de dimensions non supervisée. La base de cette réduction est que PCA prend en compte les dimensions dont les variances sont les plus importantes. Ce-ci est mathématiquement équivalent à trouver la meilleure approximation à rang minimum -dans le sens de la norme L2- des données via une SVD (Eckart et Young, 1936). Néanmoins, cette représentation par SVD -qui est a pour but la réduction du bruit- à elle seule n'est pas assez adéquate pour représenter et expliquer l'efficacité de PCA.

1.1.1. Principe

L'idée principale de la décomposition orthogonale propre est de trouver un ensemble de vecteurs de base orthonormaux et ordonnés dans un sous espace sans perte conséquente d'information où l'on exprimera de manière optimale les vecteurs aléatoires de données en utilisant les 'l' premiers vecteurs de base obtenus.

C'est une méthode qui souligne les similarités ainsi que les différences des données dans un espace à grandes dimensions où l'on n'a pas le luxe de la représentation graphique en identifiant la dépendance des structures des observations stochastiques multidimensionnelles dans le but d'avoir sa description compacte. Elle est communément considérée comme un outil de visualisation des données, cependant c'est aussi un moyen :

- de décorréler ces données ; dans le nouvel espace, constitué des nouveaux axes.
- de débruiter ces données, en considérant que les axes que l'on oublie sont des axes bruités.

Cette technique peut être vue de deux points de vue : celui de maximiser la variance ou encore celui de minimiser l'erreur quadratique moyenne. Les données représentées dans le nouvel espace sont appelées composantes principales, Elles sont décorrélées et ordonnées dans le sens décroisant de la variance.

$$M = \begin{bmatrix} X_{1,1} & . & . & . & X_{N,1} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ X_{1,K} & . & . & . & X_{N,K} \end{bmatrix} \quad (2.1)$$

Il est usuel d'appliquer une analyse en composantes principales sur un ensemble de « *N* » variables aléatoires $X_1, \ldots, X_N$ connues de dimension « *K* ». Ces « *N* » variables aléatoires peuvent être structuré dans une matrice *M* à *K* lignes et *N* colonnes.

Chaque variable aléatoire $X_n = (X_{n,1}, \ldots, X_{n,K})'$ a une moyenne $\bar{X}_n$ et un écart type $\sigma_{Xn.}$

1.1.2. Traitement des données

On notera *« g »* le vecteur $(\overline{X_1},...,\overline{X_N})$ qui représente le centre de gravité du nuage de points**.**

Les données à traiter sont couramment centrées (normalisées) sur le centre de gravité : La matrice *M* deviendra :

$$\bar{M} = \begin{bmatrix} X_{1,1} - \bar{X}_1 & \cdots & X_{N,1} - \bar{X}_N \\ \vdots & \ddots & \vdots \\ X_{1,K} - \bar{X}_1 & \cdots & X_{N,K} - \bar{X}_N \end{bmatrix} \quad (2.2)$$

Ces mêmes données peuvent être aussi *réduites* ou *blanchies*, alors la matrice *M* s'écrira:

$$\tilde{M} = \begin{bmatrix} \frac{X_{1,1}-\bar{X}_1}{\sigma(X_1)} & \cdots & \frac{X_{N,1}-\bar{X}_N}{\sigma(X_N)} \\ \vdots & \ddots & \vdots \\ \frac{X_{1,K}-\bar{X}_1}{\sigma(X_1)} & \cdots & \frac{X_{N,K}-\bar{X}_N}{\sigma(X_N)} \end{bmatrix} \quad (2.3)$$

Le fait de réduire ou non le nuage de points $(X_1, \ldots, X_N)$ dépend du modèle et du devenir des données:

- Quand les données ne sont pas réduites, toute variable à forte variance (un bruit par exemple) va majorer toutes les autres et faussera par la suite les résultats d'une PCA.
- Et quand les données sont réduites, toutes les variables vont se retrouver avec une variance apparente égale, ce qui rendra le bruit équivalent à une variable informative.

Dans ce cas, le choix de cette normalisation ou encore de ce blanchissement dépendra des données en premier lieu, et puis des applications postérieures des résultats de PCA.

1.1.3. Calcul des covariances et des corrélations

Le calcul des matrices de covariances et de corrélations est classique, du moment où les données sont déjà structurées en matrices et éventuellement normalisées ou réduites, il faut juste multiplier celle-ci par sa transposé :

- la matrice de variance-covariance des X_1, …, X_N si « *M* » est juste normalisée :

$$\text{Covariance} = {}^{1}\!/_{K} * \overline{M}' * \overline{M} \tag{2.4}$$

- la matrice de corrélation des X_1, …, X_N si « *M* » est réduite.

$$\text{Correlation} = {}^{1}\!/_{K} * \widetilde{M}' * \widetilde{M} \tag{2.5}$$

Ces deux matrices sont carrées (de taille *N*x*N*), symétriques, et réelles. Elles sont donc diagonalisables, dans une base orthonormée.

1.1.4. Projection

Comme cité ci-dessus, l'analyse en composantes principales vise à projeter les données d'un espace initial vers un second espace, de dimension inférieure, tel que son premier axes « *u* » soit issu d'une combinaison linéaire des X_n, et de manière à ce que la variance de la projection des points du nuage sur cet axe soit maximale. La projection de l'échantillon des *X* sur *u* « π » s'écrit :

$$\pi_u(M) = M * u \tag{2.6}$$

La variance de cette projection « $\pi_u(M)$ » vaudra donc :

$$\pi_u(M)' * {}^{1}\!/_{K} * \pi_u(M) = u' * \underbrace{M' {}^{1}\!/_{K} * M}_{C} * u \tag{2.7}$$

où *C* est la matrice de covariance.

Ayant vu précédemment que « *C* » était diagonalisable dans une base orthonormée, c'est cette diagonalisation qui va nous permettre en outre de voir que la variance exprimée par le $k^{ème}$ vecteur

propre valait λ_k. Et finalement, que la question de PCA nous ramène à un problème de diagonalisation de la matrice de corrélation. On notera « *P* » la matrice de changement de base associé et Δ la matrice diagonale formée de son spectre :

$$\pi_u(M)'*\frac{1}{K}*\pi_u(M) = u'*P'*\Delta*P*u = (P*u)'\Delta\underbrace{(P*u)}_{v} \tag{2.8}$$

À partir de cette reformulation, on cherchera désormais le vecteur « *v* » maximisant $v'*\Delta*v$, où $\Delta = Diag(\lambda_1,...,\lambda_N)$ est une matrice diagonale dont les valeurs sont rangés dans un ordre décroissant. Là, il sera très facile de constater que le premier vecteur unitaire vérifiera $v'*\Delta*v = \lambda$

Formellement et mathématiquement, on va utiliser les multiplicateurs de Lagrange « α » pour prouver ce résultat, et ce en maximisant la variance des données projetées sur *u* sous la contrainte que *u* soit de norme 1:

$$L(u,\alpha) = u'*C*u - \alpha(u'*u - 1) \tag{2.9}$$

En résolvant cette équation de Lagrange, deux résultats nous importent : en premier lieu « *u* » est un vecteur propre de *C* associé à la valeur propre λ_1 ; et en suite, il est de norme « 1 ». La valeur propre de la matrice de covariance *C* : « λ_1» étant la variance sur le premier axe de PCA.

On poursuit la recherche du deuxième axe de projection *w* sur le même principe en imposant qu'il soit orthogonal à *u*, ainsi de suite, jusqu'à l'obtention des « l » vecteurs propres recherchés.

1.1.5. L'algorithme

Dans un premier temps les données sont normalisées en soustrayant la moyenne de chaque colonne à tous ces éléments ainsi que celle de chaque ligne, aussi, à tous ces éléments ce qui nous donnera une moyenne nulle.

Ensuite nous calculerons la matrice de covariance de l'ensemble. Puisque celle-ci est carrée et diagonalisable ; il est possible de calculer ses valeurs et vecteurs propres, et par ce procédé les lignes qui caractérisent les données -*les vecteurs propres*- ont pu être extraites. Par la suite, les données seront exprimées en fonction de ces vecteurs propres qui seront classées dans l'ordre descendant de leur valeur propre ; ils seront donc, les nouveaux axes de représentation des données. Les premiers vecteurs de valeurs propres importantes donneront accès aux informations les plus pertinentes à dimension réduite.

On choisira ensuite, les composantes principales qui formeront « *la matrice caractéristique*» de transition qui sera composée des « l » premier vecteurs propres afin de prendre en compte les informations les plus signifiantes :

$$la\ matrice\ caractéristique = [eig_1, eig_2, ..., eig_l] \quad (2.10)$$

L'étape suivante sera de changer le repère des données et ce en transposant la matrice caractéristique et en la multipliant à gauche par les données dans l'ancien repère transposées :

$$Les\ données\ réduites = (la\ matrice\ caractéristique)' * \overline{M}' \quad (2.11)$$

L'étape finale est de récupérer les données originales après un éventuel traitement avec la perte d'information dû à la compression :

$$(Les\ données\ originales - la\ moyenne)' = (la\ matrice\ caractéristique)^{-1} * les\ données\ réduites \quad (2.12)$$

Vu que la matrice caractéristique n'est pas carrée à cause des vecteurs propres éliminés, il n'est pas possible de calculer l'inverse ; et là, il suffira de prendre la transposé car les éléments de la matrices sont des vecteurs propres unitaire de notre base de données.

L'équation finale après l'addition de la moyenne substituée initialement sera :

$$(Les\ données\ originales)' = (la\ matrice\ caractéristique)^{T} * les\ données\ réduites + la\ moyenne \quad (2.13)$$

Algorithme de PCA

- Calcul de $\overline{M}$.
- A=cov $\overline{M}'$.
- Calcul des valeurs et vecteurs propres.
- Calcul de « l »-le nombres de composantes à garder.
- Calcul de $la\ matrice\ caractéristique = [eig_1, eig_2, ..., eig_l]$.
- $Les\ données\ réduites = (la\ matrice\ caractéristique)' * \overline{M}'$.
- $(Les\ données\ originales - la\ moyenne)' = (la\ matrice\ caractéristique)^{-1} * les\ données\ réduites$.
- $(Les\ données\ originales)' = (la\ matrice\ caractéristique)^{T} * les\ données\ réduites + la\ moyenne$.

1.2. La décomposition en valeur singulière

Elle est aussi considérée comme méthode de décomposition orthogonale propre, Klema et Laub [17] ont indiqué que la SVD a été établi pour les matrices carrées réelles dans les années 1870 par Beltrami Jordan. Pour les matrices carrées complexes en 1902 par Autonne, et pour les matrices rectangulaire en général in 1939 by Eckart et Young. La SVD peut être vue comme une extension de la décomposition en valeurs propres dans le cas des matrices non carrées.

1.2.1. Principe

L'idée essentielle de la SVD est de décomposer la matrice de données en trois matrices simples : deux orthogonales et une diagonale. Du fait qu'elle produise une estimation aux moindres carrés de la matrice de données de même dimension et d'un rang inférieur, elle est équivalente à PCA, et importante autant qu'elle.

L'un des avantages de la SVD est son pouvoir de réduction des données après leur blanchissement. En effet, cette technique fournit une description plus compacte des données contenues dans une matrice, exprimée par les premiers modes statistiques. Elle peut être considérée comme une méthode permettant de construire une partition de la variance d'une base de données, c'est à dire qu'elle fournit la base orthogonale qui maxime la variance au sens des moindres carrés. Ceci signifie que le mode « 1 » contient une variance maximale et qu'il contient la structure la mieux corrélée. De manière générale, les premiers modes de la décomposition devraient préférentiellement capter les structures cohérentes et les derniers modes les structures aléatoires du champ, c'est à dire sans structure particulière.

La décomposition en valeurs singulières utilise la décomposition en valeur propre d'une matrice semi définie positive obtenue par la multiplication d'une matrice par sa transposé, pour dériver une décomposition similaire applicable à toutes les matrices rectangulaires composées de nombres réels.

1.2.2. La décomposition

Formellement, si « A » est une matrice rectangulaire, son SVD la décompose comme suit :

$$A = SUV^t \tag{2.14}$$

S : les vecteurs propres normalisés de la matrice AA^t, c'est-à-dire $S^tS = I$.

Les colonnes de « S » sont les vecteurs singuliers de gauche de A.

V : les vecteurs propres normalisés de la matrice A^tA, c'est-à-dire $V^tV = I$.

Les colonnes de « V » sont les vecteurs singuliers de droite de A.

U : la matrice diagonale des valeurs singulières.

$U = \lambda^{\frac{1}{2}}$ avec Λ est la matrice diagonale des valeurs propres de la matrice AA^t et la matrice A^tA.

La SVD a l'importante propriété de donner la meilleure approximation d'une matrice rectangulaire par une autre matrice de même dimension mais de rang inférieur, au sens des moindres carrées. Précisément si « A » est de dimension [IxJ] et de rang « L », donc « A » a « L » valeurs singulières non nulles.

En plus de la propriété de réduction de dimension, la SVD a l'avantage de pouvoir estimer l'inverse de n'importe quelle matrice qu'elle soit carrée ou rectangulaire, et surtout qu'elle soit singulière ou pas.

La clef de l'interprétation de la SVD est l'examen de la distribution des poids (les valeurs singulières). Cette dernière renseigne sur le degré de redondance des données, et fournit également des informations importantes, telles que les symétries par exemple. L'ordre décroissant de ces poids nous permet de dire que les premiers modes contiennent les propriétés principales des données considérées. Plus exactement, ce sont les modes qui vont capter la majeure partie de la variance globale des données.

1.2.3. Calcul du pseudo inverse

La décomposition en valeurs singulières permet le calcul de l'inverse d'une matrice et surtout de son pseudo inverse. En effet, le pseudo inverse d'une matrice *M* est souvent difficile et prend beaucoup de temps à être estimé directement ; mais connaissant sa décomposition en valeurs singulières $A = SUV^t$, la tache est facilitée, puisque l'inverse d'une matrice orthonormale est sa transposée Puisque **U** est diagonale**,** $\mathbf{U}^{-1}$ est aussi diagonale. Son expression est donné par :

$$A^{-1} = (V^t)^{-1} * U^{-1} * S^{-1} = V * U^{-1} * S^t$$

$$A^{+} = V * U^{+} * S^t \tag{2.15}$$

avec U^+ l'inverse de U où tout coefficient non nul est remplacé par son inverse. Le pseudo inverse lui-même permet de résoudre la méthode des moindres carrés.

1.2.4. Approximations de matrices

Définition: *la distance de Frobenius* est une mesure standard de distance entre deux matrices Σ_1 *et* Σ_2 de même dimensions [n x n] qui se définie par

$$\begin{aligned} F(\Sigma_1, \Sigma_2) &= \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(s_{ij}^1 - s_{ij}^2)^2} \\ &= \sqrt{tr\left((\Sigma_1 - \Sigma_2)(\Sigma_1 - \Sigma_2)^T\right)} \end{aligned} \tag{2.16}$$

Il en général intéressant de pouvoir approximer une matrice « A » à partir d'une matrice $\tilde{A}$ ayant un rang donné, égal à *r,* pour le besoin d'une application. La résolution de ce problème, lorsqu'il s'agit

de minimiser la distance au sens de Frobenius entre A et $\widetilde{A}$ en gardant $rang(\widetilde{A}) = r$, est la décomposition en valeurs singulières de *A*, c'est-à-dire :

$$\widetilde{A} = V * \widetilde{U} * S^t \tag{2.17}$$

avec $\widetilde{U}$ égale à U, si ce n'est qu'elle ne contient que les *r* plus grandes valeurs singulières, les autres étant remplacées par 0.

Ainsi, $\widetilde{A}$ matrice de rang *r*, est la meilleure approximation de *A* au sens de la norme de Frobenius quand $\sigma_i = s_i (i = 1,...,r)$ où s_i sont le valeurs singulières de $\widetilde{A}$. Alors, ses valeurs singulières sont les mêmes que celles de *A*.

Si $\widetilde{S}_K$ (respectivement $\widetilde{V}_K$ et $\widetilde{U}_K$) sont construites à partir des K premières colonnes de S (respectivement V et U), la matrice A reconstruite a partir des K premières sera :

$$\widetilde{A}_{[K]} = \widetilde{S}_{[K]} \widetilde{U}_{[K]} \widetilde{V}^t_{[K]} = \sum_k^K \delta_k s_k v_k^t \tag{2.18}$$

où δ_k est la k^{eme} valeur singulière.

La matrice obtenu est dite optimale dans le sens des moindres carrées pour les matrices de rang K parce qu'elle satisfait la condition suivante :

$$\left\| \mathrm{A} - \mathrm{A}_\mathrm{K} \right\|^2 = trace\left\{ (A - A_K)(A - A_K)^t \right\} = \min_x \left\| A - X \right\|^2 \tag{2.19}$$

Pour un ensemble de matrices X de rang égale ou inférieur à K, la qualité de la reconstruction est donnée par le quotient de la somme des K premières valeurs propres par la somme de toutes celles-ci.

1.2.5. Interprétation géométrique de la SVD

La SVD peut être interprétée comme une transformation géométrique (une rotation) d'un repère orthogonal dans une nouvelle base ortho normale. Cette rotation géométrique sera précédée généralement, le cas échéant, d'une translation si nous centrons nos données, c'est-à-dire si nous procédons au retrait de la moyenne (suivant le type de données analysées).

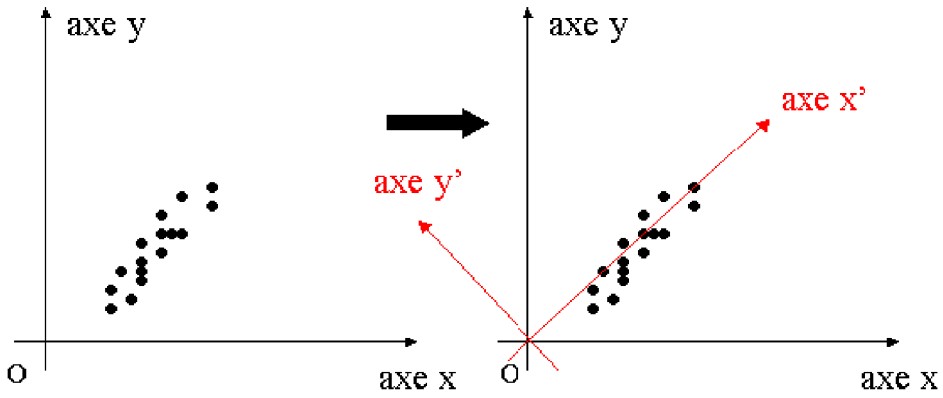


Fig. 2.1. Approche géométrique de la SVD : la SVD est équivalent à un changement de référentiel.

Dans un espace de dimension quelconque (supérieure ou égale à 3), la SVD revient à approximer le nuage de points par un hyperplan.

1.2.6. L'algorithme

Avant d'être décomposé en valeur singulière, les données subissent en premier lieu une normalisation de manière à avoir une moyenne nulle et une déviation standard égale à « 1 ».

Puis nous passerons à la décomposition proprement dite, de la matrice image, en trois matrices *S* et *V* qui seront des matrices orthogonales carrées et U une matrice rectangulaire de même dimension que les données initiales :

$$A = SUV^t$$

Elle sont tronquées de manière à garder les valeurs singulière les plus significatives et leurs vecteurs propres correspondants on obtiendra ainsi de nouvelles matrices qui serviront à reconstituer l'essentiel des données :

$$\widetilde{A}_{[K]} = S_{[K]} U_{[K]} V^t_{[K]} = \sum_{k}^{K} \delta_k s_k v^t_k$$

Cette approximation de la matrice initiale nous a permis de redimensionné notre base de données de manière à garder uniquement les informations qui nous intéressent.

