Université de Montréal

# Apprentissage semi-supervisé par réduction de dimensionnalité non linéaire

par
François Payette
Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de
Maîtrise ès sciences (M.Sc.)
en Informatique

Août, 2004

# Université de Montréal

Faculté des études supérieures

Ce mémoire intitulé:

## Apprentissage semi-supervisé par réduction de dimensionnalité non linéaire

présenté par:

François Payette

a été évalué par un jury composé des personnes suivantes:

Yoshua Bengio
(président-rapporteur)

Balázs Kégl
(directeur de recherche)

Douglas Eck
(membre du jury)

Mémoire accepté le:
16 Novembre 2004

# Résumé

**Mots-clés :** Statistiques, apprentissage statistique, algorithmes, réduction de dimensionnalité non linéaire, classification, apprentissage semi- supervisé, apprentissage non supervisé, variété non linéaire, échelonnement multidimensionnel, noyaux, Isomap, LLE, forage de données, méthodes itératives.

Les algorithmes de réduction de dimensionnalité, particulièrement la famille qui effectuent une réduction non-linéaire sont un sujet très actuel en apprentissage statistique. Nous présentons ici leur lien avec les méthodes à noyau et plus particulièrement leur dépendance sur l'extraction de valeurs et vecteurs propres d'une matrice d'adjacence. Nous nous sommes intéressés de près à une de ces méthodes, Isomap, et à son intérêt à des fins d'apprentissage semi supervisé. Nous proposons ici deux variantes de celle-ci qui intègrent une méthode de classification : RISIMAP et Isostrech. Ce dernier propose une distorsion de la variété dictée par les étiquettes des points. Nous avons appliqué ces deux algorithmes à un ensemble de données oncologiques de gènes exprimés en très haute dimensionnalité pour un nombre restreint d'individus, et à un ensemble de données issu de recherches en météorologie afin de valider empiriquement leur utilité.

# Summary

**Keywords :** Statistics, machine learning, algorithmics, non-linear dimensionality reduction, classification, semi-supervised learning, non-supervised learning, non-linear manifold, multidimensional scaling, kernels, Isomap, LLE, data-mining, iterative methods.

Dimensionality reduction algorithms, especially the family that computes a non-linear reduction are one of machine learning's hottest topics. We present here their link with kernel methods as well as their reliance on extraction of eigenvalues and eigenvectors of and adjacency matrix. We have investigated thoroughly one of these methods : Isomap, we have concentrated on its applications in semi-supervised learning. We propose here two variants of it that integrate a classification method : RISIMAP and Isostretch. The latter suggests a distortion of the manifold as dictated by the labels of a pair of points. We have applied these two algorithms to a cancer research dataset of genes expressed in very high dimensionality space for a relatively small sample size, as well as to a dataset from climatology research to empirically validate their potential.

# Table des matières

# Liste des figures

# Liste des tableaux

*À mon épouse*

# Remerciements

J'aimerais d'abord remercier mon directeur de recherche Balazs Kegl pour l'encadrement de ce travail. Merci aussi à Yoshua Bengio pour les discussions stimulantes et conseils judicieux, c'est un modèle de chercheur infatigable. Je remercie sincèrement Christian Jauvin pour ses critiques constructives ainsi que les autres membres du LISA avec qui j'ai fraternisé à l'occasion. Un merci aussi à Vincent Côté-Roy pour ses critiques et suggestions.

Je tiens à mentionner le soutien de IRIS-PRECARN ; la poursuite d'études supérieures est un chemin ardu pour plusieurs sur le plan financier, et ces organismes peuvent donner un bon coup de main.

Un merci spécial à mes parents pour leur amour et pour avoir stimulé et encouragé mon envie de découvrir. Finalement, un merci profond à mon épouse ; en particulier pour son support de chaque instant lors de ce travail de longue haleine, mais aussi *d'être* ; elle est la source de mon inspiration, de mon énergie et de ma joie de vivre. Merci à Dieu de l'avoir mise sur ma route.

Chapitre 1

# Introduction

La définition généralement acceptée d'apprentissage, à l'instar de celle d'intelligence est un concept foncièrement humain. Malgré cela, même pour les être humains il est difficile, voire tendancieux de quantifier l'intelligence ; des tests généraux de quotient intellectuel dépendant d'un minimum de connaissances existent, mais produisent des résultats imprécis et parfois erronés. La définition de l'intelligence est elle-même relativement floue ; il semble cependant valable de supposer que l'apprentissage est un prérequis à celle-ci.

L'apprentissage est plus facile à évaluer : plutôt que de quantifier le potentiel on mesure le rendu pour une tâche précise. Les professeurs de ce monde ont développé, au grand dam de certains, des méthodes relativement exactes pour l'évaluer chez l'humain. Il s'ensuit que lorsqu'on mesure l'apprentissage de systèmes adaptifs, on utilise des outils, des métaphores et des structures conceptuelles profondément humains.

La recherche en intelligence artificielle est un domaine en ébullition depuis quelques décennies ; l'évolution de ses théories fondamentales et de ses postulats ainsi que celle de la psychologie cognitive humaine a induit l'émergence de reformulations et de spécialisations du champ. L'apprentissage statistique en est une très mathématique tentant de dégager des algorithmes exhibant des

propriétés désirables de rétention vérifiable d'information.

Il est possible de dégager deux forces sous-tendant le processus d'apprentissage chez l'homme. Premièrement le cerveau humain possède un formidable pouvoir de synthèse : même aux niveaux relativement bas et automatiques des processus cognitifs comme ceux prenant place dans le pré cortex et dans le cortex visuel secondaire, la quantité gigantesque d'information transmise à chaque centième de seconde par les $10^6$ nerfs optiques est traitée, réduite et reformulée ; des données spatiales et de reconnaissances de formes en sont inconsciemment extraites. Un visage vu furtivement quelques dixièmes de secondes, une voix et le processus est enclenché sans que les fonctions cognitives supérieures soient gênées ou même que le sujet soit conscient du travail herculéen d'analyse qui est effectué. Comme nous allons voir dans les chapitres suivants, les algorithmes de réduction de dimensionnalité peuvent être perçus comme des méthodes qui tentent de reproduire cette synthèse, mais pour des données quelconques.

Le constat est que le cerveau humain possède des sous-systèmes très efficaces de traitement de signal à large bande pour des données visuelles, auditives, olfactives et tactiles s'apparentant à la réduction de dimensionnalité que nous allons étudier dans ce document ; mais d'autres tâches du même type requièrent de lui beaucoup plus de traitement par ses processus cognitifs supérieurs, manipulations pour lesquelles ces derniers ne sont pas spécialisés. Les algorithmes que nous allons voir n'ont pas cette limitation ; la rapidité de traitement est la même que les données soient des imagettes en 76000 dimensions[1] ou des données génétiques de dimensionnalité 40000[2] ; cela rend ces algorithmes très intéressants pour une foule d'applications. Il va donc sans dire que pour des tâches pour lesquelles le cerveau humain possède du matériel spécialisé comme l'extraction de visages les modèles robotiques actuels ne sont pas (encore ?) de taille. Mais il n'en est pas de même pour d'autres ensembles de données s'inscrivant dans des espaces langagiers non familiers à l'humain.

La généralisation est la dynamique complémentaire se manifestant de façon

[1] 320*240 pixels
[2] gènes exprimés

simultanée à tous les niveaux de la cognition humaine. Cette reconnaissance de formes qui associe un nouvel exemplaire à un concept ou un ensemble est, depuis toujours, au coeur même du questionnement des philosophes sur l'être humain : du monde des idées de Platon au sens par l'usage de Wittgenstein en passant par les postulats cartésiens, leur interrogations s'articulent souvent autour des mécanismes de généralisation et d'analyse de la pensée humaine.

Les psychologues par les neurosciences cognitives étudient aussi les tenants et aboutissants de ces mécanismes de généralisation et de catégorisation sémantique qui se déroulent dans le cortex cérébral, leurs théories tentent de modéliser le fonctionnement de ce processus. Les travaux de Broadbent, Sperling, Miller, Galanter et Pribram et de nombreux autres sont autant d'avancées en psychologie cognitive permettant la construction de modèles qui expliquent les processus cérébraux. L'extrait suivant de *A Study of Thinking* de Bruner, Goodnow et Austin met en contexte le travail phénoménal de l'organe qui nous rend humains :

> *Commençons par ce qui semble un paradoxe. Le monde des expériences de tout homme normal est composé d'un formidable ensemble de différents objets, événements, individus, impressions, tous discernables. Mais si nous devions utiliser l'ensemble de nos capacités pour relever les différences entre chaque chose et répondre à chaque événement comme s'il était unique, nous serions bien vite submergés par la complexité de notre environnement. La solution à cette apparent paradoxe- l'existence de capacités de discernement qui, pleinement utilisées, nous rendent esclaves de la singularité- repose sur la capacité de l'homme à catégoriser. Catégoriser, c'est rendre équivalentes différentes choses discernables, regrouper des objets et des événements dans des classes, et y répondre selon leur statut de membre d'une classe plutôt qu'en fonction de leur singularité.*

Pour les systèmes adaptifs, le pouvoir de généralisation est l'étalon de référence pour mesurer l'apprentissage ; cette généralisation évaluée empiriquement doit être rédigée dans un formalisme pour lequel il est facile de

mesurer l'erreur ; de plus celle-ci doit être statistiquement stable pour être validée.

Le sujet du présent mémoire s'inscrit au carrefour des deux axes orientant notre compréhension de l'apprentissage : à la fascinante synthèse d'information que la réduction de dimensionnalité non linéaire ( que nous allons voir au chapitre 4 ) effectue, nous combinons des modèles de généralisation qui tentent d'apprendre la classification d'ensembles d'observations, dans l'optique d'ajuster la tâche de synthèse afin de faciliter cette généralisation. Nous allons présenter au chapitre 6 de ce document deux propositions en ce sens : RISIMAP et Isostretch.

Nous allons commencer par introduire au chapitre 2 la terminologie et les concepts qui supporte l'évaluation de la généralisation telle qu'elle est mesurée en apprentissage statistique. Nous allons ensuite passer en revue certaines méthodes de réduction de dimensionnalité linéaire, non linéaire, les implications algébriques et algorithmiques de celles-ci pour enfin présenter nos modèles et leurs résultats sur des ensembles issus du génie génétique et de la météorologie. Nous conclurons par une récapitulation des contributions de ce mémoire, pour par la suite poser un bref regard sur des pistes nouvelles et implications qui pourraient découler de la compréhension actuelle de notre sujet.

## 1.1 Définitions et notation

Dans ce mémoire nous nous efforcerons de présenter le formalisme relatif aux modèles et équations sous-tendant le propos avec le plus de concision et de simplicité possible. Toutefois, pour éviter la possibilité de confusion quand à ce formalisme, dans ce chapitre nous énumérerons l'ensemble des symboles et abstractions utilisés dans ce mémoire. Le lecteur familier avec celui de l'apprentissage statistique est prié de se soustraire à la lecture des définitions qui suivent et passer sans tarder au chapitre suivant.

**Tableau 1.1** – *Symboles et notation*

| Symbole | Signification |
|---|---|
| $\mathbb{R}$ | ensemble des nombres réels. |
| $\mathbb{R}^d$ | espace à d dimensions réelles. |
| $S_n$ | ensemble de n variables aléatoires représentant n observations. |
| $\lambda_i$ | *i-ème eigenvalue*, ou valeur propre. |
| $\boldsymbol{v}$ | un vecteur, dénoté en gras. |
| $\Sigma$ | matrice de covariance. |
| $\Lambda$ | matrice diagonale, généralement de valeurs propres. |
| $E[X]$ | espérance de X. |
| $\mathbb{X}$ | ensemble des observations X. |
| $\epsilon$ | utilisé pour dénoter un voisinage de taille donnée. |
| $\delta_{ij}$ et $d_{ij}$ | utilisé pour dénoter la distance entre les deux points $i$ et $j$. |
| $\theta$ | coefficient de rotation. |
| $\phi$ | angle de transformation par rotation. |
| $I$ | matrice identité. |
| $\widehat{R}(f, S_n)$ | erreur empirique de la fonction $f$ sur $S_n$. |
| log, lg, ln | logarithme en base 10, logarithme en base 2, logarithme naturel. |
| $\ll$ | plus petit par plusieurs ordres de grandeurs. |
| $\tilde{X}_i$ | reconstruction de la $i$-ième observation. |
| $\lvert C_j \rvert$ | la cardinalité de l'ensemble $C_j$. |
| $\dot{Y}_i$ | transformation par rotation, réflexion ou translation. |
| $\xi_i$ | $i$-ième vecteur propre de $\Sigma$. |
| $\tau$ | opérateur du centroïde |
| $\eta_d$ | le volume de la boule unitaire dans l'espace euclidien de $d$. |
| $\Phi(Y)$ | fonction de projection. |
| $K(X, X')$ | fonction de noyau |

CHAPITRE 2

# Concepts préliminaires d'apprentissage statistique

Pour bien situer le sujet du présent mémoire nous allons dans ce chapitre spécifier les concepts relatifs au champ des systèmes adaptifs qui délimitent le cadre dans lequel ce mémoire s'inscrit. Nous allons spécifier ce qui est entendu par apprentissage statistique ; soit les tâches de régression, d'estimation de densité et de classification. Par la suite nous allons spécifier les trois types d'apprentissage possibles : soit l'apprentissage supervisé, non supervisé et semi-supervisé. Nous allons finalement rappeler comment l'apprentissage est évalué et quelle est la principale difficulté inhérente aux données en haute dimensionnalité.

## 2.1 Tâches d'apprentissage

La description des différentes tâches d'apprentissage qui suivent présuppose certaines notions : nous notons $S_n = \{Z_1, Z_2, \ldots, Z_n\}$ un ensemble de $n$ variables aléatoires indépendantes, distribuées de façon identique qui représentent

autant d'observations d'un phénomène dont la distribution (possiblement continue) $p(Z)$ est inconnue.

### 2.1.1 Régression

Les problèmes de régression en intelligence artificielle ont l'objectif commun d'apprendre(de découvrir) la fonction génératrice d'un phénomène à partir d'un ensemble fini et restreint d'observation de celui-ci. On tente donc d'estimer une fonction continue à partir de $n$ observations en $d$ dimensions.

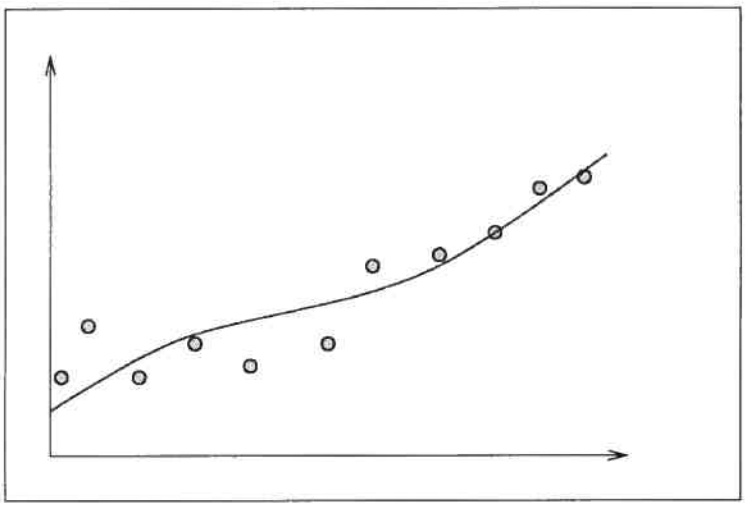


**Figure 2.1** – *Un exemple de régression pour des points dans un espace à une dimension*

On cherche donc la fonction $f(x)$ de régression :

$$f(x) = E(Y|X = x)$$

Formellement, Z est un couple de deux variables aléatoires X et Y, X est le vecteur de caractéristiques observées (de taille $d$, la dimensionnalité du

problème), et Y est la valeur réelle qui est estimée ; la régression.

$$Z = (X, Y) \qquad X \in \mathbb{R}^d \quad Y \in \mathbb{R}$$
$$(X, Y) \in \mathbb{R}^d \times \mathbb{R}$$

Dans la figure 2.1 on peut voir la modélisation d'un phénomène en une dimension à partir d'un nombre fini d'observations.

Par exemple, étant donné des mesures de hauteur et de circonférence pour un ensemble d'érables, nous pourrions par régression apprendre une fonction continue exprimant l'âge d'un érable ( nombre de cercles concentriques dans une coupe transversale du tronc). Nous pourrions estimer (avec une marge d'erreur mesurable pour un écart de confiance donné) l'âge de n'importe quel érable étant donné sa hauteur et sa circonférence.

### 2.1.2 Classification

Les tâches de classification se définissent par l'objectif primaire d'identifier l'appartenance d'un point à un ensemble discret de classes, étant donné des exemples de chacune de ces classes. A l'inverse de la régression qui tente de trouver une fonction continue passant par (ou le plus près possible) chaque Y estimé, la classification tente d'assigner une étiquette parmi les $k$ étiquettes possibles. Dans le cas plus simple et plus étudié où il n'y a que deux classes, traditionnellement on remplace les étiquettes par $-1$ pour la première classe et 1 pour la seconde.