Algorithme de SVD

- Calcul de $\widetilde{M}$.
- Calcul des valeurs singulières et vecteurs propres à gauche et à droite.
- Calcul de S, U et V.
- Calcul de « K » :le nombres de valeurs singulières à garder.

- Tronquer les matrices à K.
- Calcul de la nouvelle matrice « approximée »par $\widetilde{A}_{[K]} = S_{[K]} U_{[K]} V^t_{[K]} = \sum_{k}^{K} \delta_k s_k v^t_k$.

1.3. L'analyse en composantes indépendantes

Au même titre que la SVD, l'ICA est une technique statistique d'analyse de données multidimensionnelles [18]. Comme la SVD, cette technique est dédiée à la recherche de projections significatives d'une distribution spatiale de données.

En SVD sont recherchées les directions orthogonales de l'espace des données porteuses du maximum d'information au sens de la maximisation de l'énergie du signal. Les modes statistiques (ou composantes principales) désignent les composantes des données projetées le long de ces directions. Les propriétés statistiques des données exploitées par la SVD, formalisées par la matrice de covariance, se limitent au second ordre. La SVD réalise ainsi une identification de la structure de la dépendance corrélative d'une distribution de données [19]. L'introduction du concept de l'ICA [19] vise précisément à dépasser cette limitation. En ICA, sont recherchées les directions génératrices d'un espace de données statistiquement indépendantes. Les composantes projectives des données le long des directions génératrices sont alors dénommées composantes indépendantes, en raison de leur propriété d'indépendance statistique. Comparativement à la SVD, l'ICA exploite les propriétés statistiques des données d'ordre supérieur à 2, en plus de celles d'ordre inférieur ou égal à 2. L'ICA se présente donc comme une extension de la SVD.

Si les statistiques d'ordre 2 permettent de caractériser pleinement les distributions de données gaussiennes et exploitent la notion de décorrélation, elles s'avèrent insuffisantes à la caractérisation de toute autre distribution de données dont la description complète nécessite des statistiques d'ordre supérieur à 2 [19] qui exploitent plus complètement l'indépendance des sources.

L'analyse en composantes indépendantes (ACI) est souvent illustrée par le problème de la « cocktail party » : *d* personnes tiennent conversation dans un salon dans lequel *d* microphones d'enregistrement sont installés. Nous avons *d* signaux vocaux (sources) que l'on suppose indépendants et que l'on veut retrouver à partir des *d* enregistrements.

1.3.1. Principe

Cependant, l'idée de rechercher des composantes indépendantes était trop générale pour rester confinée dans son domaine d'origine : le traitement des signaux et images. Comme ça a été cité

précédemment que ICA était une extension de la SVD, d'un point de vue statistique elle peut être vue aussi comme une extension de l'analyse en composantes principales (PCA), car SVD et PCA partent du même principe. En fait, si PCA cherche à extraire des variables décorrélées, en se limitant à imposer la contrainte d'indépendance aux statistiques d'ordre deux (matrice de covariance), l'ICA en revanche, cherche l'indépendance des statistiques d'ordre supérieur à deux des variables mesurées.

L'ICA est aussi connectée, en tant que méthode d'exploration de données, aux idées de la méthode dite projection poursuite (PP) [18]. En utilisant la méthode PP, on cherche des directions intéressantes dans un nuage de données multidimensionnelles, en arguant du fait que les structures intéressantes sont celles qui font apparaître le plus l'aspect non gaussien des données projetées.

Il existe un lien très significatif entre dépendance et corrélation d'une variable aléatoire dans le cadre du modèle linéaire de l'ICA. En fait, la corrélation ne dépend que de la matrice de covariance. Elle représente, en quelque sorte, la partie gaussienne de l'information mutuelle qui existe entre la vraie distribution de la variable aléatoire et sa distribution estimée. L'aspect non gaussien d'une variable aléatoire est la divergence qui existe entre sa distribution et sa meilleure approximation gaussienne. Ainsi, l'ICA, qui cherche à expliquer une variable aléatoire multidimensionnelle *X*, en termes de composantes linéairement et statistiquement indépendantes, peut être reformulée comme étant la minimisation de la corrélation entre les composantes de *X* tout en maximisant l'aspect non gaussien de chaque composante. Dans les sous-sections suivantes, nous donnons les mesures de l'indépendance statistique qui peuvent être utilisées dans la mise en oeuvre de l'ICA. Ces mesures sont utilisées dans l'élaboration des critères d'optimisation des algorithmes de l'ICA. Ces critères sont appelés fonctions de contrastes [18].

L'ICA repose sur le modèle suivant [18] :

$$\mathrm{y} = A\mathrm{s} \tag{2.20}$$

où « y » est le vecteur des observations, « *A* » est la matrice de mélange à définir et « s » sont les sources à déterminer, supposées indépendantes et non gaussiennes.

Le théorème de la limite centrale affirme que la somme de variables aléatoires indépendantes tend, sous certaines conditions, vers la distribution gaussienne. De plus, la somme de deux variables aléatoires indépendantes est généralement plus proche de la gaussienne que n'importe laquelle des deux variables. Les observations qui sont des combinaisons linéaires des sources sont alors davantage gaussiennes que ces dernières, et l'estimation de $W = A^{-1}$ peut être basée sur la maximisation de la propriété non gaussienne des sources. Une mesure de cette propriété s'impose donc. La plus classique est le Kurtosis qui est le cumulant normalisé d'ordre 4, mais il reste peu robuste

$$Kurt(y) = E\{y^4\} - 3 * E\{y^2\}^2 \tag{2.21}$$

Des valeurs strictement négatives du Kurtosis sont obtenues avec des distributions sous gaussiennes (plus aplaties que la gaussienne) tandis que les distributions sur gaussiennes (plus piquées que la gaussienne) donnent des valeurs strictement positives. Malgré sa simplicité, le Kurtosis a l'inconvénient d'être très sensible aux données aberrantes.

La seconde mesure trouve ses bases dans la théorie de l'information. Il est établi que pour un ensemble de variables aléatoire de même variance, celle qui suit la loi gaussienne maximise l'entropie donnée par :

$$H(y) = -\int f_y(u) * \log f_y(u) du \tag{2.21}$$

où $f_y(y)$ est la densité de y. Cette remarque permet d'introduire une mesure de la propriété non gaussienne appelée néguentropie définie par :

$$J(y) = H(y_g) - H(y) \tag{2.22}$$

où y_g est un vecteur aléatoire gaussien de même covariance que y. La néguentropie est égale à la divergence de Kullback-Leibler entre $f_y(y)$ et $f_{y_g}(y_g)$. Etant donnée la difficulté d'évaluer « J » d'après cette équation, cette quantité est en pratique approximée. Une méthode classique d'approximation est:

$$j(y) = \frac{1}{12} E\{y^3\}^2 - \frac{1}{48} * Kurt(y)^2 \tag{2.23}$$

Cet estimateur basé sur le kurtosis demeure peu robuste. Hyvärinen [19] a proposé une approximation robuste de la néguentropie :

$$j(y) = \left[E\{G(y)\} - E\{G(y_g)\}\right]^2 \tag{2.24}$$

où G est une fonction non quadratique1puisqu'alors J serait trivialement nul pour toutes les distributions. La maximisation de J permet de retrouver la matrice de projection $W = A^{-1}$

Comme l'indépendance implique la non corrélation, l'espace de recherche est restreint aux composantes décorrélées. Cela est mis en pratique grâce à une procédure de blanchissement des observations préalable à l'optimisation de *J*. Cette opération n'est autre qu'une ACP sous la contrainte d'une matrice de covariance identité pour les vecteurs transformés. Elle est donnée par:

$$\tilde{y} = U * \Lambda^{-\frac{1}{2}} * U^t * y \tag{2.25}$$

où $\tilde{y}$ est le vecteur blanchi, Λ est la matrice des valeurs propres de la matrice de covariance de « y » et U est la matrice des vecteurs propres de la matrice de covariance de « y ».

Hyvärinen [19] a proposé une approximation robuste de la néguentropie et un algorithme rapide pour sa maximisation (FastICA). Nous avons choisi la version FastICA avec décorrélation

déflationniste qui recherche les vecteurs colonnes de W l'un après l'autre en favorisant les premiers. Elle se base sur la procédure d'orthogonalisation de Gram-Schmidt qui contraint le vecteur colonne w_i à appartenir à l'espace orthogonale aux (i- 1) vecteurs déjà déterminés.

1.3.2. L'algorithme

Comme l'indépendance implique la décorrélation, la maximisation de la néguentropie se fait sur l'espace réduit des données décorrélées par une PCA initiale où toutes les valeurs propres sont gardées.

L'algorithme va suivre les étapes suivantes après un blanchissement des données :

Algorithme de ICA

- Calcul $y = \widetilde{M}$.
- Choisir un vecteur de poids initial et aléatoire **w.**
- calculer $\mathbf{w}^+$ tel que :

$$\mathbf{w}^+ = E\{\mathbf{y}^* g(\mathbf{w}^T * y)\} - E\{g_0(\mathbf{w}^T * y)\}\mathbf{w}.$$

- Puis $\mathrm{w} = \mathrm{w}^+ \Big/ \left\|\mathrm{w}^+\right\|$.
- S'il n'y a pas de convergence retourner à 2.

Un choix judicieux de « g » serait :

$$g(u) = \tanh(a_1 u) \quad \text{ou} \quad g(u) = u \exp(-u^2/2)$$

Où $1 \leq a1 \leq 2$, on aura plus tendance à le choisir égale à 1.

Il faut noter que la convergence veut dire que l'ancienne et la nouvelle valeur de **w** aient la même direction. Il est nécessaire que les vecteurs convergent au même point, du moment où **w** et –**w** définissent la même direction, c'est parce que les composantes indépendantes peuvent être uniquement définies par un signe multiplicatif.

1. Les méthodes supervisées

Contrairement aux méthodes non supervisées, celles-ci imposent des limites à l'analyse des données sous certaines contraintes du modèle. Nous énumérons quelques unes : l'analyse discriminante linéaire (LDA), l'algorithme de classification à erreur minimales (MCE) et les séparateur à larges marges (SVM).

2. L'analyse discriminante linéaire

Beaucoup des techniques de classification deviennent peu appropriées ou inefficaces à cause de la haute dimensionnalité des données. L'analyse linéaire discriminante de Fisher (LDA) est une méthode bien connue et très utilisée dans la phase d'apprentissage d'un classificateur. L'analyse discriminante linéaire doit son nom au fait qu'elle réalise des séparations linéaires entre les classes. Proposée par Ronald A. Fisher en 1936 [20], l'Analyse Factorielle Discriminante - *Fisher Discriminant Analysis (FDA)* - appelée aussi analyse discriminante linéaire de Fisher s'applique lorsque les classes des individus sont connues et essaie de trouver un sous-espace vectoriel de faible dimension dans lequel les moyennes des attributs des classes (centroïdes des nuages d'échantillons des classes) sont dispersées par rapport à la covariance interne de chaque classe (tailles des nuages) et donc maximise la variance inter-classe. Elle projette les données sur ce sous-espace, en évitant aussi des effets collatéraux comme des possibles pertes d'information. Le problème consiste alors à trouver la projection linéaire optimale qui satisfait tous ces critères. Une base de cet espace est obtenue en appliquant une Analyse en Composantes Principales sur les centroïdes des différentes classes pondérés par l'effectif de la classe correspondante avec Σ^{-1} : la matrice de covariance inversé comme métrique. On conservera, au plus, $(C - 1)$ axes discriminants où « C » est le nombre de classes.

2.1. Principe

L'idée est de trouver une projection des échantillons, sur une droite qui sépare le mieux possible les classes. Le critère de séparation est exprimé par la maximisation du rapport des variances inter classes et intra classes dans cette projection. Autrement dit, la distribution de données d'une même classe devrait être compacte, ce qui reviendrait à minimiser la dispersion autour de la moyenne, ainsi qu'à la maximiser entre les classes.

2.1.1 L'analyse discriminante linéaire orientée ligne

La discrimination entre données est réalisée par apprentissage supervisé. En effet les données sont préclassifiées, et il s'agit d'optimiser cette classification en appliquant une projection linéaire à droite de la matrice de données X ($n \times m$) sur une matrice $P(m \times d)$ telle que:

$$W = X.P \tag{3.1}$$

Où la matrice W ($n \times d$) est dite matrice caractéristique (ou matrice de la signature associée aux données dans X).

La matrice de projection « P » est obtenue en maximisant le critère de Fisher généralisé suivant:

$$P^* = \underset{P \in R^{(mxd)}}{arg\ max} = \frac{\left| P^T * S_b * P \right|}{\left| P^T * S_w * P \right|} \tag{3.2}$$

Où S_b est la matrice de covariance inter classes et S_w est la matrice de covariance intra classes données respectivement par:

$$S_b = \sum_{c=1}^{C} n_c (\overline{X}_c - \overline{X})^T (\overline{X}_c - \overline{X}) \tag{3.4}$$

$$S_w = \sum_{c=1}^{C} \sum_{i=1}^{n_c} (\overline{X}_i - \overline{X}_c)^T (\overline{X}_i - \overline{X}_c) \tag{3.5}$$

avec $\overline{X}_c$: matrice moyenne de la classe c,

X : matrice moyenne totale (de toutes les matrices de la base d'apprentissage).

Le résultat de l'optimisation du critère de Fisher est une matrice « P » dont les colonnes sont les vecteurs propres de la matrice $S_w^{-1} * S_b$ associées aux « d » plus grandes valeurs propres.

2.1.2. L'analyse discriminante linéaire orientée colonnes

Dans ce cas, la projection de chaque matrice de données « X » est effectuée à gauche telle que $Z = L*T$. « X » est la matrice caractéristique de dimension ($d \times m$) où L ($n \times d$) est la matrice de projection optimale maximisant le critère de Fisher généralisé :

$$L^* = \underset{L \in R^{(nxd)}}{arg\ max} = \frac{\left| L^T * \Sigma_b * L \right|}{\left| L^T * \Sigma_w * L \right|} \tag{3.6}$$

ayant pour colonnes, les *d* premiers vecteurs propres de la matrice $\Sigma_w^{-1}\Sigma_b$ associés aux «*d* » plus grandes valeurs propres. Σ_w *et* Σ_b désignent respectivement les matrices intra et inter classes généralisées de la base d'apprentissage:

$$\Sigma_w = \sum_{c=1}^{C}\sum_{i=1}^{n_c}(X_i - \overline{X}_c)(X_i - \overline{X}_c)^T \tag{3.7}$$

$$\Sigma_b = \sum_{c=1}^{C} n_c * (\overline{X}_c - \overline{X})(\overline{X}_c - \overline{X})^T \tag{3.8}$$

2.2. Ses limites

LDA utilise des séparations linéaires pour la discrimination entre les échantillons, des combinaisons linéaires des variables indépendantes. L'analyse discriminante linéaire peut être utilisée pour la classification des données qui sont linéairement séparable, si c'est le cas, c'est une indication que les caractéristiques choisies pour la classification sont discriminatoires; figure 3.1 (a). L'exemple de la figure 3.1 (b) illustre le cas où LDA ne donnera pas de bons résultats de classification car les deux classes se chevauchent.

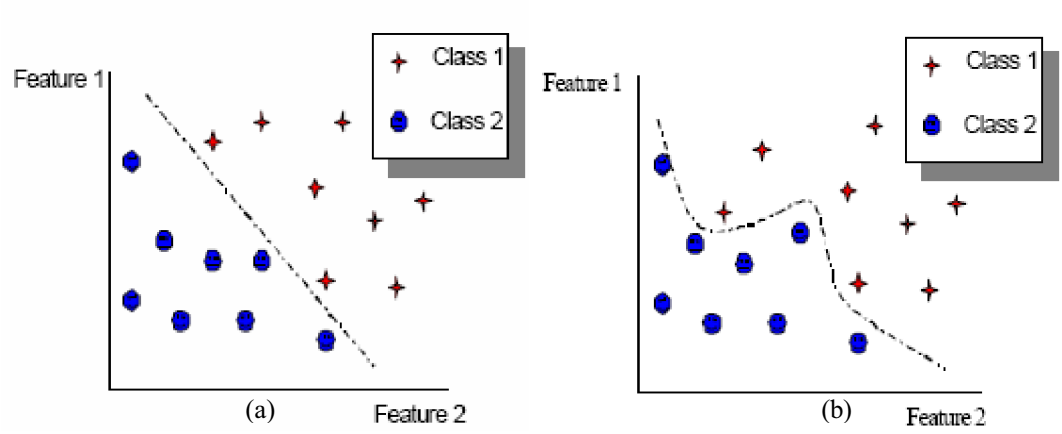


Fig.3.1 : les séparation par LDA

L'analyse discriminante linéaire est fréquemment utilisée car elle offre un bon compromis entre pertinence et complexité. D'autre part, elle fournit des résultats robustes aux fluctuations de la normalité des classes et d'égalité des matrices de variance. Pour ces raisons, elle doit être considérée comme une méthode de référence.

1. Les séparateurs à larges marges

Les machines à vecteurs de support ou séparateurs à vaste marge (*Support Vector Machine,* SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre

des problèmes de discrimination et de régression. Les SVM sont une généralisation des classifieurs linéaires. Les SVM sont une famille de classifieurs binaires développés dans les années 1990 à partir des considérations théoriques de Vladimir Vapnik sur le développement d'une théorie statistique de l'apprentissage : la Théorie de Vapnik-Chervonenkis [21] et connaissent un franc succès dans la communauté du « *machine learning* ». Les SVM ont rapidement été adoptés pour leur capacité à travailler avec des données de grandes dimensions, le faible nombre de paramètres, le fait qu'ils soient bien fondés théoriquement, et leurs bons résultats en pratique.

Les SVM ont été appliqués à de très nombreux domaines (bioinformatique, recherche d'information, vision par ordinateur, finance…). Selon les données, la performance des machines à vecteurs de support est de même ordre, ou même supérieure, à celle d'un réseau de neurones ou d'un modèle de mixture gaussienne. Les séparateurs à vastes marges sont des classifieurs qui reposent sur deux idées clés, qui permettent de traiter des problèmes de discrimination non linéaire, et de reformuler le problème de classement comme un problème d'optimisation quadratique.

La première idée clé est la notion de *marge maximale*. La marge est la distance entre la frontière de séparation et les échantillons les plus proches. Ces derniers sont appelés *vecteurs supports*. Dans les SVM, la frontière de séparation est choisie comme celle qui maximise la marge. Ce choix est justifié par la théorie de Vapnik-Chervonenkis (ou théorie statistique de l'apprentissage). Le problème est de trouver cette frontière séparatrice optimale, à partir d'un ensemble d'apprentissage. Ceci est fait en formulant le problème comme un problème d'optimisation quadratique, pour lequel il existe des algorithmes connus.

Afin de pouvoir traiter des cas où les données ne sont pas linéairement séparables, la deuxième idée clé des SVM est de transformer l'espace de représentation des données d'entrées en un espace de plus grande dimension (possiblement de dimension infinie), dans lequel il est probable qu'il existe une séparatrice linéaire. Ceci est réalisé grâce à une fonction noyau, qui doit respecter certaines conditions, et qui a l'avantage de ne pas nécessiter la connaissance explicite de la transformation à appliquer pour le changement d'espace. Les fonctions noyau permettent de transformer un produit scalaire dans un espace de grande dimension, ce qui est coûteux, en une simple évaluation ponctuelle d'une fonction. Cette technique est connue sous le nom de kernel trick.