Formellement, nous avons un couple Z de deux variables aléatoires X et Y, X est le vecteur de caractéristiques (de taille $d$) et Y est l'étiquette associée.

$$Z = (X, Y) \qquad X \in \mathbb{R}^d \quad Y \in \{1, 2, \ldots, k\} (ou \{-1, 1\})$$
$$(X, Y) \in \mathbb{R}^d \times \{1, 2, \ldots, k\} (ou \{-1, 1\})$$

Dans le cas à deux classes nous avons :

$$P(Y = 1|X = x) = 1 - P(Y = -1|X = x).$$

La classification optimale (de Bayes) est la fonction $f(x)$ qui vaut 1 ou $-1$ selon :

$$f(x) = \begin{cases} 1 & \text{si } P(Y = 1|X = x) > P(Y = -1|X = x), \\ -1 & \text{sinon.} \end{cases}$$

Notons que si on avait plus de deux classes, il est possible de reproduire ce cas avec un nombre fini de problèmes à deux classes. Donc étudier le cas à deux classes, sur le plan théorique, englobe aussi les problèmes à plus de deux classes.

Si on connaissait la distribution des probabilités cette classification serait très simple à évaluer, mais ce n'est pas le cas ; c'est ici le problème central de l'apprentissage. On a les données $S_n$ qui sont, on le rappelle, un ensemble de $n$ variables aléatoires indépendantes, distribuées de façon identique à partir desquelles nous tentons d'établir cette classification.

Dans la figure 2.2 on peut voir un exemple d'une surface de décision apprise qui sépare les deux classes de points donnés.

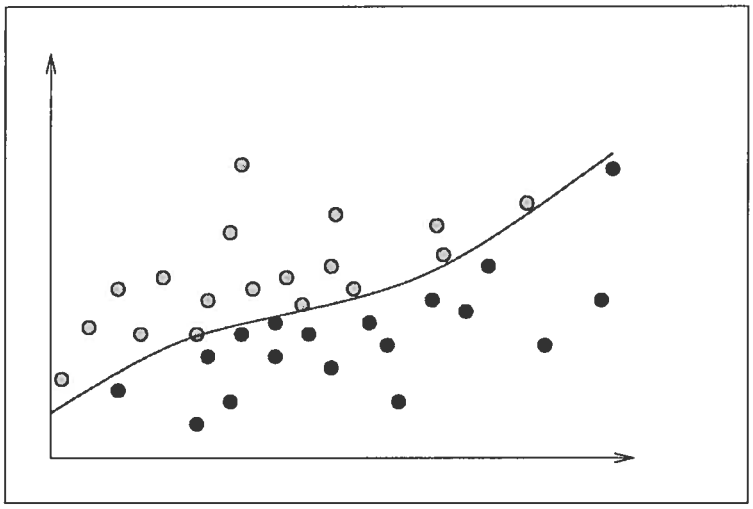


**Figure 2.2** – *Un exemple de surface de décision pour des points appartenant à 2 classes dans un espace à 2 dimensions*

Pour reprendre une variation de l'exemple cité dans la section précédente sur la régression, étant donné des observations de hauteur, de couleur d'écorce, de couleur de feuillage et de circonférence d'un ensemble de bouleaux et d'érables(les étiquettes), un modèle de classification entraîné sur cet ensemble pourrait nous prédire l'appartenance à l'une ou l'autre de ces classes selon les caractéristiques de hauteur, de couleur d'écorce, de couleur de feuillage et de circonférence d'un spécimen donné.

### 2.1.3 Estimation de densité

Étant donné un ensemble d'observations qui ne sont pas qualifiées ou quantifiées de manière particulière, on tente d'apprendre la densité de la distribution dans l'espace généré par les caractéristiques de ces observations. Tous les points sont considérés comme étant des exemples du même phénomène, on tente de modéliser les caractéristiques statistiques de la distribution de ces points. La figure 2.3 montre un exemple d'une fonction de distribution apprise étant donné des points ayant une dimension(caractéristique).

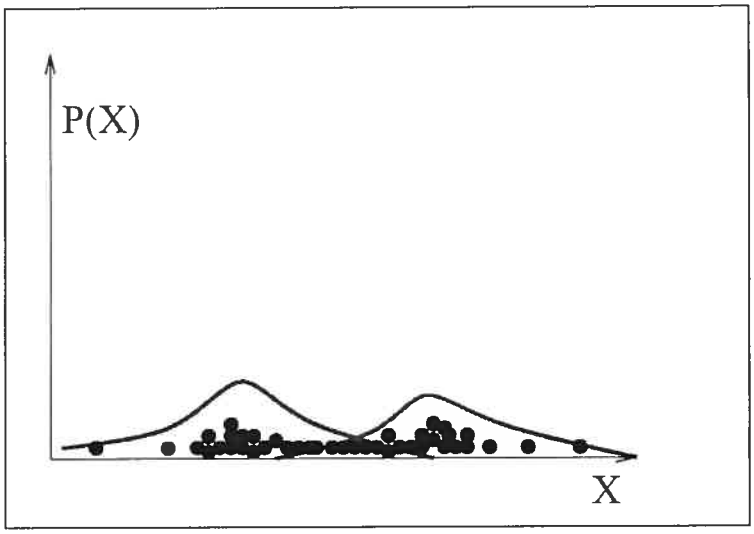



**Figure 2.3** – *Un exemple d'estimation de densité apprise sur des points ayant seulement une dimension*

Pour prendre un exemple dans la même famille que les exemples cités dans les 2 sections précédentes, étant donné les mesures de superficie moyenne des feuilles pour tous les arbres d'une forêt, on pourrait apprendre l'estimation de densité de cette variable afin de dire combien d'espèces il y a dans cette forêt (nombre de modes), la taille moyenne des feuilles de chaque espèce, etc.

## 2.2 Types d'apprentissage

### 2.2.1 Apprentissage supervisé

L'utilisation ou l'inexistance d'instructions extérieures qualifiant ou quantifiant les données à apprendre définit une facette majeure de l'apprentissage statistique nous permettant de classifier les différents systèmes adaptifs. La classification et la régression sont des types d'apprentissage supervisés. Si la méthode utilise une information de qualification(les étiquettes des points) ou une information de quantification(une valeur réelle), on dit d'elle qu'elle fait de l'apprentissage supervisé ; l'apprentissage se concentre à bien prédire cette information de qualification ou de quantification pour de nouveaux exemples. La personne qui supervise l'apprentissage de l'algorithme donne donc à celui-ci, à un moment donné dans la phase d'apprentissage, une étiquette de classification(ou une valeur réelle de quantification) qui a été choisie par une source fiable : généralement un spécialiste dans le domaine des données du problème traité. Dans la phase de rappel (ou de test), la qualité et la capacité de généralisation du modèle est évaluée en observant ses prédictions sur de nouveaux exemples. La figure 2.2 est un exemple d'apprentissage supervisé.

Par exemple dans une application médicale d'un modèle de classification supervisée, le superviseur formant le modèle à partir de radiographies de colonnes vertébrales spécifierait avec chaque image une étiquette de classification (de scoliose ou de condition normale par exemple) établie par un orthopédiste.

### 2.2.2 Apprentissage non-supervisé

Si aucune information qualificative ou quantitative n'est fournie avec les données d'entraînement, on qualifie de non-supervisées les méthodes d'apprentissage statistique qui traitent ces données. Les algorithmes qui appartiennent à cette classe de méthodes d'apprentissage ne font généralement qu'apprendre un modèle qui souligne des caractéristiques de la distribution des données. Le nombre d'agrégats de points (ou modes) qui semblent être des classes parmi les points donnés est un exemple d'apprentissage, [1] ou des estimations pour certains paramètres statistiques de la distribution en serait un autre. L'estimation de densité est une méthode non supervisée. La figure 2.3 pourrait être le rendu d'une caractérisation non supervisée d'un ensemble de données ayant une seule dimension.

### 2.2.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé consiste plutôt à fournir une information qualificative (des étiquettes de classification) ou une information quantitative (valeur réelle à des fins de régression) pour seulement un sous-ensemble des points d'entraînement soumis à l'algorithme d'apprentissage. L'algorithme peut quand même extraire de l'information sur la nature du ou des phénomène(s) observé(s) à partir des vecteurs qui sont fournis sans information de qualification ou de quantification.

Dans la figure 2.4 on peut voir une surface de décision apprise à des fins de classification étant donné certains points étiquetés comme appartenant à la première ou à la deuxième classe et certains point dont l'étiquette n'est pas révélée.