3.1. Principe

Les SVM peuvent être utilisés pour résoudre des problèmes de discrimination, c'est-à-dire décider à quelle classe appartient un échantillon, ou de régression, c'est à dire prédire la valeur numérique d'une variable. La résolution de ces deux problèmes passe par la construction d'une fonction f qui à un vecteur d'entrée x fait correspondre une sortie $y = f(x)$.

On se limite pour l'instant à un problème de discrimination à deux classes (discrimination binaire), c'est à dire $y \in \{-1,1\}$, le vecteur d'entrée x étant dans un espace X muni d'un produit scalaire. On peut prendre par exemple $X = R^N$.

Pour rappel, le cas simple est le cas d'une fonction discriminante linéaire, obtenue par combinaison linéaire du vecteur d'entrée $x = (x_1,...,x_N)^T$: $h(x) = w^T * x + w_0$

Il est alors décidé que x est de classe 1 si $h(x) \geq 0$ et de classe -1 sinon. C'est un classifieur linéaire.

La frontière de décision $h(x) = 0$ est un hyperplan, appelé *hyperplan séparateur*, ou *séparatrice*. Rappelons que le but d'un algorithme d'apprentissage supervisé est d'apprendre la fonction h(x) par le biais d'un ensemble d'apprentissage :

$$\{(x_1,l_1),(x_2,l_2),...,(x_p,l_p)\} \in R^N \times \{-1,1\} \tag{3.9}$$

où les l_k sont les labels des classes, p est la taille de l'ensemble d'apprentissage, N la dimension des vecteurs d'entrée. Si le problème est linéairement séparable, on doit alors avoir : $l_k h(x_k) \geq 0 \quad 1 \leq k \leq p$ autrement dit $l_k(w^T x_k + w_0) \geq 0 \quad 1 \leq k \leq p$

On se place désormais dans le cas où le problème est linéairement séparable. Même dans ce cas simple, le choix de l'hyperplan séparateur n'est pas évident. Il existe en effet une infinité d'hyperplans séparateurs, dont les performances en apprentissage sont identiques, mais dont les performances peuvent être très différentes. Pour résoudre ce problème, il a été montré qu'il existe un unique hyperplan optimal, défini comme l'hyperplan qui maximise la marge entre les échantillons et l'hyperplan séparateur.

Il existe des raisons théoriques à ce choix. Vapnik a montré que la capacité des classes d'hyperplans séparateurs diminue lorsque leur marge augmente.

La marge est la distance entre l'hyperplan et les échantillons le plus proches. Ces derniers sont appelés *vecteurs supports*. L'hyperplan qui maximise la marge est donné par :

$$arg \ \max_{w,w_0} \ \min_k \left\{ \left\| x - x_k \right\| : x \in R^N \ , w^T * x + w_0 = 0 \right\} \tag{3.10}$$

Il s'agit donc de trouver w et w_0 remplissant ces conditions, afin de déterminer l'équation de l'hyperplan séparateur : $h(x) = w^T x + w_0 = 0$

3.2. Recherche de l'hyperplan optimal

La marge est la plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur qui satisfait la condition de séparabilité (à savoir $l_k(w^T x_k + w_0) \geq 0$ comme expliqué précédemment). La distance d'un échantillon x_k à l'hyperplan est donnée par sa projection orthogonale sur l'hyperplan:

$$\frac{l_k(w^T x_k + w_0)}{\|w\|} \tag{3.11}$$

L'hyperplan séparateur (w,w_0) de marge maximale est donc donné par:

$$arg \ \max_{w,w_0} \left\{ \frac{1}{\|w\|} \min_k \left[l_k(w^T x_k + w_0) \right] \right\} \tag{3.12}$$

Afin de faciliter l'optimisation, on choisit de normaliser w et w_0, de telle manière que les échantillons à la marge ($x^+_{m\arg e}$ pour les vecteurs supports sur la frontière positive, et $x^-_{m\arg e}$ pour ceux situés sur la frontière opposée) satisfassent :

$$\begin{cases} w^T x^+_{m\arg e} + w_0 = 1 \\ w^T x^-_{m\arg e} + w_0 = -1 \end{cases} \tag{3.13}$$

d'où pour tous les échantillons, k=1,…,p $\qquad l_k(w^T x_k + w_0) \geq 0$ (3.14)

Cette normalisation est parfois appelée la forme canonique de l'hyperplan, ou *hyperplan canonique*. Avec cette mise à l'échelle, la marge vaut désormais $1/\|W\|$ il s'agit donc de maximiser $\|W\|^{-1}$ La formulation dite *primale* des SVM s'exprime alors sous la forme suivante

$$Minimiser \frac{1}{2}\|w\|^2 sous\ les\ contra\,\text{int}\,es\ \ l_k\left(w^T x_k + w_0\right) \geq 1 \tag{3.15}$$

Ceci peut se résoudre par la méthode classique des Multiplicateur de Lagrange, où le lagrangien est donné par : $$L(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{k=1}^{p} \alpha_k \left\{ l_k (w^T x_k + w_0) - 1 \right\} \tag{3.16}$$

Le lagrangien doit être minimisé par rapport à w et w_0, et maximisé par rapport à α. En annulant les dérivées partielles du lagrangien, selon les conditions de Kuhn-Tucker, on obtient :

$$\begin{cases} \sum_{k=1}^{p} \alpha_k l_k x_k = w^* \\ \sum_{k=1}^{p} \alpha_k l_k = 0 \end{cases} \tag{3.17}$$

Les conditions de Kuhn-Tucker

Si $\overline{w}$ est un minimum local de $f(w) = \|W\|^{-1}$, et si le condition de qualification des contraintes g_i et h_i sont vérifiées, alors il existe $\overline{\alpha}_1,...,\overline{\alpha}_p, \overline{l}_1,...,\overline{l}_p$ tels que :

- $\overline{l}_0 \nabla f(\overline{w}) + \sum_{i=1}^{p} \overline{\alpha}_i \nabla h_i(\overline{w}) + \sum_{j=1}^{p} \overline{l}_j \nabla g_i(\overline{w}) = 0$.
- $\overline{l_j} \geq 0$ pour $j \geq 0$ et $\overline{l}_j g_i(\overline{w}) = 0$ pour $j \geq 0$.

Les qualifications de contraintes sont :

- il existe d tel que $\nabla h_i(\overline{w})^t * d = 0$ pour tout i.
- et $\nabla g_j(\overline{w})^t * d \prec 0$ pour tout $j \in J(\overline{w})$ et $\nabla h_i(\overline{w})$ sont linéairement indépendantes.

Pour notre cas les contraintes d'égalités n'existe pas.

En réinjectant ces valeurs dans l'équation de Lagrange, on obtient la formulation duale :

$$Maximiser\ \ \widetilde{L}(\alpha) = \sum_{k=1}^{p} \alpha_k - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j l_i l_j x_i^T x_j \tag{3.18}$$

$$sous\ les\ contra\,\text{int}\ es\ \alpha_k \geq 0, et \sum_{k=1}^{p} \alpha_k l_k = 0 \tag{3.19}$$

Ce qui donne les multiplicateurs de Lagrange optimaux α_k^*.

Afin d'obtenir l'hyperplan solution, on remplace w par sa valeur optimale w^*, dans l'équation de l'hyperplan *h*(*x*), ce qui donne :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k (x.x_k) + w_0 \tag{3.20}$$

3.3. Conséquences

Il y a trois remarques intéressantes à faire à propos de ce résultat. La première découle de l'une des conditions de Kuhn-Tucker, qui donne :

- $\alpha_k [l_k h(x_k) - 1] = 0 \quad 1 \leq k \leq p$ d'où : $\begin{cases} \alpha_k = 0 \\ l_k h(x_k) = 1 \end{cases}$

Les seuls points pour lesquels les contraintes du lagrangien sont actives sont donc les points tels que $l_k h(x_k) = 1$, qui sont les points situés sur les hyperplans de marges maximales. En d'autres termes, seuls les *vecteurs supports* participent à la définition de l'hyperplan optimal.

- La deuxième remarque découle de la première. Seul un sous-ensemble restreint de points est nécessaire pour le calcul de la solution, les autres échantillons ne participant pas du tout à sa définition. Ceci est donc efficace au niveau de la complexité. D'autre part, le changement ou l'agrandissement de l'ensemble d'apprentissage a moins d'influence que dans les autres modèles, où tous les points participent à la solution. En particulier, le fait d'ajouter des échantillons à l'ensemble d'apprentissage qui ne sont pas des vecteurs supports n'a aucune influence sur la solution finale.
- La dernière remarque est que l'hyperplan solution ne dépend que du produit scalaire entre le vecteur d'entrée et les vecteurs supports. Cette remarque est l'origine de la deuxième innovation majeure des SVM: le passage par un espace de redescription grâce à une fonction noyau

En pratique, on ne connaît pas la transformation φ. On construit plutôt directement une fonction noyau. Celle ci doit respecter certaines conditions, elle doit correspondre à un produit

scalaire dans un espace de grande dimension. L'exemple le plus simple de fonction noyau est le noyau linéaire : $$K(x_i, x_j) = x_i^T * x_j \quad (3.21)$$

On se ramène donc au cas d'un classifieur linéaire, sans changement d'espace. Les noyaux usuels employés pour les SVM sont:

- le noyau polynomial : $K(x_i, x_j) = (x_i^T * x_j)^d$ (3.22)
- le noyau gaussien : $K(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$ (3.23)

3.4. Cas multi classes

Plusieurs méthodes ont été proposées pour étendre les SVM au cas où plus de deux classes sont à séparer. Ces schémas sont applicables à tout classifieur binaire, et ne sont donc pas spécifiques aux SVM. Les deux plus connues sont appelées *one versus all* et *one versus one*. Formellement, les échantillons d'apprentissage et de test peuvent ici être classés dans M classes $\{C_1, C_2, ..., C_M\}$.

La méthode *one-versus-all* (appelée parfois *one-versus-the-rest*) consiste à construire M classifieurs binaires en attribuant le label 1 aux échantillons de l'une des classes et le label -1 à toutes les autres. En phase de test, le classifieur donnant la valeur de confiance (e.g la marge) la plus élevée remporte le vote.

La méthode *one-versus-one* consiste à construire $M(M-1)/2$ classifieurs binaires en confrontant chacune des M classes. En phase de test, l'échantillon à classer est analysé par chaque classifieur et un vote majoritaire permet de déterminer sa classe. Si l'on note x_t l'échantillon à classer et $h_{ij}(.)$ le classifieur SVM séparant la classe C_i et la classe C_j et renvoyant le label attribué à l'échantillon à classer, alors le label attribué à x_t peut formellement se noter $Card(\{h_{i,j}(x_t)\} \cap \{k\}; i, j, k \in [1, M], i < j$. C'est la classe qui sera le plus souvent attribuée à x_t quand il aura été analysé par tous les h_{ij}.

4. Une représentation séparatrice basée sur la distance

Il existe quelques problèmes dans les représentation de PCA et LDA. On citera par exemple le manque de séparabilité optimale dans les données à dimension réduite mais aussi le

calcul du nombre total de vecteurs propres à retenir pour la transformation. Afin de résoudre ces problèmes. Une nouvelle représentation appelée DBSR que l'on va exposé (Distance-based separator representation) a été proposée dans [22].

4.1. Principe

Avant la description, la figure 3.2 illustre la motivation première de la méthode. Dans la partie (a) de celle-ci, supposons que les deux classes peuvent être séparées linéairement, une ligne droite pourrait les diviser. Lorsqu'une autre classe est rajoutée au plan, une troixième lignes est nécessaire pour ce faire. Donc, quand il y a « k » classes, il faudra « k » lignes pour les séparer. En utilisant l'algorithme one versus one SVM sur l'exemple de la figure 3.2. (a2) et (b2) sont les résultats de l'utilisation de la représentation PCA, alors que (a3) et (b3) sont ceux de DBSR.

Nous pouvons voir à partir de figure3.3 (b2) que l'une des lignes séparatrices du classifieur PCA est très proches de l'une des classes. Bien que PCA arrive à séparer correctement les classes dans ces deux simples exemples, si le nombre de classes augmente, elle risque de produire des erreurs.

Notre but est de fournir une meilleure habilité à la discrimination et d'avoir des résultats valides même dans cas complexes en utilisant DBSR.

Dans la figure3.3. Nous illustrons une représentation à 3 classes par DBSR, où C_1, C_2, C_3 sont les différentes classes et y_1, y_2, y_3 les hyperplan .le vecteur échantillon à trois classes $\vec{x}$ est représenté : $\vec{x} = (f_1, f_2, f_3)^T$

où f_i est la distance euclidienne du i^{ème} séparateur

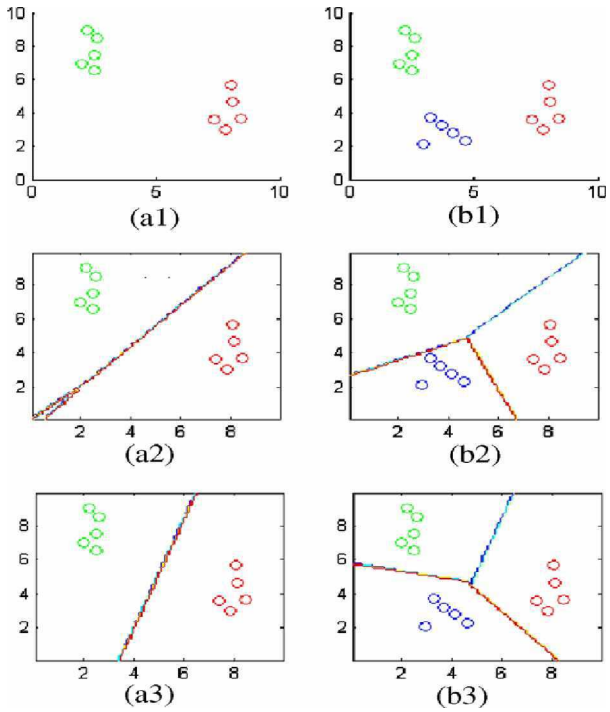


Fig.3.2. Exemple d'une classification: (a). 2class case, (b) 3 class case

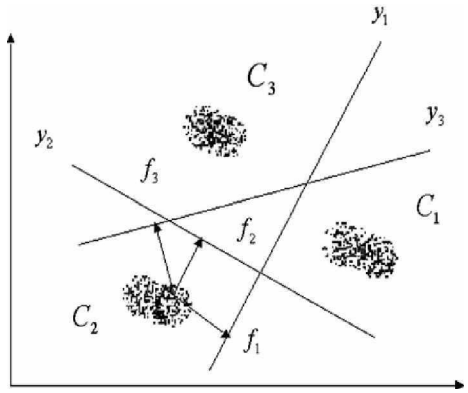


Fig.3.3. La nouvelle représentation $\vec{x} = (f_1, f_2, f_3)^T$

Chaque hyperplan one versus all multi class SVM satisfait les condition des séparateurs. on peut l'utiliser pour construire la nouvelle structure de manière à ce que la représentation soit $\vec{x} = (f_1, f_2, ..., f_k)^T$ dans le problème à k classes, alors

$$f_i = (w^i)^T \phi(\vec{x}) + b^i \tag{3.24}$$

Notons que $f_i = (w^i)^T \phi(\vec{x}) + b^i = 0$ est le i^{ème} hyperplan.

Un échantillon $\vec{x}$ de la classe « i », sert tous les SVMs. On obtient un nouveau vecteur de distance $\vec{f}_d(\vec{x}) = (f_d^1, f_d^2, ..., f_d^k)^T \in R^k$ qui présente l'échantillon $\vec{x}$ dans l'espace R^k. Finalement le vecteur de sortie $\vec{Y}(\vec{x}) = (y_1, y_2, ..., y_k)^T = \vec{f}_d(\vec{x}).W$ où W est une matrice de

poids de dimension k x k, y_i=1 et $y_{i\neq j} = 0$. Puisque $\vec{Y}(\vec{x})$ et $\vec{f}_d(\vec{x})$ sont connus, la matrice de poids peut être obtenue par

$$W = F_d^+ Y = \left(F_d^T F_d\right)^{-1} F_d^T Y \tag{4.25}$$

où F_d et Y sont dérivés par tout les échantillons, et F_d^+ et le pseudo inverse de F_d.

Les transformations traditionnelles transposent les vecteurs d'image originaux à un autre espace de dimension plus grande avec un point origine fixé. Mais la méthode proposée les transpose dans un système de coordonnées cinétiques dont le degré est relié au nombre de classes.

4.2. L'algorithme

Il y a trois étapes dans le procédé [22], la première étape est la phase constructrice, la seconde étape est la phase d'apprentissage, et la dernière phase est celle des tests.

L'algorithme va se déroulé comme suit :

- *La construction de tous les SVMs* : La dimension des échantillons est réduite de R^m vers R^n par une PCA. Le vecteur $\vec{x} \in R^n$ sera l'entrée du procédé. En supposant que chaque classe a « l » échantillon dans l'ensemble d'apprentissage, pour la i^{eme} SVM, M_i^1, tout les échantillons de classe i et n'importe quels autres de toutes les autres classes $j \neq i$ sont utilisés afin d'obtenir l'ensemble d'apprentissage T_i^1.
- *L'apprentissage* : tout ensemble M_i^1 sera entraîné.
- *Le test*: tout ensemble d'apprentissage sera testé,
    - s'il existe des échantillons mal classé, on les ajoute à T_i^1 avec des labels négatifs pour avoir un nouvel ensemble d'apprentissage T_i^2, donc une nouvelle SVM M_i est obtenue.
    - Sinon $M_i^1 = M_i$. Il est à noter que M_i peut améliorer sa généralisation à travers une telle procédure, et les SVM est entraînée par touts les échantillons d'apprentissage une seule fois. Après la création de tout le $M_i^{'}s$, la sortie cible est créée.

En plus du fait que cette structure garde les mêmes règles que le système, un autre intérêt majeur est que la dimension du nouvel espace correspond exactement au nombre de classes.

L'objectif de la phase d'apprentissage est de calculer la matrice de poids. Après l'apprentissage de tout l'ensemble, le système génère :

$$F_d W = Y \tag{4.26}$$

$$\text{où } F_d = \left(\overrightarrow{f_d^1}, \overrightarrow{f_d^2}, ..., \overrightarrow{f_d^{k+1}} \right) \tag{4.27}$$

$$\text{et } Y = \left[\vec{Y}(\vec{x}_{1,1}), \vec{Y}(\vec{x}_{1,2}), ...\vec{Y}(\vec{x}_{k,l}) \right] \tag{4.28}$$

Le vecteur de sortie ne contiendra qu'un seul « 1 » correspondant au nœud qui identifiera la classe désirée. On obtient alors $\vec{Y}(\vec{x}_{i,m}) = (y_1, y_2, ..., y_k)$ où $y_i = 1$ et $y_{i \neq j} = 0$. Une fois F_d et Y connus, la matrice de poids sera obtenu par l'équation (4.25) alors le système sera complètement établi.

La phase de test, pour échantillon inconnu $\vec{x}$, le système donnera un vecteur $\vec{Y}(\vec{x}\ )$ où l'emplacement de la valeur maximum des nœuds de sortie est donné par l'équation:

$$\text{classe de } \vec{x} = \arg \max_{i=1,2,...,k} y_i(\vec{x}) \tag{4.29}$$

et indiquera la classe reconnue et souhaitable.

Dans ce simple cas, le séparateur linéaire n'est pas nécessaire. Une extension de DBSR utilisant les SVM non linéaires. Mais la fonction de distance est la même.

Introduction

Dans ce chapitre nous allons exposer une méthode utilisée pour la classification d'images puis comparer les résultats obtenus sans et avec l'étape de prétraitement. Il s'agit dans notre cas d'une réduction de dimensions. Pour que l'algorithme de classification obtienne une convergence plus rapide ainsi qu'un regroupement plus pertinent des pixels de chaque classe.

1. L'analyse en composantes principales

Nous avons appliqué l'analyse en composantes principales sur différentes images pour récupérer les informations les plus pertinentes de celles-ci.

Il s'agira de faire une projection dans un nouvel espace [23] où les différentes directions (les nouveaux axes) seront les vecteurs propres de la matrice de covariance de l'image elle-même. De ces axes qui sont orthonormaux, nous ne garderons que ceux dont les valeurs propres sont les plus importants. Et donc ceux qui contiennent l'information le plus pertinente, puisque la sélection peut se faire en minimisant l'erreur quadratique donnée par l'équation ci-dessous [24] :

$$\varepsilon^2(l) = E\left\{\|x - x(l)\|^2\right\} = E\left\{\left\|\sum_{i=l+1}^{m} y_i * \alpha_i\right\|^2\right\} = \sum_{i=l+1}^{m} E\left\{y_i^2\right\} \tag{4.1}$$

où $x \in R^m$ est un vecteur aléatoire, et $y_1, y_2, ..., y_m \in R$ sont les « m » composantes principales, les exigences de PCA exprime y_1 comme une combinaison linéaire du vecteur aléatoire original, donc :

$$y_1 = \sum_{i=1}^{m} \alpha_{i1} * x_i = \alpha_1^T * x \tag{4.2}$$

où $\alpha_1 = (\alpha_{11}, \alpha_{21}, ..., \alpha_{m1})^T$ est un vecteur de constantes.

Après un certain nombre de calcul et une étape d'optimisation par les équation de Lagrange sur les différentes variances de la première composante principale, il sera prouvé que α_1 correspond au premier vecteur propre de la matrice de covariance des données originales, et que le paramètre de l'équation de Lagrange λ_1 est la valeur propre correspondant à ce vecteur propre. En poursuivant à faire les calculs pour le reste des composantes principales, l'erreur s'écrira :

$$\varepsilon^2(l) = \sum_{i=l+1}^{m} \lambda_i \tag{4.3}$$

L'erreur sera choisie empiriquement afin de donner le nombre d'axes optimal pour une meilleure représentation des données avec des pertes insignifiantes, l'algorithme sera appliqué sur une image de synthèse puis sur des images réelles.

Le nombre optimal de composantes principales à garder est estimé en utilisant l'équation (4.4) que les auteurs ont donnés dans [25]. Ce même paramètre a été utilisé comme nombre optimal de classes sous

certaines hypothèses de redondance dans les images d'origine pour accélérer le processus de classification sur des données réduites et restructurées:

$$k = \arg\min\left\{\alpha.n \le \sum_{l=1}^{k}\delta_l^{\,2}\right\} \qquad \text{où } \|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}a_{ij}^2 = n = \sum_{l=1}^{rank(A)}\delta_l^{\,2} \tag{4.4}$$

où δ_i est la ième valeur singulière de la matrice de covariance et α est paramètre ajustable (le « F » correspond a la norme de Frobenius).

2. La décomposition en valeurs singulières

Elle a été exploitée de la même manière que PCA afin d'avoir une représentation de l'image ne contenant que l'essentiel des information (dans un but de compression) pour évité la redondance et ainsi accéléré le processus de classification, ses propriétés seront aussi utilisée pour le calcule du nombres optimale de classes pour la classification sous un nombre d'hypothèse imposées sur les image initiales, et ce en minimisant l'erreur quadratique.

Donc, par souci de trouver une base orthonormée optimale pour la nouvelle représentation des données, plusieurs manœuvres de calcul et d'optimisation ainsi que les équations de Lagrange ont été utilisées dans [24] pour arriver a prouver que l'erreur à minimiser dans une SVD est donnée par l'équation qui suit :

$$\varepsilon^2(l) = tr(\Lambda) = \sum_{j=l+1}^{m}\sigma_j^{\,2} \tag{4.5}$$

où Λ est une matrice diagonale dont les éléments sont les valeurs singulières σ_j portées au carrée de la matrice de données *« X »*. Et ce ci est prouvé uniquement lorsque les vecteurs singuliers à droite de la matrice de données transposée *« X^T »* sont pris comme base de vecteurs orthonormée et c'est à ce moment que l'erreur est la sommation des (m-l) valeurs singulières de cette même matrice *« X^T »*.

2.1. Propriétés de la décomposition en valeurs singulière

Nous allons mettre en évidence quelques unes des propriétés de la décomposition en valeur singulières de matrices qui ont une ou plusieurs colonnes identiques afin de pouvoir les exploiter pour accélérer la convergence de l'algorithme de classification [25] :

Propriété-1- : Si « *A* » est une matrice de rang « *r* » avec une décomposition en valeurs singulières telle que :

$$A = S * U * V^T \tag{4.6}$$

alors, la distance Euclidienne entre deux quelconques vecteurs de « A » est égale à la distance Euclidienne pondérée entre les colonnes correspondantes de « V^T» où le poids de pondération est la valeur singulière σ_k ,ce qui veut dire que:

$$\left\|a_i - a_j\right\|^2 = \sum_{k=1}^{r} \sigma_k^2 (v_{ik} - v_{jk})^2 \tag{4.7}$$

où $A = [a_1, a_2, ..., a_n]$ et $V = [v_1, v_2, ..., v_n]$.

Propriété-2- : Supposons que « A » soit une matrice [m x n] de rang « r », et que $V = [v_1, v_2, ..., v_n]$ soit la matrice singulière à droite de « A » de dimension [n x n] et $V^T = [\phi_1, \phi_2, ..., \phi_n]$, alors, les « r » premiers éléments de la i^{ème} et de la j^{ième} colonnes de « V^T» sont égales : $\phi_{ki} = \phi_{kj}$ *pour k = 1,2,...,r*

Propriété-3- : Supposons que $A_1 = [a_1, a_2, ..., a_n]$ soit une matrice de dimension [m *x* n] où $m \geq n$, de rang « n », c'est-à-dire que les colonnes de A_1 sont linéairement indépendantes. a_0 est vecteur de longueur m qui lui aussi est linéairement indépendant des colonnes a_i de A_1 ; et A_2 une matrice de dimension [m *x* d] de rang « 1 » formée par la répétition de a_0 « d » fois : $A_2 = [a_0, a_0, ..., a_0]$.

Supposons aussi que « A » soit une matrice de dimension [m x (n+d)], formée par A_1 et A_2 comme suit : $A = \left[A_1 | A_2\right]$ avec une SVD tel que $A = S * U * V^T$ .On suppose encore que V^T est partitionnée à son tour comme suit :

$$V^T = \begin{bmatrix} \phi_0 & \phi_1 \\ \phi_3 & \phi_2 \end{bmatrix} \tag{4.8}$$

où ϕ_0 et ϕ_3 ont n colonnes comme c'est le cas pour A_1 ; ϕ_1 et ϕ_2 ont d colonnes : le nombre de celle de A_2 .ϕ_0 et ϕ_1 ont n+1 lignes, qui est le rang de A.

En divisant V^T comme cité ci-dessus, on aura :

$$\begin{aligned} \phi_3 &= 0 \\ \left[1, 1, ..., 1\right] * \phi_2^T &= 0 \end{aligned} \tag{4.9}$$

C'est-à-dire que la somme des éléments de chaque ligne de ϕ_2 est nulle.

Propriété-4- : Nous gardons les mêmes suppositions que celles de la ***propoiété-3-***, puis ajoutons que V: la matrice singulière de droite est partitionnée de la même manière que dans cette propriété avec $\phi_1 = \left[w_0, w_0, ..., w_0\right]$, il en résulte que la norme Euclidienne portée au carrée de w_0, est : $\left\|w_0\right\|^2 = 1/d$ où d est le nombres de fois que w_0 se répète dans ϕ_1 .

Nous avons tenté de modifier quelques étapes de la procédure proposée dans [25] afin de l'utiliser dans notre travail, l'algorithme modifié se résume dan les étapes qui suivent :

- La normalisation de l'image en niveau de gris.
- Sa décomposition en valeurs singulières.
- La génération d'une nouvelle matrice $\hat{A}$ de telle manière à ce que ses colonnes soient $[\sigma_l v_{il}]^T$ *pour* $l = 1,2,...,k$ classées dans l'ordre ascendant de leur norme.
- La reconstruction de l'algorithme des c- moyennes floues sur la nouvelle matrice, avec un choix empirique du nombre de classes. Le sélection, quand à elle du nombre « l » de colonnes de $\hat{A}$ -qui est, en quelque sorte, le nombre de valeur singulière à garder- se fait en utilisant l'inégalité suivante :

$$k = \arg\min\left\{\alpha * \|A\|_F^2 \le \sum_{l=1}^{k} \delta_l^{\,2}\right\}$$
$$\|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2 = \sum_{l=1}^{rank(A)} \delta_l^{\,2} \qquad (4.10)$$

  Nous avons modifié le paramètre α que l'on peut considérer comme un pourcentage de la variance totale de l'image.
- la restitution de l'image après classification.

3. L'analyse discriminante linéaire

Afin de comparer nos résultats des approches non supervisées avec l'une de celles qui ne le sont pas, on a tenté d'implémenter LDA, et d'utiliser son coté discriminatif sur les pixels de l'image un par un. Après une classification initiale par FCM, nous avons essayé de calculer les matrices de covariance inter et intra classe selon les formules données dans le chapitre III. Ses éléments étaient les variances du niveau de gris de chaque pixel par rapport à tout les autres. Le problème de dimension des matrices s'est posé. La saturation de la mémoire virtuelle ne nous a pas permis d'exploiter cette méthode jusqu'au bout.

4. FCM

Nous avons essayé de faire une classification directement ; sans prétraitement; puis nous avons comparé celle ci avec une classification où les images ont subi par une phase de réduction :

- du point de vue temps d'exécution pour aboutir à la convergence
- et du point de vue de la qualité de la classification.

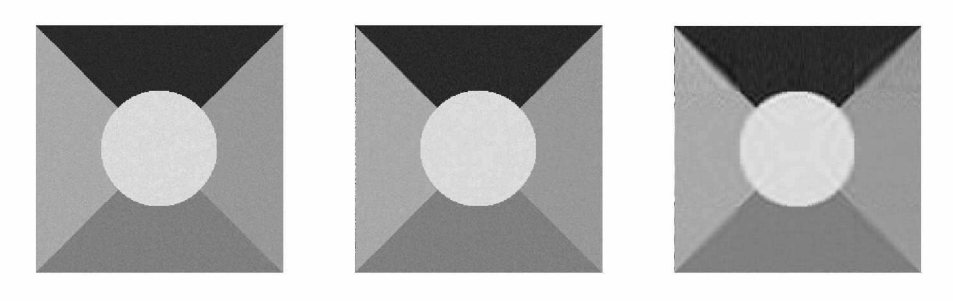
Pour la phase de défuzzification, nous avons choisi la méthode par maximum d'appartenance.

5. Les résultats sur les images

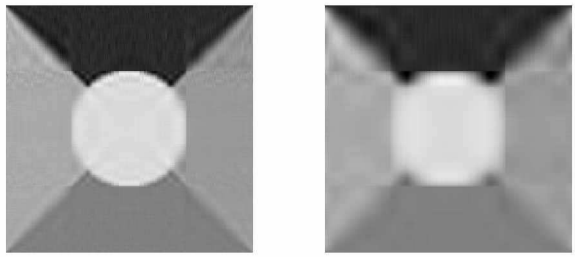
5.1. Les images de synthèse

a. Traitement par SVD

Les images sont décomposées par une SVD puis tronquées à « l » valeurs singulière pour être réduites avec plusieurs valeur de α dans l'équation $k = \arg\min\left\{\alpha.n \le \sum_{l=1}^{k} \delta_l^2\right\}$



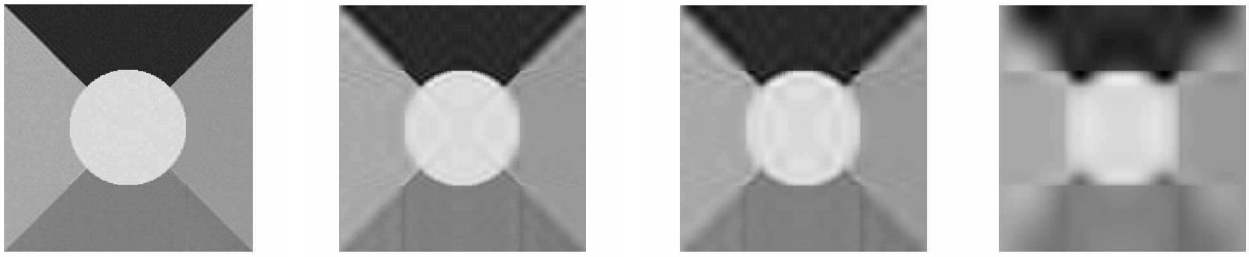
a: image originale **b:** $\alpha = 0.99$ et l=106 **c:** $\alpha = 0.90$ et l=15



d: $\alpha = 0.89$ et l=13 **e:** $\alpha = 0.8$ et l=05

Fig.4.1. Reconstruction d'une une image de synthèse par une SVD

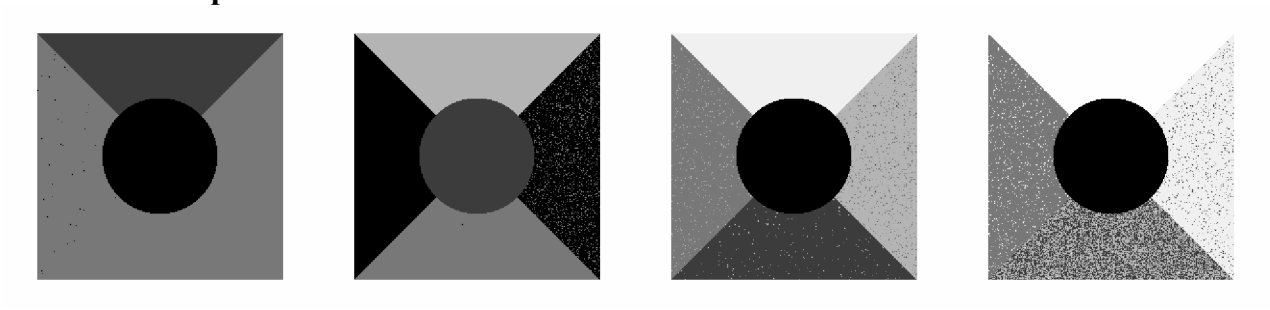
b. Traitement par PCA



a: $\alpha = 0.99$ et l= 41 **b:** $\alpha = 0.97$ et l= 11 **c:** $\alpha = 0.96$ et l= 08 **d:** $\alpha = 0.90$ et l= 03

Fig.4.2. Reconstruction d'une une image de synthèse par une PCA

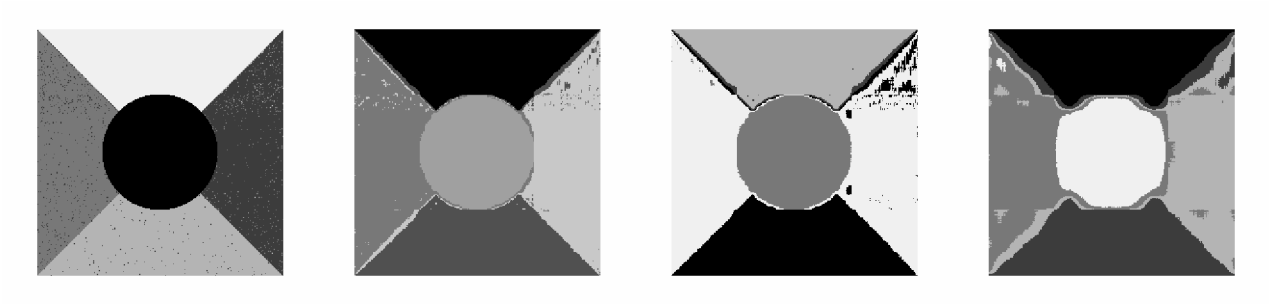
c. Traitement par FCM



a: C=3 **b:** C=4 **c:** C=5 **d:** C=6

Fig.4.3. Classification d'une image de synthèse par une FCM

d. Traitement par SVD+FCM

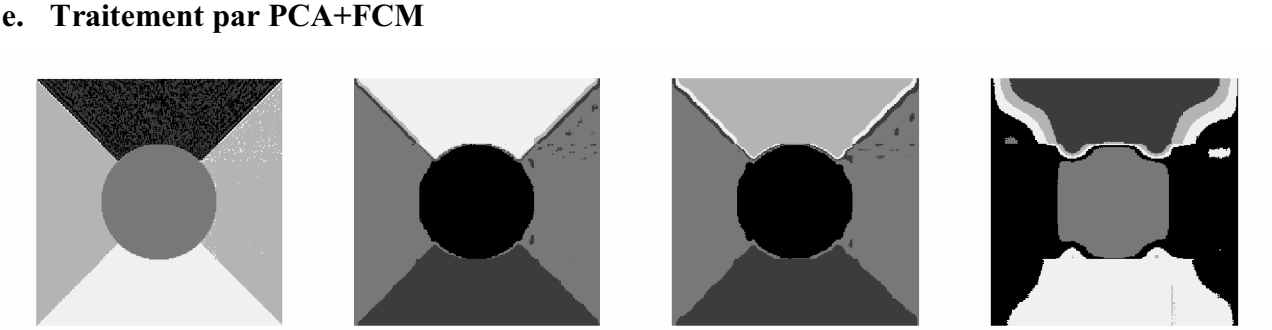


C=5

a: $\alpha = 0.99$ et l=106 **b:** $\alpha = 0.90$ et l=15 **c:** $\alpha = 0.89$ et l=13 **d:** $\alpha = 0.80$ et l=05

Fig.4.4. Classification d'une image de synthèse après une SVD+FCM

e. Traitement par PCA+FCM



C=5

a: $\alpha = 0.99$ et l=41 **b:** $\alpha = 0.97$ et l=11 **c:** $\alpha = 0.96$ et l=08 **d:** α=0.90 et l=03

Fig.4.5. Classification d'une image de synthèse par une PCA+FCM

	L'erreur maximum tolérée	La dimension	Le nombre de classes supposé	Le rapport nbr de classes / dimension
SVD	90%	15	05	=1/3
PCA	97%	11		≈1/2

Tab.4.1. Rapport du nombre de classes à la dimension sur une image de synthèse.