Dans les chapitres suivants nous verrons plusieurs exemples et variations d'algorithmes d'apprentissage semi-supervisé. À des fin de complétude, nous allons reprendre l'exemple des arbres dans la forêt ; un apprentissage semi supervisé s'effectuerait sur les caractéristiques d'arbres dont seulement un sous-ensemble est identifié comme étant d'une des 2 classes présentes, soit

[1] on appele les méthodes des méthodes de partitionnement ou "clustering"

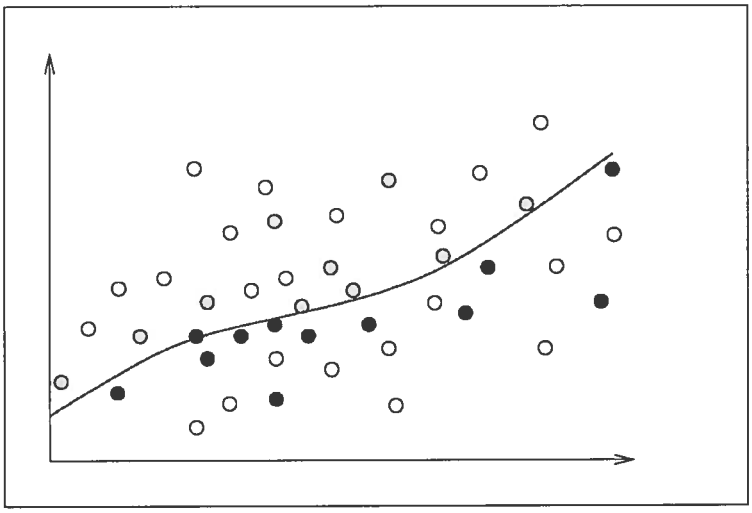


**Figure 2.4** – *Un exemple une fonction discriminante apprise de manière semi-supervisée sur des points exprimés dans un espace à 2 dimensions*

les bouleaux et les érables ; nous avons un troisième ensemble d'échantillons qui ne sont pas identifiés comme étant d'une espèce ou de l'autre.

## 2.3 Évaluation de l'apprentissage

Pour évaluer de façon objective des modèles produits par différentes méthodes d'apprentissage machine, il est primordial d'avoir des outils impartiaux qui permettent de comparer la qualité de la généralisation apprise par ces modèles. Pour évaluer la qualité d'une généralisation, il faut premièrement quantifier *l'erreur de généralisation*, notée $R$. Celle-ci est définie comme :

$$R(f) = E[L(Z, f)]^2 = \int L(Z, f)p(Z)dz$$

[2]Pour un scalaire $x$ et toute fonction $f$, $E[f(x)] = \int_{x \in \mathbb{R}}(x)P(x)dx$ où P(x) est la fonction de densité de probabilité

où $L(Z, f)$ est la qualité de la solution sur un point, la mesure de la perte de $f$ sur la variable aléatoire $Z$ ; des définitions précises de $L$ pour la classification et la régression suivent dans les prochaines sections.

Évidemment, ne connaissant pas la distribution $p(Z)$, on ne peut trouver l'erreur de généralisation réelle. Ainsi on doit utiliser une estimation bruitée de celle-ci qu'on appelle *erreur empirique* notée :

$$\widehat{R}(f, S_n) = \widehat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} L(Z_i, f)$$

cette estimation est valide puisque :

$$R(f) = \lim_{n \to \infty} \widehat{R}(f, S_n) = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} L(Z_i, f).$$

### 2.3.1 Évaluation de la classification

Pour calculer l'estimation de l'erreur de généralisation dans le cas de la classification, on compte simplement le nombre d'erreurs que le modèle produit. On a donc une fonction de perte :

$$L(Z, f) = L\big((X, Y), f\big) = \begin{cases} 1 & \text{si } f(X) \neq Y, \\ 0 & \text{sinon.} \end{cases}$$

Le risque empirique dans ce cas est tout simplement le ratio du nombre d'erreur sur le nombre de total points : l'erreur de Bayes minimise ce risque, mais puisque nous n'avons pas la distribution nous ne pouvons qu'estimer cette erreur.

### 2.3.2 Évaluation de la régression

L'erreur quadratique moyenne est généralement utilisée pour évaluer la qualité de généralisation dans le cas de la régression. Celle-ci s'exprime pour

une variable aléatoire par la fonction de perte suivante :

$$L(Z, f) = L\big((X, Y), f\big) = (f(X) - Y)^2$$

La fonction de régression $E[Y|X = x]$ minimise le risque.

### 2.3.3 Évaluation de l'apprentissage non supervisé

L'évaluation de la qualité de la généralisation d'un modèle est moins intuitive dans le cas où les échantillons ne portent pas d'étiquette. On ne peut calculer de précision de rappel ou d'erreur quadratique moyenne. Pour ces apprentissages, nous devons plutôt nous fier à des indicateurs statistiques comme le nombre de points de l'ensemble se trouvant à l'intérieur d'un intervalle de confiance donné, ou calculer la variance résiduelle par exemple, ou la vraisemblance :

$$L(Z, f) = -\log f(Z)$$

### 2.3.4 Validation Simple

La première étape nécessaire à l'évaluation d'une méthode d'apprentissage statistique est de définir un ensemble d'entraînement et un ensemble d'évaluation, appelé communément l'ensemble de *test*. Si on donne l'ensemble des points à un algorithme d'apprentissage, on ne pourra évaluer son pouvoir de généralisation(ce que l'apprentissage est réellement) puisqu'une méthode pourrait tout simplement faire du "par coeur" et apprendre la classe de chaque point (incluant ceux de test) dans une table et régurgiter l'étiquette de chacun de ces points sur demande. Dans une telle situation, nous ne serions définitivement pas en présence d'apprentissage. En définissant deux ensembles, un de test et un d'entraînement, nous cachons une partie des exemples pendant la phase d'apprentissage afin d'évaluer la qualité de la modélisation qui a été effectuée pendant la phase de test. On note l'ensemble de test $S_m$ :

$$S_m = \{Z_{n+1}, Z_{n+2}, \ldots, Z_{n+m}\}.$$

L'estimation de l'erreur de généralisation est évaluée sur $m$ points indépendants de l'ensemble d'entraînement, elle est notée :

$$R(f_n) \approx \widehat{R}_m(f_n) = \frac{1}{m} \sum_{i=n+1}^{n+m} L(f_n, Z_{n+i})$$

### 2.3.5 Validation croisée

La principale faille de la validation simple est qu'elle est sujette à de la variance induite de plusieurs sources ; on se doit de minimiser cette variance si on veut que l'évaluation de l'erreur de généralisation soit statistiquement stable. Les variables aléatoires formant l'ensemble d'entraînement ont elles aussi une variance, de même que celles formant l'ensemble de test ; ces deux variances peuvent induire une variabilité à la validation simple. [3] Pour diminuer la variance de l'estimation d'erreur de généralisation on utilise la validation croisée.

La validation croisée consiste à faire $k$ phases d'apprentissage et de test ; on divise l'ensemble de données en $k$ sous-ensembles. On note le $j$-ième de ces sous-ensembles de test :

$$\begin{aligned} q &= \frac{n}{k} \\ S_q^{(j)} &= \{Z_{(j-1)q+1}, Z_{(j-1)q+2}, \ldots, Z_{jq}\} \end{aligned}$$

et son ensemble d'entraînement par :

$$S_{n-q}^{(j)} = S_n \backslash S_q^{(j)}$$

On entraîne et on teste successivement l'algorithme sur chacun des $k$ sous-ensembles composé de $k-1$ des divisions de l'ensemble original et on évalue la précision en utilisant le $k$-ième sous-ensemble comme ensemble de test. On note $f_{n-q}^{(j)}$ la $j$-ième fonction entraînée sur $S_{n-q}^{(j)}$. L'estimation de l'erreur de

[3] Il est possible que l'algorithme d'apprentissage ait une source interne aléatoire, celle-ci peut être source de variance.

test pour cette permutation est notée :

$$\widehat{R}_q^{(j)}(f_{n-q}^{(j)}) = \frac{1}{q} \sum_{i=(j-1)q+1}^{jq} L(f_{n-q}^{(j)}, Z_i).$$

Ensuite on fait la moyenne entre ces $k$ évaluations de l'erreur pour avoir une estimation précise du pouvoir de généralisation de notre modèle.

$$R(f_n) \approx \widehat{R}_{cr(k)}(f_n) = \frac{1}{k} \sum_{j=1}^{k} \frac{1}{q} \sum_{i=(j-1)q+1}^{jq} L(f_{n-q}^{(j)}, Z_i)$$

Cette forme de validation croisée se nomme aussi la "k-fold cross-validation".