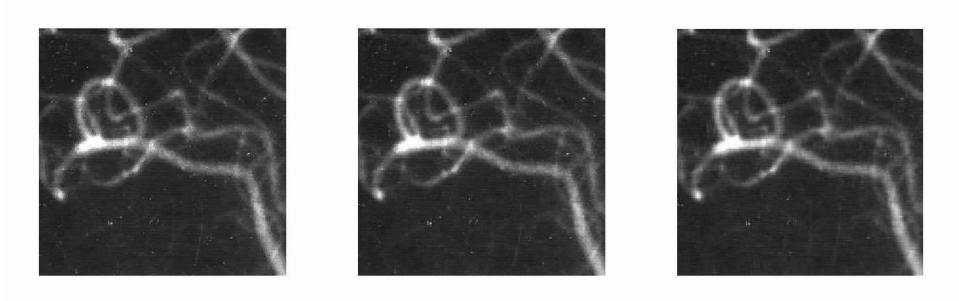
Sur le tab.4.1. nous avons reporté les résultat obtenus sur une image de synthèse afin de définir le rapport entre le nombre de classes et la dimension de l'espace de projection de l'image, vue que sur ce type d'images le nombre de classe est connue. Il en est ressorti que ce rapport était :

- Au tiers du nombre de valeurs singulières utilisées pour reconstitution d'une image de bonne qualité après l'application d'une décomposition en valeurs singulières.
- A la moitié du nombre de composantes principales pour une reconstitution non altérée d'une image après une analyse en composantes principales.

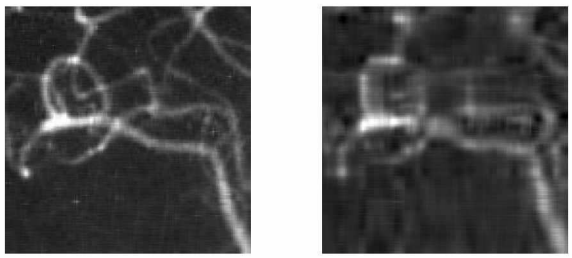
Pour le cas d'images de synthèse, où le nombre de classes est connu d'emblée nous n'avons fait que situer la détérioration acceptable de l'image sur une marge maximum d'erreur tolérée. Le diviseur obtenu pour SVD est celui obtenue pour PCA en y additionnant «1».

5.2. Les images médicales

a. Traitement par SVD

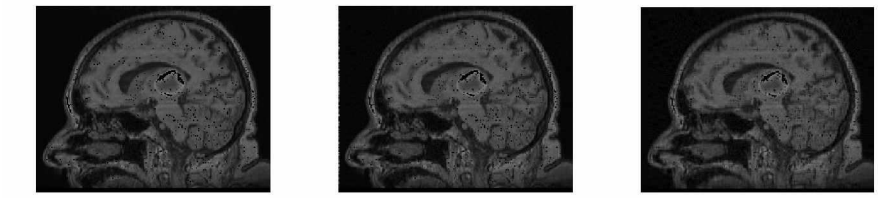


a: image originale **b:** $\alpha = 0.99$ et l=88 **c:** $\alpha = 0.97$ et l=45

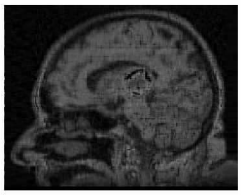
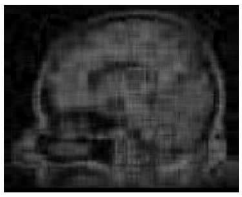


d: $\alpha = 0.95$ et l=27 **e:** $\alpha = 0.75$ et l=08

Fig.4.6. Reconstruction d'une une image d'angiographie après une SVD



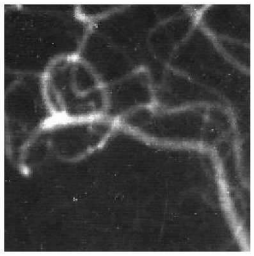
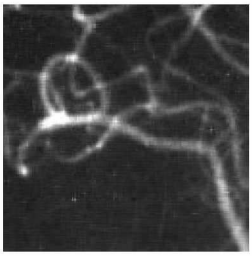
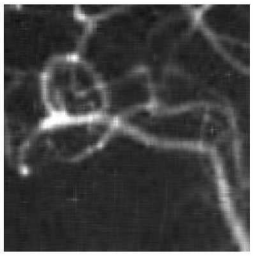
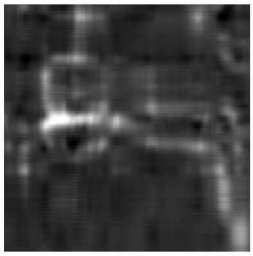
a: image originale **b:** $\alpha = 0.99$ et l=95 **c:** $\alpha = 0.95$ et l=54

d: $\alpha = 0.91$ et l=33 **e:** $\alpha = 0.75$ et l=10

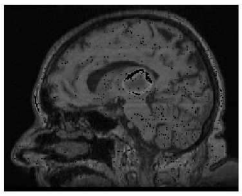
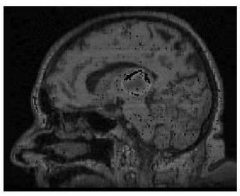
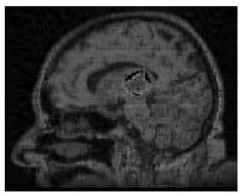
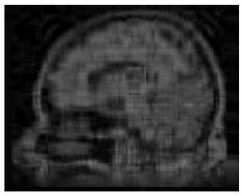
Fig.4.7. Reconstruction d'une une image de crâne après une SVD

b. Traitement par PCA

a: $\alpha = 0.99$ et l= 76 **b:** $\alpha = 0.97$ et l=32 **c:** $\alpha = 0.95$ et l= 20 **d:** $\alpha = 0.75$ et l= 06

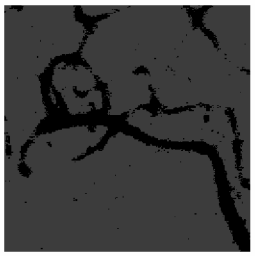
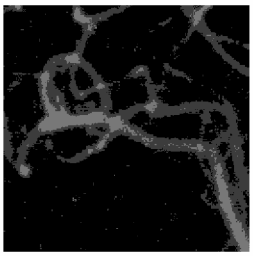
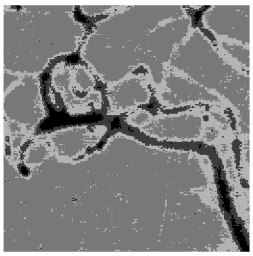
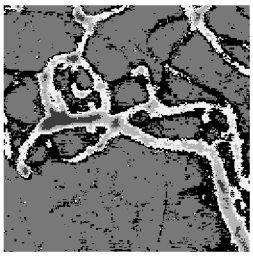
Fig.4.8. Reconstruction d'une une image d'angiographie après une PCA

a: $\alpha = 0.99$ et l= 96 **b:** $\alpha = 0.95$ et l= 54 **c:** $\alpha = 0.88$ et l= 28 **d:** $\alpha = 0.75$ et l= 11

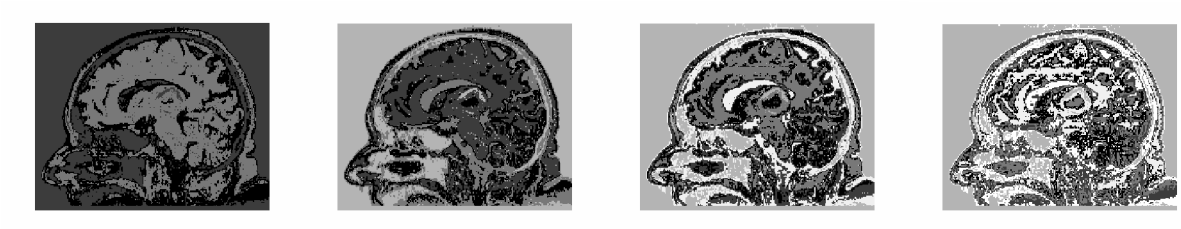
Fig.4.9. Reconstruction d'une une image de crâne après une PCA

c. Traitement par FCM

a: C=2 **b:** C=3 **c:** C=4 **d:** C=5

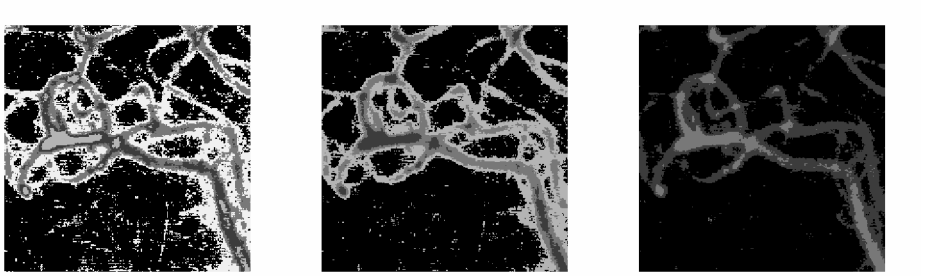
Fig.4.10. Classification d'une image d'angiographie après une FCM**.**



a: C=3 **b:** C=4 **c:** C=5 **d:** C=6

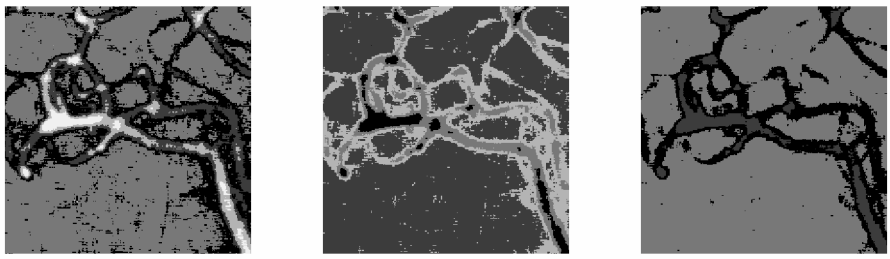
Fig.4.11. Classification d'une image de crâne après une FCM

d. Traitement par SVD+FCM



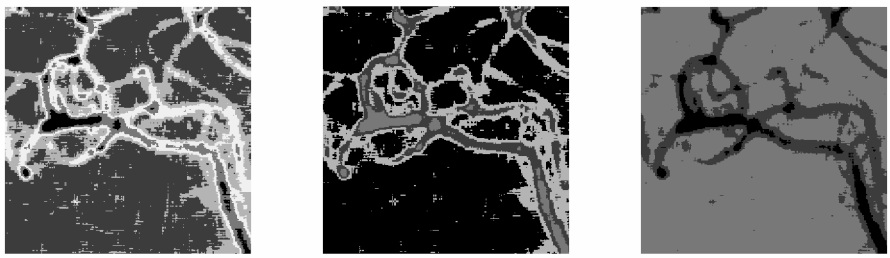
α =0.99 l=88

a: C=5 **b:** C=4 **c:** C=3



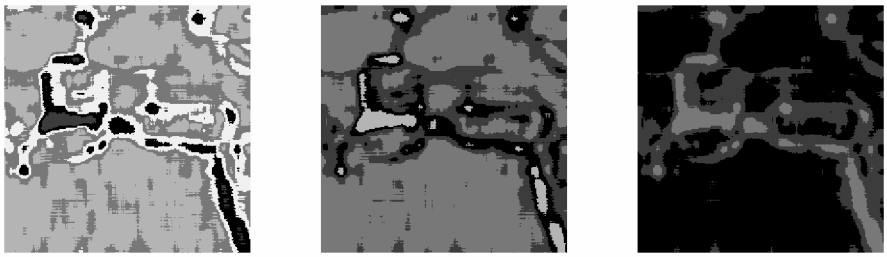
α =0.97 l=45

d: C=5 **e:** C=4 **f:** C=3



α =0.95 l=27

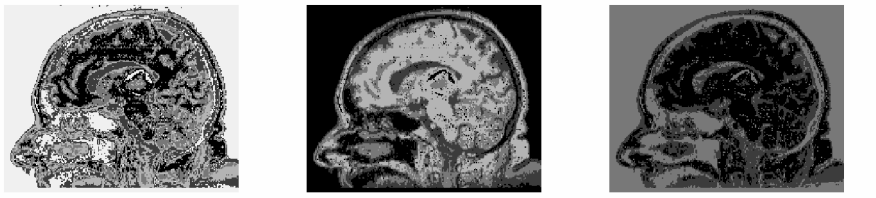
g: C=5 **h:** C=4 **i:** C=3



α =0.75 l=08

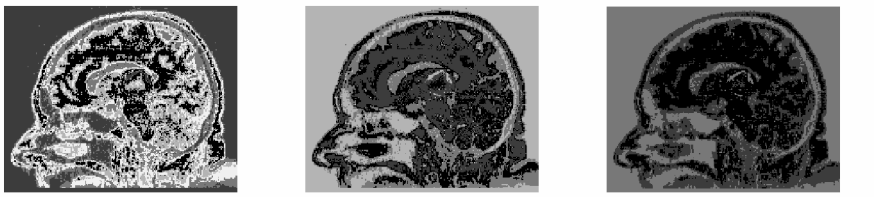
j: C=5 **k:** C=4 **l:** C=3

Fig.4.12. Classification d'une image d'angiographie après une SVD+FCM



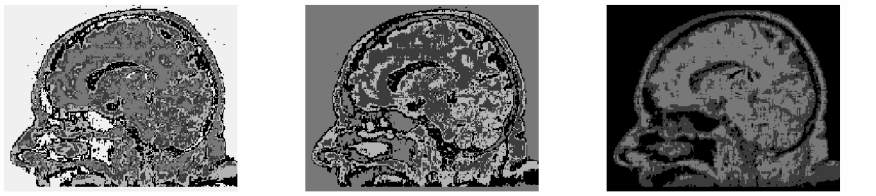
α =0.99 l=95

a: C=5 **b:** C=4 **c:** C=3



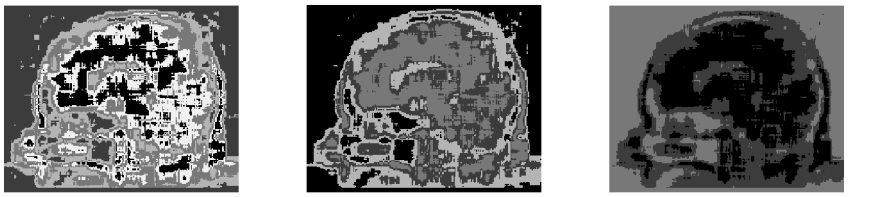
α =0.95 l=51

d: C=5 **e:** C=4 **f:** C=3



α =0.91 l=33

g: C=5 **h:** C=4 **i:** C=3

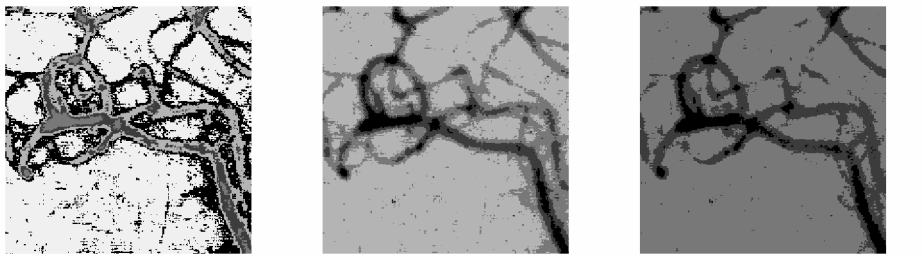


α =0.75 l=10

j: C=5 **k:** C=4 **l:** C=3

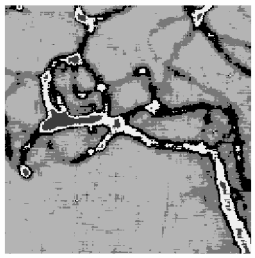
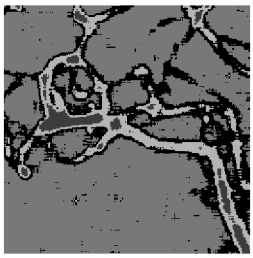
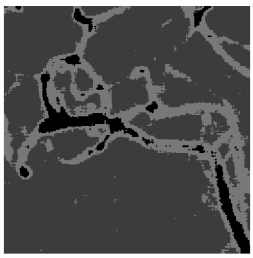
Fig.4.13. Classification d'une image de crâne après une SVD+FCM

e. Traitement par PCA+FCM



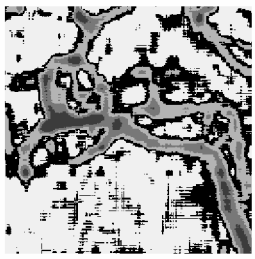
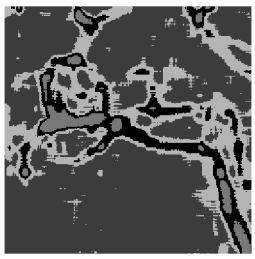
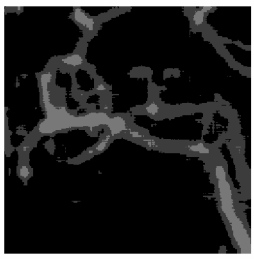
α =0.99 l=76

a: C=5 **b:** C=4 **c:** C=3

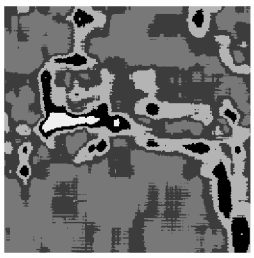
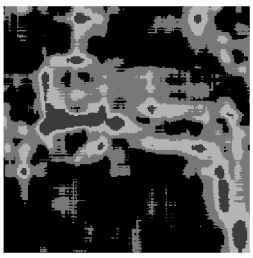
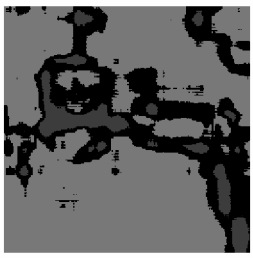
α =0.97 l=32

d: C=5 **e:** C=4 **f:** C=3

α =0.95 l=13

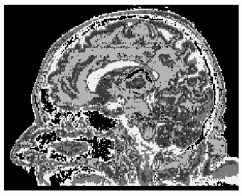
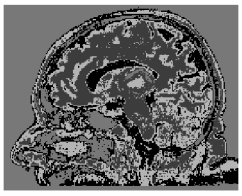
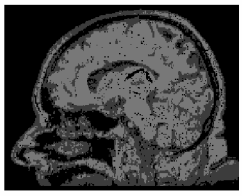
g: C=5 **h:** C=4 **i:** C=3

α =0.75 l=06

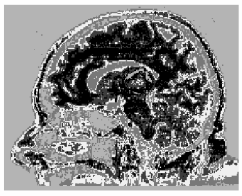
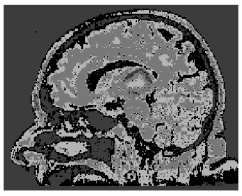
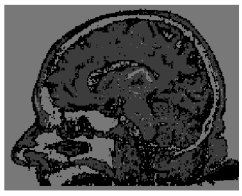
j: C=5 **k:** C=4 **l:** C=3

Fig.4.14. Classification d'une image d'angiographie après une PCA+FCM

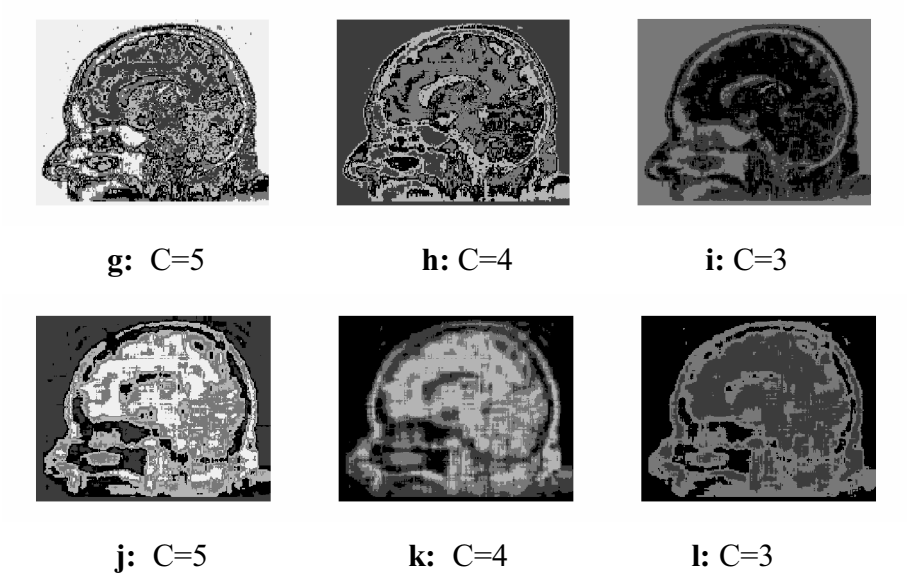
α =0.99 l=96

a: C=5 **b:** C=4 **c:** C=3

α =0.95 l=54

d: C=5 **e:** C=4 **f:** C=3



α =0.88 l=28

g: C=5 **h:** C=4 **i:** C=3

α =0.75 l=11

j: C=5 **k:** C=4 **l:** C=3

Fig.4.15. Classification d'une image de crâne après une PCA+FCM

	L'image	L'erreur maximum tolérée	La dimension	Le nombre de classes supposé	Le rapport nbr de classes / dimension
SVD	Angiographie	95%	27	04	≈1/6
	Crâne	91%	33	05	≈1/6
PCA	Angiographie	95%	20	04	≈1/5
	Crâne	88%	28	05	≈1/5

Tab.4.2. Rapport du nombre de classes à la dimension sur les images médicales

Le Tab.4.2. comporte les résultats obtenus pour les images médicales. Nous avons appliqué une décomposition en valeurs singulières ainsi qu'une analyse en composantes principales avec des taux d'erreur variables afin de pouvoir estimer le nombre de classes pour une détérioration acceptable maximale. Nous avons aussi procédé à une FCM en variant le nombre de classes afin d'estimer celui du meilleur résultat. Enfin nous avons fait le rapport entre celui-ci et le nombres de valeurs singulières à partir duquel on obtient la bonne reconstitution ; puis ce même rapport, mais par rapport aux nombres de composantes principales prises en compte toujours dans but de la reconstruction optimale.