On peut pousser l'idée de validation croisée à son extrême et faire égaler $k$ au nombre d'exemples qu'il y a dans notre ensemble de points. On entraînerait donc autant de modèles qu'il y de points sur tout les points sauf un et on évaluerait la qualité du modèle en prédisant l'étiquette du point soustrait à l'ensemble d'entraînement. Cette forme de validation croisée se nomme la "leave one out cross-validation". Puisque le cycle d'entraînement est généralement coûteux en temps de calcul, on cherche plutôt à trouver un $k$ le plus petit possible qui ramène la variance des observations de test à un niveau acceptable ; cette variance est un indicateur qui ne vaut plus rien dans le cas de la "leave one out cross-validation" parce que la variabilité dans ce cas est extrême ; par exemple dans le cas de la classification le taux d'erreur de rappel est 100% ou 0%.

## 2.4 Difficultés d'apprentissage

### 2.4.1 Malédiction de la dimensionnalité

L'expression "Curse of dimensionality" (BELLMAN 1961) fait référence à la croissance exponentielle du volume de l'espace en fonction du nombre de dimensions du problème. Cette expansion de l'espace de recherche peut induire

aux modèles le problème de concentrer leur apprentissage sur des parties de cet espace qui ne contiennent pas réellement d'information susceptible d'être généralisée.

L'augmentation exponentielle du nombre d'exemples nécessaires afin de bien caractériser cet espace est un corollaire de ce problème. Dans beaucoup de cas, la taille des ensembles données implique des capacités en temps de calcul et en espace mémoire telles que les méthodes conventionnelles ne peuvent traiter ces problèmes directement étant donné les capacités computationelles actuelles. De plus, le nombre de paramètres de ces méthodes risque de causer un sur-apprentissage(voir 2.4.2) et nuire au potentiel de généralisation de l'algorithme. Pour obtenir des résultats statistiquement stables, lesquels sont essentiels à la reproductibilité d'une expérience, un *sine qua non* en science, les dimensions et/ou le nombre de points traités doivent être restreints. Une manière de mitiger l'impact de ce problème est d'utiliser les techniques de réduction de dimensionnalité, méthodes que nous allons voir dans les 2 prochains chapitres.

L'augmentation du nombre de points, indépendamment de la réduction possible de la dimensionnalité, peut avoir de grands impacts sur la faisabilité et la capacité d'un modèle ; la quantité de mémoire ou de temps de calcul peut être une borne limitative assez difficile à éviter pour l'instant, comme nous allons le voir au chapitre 5.

### 2.4.2 Surapprentissage et sous-apprentissage

Il est possible que certains modèles développent de mauvais plis au moment de l'apprentissage. Une mauvaise sélection de paramètres régissant ces modèles peut avoir un effet pervers ; le modèle peut tendre à apprendre "par coeur" les étiquettes de l'ensemble d'entraînement si on pénalise trop l'erreur sur cet ensemble. Ce faisant, le modèle devient particulièrement sous optimal quand à la généralisation ; l'apprentissage est donc beaucoup moindre, et l'erreur de test devrait indiquer cet état dégénéré.

La situation inverse est possible : les paramètres du modèle peuvent plutôt influencer l'algorithme à laisser passer trop d'erreurs et faire une généralisation

trop *générale*. Un bon apprentissage est aussi une question d'équilibre dans les paramètres, certains paramètres peuvent avoir plus d'influence que d'autres sur celui-ci.

Dans les problèmes à très haute dimensionnalité, même les modèles linéaires de classification les plus simples ont trop de paramètres et tendent à surapprendre les points ; la réduction de dimensionnalité peut être utilisée pour palier à cette situation avec de résultats relativement bons, comme nous allons voir aux chapitres 3 et 4.

### 2.4.3 Introduction de biais

La sélection de paramètres peut être un vecteur par lequel l'expérimentateur introduit un biais dans la procédure. Tout biais doit être reconnu et identifié si on veut prétendre à un résultat scientifique, reproductible et objectif. On doit tenter de l'éliminer quand c'est possible.

L'introduction involontaire ou inconsciente de bias est une plaie qui mine plusieurs disciplines scientifiques, et tous les moyens possibles doivent être mis en oeuvre pour tenter de conscrire cette dangeureuse tendance, il en va de la santé et de la solidité des dites disciplines ou expériences.

CHAPITRE 3

# Réduction de dimensionnalité linéaire

Un problème récurrent rencontré par les concepteurs de systèmes adaptifs est la malédiction de la dimensionnalité (voir 2.4.1 ). L'application directe de méthodes d'apprentissage statistique praticables au point de vue du temps de calcul et de l'analyse statistique pour des problèmes exprimés dans des espaces à basse dimensionnalité devient rapidement impraticable quand le nombre de dimensions augmente ; le pouvoir de généralisation en souffre généralement aussi. Afin de remédier à ce problème, des méthodes de réduction de dimensionnalité ont été mises au point. Au cours de ce chapitre nous allons voir quelques méthodes de la première itération de solutions qui s'appliquent à ce problème ; celles-ci accomplissent une réduction linéaire de dimensionnalité.

## 3.1 Analyse en composantes principales

L'analyse en composantes principales(HOTELLING 1933) sous sa première forme [1] (aussi appelée la transformée de Karhunen-Loève) est une méthode non supervisée de réduction de dimensionnalité linéaire. On peut en voir un exemple très simple dans la figure 3.1. De façon intuitive, cette méthode réduit l'espace dans lequel s'inscrit un problème en *projetant* les points sur un sous-espace de $d$ combinaisons linéaires des $D$ dimensions originales ; le critère de sélection étant celles qui maximisent la variance des données du problème.

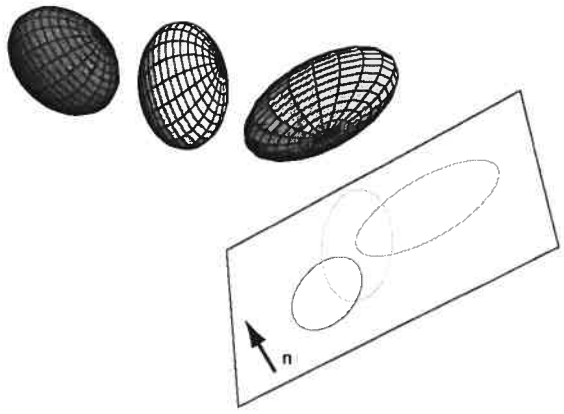



**Figure 3.1** – *Un exemple d'analyse en composantes principales, on projette 3 distributions gaussiennes en 3 dimensions sur un espace à 2 dimensions (tiré de (DUDA, HART et STORK 2000) )*

Une description symbolique explicite de la méthode suit : On dénote par $X = (X_1, X_2, \ldots, X_n) \in \mathbb{R}^D$ l'ensemble des observations aléatoires (des vecteurs en dimension D) qui ont été préalablement ajustées pour avoir une moyenne $E[X] = 0$, et $Y = (Y_1, Y_2, \ldots, Y_n) \in \mathbb{R}^d$ leur projection dans un espace à $d$ dimensions résultant d'une transformation linéaire Y = WX. L'objet de la méthode est de trouver le W qui annule le maximum de corrélation

[1] Des itérations subséquentes emploient des techniques non linéaires, comme l'ACP à noyaux.

entre les points projetés Y, qui préserve le maximum de variance et qui, conséquemment, minimise l'erreur de reconstruction. La matrice de covariance des observations originales $X$ est dénotée par $\mathbf{\Sigma}$ et celle des points projetés par $\mathbf{\Lambda}$. Cette matrice de covariance $\mathbf{\Lambda}$ est une matrice diagonale puisque la corrélation qui existe entre les points X est éliminée entre les points Y par la projection.

$$\mathbf{\Sigma} = X^T X \qquad (moyenne \quad E[X] = 0) \tag{3.1}$$

$$\mathbf{\Lambda} = Y^T Y = W X^T X W^T = W \mathbf{\Sigma} W^T \qquad (Y = WX) \tag{3.2}$$

$$\mathbf{\Sigma} W^T = W^T \mathbf{\Lambda} \qquad (W^T = W^{-1}) \tag{3.3}$$

$$\mathbf{\Sigma} \boldsymbol{w_i} = \lambda_i \boldsymbol{w_i} \tag{3.4}$$

Soulignons que 3.3 tient si $d = D$, puisque $W$ devient une matrice de rotation. Puisque $\mathbf{\Lambda}$ doit être diagonale, les colonnes de $W^T$ contiennent les vecteurs propres (*eigenvectors*), et la diagonale de $\mathbf{\Lambda}$ contient les valeurs propres $\lambda_i$ (*eigenvalues*). On choisit les $d$ vecteurs propres $\boldsymbol{w_i}$ qui ont les plus grandes valeurs propres $\lambda_i$ pour construire la matrice de projection W afin de préserver le maximum de variance.

On peut voir dans (KUNG et DIAMANTARAS 1996) que cette opération de maximisation de la variance est équivalente à la minimisation de l'erreur quadratique moyenne de reconstruction. Celle-ci est calculée à partir des observations reconstruites $\tilde{X}_i$.

$$\tilde{X}_i = W^T Y_i = W^T W X_i \tag{3.5}$$

$$R = \sum_{i=1}^{n} \|X_i - \tilde{X}_i\|^2 \tag{3.6}$$

L'analyse en composantes principales maximise la variance de la projection, mais puisqu'elle n'est pas supervisée, elle ne peut utiliser d'information de qualification ou de quantification, donc la projection qu'elle effectue peut rendre la distinction entre deux classes dans l'ensemble $X$ plus difficile.

## 3.2 Le Discriminant linéaire de Fisher

Dans la figure 3.1 on peut imaginer une procédure qui permettrait de trouver une projection linéaire qui ne maximiserait pas la variance entre les points projetés [2], mais qui maximiserait plutôt la distance entre les classes projetées ; cette méthode s'appelle le discriminant linéaire de Fisher(FISHER 1936), c'est une méthode d'apprentissage supervisé, la figure 3.2 en est un exemple.

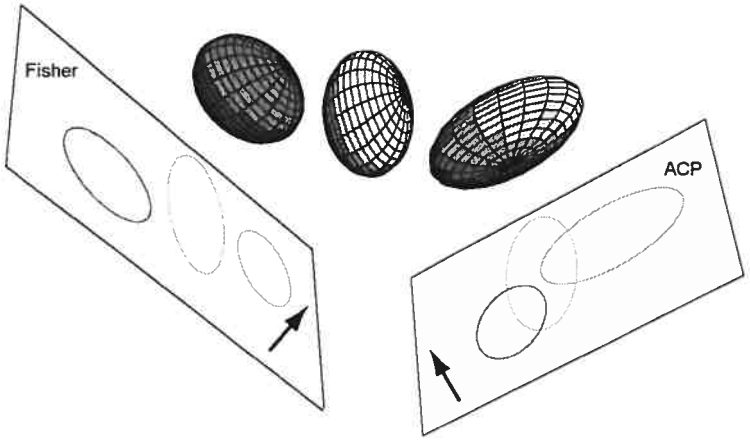



**Figure 3.2** – *Deux exemples de projections : on projette 3 distributions gaussiennes en 3 dimensions sur un espace à 2 dimensions (tiré de (*DUDA, HART *et* STORK *2000) )*

Il est possible de réduire la dimensionnalité d'un problème à $D$ dimensions à *une* seule dimension ; il est seulement nécessaire de faire une transformation linéaire qui projette les points sur une droite. Comme mentionné au paragraphe précédent, si cette droite est choisie arbitrairement, ou en sélectionnant la droite comme étant normale à la direction qui maximise la variance des points, nous n'avons aucune garantie que la projection d'agrégats bien séparable dans l'espace à $D$ dimensions nous donnera une projection

[2] ni ne minimiserait l'erreur quadratique de reconstruction

qui est facilement séparable. Dans la figure 3.3 on peut voir 2 projections sur des droites qui donnent des résultats complètement différents au niveau de la séparabilité, la tâche de trouver l'orientation de projection qui donne une bonne séparabilité est l'analyse du discriminant linéaire. Le Discriminant Linéaire de Fisher donne une matrice de projection $W$ qui déforme la distribution des points afin de maximiser la séparabilité des classes ; celle-ci est définie comme le ratio entre la matrice d'étalement interclasse et celle d'étalement intra classe.

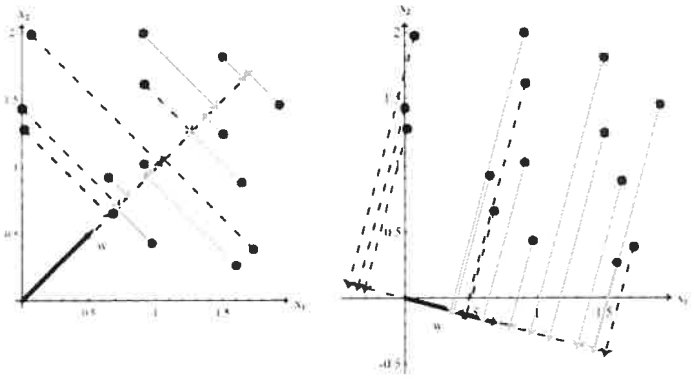


**Figure 3.3** – *2 projections linéaires de points dans un espace à 2 dimensions sur une droite ; la projection de droite résulte en une meilleure séparation entre les classes (tiré de (*DUDA, HART *et* STORK *2000) )*

Une description symbolique explicite de la méthode suit : On note par $Z = (Z_1, Z_2, \ldots, Z_n) \in \mathbb{R}^D$ l'ensemble des observations aléatoires (des vecteurs en dimension $D+1$) ; chacune de ces variables aléatoire est composée du vecteur d'observation $X$(de dimension $D$) et d'une variable aléatoire $L$ [3] qui représente l'étiquette (parmi les $k$ étiquettes possibles) de l'observation $X$. La variable $L$ prend donc une valeur de 1 à $k$ : nous référons à l'ensemble des observations qui prennent la valeur $k$ par $C_k$ ; toutes les variables aléatoires

[3] nous remplaçons ici Y qui dénotait la variable aléatoire d'étiquette par L parce que Y est utilisé précédemment et subséquemment pour dénoter la projection dans l'espace à dimension $d$.

sont donc distribuées entre les sous-ensembles $C_1, C_2, \ldots, C_k$ des données originales. La moyenne de tous les $X$ est notée par $\mu_X$, chaque moyenne des $k$ classes $C_1, C_2, \ldots, C_k$ par $\mu_{Xj}$, la cardinalité de $C_j$ par $|C_j|$, la matrice d'étalement inter-classe par $S_B$[4] et celle intra-classe par $S_W$[5]. La projection est notée Y, et on cherche une matrice W qui répond aux critères d'étalement des observations.

$$Y = W^T X \qquad W \text{ est une matrice de projection} \tag{3.7}$$

$$\mu_X = \frac{1}{n}\sum_{i=1}^{n} X_i \qquad \mu_{Xj} = \frac{1}{|C_j|}\sum_{X_i \in C_j} X_i \tag{3.8}$$

$$S_B = \sum_{j=1}^{k} |C_j|(\mu_{Xj} - \mu_X)(\mu_{Xj} - \mu_X)^T \tag{3.9}$$

$$S_W = \sum_{j=1}^{k} \sum_{X_i \in C_j} (X - \mu_{Xj})(X - \mu_{Xj})^T \tag{3.10}$$

$$On\ maximise\ J(W) : \quad J(W) = \frac{W^T S_B W}{W^T S_W W} \tag{3.11}$$

L'expression 3.11 est connue comme le quotient généralisé de Rayleigh, Les vecteurs propres dans W sont les vecteurs propres généralisés de $S_B S_W^{-1}$. Si on veut projeter dans une dimension $d \ll D$ on sélectionne les d vecteurs propres qui ont les plus grandes valeurs propres pour avoir une matrice $W_d = [\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_d}]$.

## 3.3 Échelonnement multidimensionnel

Le critère que l'ACP (voir 3.1) tente de minimiser est l'erreur de reconstruction (la somme des distances entre un point $X_i$ et sa projection $\tilde{X}_i$, voir équation 3.5) ; l'échelonnement multidimensionnel (aussi appelé *positionnement multidimensionnel*, ou MDS pour "Multidimensional Scaling") tente

[4]B pour *between classes*
[5]W pour *within*

plutôt de construire une configuration en $p$ dimensions [6] ou $p \ll D$ qui minimise la différence entre la distance entre chaque objet dans l'espace original en $D$ dimensions et celle entre leurs images en $p$ dimensions. Cette méthode est non supervisée, puisqu'elle n'utilise pas les étiquettes des points.

La notion qu'à partir de seulement la matrice de distances entre des observations dans un espace euclidien on peut trouver des coordonnées pour ces points qui préservent ces distances a été introduite dans l'article (YOUNG et HOUSEHOLDER 1938) ; l'utilisation de cette méthode pour effectuer un échelonnement a été proposée dans l'article (TORGERSON 1952) et c'est l'article (GOWER 1966) qui l'a mise en valeur. Il est important de noter que les coordonnées des observations résultant de la reconstruction sont l'équivalent d'une transformation $T$ par la rotation, la réflexion ou la translation de la solution produite ; la distance entre les points ne contient aucune information quant à l'orientation de l'espace reconstruit.

$$\dot{Y}_i = TY_i + b \tag{3.12}$$

Le livre "Multidimensional Scaling" de (COX et COX 2001) consacre un chapitre entier à l'analyse de procrustes, la technique pour associer une configuration à une autre, par exemple pour comparer le résultat d'un MDS à un autre.