Finalement, le rapport est passé du tiers au sixième pour SVD et de la moitié au cinquième pour PCA lorsqu'il s'agit d'images médicales.

Nous avons aussi appliqué une classification par FCM après une réduction préalable des images dans le but d'accélérer le processus. Le résultat est assez bon lorsqu'on ne descend pas au dessous de l'erreur considérée et tolérée pour la reconstruction optimale, et même que les classe apparaissent mieux après la dite réduction.

5.3. Les images de scène

a. Traitement par SVD



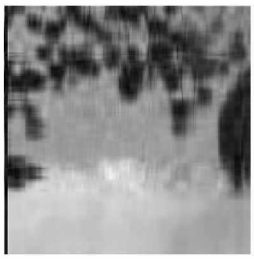
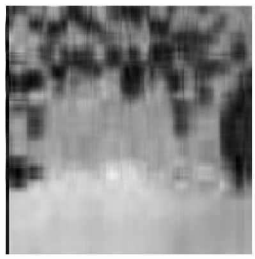
a: image originale **b:** $\alpha = 0.99$ et l=81 **c:** $\alpha = 0.95$ et l=31


d: $\alpha = 0.90$ et l=17 **e:** $\alpha = 0.80$ et l=08

Fig.4.16. Reconstruction d'une image de scène après une SVD



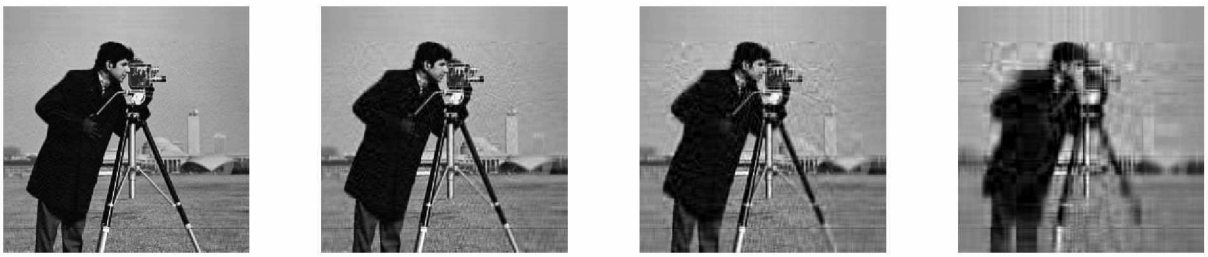
a: image originale **b:** $\alpha = 0.99$ et l=50 **c:** $\alpha = 0.97$ et l=26

d: $\alpha = 0.91$ et l=11 **e:** $\alpha = 0.85$ et l=08

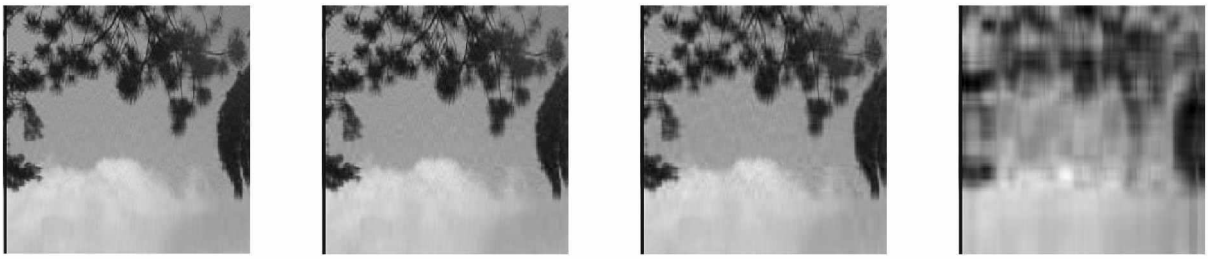
Fig.4.17. Reconstruction d'une image de paysage après une SVD

b. Traitement par PCA



a: $\alpha = 0.99$ et l= 72 **b:** $\alpha = 0.97$ et l= 42 **c:** $\alpha = 0.94$ et l=26 **d:** $\alpha = 0.85$ et l=11

Fig.4.18. Reconstruction d'une image de scène après une PCA



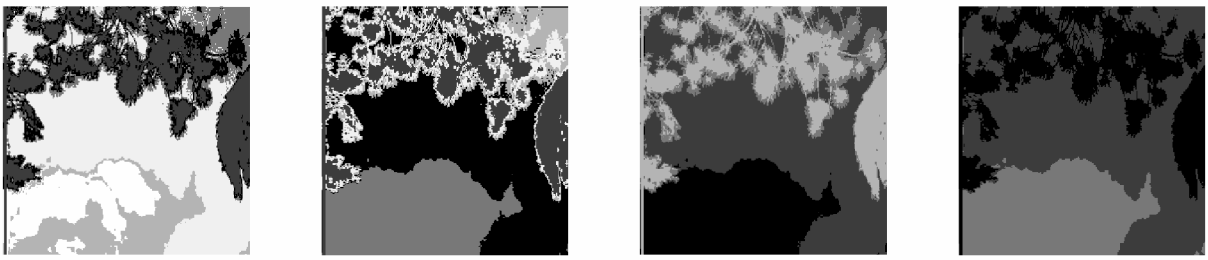
a: $\alpha = 0.99$ et l= 39 **b:** $\alpha = 0.985$ et l=31 **c:** $\alpha = 0.975$ et l= 21 **d:** $\alpha = 0.85$ et l= 05

Fig.4.19. Reconstruction d'une image de paysage après une PCA

c. Traitement par FCM

a: **C=6** b: **C=5** c: **C=4** **e:** C=3

Fig.4.20. Classification d'une image de scène par une FCM

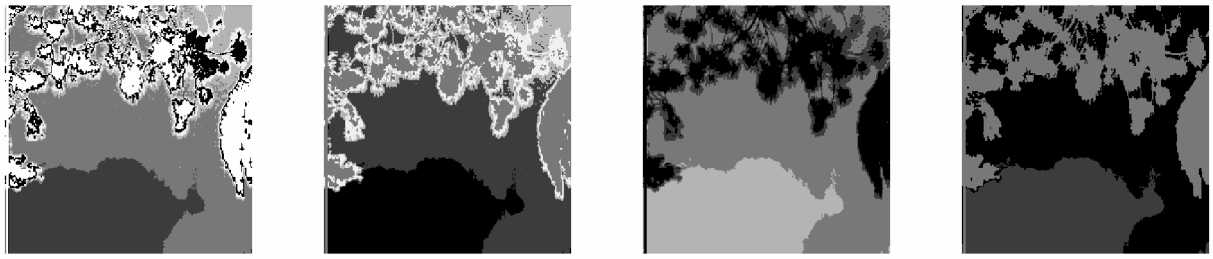


a: C=6 **b:** C=5 **c:** C=4 **e:** C=3

Fig.4.21. Classification d'une image de paysage par une FCM

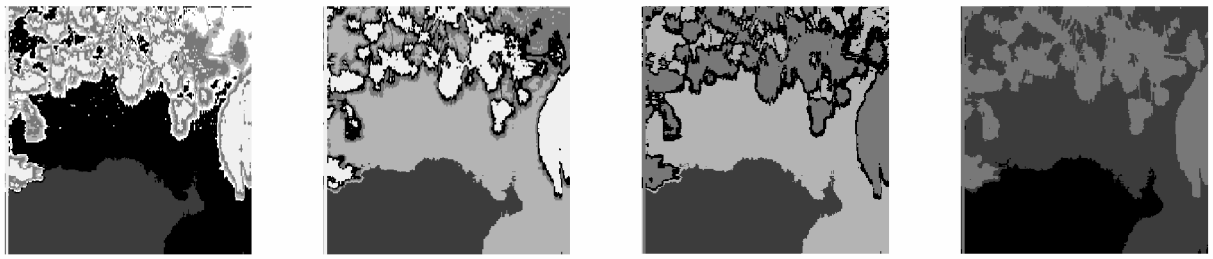
d. Traitement par SVD+FCM

Fig.4.22. Classification d'une image de scène par une SVD+ FCM



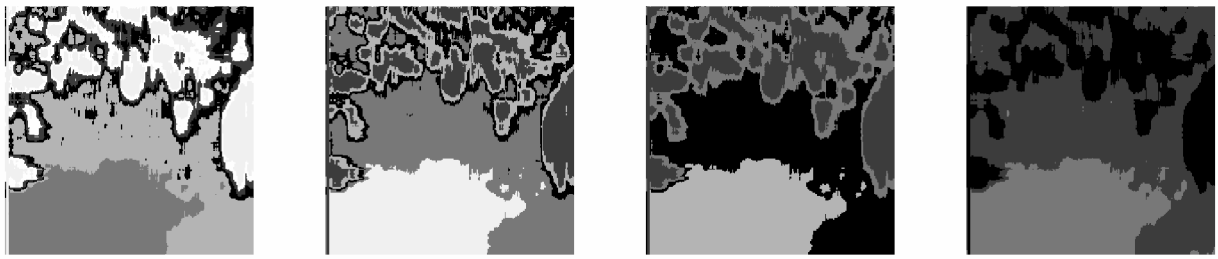
α =0.99 l=50

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3



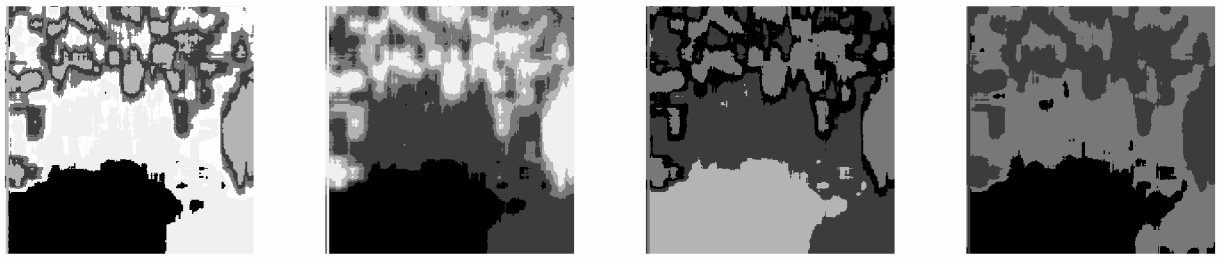
α =0.97 l=26

e: C=6 **f:** C=5 **g:** C=4 **h:** C=3



α =0.91 l=11

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3

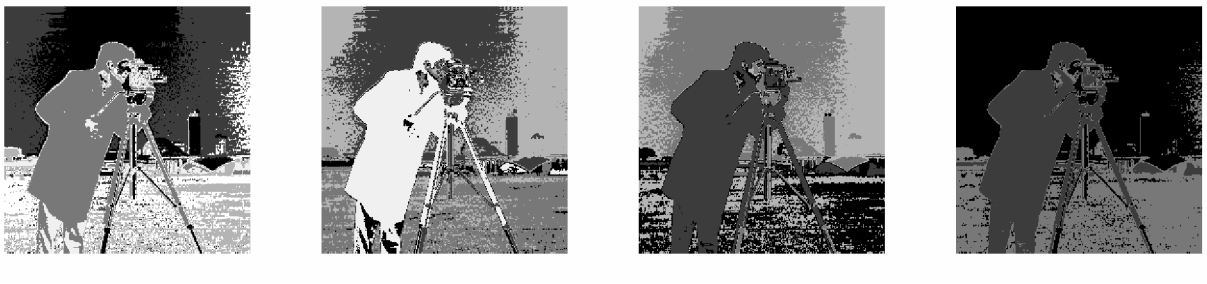


α =0.85 l=08

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.23. Classification d'une image de paysage par une SVD+FCM

e. Traitement par PCA+FCM

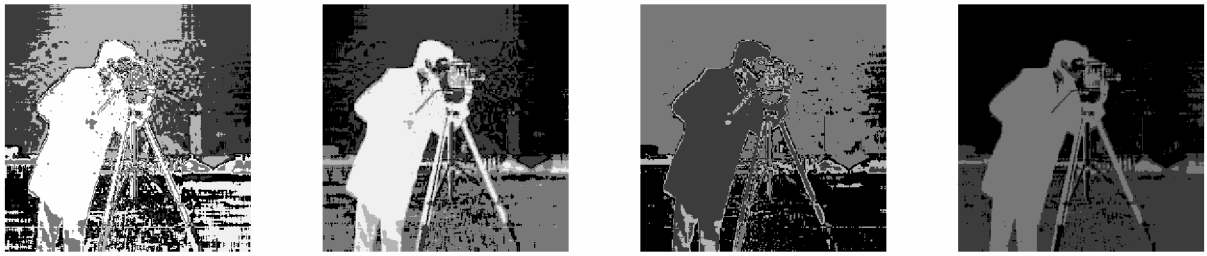


α =0.99 l=72

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3

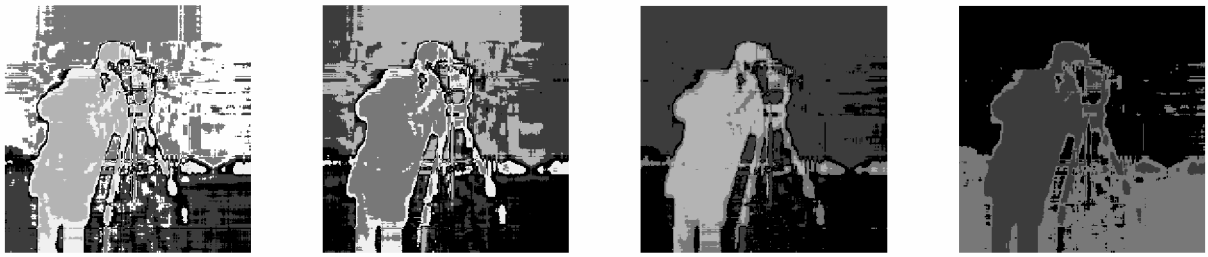
α =0.97 l=42

e: C=6 **f:** C=5 **g:** C=4 **h:** C=3



α =0.94 l=26

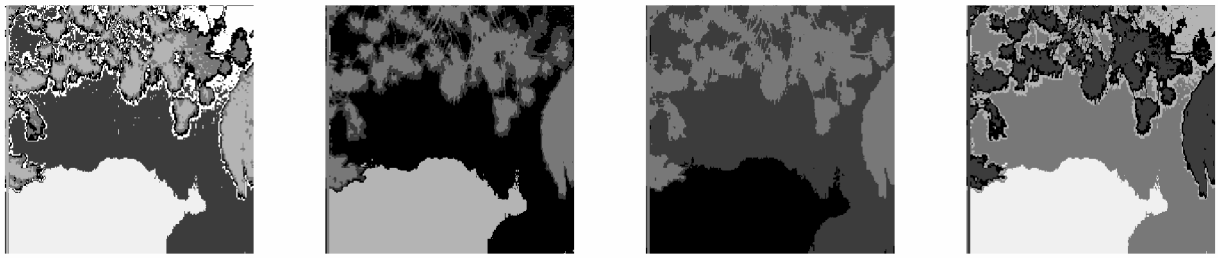
i: C=6 **j:** C=5 **k:** C=4 **l:** C=3



α =0.85 l=11

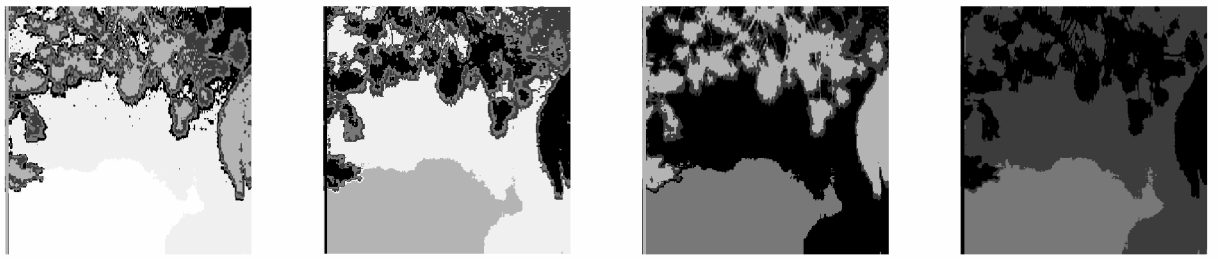
m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.24. Reconstruction d'une image de scène après une PCA+ FCM.



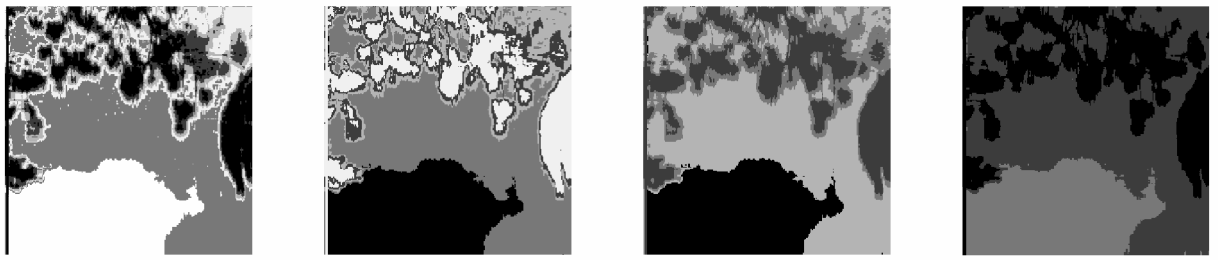
α =0.99 l=39

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3



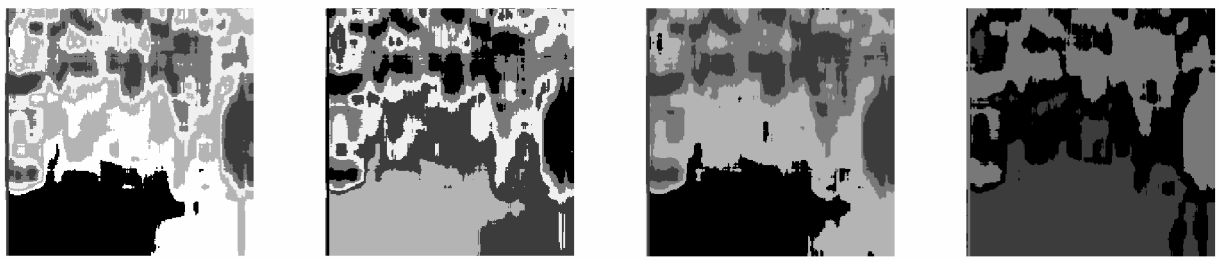
α =0.985 l=31

e: C=6 **f:** C=5 **g:** C=4 **h:** C=3



α =0.975 l=21

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3



α =0.85 l=05

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.25. Classification d'une image de paysage après une PCA+FCM

	L'image	L'erreur maximum tolérée	La dimension	Le nombre de classes supposé	Le rapport nbr de classes / dimension
SVD	Scène	95%	33	05	≈1/6
	Paysage	97%	26	04	≈1/6
PCA	Scène	94%	26	05	≈1/5
	Paysage	97.5%	21	04	≈1/5

Tab.4.3. Rapport du nombre de classes à la dimension sur les images de scène

Le Tab.4.3. montre les mêmes résultats que ceux obtenus sur les images médicales- Tab.4.2. Les images de scènes se comportent de la même manière que ces dernières car on remarque que le rapport du nombre de classes/dimension est identique, sauf que l'erreur considérée est un peu plus petite que dans le cas précédent. Ceci est dû à la différence entres les classes –la variance entre elles- que l'on voit plus importante sur celles-ci. Quand à la classification que l'on a considérée après la SVD et L'ACP, les classes sont tout aussi distinctes pour la marge de l'erreur acceptée.

5.4. Les portraits

a. Traitement par SVD

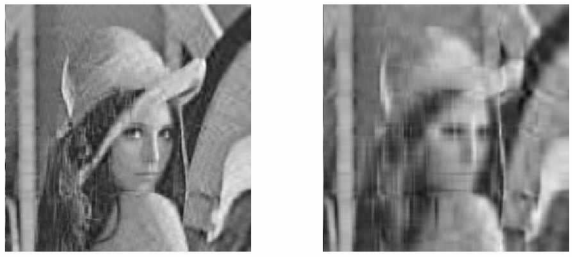


a: image originale **b:** $\alpha = 0.99$ et l=50 **c:** $\alpha = 0.98$ et l=31


d: $\alpha = 0.95$ et l=16 **e:** $\alpha = 0.85$ et l=06

Fig.4.26. Reconstruction d'un portrait après une SVD

a: image originale **b:** $\alpha = 0.99$ et l=68 **c:** $\alpha = 0.95$ et l=28



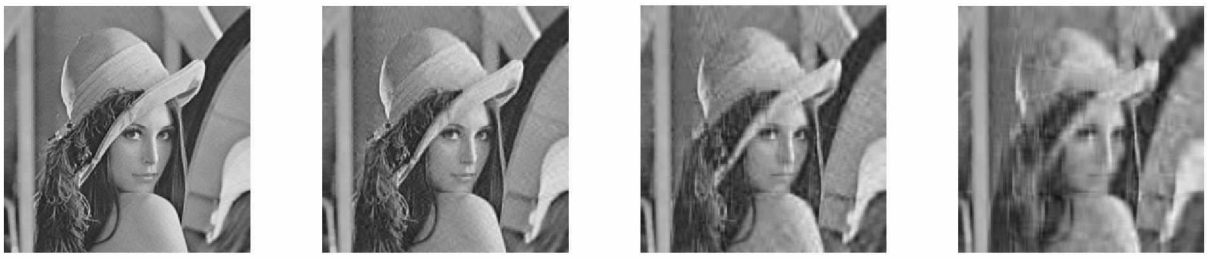
d: $\alpha = 0.94$ et l=25 **e:** $\alpha = 0.85$ et l=11

Fig.4.27. Reconstruction d'un portrait de Léna après une SVD

b. Traitement par PCA

a: $\alpha = 0.99$ et l= 52 **b:** α =0.97 et l= 26 **c:** α =0.95 et l= 18 **d:** α =0.85et l= 06

Fig.4.28. Reconstruction d'un portrait après une PCA



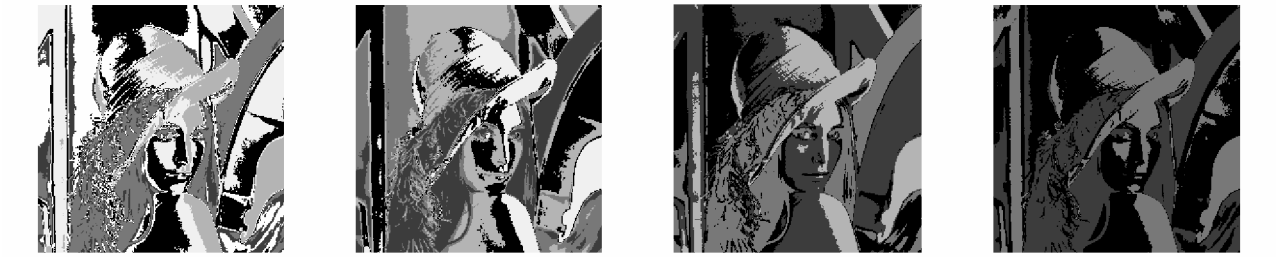
a: $\alpha = 0.99$ et l= 73 **b:** α =0.97 et l= 44 **c:** α =0.92 et l= 23 **d:** α =0.85et l= 13

Fig.4.29. Reconstruction d'un portrait de Léna après une PCA

c. Traitement par FCM

a: C=6 **b: C=5** **c: C=4** **e: C=3**

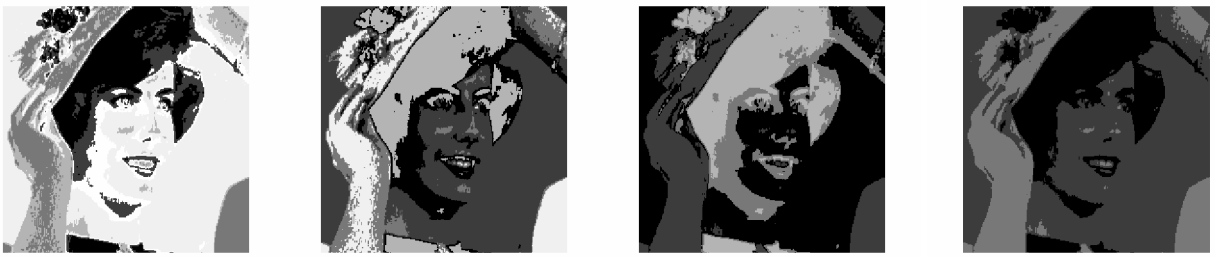
Fig.4.30. Classification d'un portrait par une FCM



a: **C=6** b: **C=5** c: **C=4** **e:** C=3

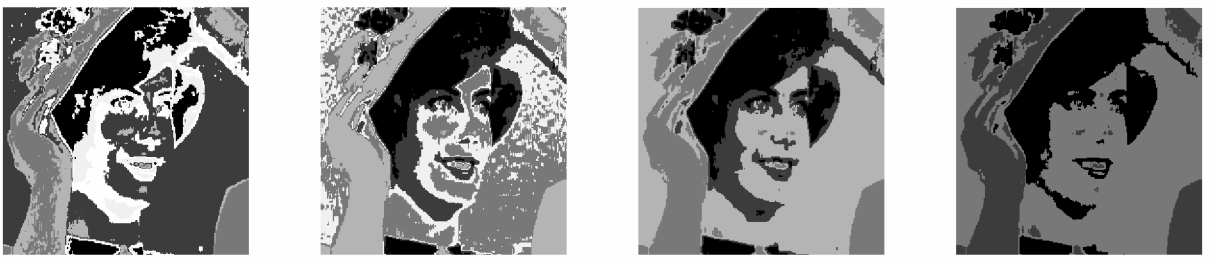
Fig.4.31. Classification d'un portrait de Léna par une FCM

d. Traitement par SVD+FCM



α =0.99 l=50

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3



α =0.98 l=31

e: C=6 f**:** C=5 **g:** C=4 h**:** C=3




α =0.95 l=16

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3




α =0.85 l=06

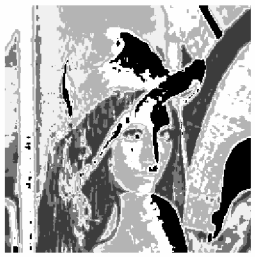
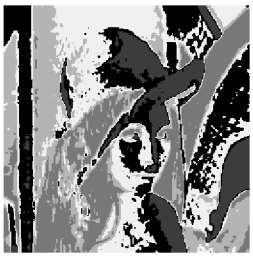
m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.32. Classification d'un portrait par une SVD+ FCM




α =0.99 l=68

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3

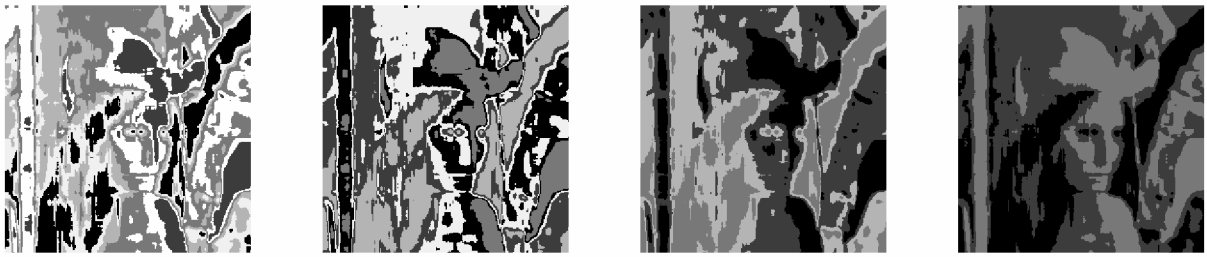
α =0.95 l=28

e: C=6 f**:** C=5 **g:** C=4 h**:** C=3




α =0.94 l=25

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3

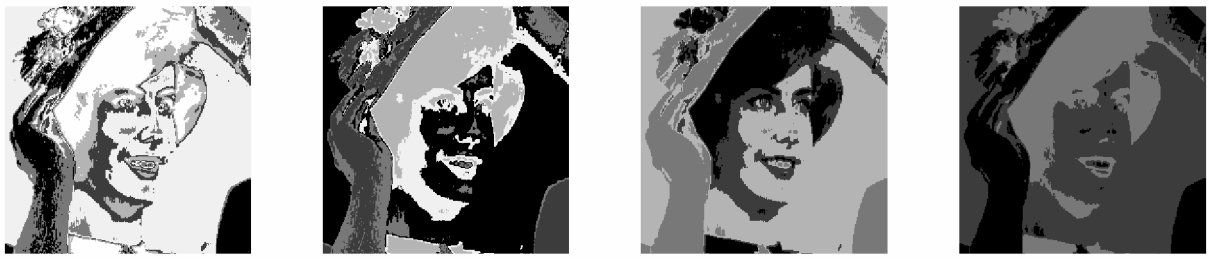


α =0.85 l=11

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

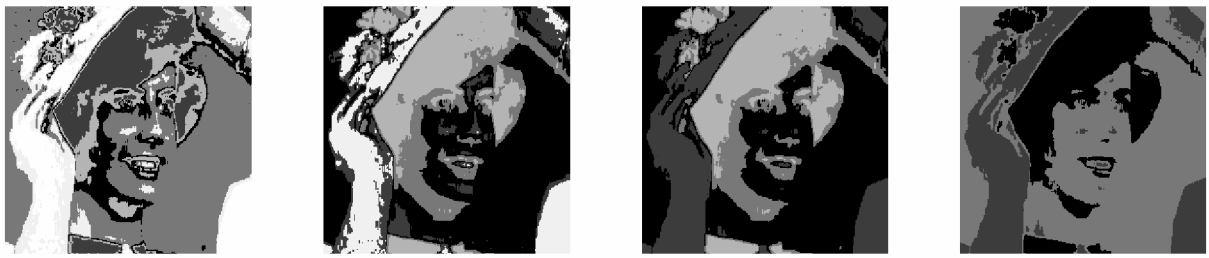
Fig.4.33. Classification d'un portrait de Léna par une SVD+ FCM

e. Traitement par PCA+FCM



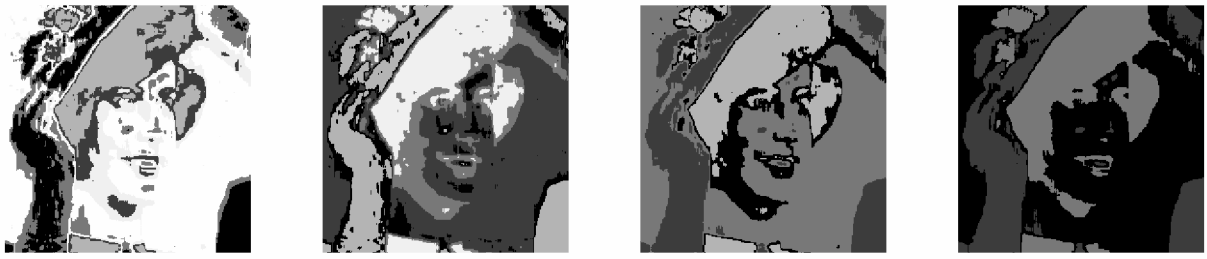
α =0.99 l=52

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3



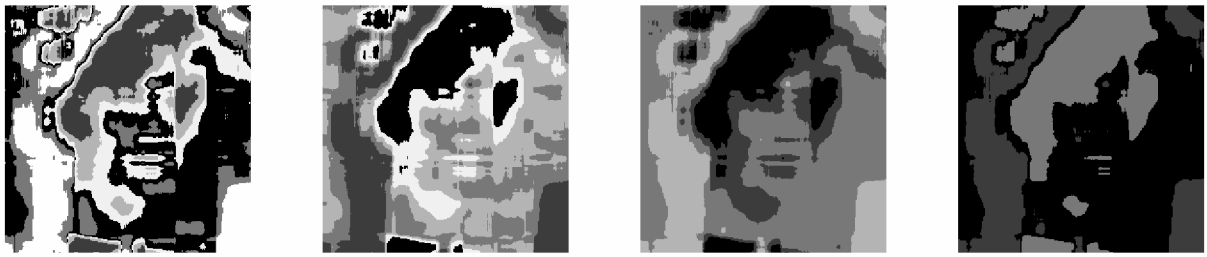
α =0.97 l=27

e: C=6 f**:** C=5 **g:** C=4 h**:** C=3



α =0.95 l=18

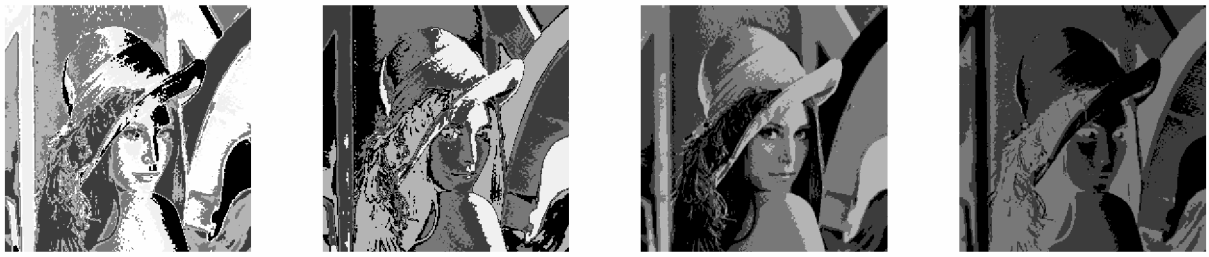
i: C=6 **j:** C=5 **k:** C=4 **l:** C=3



α =0.85 l=07

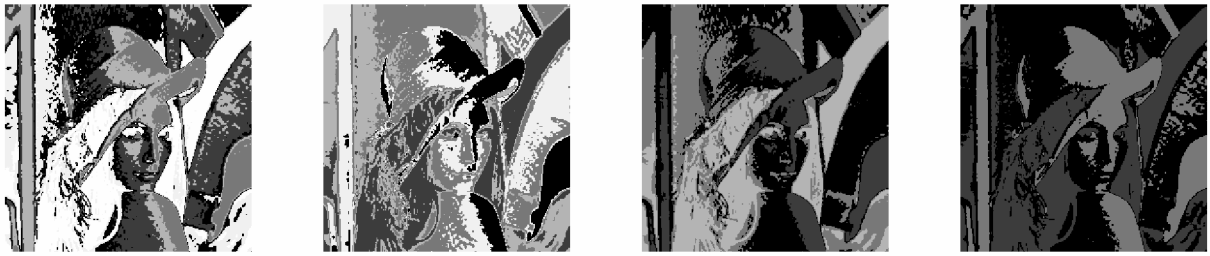
m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.34. Classification d'un portrait par une PCA+ FCM.



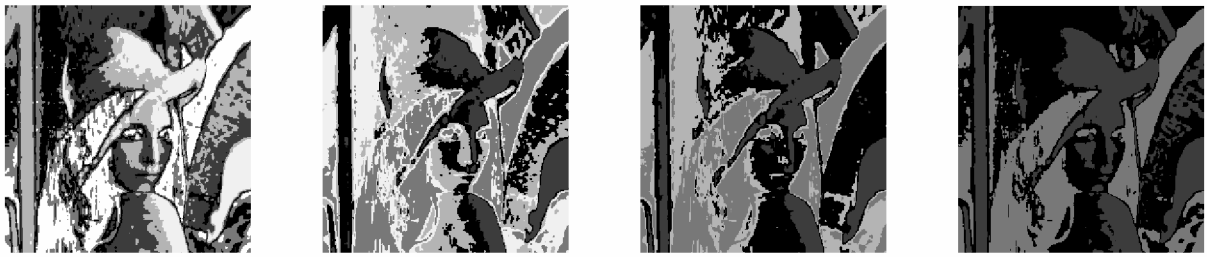
α =0.99 l=73

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3



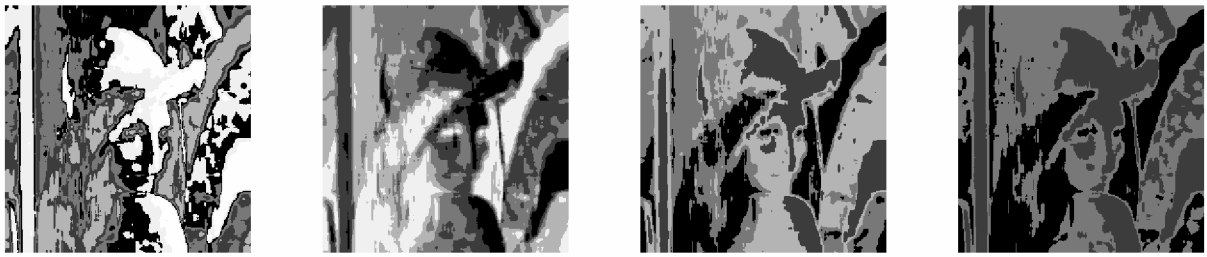
α =0.97 l=44

e: C=6 f**:** C=5 **g:** C=4 h**:** C=3



α =0.92 l=23

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3



α =0.85 l=13

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.35. Classification d'un portrait de Léna par une SVD+ FCM .