Il faut noter de plus que des distances euclidiennes dans la matrice d'adjacence ne sont pas des prérequis au MDS. Tout genre de dissimilitude entre les objets peut y être exprimé en autant que cette matrice soit positive et semi-définie (et symétrique, puisque la distance d'un point à une autre est invariante à la direction de mesure). Dans la figure 3.4 on peut voir un exemple de MDS.

Formellement, on cherche à trouver une configuration de points $Y_i$ pour laquelle les distances $d_{ij}$ entre ces points est la plus proche possible des $\frac{n(n-1)}{2}$ distances $\delta_{ij}$ entre les observations originales $X_{ij}$. Puisqu'on ne peut généralement pas espérer trouver une configuration où $\delta_{ij} = d_{ij}$ pour tous les $i$ et $j$,

[6] nous utilisons p pour noter la dimension d'échelonnement parce que $d_{ij}$ est utilisé dans cette méthode pour indiquer l'approximation de distances entre 2 points

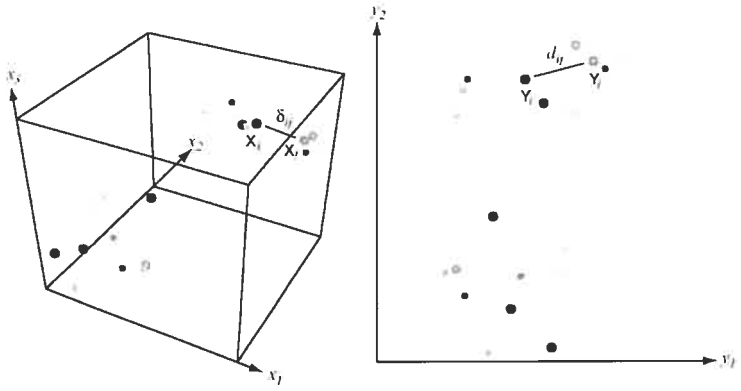


**Figure 3.4** – *Exemple d'échelonnement multidimensionnel. La différence entre la distance $\delta_{ij}$ et la distance $d_{ij}$ entre les points $X_i$ et $X_j$ dans l'espace à 3 dimensions et les points $Y_i$ et $Y_j$ dans l'espace à 2 dimensions est minimisée. (tiré de (*DUDA, HART *et* STORK *2000) )*

nous avons besoin d'un critère pour choisir laquelle des approximations est la meilleure ; ces différents critères possibles donnent leur nom à certaines variantes de MDS ; nous allons détailler ici l'échelonnement classique, parfois aussi appelé analyse en coordonnées principales, ou échelonnement métrique ; mais ce dernier terme désigne une approche plus générale. La description sommaire de l'algorithme suit, les différents symboles utilisés ainsi que leur signification mathématique suivra.

La matrice B est le résultat d'un centrage à l'origine obtenu à l'aide du centroïde de la configuration de points dans l'espace $D$ ; cette procédure doit être faite pour éviter que la solution soit indéterminée à cause d'une translation arbitraire.

Nous avons donc pour des points $i$ et $j$ la distance carrée :

$$d_{ij}^2 = (X_i - X_j)^T(X_i - X_j) = |X_i - X_j|^2 \tag{3.13}$$

**Tableau 3.1** — *Algorithme MDS*

| | |
|---|---|
| 1. | Calculer les distances euclidiennes $\Delta = [\delta_{ij}]$ |
| 2. | Calculer $A = [-\frac{\delta_{ij}^2}{2}]$ |
| 3. | Calculer $B = [a_{ij} - a_{i.} - a_{.j} + a_{..}]$ |
| 4. | Trouver les valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ et leur vecteurs propres respectifs $\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_n}$ normalisés de façon à ce que $\boldsymbol{y_i^T y_i} = \lambda_i$. |
| 5. | Choisir un nombre p approprié de dimensions. On peut utiliser le ratio $\frac{\Sigma_{i=1}^{p} \lambda_i}{\Sigma i=1^{n-1} \lambda_i}$ (où on ignore les $\lambda_i$ négatifs s'il y en a) |
| 6. | Les coordonnées des n points dans l'espace à p dimensions sont les $y_{ij}$ tel que $i = 1, 2, \ldots, n$ et $j = 1, 2, \ldots, p$. |

La matrice $B$ des produits intérieurs est : $B = [b_{ij}] = X_i^T X_j$. On veut que :

$$\sum_{i=1}^{n} X_{ij} = 0 \quad (j = 1, 2, \ldots, D)$$

Pour trouver B, nous avons de 3.13 que :

$$d_{ij}^2 = X_i^T X_i + X_j^T X_j - 2X_i^T X_j \tag{3.14}$$

$$donc, \quad \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 = \frac{1}{n}\sum_{i=1}^{n} X_i^T X_i + X_j^T X_j \tag{3.15}$$

$$\frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 = X_i^T X_i + \frac{1}{n}\sum_{j=1}^{n} X_j^T X_j \tag{3.16}$$

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 = \frac{2}{n}\sum_{i=1}^{n} X_i^T X_i. \tag{3.17}$$

En remplaçant dans 3.14 nous avons :

$$b_{ij} = X_i^T X_j \tag{3.18}$$

$$= -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2\right) \tag{3.19}$$

$$= [a_{ij} - a_{i.} - a_{.j} + a_{..}] \tag{3.20}$$

$B = [a_{ij} - a_{i.} - a_{.j} + a_{..}]$ est l'équation de centrage,

$$a_{j.} = \frac{1}{n}\sum_{j=1}^{n} a_{ij}$$

est le vecteur des moyennes des colonnes,

$$a_{.i} = \frac{1}{n}\sum_{i=1}^{n} a_{ij}$$

est le vecteur des moyennes des lignes, et

$$a_{..} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}$$

est la moyenne de tous les points.

$A = [a_{ij}]$ contient donc $-\frac{\Delta}{2} = -\frac{d_{ij}^2}{2}$.

On peut définir la matrice des produits intérieurs B par :

$$B = HAH$$

où $H$ est la matrice de centrage

$$H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$$

où $\mathbf{1} = [1, 1, \ldots, 1]$ est un vecteur de $n$ 1.

Cette matrice $B$ de produits intérieurs peut aussi être exprimée comme

suit :

$$B = XX^T$$

où $X = [X_1, X_2, \ldots, X_n]$ est la matrice $n \times D$ des coordonnées. Le rang de $B$ noté $r(B)$ est :

$$r(B) = r(XX^T) = r(X) = D$$

$B$ est donc symétrique, semi-définie et de rang $D$ ; elle a donc $D$ valeurs propres non négatives et $n - D$ valeurs propres égales à zéro. On peut donc écrire $B$ en termes de sa décomposition spectrale :

$$B = V\Lambda V^T$$

où $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ est la matrice diagonale des valeurs propres, et $V = [\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}]$ est la matrice des vecteurs (ou fonctions) propres correspondantes, normalisés tel que $\boldsymbol{v_i}^T\boldsymbol{v_i} = 1$ ; pour que $\boldsymbol{v_i}^T\boldsymbol{v_i} = \lambda_i$ nous multiplions les vecteurs propres par $\Lambda^{\frac{1}{2}}$.

$$Y = V\Lambda^{\frac{1}{2}}$$

Il est intéressant de noter que lorsque toutes les mesures de dissimilitudes sont euclidiennes, à la page 43(section 2.2.7) du livre (Cox et Cox 2001) les auteurs démontrent qu'il y a une relation de dualité entre l'analyse en composante principale et l'échelonnement classique.

De X nous obtenons la matrice de covariance des exemples $\Sigma = \frac{1}{(n-1)}X^TX$ [7]. Les composantes principales sont obtenues en trouvant les valeurs propres $\boldsymbol{\mu_i}$ et les vecteurs propres $\xi_i$ de $\Sigma$ ; la $i$ème composante principale est donnée par $y_i = \xi_i^T X$. On peut voir (Chatfield et Collins 1980) ou (Mardia, Kent, et Bibby 1979) pour un exemple et une démonstration formelle.

Nous avons que :

$$B = XX^T$$

Il est connu que les valeurs propres de $XX^T$ sont les mêmes que celles de

[7] nous assumons que les données ont été modifiées pour que les moyennes soient zéro.

$X^TX$. Si $\boldsymbol{v_i}$ est un vecteur propre de $XX^T$ nous avons :

$$XX^T\boldsymbol{v_i} = \lambda_i\boldsymbol{v_i}. \tag{3.21}$$

En prémultipliant par $X^T$,

$$(X^TX)(X^T\boldsymbol{v_i}) = \lambda_i(X^T\boldsymbol{v_i}) \tag{3.22}$$

mais :

$$X^TX\xi_i = \mu_i\xi_i \tag{3.23}$$

donc $\mu_i = \lambda_i$ et les vecteurs propres ont la relation : $\xi_i = X^T\boldsymbol{v_i}$. La relation de dualité entre l'analyse en composantes principales et l'échelonnement classique quand les distances sont euclidiennes est donc exposée.