	L'image	L'erreur maximum tolérée	La dimension	Le nombre de classes supposé	Le rapport nbr de classes / dimension
SVD	Portrait	98%	31	06	≈1/5
	Portrait de Léna	95%	28	05	≈1/5
PCA	Portrait	97%	26	06	≈1/4
	Portrait de Léna	92%	23	05	≈1/4

Tab.4.4. Rapport du nombre de classes à la dimension sur les portraits

Le Tab.4.4. contient les résultats acquis sur un autre type d'images (les portraits) où l'on distingue une différence dans les fractions même si l'erreur n'a pas vraiment changé. Celles-ci sont passées du sixième au cinquième pour l'application de la SVD et du cinquième au quart pour l'ACP. Par contre le changement s'est fait de la même manière sur les deux méthodes où learelation des diviseur reste identique.

5.5. Les images aérienne

a. Traitement par SVD

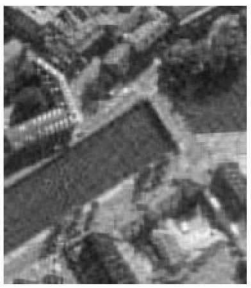
a: image originale **b:** $\alpha = 0.99$ et l=71 **c:** $\alpha = 0.97$ et l=47


d: $\alpha = 0.93$ et l=30 **e:** $\alpha = 0.8$ et l=14

Fig.4.36. Reconstruction d'une image aérienne après une SVD

b. Traitement par PCA

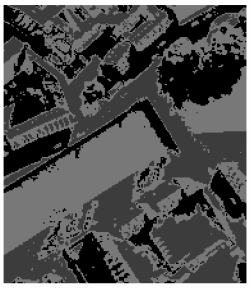
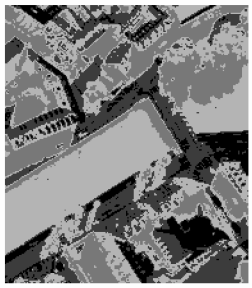
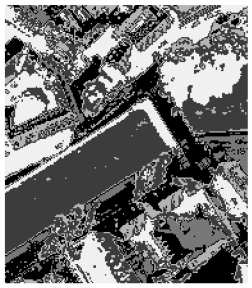
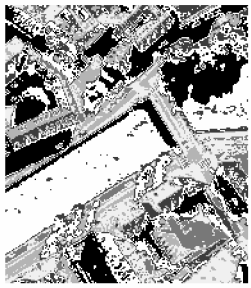





a: $\alpha = 0.99$ et l= 69 **b:** $\alpha = 0.97$ et l= 44 **c:** $\alpha = 0.93$ et l= 28 **d:** $\alpha = 0.80$ et l= 12

Fig.4.37. Reconstruction d'une image aérienne après une PCA

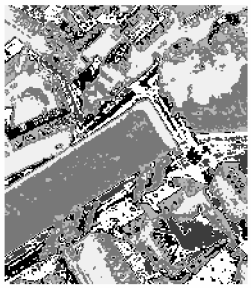
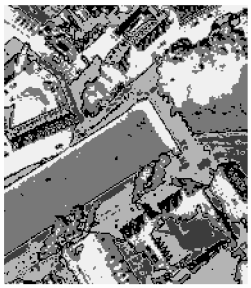
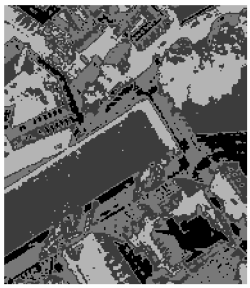
c. Traitement par FCM

a: C=3 **b:** C=4 **c:** C=5 **d:** C=6

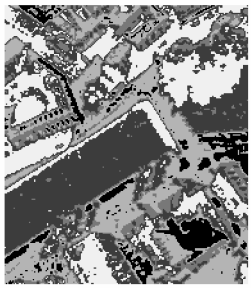
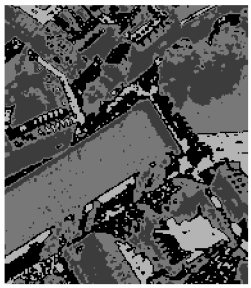
Fig.4.38. Classification d'une image aérienne par une FCM

d. Traitement par SVD+FCM

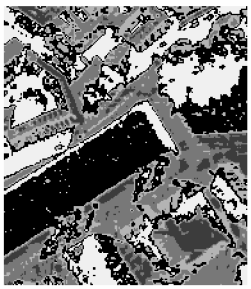
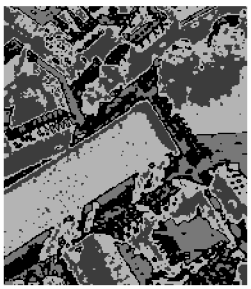
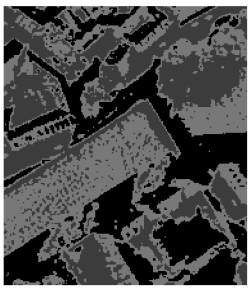

α =0.99 l=71

a: C=6 **b:** C=5 **c:** C=4 **d:** C=3

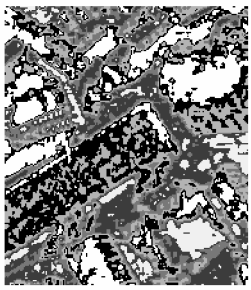
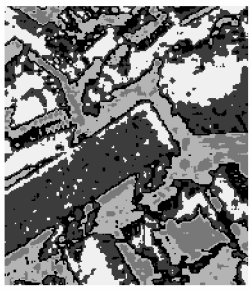
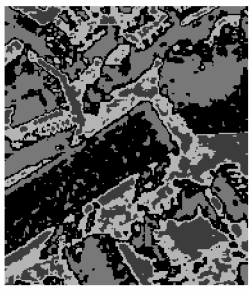
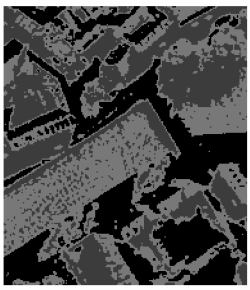

α =0.97 l=47

e: C=6 f**:** C=5 **g:** C=4 h**:** C=3

α =0.96 l=41

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3

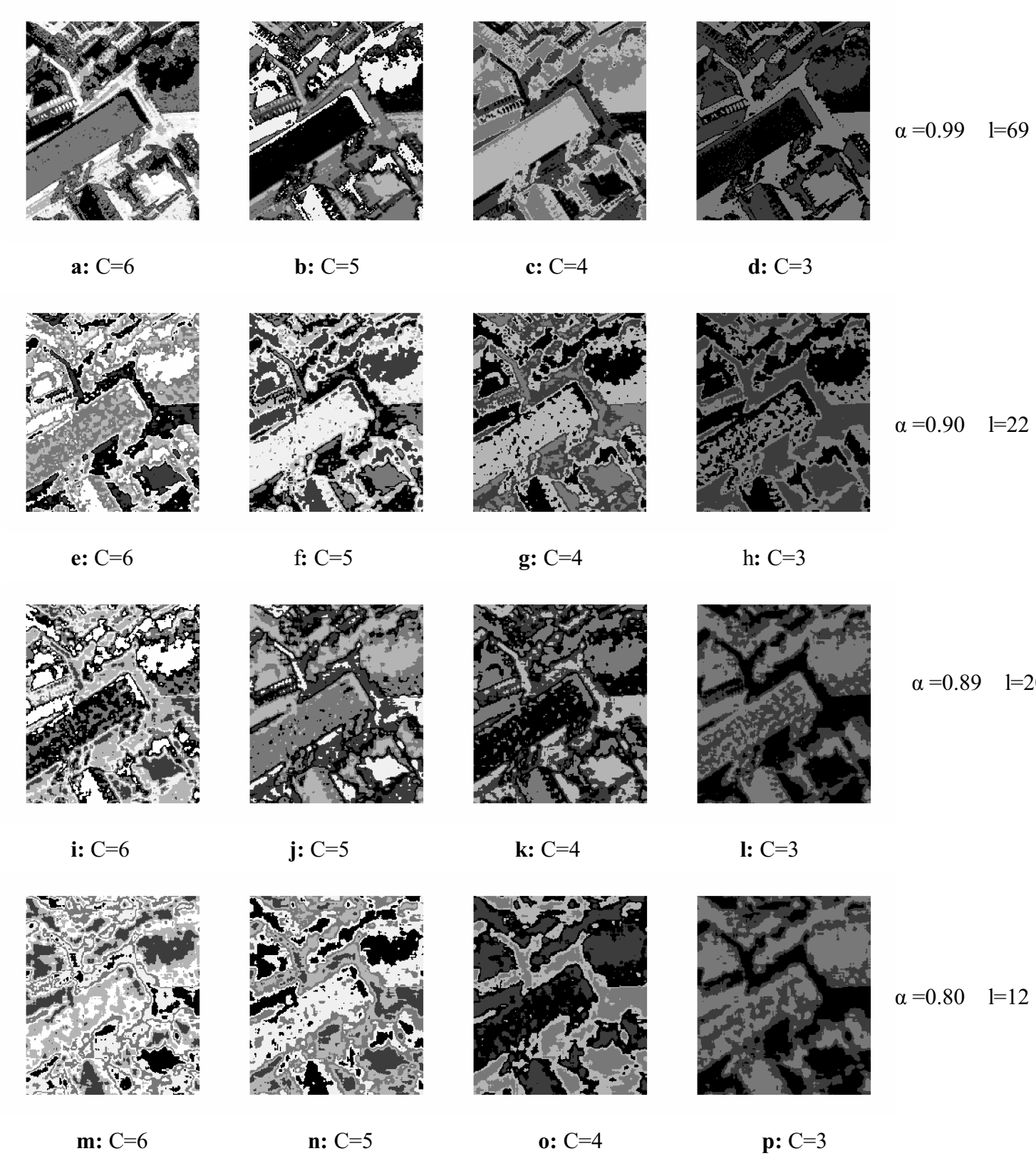





α =0.90 l=24

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.39. Classification d'une image aérienne par une SVD+ FCM.

e. Traitement par PCA+FCM



a: C=6 **b:** C=5 **c:** C=4 **d:** C=3

e: C=6 **f:** C=5 **g:** C=4 **h:** C=3

i: C=6 **j:** C=5 **k:** C=4 **l:** C=3

m: C=6 **n:** C=5 **o:** C=4 **p:** C=3

Fig.4.40. Classification d'une image aérienne par une PCA+ FCM.

	L'erreur maximum tolérée	La dimension	Le nombre de classes supposé	Le rapport dimension / nbr de classes
SVD	93%	30	05	=1/6
PCA	93%	28		≈1/5

Tab.4.4. Rapport du nombre de classes à la dimension sur une image aérienne

Le Tab.4.4 réunit les résultats obtenus à partir d'images aériennes. On retrouve des similitudes avec les images médicales dans les fractions et les erreurs dues au taux de variance semblable sur les deux types.

Dans le cadre de cette thèse, nous avons pu identifier le problème lié à la réduction de dimension dans le cadre de l'apprentissage non supervisé. Dans ce contexte, l'évaluation de ce paramètre est un enjeu majeur, car il nous est délicat et ardu de définir ce qui est opportun de ce qui ne l'est pas puisqu'on est dépourvu d'une référence. En général toute approche qui a pour but la sélection de variables peut s'appuyer sur la définition de la pertinence d'un sous-ensemble de variables, et on peut dire qu'un sous ensemble est pertinent du moment où il participe au jaillissement d'une organisation en groupe d'un ensemble d'objet. Cependant, ce-ci n'est qu'une amorce de la résolution du problème. Hélas Il n'existe pas de consensus autour d'un critère de choix adéquat ; et il subsiste alors, à être inaccessible.

Alors que l'un des enjeux actuels le plus prépondérant est la capacité à traiter de grandes masses de données, on a pensé à une première approche consistant à obtenir les images réduites afin de faciliter la classification. Nous avons utilisé le traitement de données comme l'analyse en composantes principales et la décomposition en valeurs singulières dans ce but de réduction en premier lieu, et surtout pour estimer un facteur crucial pour l'étape de classification –qui est le nombre de classes- en second lieu.

Après synthèse des résultats obtenus, par l'utilisation de l'analyse en composantes principales ainsi que la décomposition en valeurs singulières pour l'estimation du nombre des classes des images, il a été observé que la garantie une bonne reconstitution de l'image en tolérant une marge d'erreur maximum permet de dire que la dimension de l'image dans l'espace de projection est un multiples du nombre de classes de l'image. Le choix est donc assez subjectif puisqu'il nous revient de décider de la qualité de l'image reconstituée reste tributaire du choix heuristique des paramètres.

Quand à la reconstruction d'une segmentation après prétraitement elle permet de distinguer les classes plus clairement jusqu'à atteinte du seuil de la dimension approximée au multiple du nombre de classes. Ce multiple variera selon le type d'images. Une autre indication est que le diviseur du rapport dimension /nombres de classes pour une décomposition en valeurs singulières, est celui obtenu par l'analyse en composantes principales augmenté de un.

Au dessous du seuil de l'erreur maximale toléré,la reconstitution après la classification d'images réduites est de très mauvaise qualité. il en ressort une image complètement floue.

[1] T. Uchiyama et M.-A.Arbib, Color image segmentation using competitive learning, *Pattern Analysis and Machine Intelligence*, Vol. 16, N 12, 1994.

[2] A.Tremeau, P.Golantoni and B.Laget, On colour segmentation guided by the cooccurrence matrix*, OSA Annual Conference on Optics and Imaging in the Information Age*, pp. 30-38, 1996.

[3] A.Verikas, K.Malmqvist, L.Bergman, Colour image segmentation by modular neural network, *Pattern Recognition Letters*, 1996.

[4] P. Scheunders, A Genetic C-Means Clustering Algorithm applied to Color Image Quantization, *Pattern Recognition*, *Vol. 30, N6, pp. 859-866*, 1997.

[5] T.Géraud, P.Y.Strub et J.Darbon, Segmentation d'Images Couleur par Classification Morphologique non supérvisée*, Proceeding of the International Conference on Image and Signal Processing ICISP 2001*, Agadir, Moroc, mai 2001.

[6] S.GUÉRIF, Réduction de dimension en Apprentissage Numérique Non Supervisé, *université Paris 13 Spécialité : Informatique*, Décembre 2006.

[7] J.B. MacQueen, Some methods for classification and analysis of multivariate observations, *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, vol.1, University of California Press, Berkeley, CA, pp. 281–297*, 1967.

[8] J.C.Dunn, Well Separated Clusters and Optimal Fuzzy Partitions, *J. Cybern, Vol. 4, No. 3, pp. 95-104*, 1974.

[9] J.C.BEZDEK, Pattern Recognition with Fuzzy Objective Function Algorithms, *Plenum Press, New York,*1981.

[10] H. Hartley, Maximum likelihood estimation from incomplete data. *Biometrics,14 :174_194,* 1958.

[11] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society*, 39(1): 1–38, 1977.

[12] T. Kohonen, Self-Organisation and Associative Memory, *Springer-Verlag, 2nd Edition*, New York, 1984.

[13] A. Pandya et R.Macy, Pattern Recognition with Neural Networks in C++, *CRC Press*, 1996.

[14] J. Moreira et L.D.F. Costa, Neural-based color image segmentation and classification using self-organizing maps, *Proceedings of Anais do IX SIBGRAPI 47–54*, 1996.

[15] H.Bélanger, Réseau de Kohonen pour la détection des contours d'objets dans une image à niveau de gris. Maîtrise en technologie des systèmes M.Ing. École de technologie supérieure Université du Québec, 1998.

[16] J.L.LUMLEY ,*Stochastic tools in turbulence*. New York: Academic Press, 1970.

[17] V.C.KLEMA et A.J.LAUB, The singular value decomposition: its computation and some applications, *IEEE* T*ransactions on Automatic Control* 25, 164-176, 1980.

[18] P. Comon, Independent component analysis, a new concept ?, Signal Processing 36: 287-314, 1994.

[19] A. Hyvärinen, J.Karhunen, et E.Oja, *Independent Component Analysis*, John Wiley and Sons New York, 481 p. 2001.

[20] Ronald A. Fisher, The use of multiple measurements in taxonomic problems, *Annals of Eugenics*, 7 :179–188, 1936.

[21] V. Vapnik, The nature of statistical learning theory, *Springer-Verlag, New York*, 1996.

[22] Frank Y.Shih, Kai Zhang, A distance-based separator representation for pattern classification, *Image and Vision Computing 26.67–672*, 2008.

[23] Chris Ding, Tutorial Notes Principal Component Analysis and Matrix Factorizations for Learning*, The 22nd International Conference on Machine Learning, in Bonn, Germany*, 7-11 August 2005.

[24] Y.C.Liang, H.P.Lee, S.P.Lim, W.Z.Lin, K.H.Lee, C.G.Wu, Proper Orthogonal Decomposition And Its Applications Part I: Theory, *Journal of Sound and vibration* 252(3), 527 -544, 2002

[25] Sangkeun Lee*, Member, IEEE, and* Monson H.Hayes*, Fellow, IEEE,* Properties of the Singular Value Decomposition for Efficient Data Clustering, *IEEE Singnal Processing Letters, Vol: 11, N°11*, November 2004.

RESUME

A travers cette étude, nous avons essayé d'accroître l'efficacité de l'algorithme de classification (clusturing) -les k-moyennes floues-. Ce-ci en opérant des traitements préliminaires aux images par des techniques de réduction telle que l'analyse en composantes principales, la décomposition en valeurs singulières. Nous avons surtout utilisé les propriétés de ces méthodes et comparé leurs résultats pour l'optimisation du calcul d'un paramètre qui reste difficile à évaluer qui est le nombre optimal de classes de l'image, sur lequel repose l'efficacité et le bon résultat de l'algorithme de clustering.

ABSTRACT

Through this study, we tried to show how the efficiency of the fuzzy c-means clustering algorithm, can be improved by applying a pre-processing techniques to the image which aim to reduce its dimensions such as the principal component analysis, the singular value decomposition. And we'll mostly try to use their proprieties and compare their results to optimize the computation of a parameter which remain difficult to evaluate, we mean the optimal number of class, on which is based the efficiency of the clustering algorithm.

ملخص

من خلال هذا العمل أردنا أن نستعمل بعض الطرق الواردة من التحليل التمييزي من اجل تخفيض حجم البيانات في المعالجة المسبقة للصور.أول استعمال كان للبحث عن العدد الأمثل للفئات المستعمل في خوارزميات التقسيم بطريقة التحليل إلى قيم فردية و كذلك طريقة التحليل إلى مكونات رئيسية. ثاني استعمال مكننا من تخفيض كمية المعلومات لتمثيل الصورة بإجراء ضغط جيد.