Les différentes méthodes de réduction de dimensionnalité linéaire présentées ici ainsi que d'autres non mentionnées sont d'excellents outils qui permettent de disposer un problème statistique dans une dimension plus petite, mais elles souffrent toutes d'une défaillance majeure ; si la structure intrinsèque du sous-espace de dimensionnalité réduite que l'on nomme généralement variété[8] n'est pas de forme linéaire, ces algorithmes ne peuvent extraire aucune structure cohérente ; leur résultat sera inutilisable. Nous allons voir au chapitre suivant des méthodes qui pallient à cette déficience.

[8]que l'on nomme le *manifold* en anglais.

CHAPITRE 4

# Réduction de dimensionnalité non linéaire

Le défaut central des méthodes de réduction de dimensionnalité linéaire se résume très simplement ; elles sont limitées à performer l'extraction d'une variété en utilisant une projection linéaire. Intuitivement, ces méthodes accordent une importance croissante en fonction de la distance entre deux points ; plus le nombre de dimensions d'un problème est élevé, plus il existe de façons pour 2 points d'être grandement différents ; on peut facilement imaginer que l'inverse n'est pas vrai : le nombre de dimensions n'influent pas beaucoup sur comment 2 points proches dans un espace euclidien sont similaires. En bref, en haute dimensionnalité, dans une matrice de distances d'adjacence,[1] les courtes distances contiennent plus d'information sur le phénomène observé que les longues distances et les méthodes que nous avons vues au dernier chapitre font abstraction de cette donnée structurante. Les méthodes présentées dans ce chapitre tentent de remédier à cet état de faits.

[1] matrice de distances $\delta_{ij}$ de chaque points à chaque points.

## 4.1 Isomap

L'algorithme Isomap (TENENBAUM, DE SILVA et LANGFORD 2000) est une méthode d'apprentissage statistique non supervisée qui extrait d'un ensemble de données une structure géométrique globale par une réduction de dimensionnalité non linéaire. On tente donc de trouver des coordonnées $y_i$ pour les points $X_i$ dans une variété euclidienne de dimensionnalité $d$ moindre que $D$ qui capture les degrés de liberté intrinsèque de l'ensemble de données, et ce en minimisant l'erreur de reconstruction. Cet algorithme est une permutation de l'échelonnement multidimensionnel en ce qu'on ajoute une étape entre le calcul des distances euclidiennes de points à points et la recherche des valeurs propres de la matrice de Gram ; cette nouvelle étape introduit le concept de voisinage.

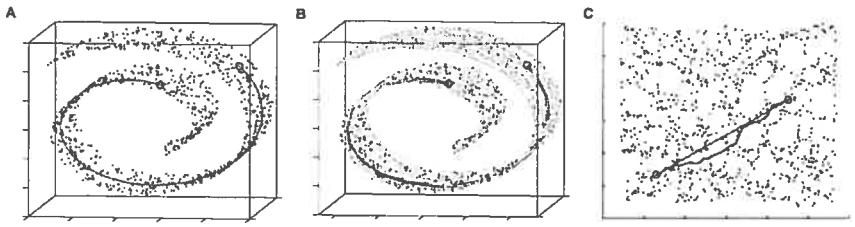



**Figure 4.1** – *On voit ici un exemple de variété qui existe dans un espace latent à plus haute dimensionnalité ; deux points qui sont relativement près au sens euclidien du terme (trait pointillé dans la figure A) ne le sont pas nécessairement en tenant compte de la distance géodésique sur la variété, qui est la réelle mesure de distance. On peut approximer cette distance géodésique en utilisant le voisinage de chaque point, en calculant la distance de tout les points à tout les points en n'utilisant que ces distances euclidiennes définies localement on trouve les plus courts chemins entre points éloignés à l'aide de ces distances. En C on voit la reconstruction en 2 dimensions du "roulé suisse", on constate l'approximation à l'aide des plus courts chemins de la vraie distance géodésique pour deux points donnés.(tiré de (*TENENBAUM, DE SILVA *et* LANGFORD *2000))*

### 4.1.1 Voisinage

On distingue généralement deux types de voisinage : soit on définit pour tous les $X_i$, un nombre $k$ de points parmi lesquels on considère que la variété intrinsèque est linéaire, et on insère dans la matrice de Gram la distance euclidienne à ce nombre fixe de $k$ voisins, soit on choisit une distance $\epsilon$ fixe, sur chaque point on fixe une boule de rayon $\epsilon$ et on considère le voisinage comme tous les points étant à l'intérieur de cette boule. Dans cette dernière définition, la cardinalité des ensembles de voisinage est plus variable d'un point à l'autre que pour le cas des $k$ plus proches voisins.

C'est ici l'étape qui différencie diamétralement Isomap de l'échelonnement multidimensionnel ; à partir *seulement* des voisinages pour chaque point, on calcule les plus courts chemins de tous les points à tout les points puis on centre à l'origine avec l'algorithme du centroïde, aussi appelé l'opérateur $\tau$, défini par :

$$\tau(A) = -\frac{HSH}{2} \tag{4.1}$$

où $S$ est la matrice des distances au carré :

$$S_{ij} = \delta_{ij}^2 \tag{4.2}$$

et $H$ est la matrice de centrage :

$$H = \delta_{ij} - \frac{1}{n}. \tag{4.3}$$

(Voir (MARDIA, KENT, et BIBBY 1979) pour plus de détails sur l'opérateur $\tau$.) C'est sur cette nouvelle matrice de Gram que l'on applique la décomposition spectrale afin de trouver les valeurs et vecteurs propres, comme on le fait pour l'échelonnement positionnel (3.3). Le tableau 4.1 présente une description de l'algorithme Isomap.

L'ensemble $Y$ de coordonnées $y_i$ dans l'espace à $d$ dimensions minimise la

**Tableau 4.1** – *Algorithme Isomap*

1. Construire le graphe de voisinage : On définit un graphe $G$ sur l'ensemble des observations, et pour tous les couples $(i, j)$ possibles, on y connecte $i$ et $j$ par la distance euclidienne $\delta_{ij}$, soit s'ils sont plus proches que le paramètre de voisinage $\epsilon$ ($\epsilon$-Isomap), ou si $j$ est un des $k$ plus proches voisins de i. ($k$ étant l'autre paramètre de voisinage possible pour la version $k$-Isomap)
2. Calculer les plus courts chemins : On initialise la matrice d'adjacence $A$ à $\delta_{ij}$ si les points $i$ et $j$ possèdent une arête dans le graphe $G$ et à $\infty$ dans le cas contraire. Pour chaque $k = 1, 2, \ldots, n$, on remplace successivement $A_{ij}$ par $\min\{A_{ij}, A_{i,k} + A_{k,j}\}$ ; la matrice $A$ contient donc les plus courts chemins entre chaque paire de points dans le graphe G étant donné les distances euclidiennes du voisinage données en 1.
3. Comme en 3. dans MDS, on applique l'opérateur $\tau$ à la matrice $A$ et on obtient la matrice $B$.
4. Trouver les valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ de $B$ et leur vecteurs propres respectifs $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}$.
5. Choisir les $d$ valeurs et vecteurs propres les plus significatifs en ordre décroissant : les coordonnées des points $y_i$ de l'incorporation dans l'espace à $d$ dimensions sont données par : $y_i = \sqrt{\lambda_p}\boldsymbol{v_n^i}$ où $\boldsymbol{v_n^i}$ est la $i$-ième composante du $p$-ième vecteur propre.

fonction de coût :

$$R(Y) = |\tau(A) - \tau(A_d)|_{L^2} \tag{4.4}$$

où $A_d$ dénote la matrice des distances euclidiennes entre les $Y_{ij}$ dans l'espace à $d$ dimensions

$$A_d(i,j) = |y_i - y_j| \tag{4.5}$$

et $|A|_{L^2}$ est la norme $L^2$ de la matrice $A$ :

$$|A|_{L^2} = \sqrt{\sum_{i,j} A_{ij}^2}. \tag{4.6}$$

Notons que si on établit que le paramètre de voisinage $k$ à $n$ (où le $\epsilon$ à $\max(\delta_{ij})$ pour la version $\epsilon$-Isomap ) Isomap est équivalent à l'échelonnement multidimensionnel classique : On peut donc en déduire qu'Isomap est une généralisation de l'échelonnement multidimensionnel.

### 4.1.2 Garantie Asymptotique de Convergence

Cette dernière constatation comporte une implication très désirable ; comme pour l'analyse en composante principale et l'échelonnement multidimensionnel qui incluent une garantie de retrouver la vraie structure de variétés linéaires étant donné un nombre suffisant de points Isomap possède la même implication asymptotique. L'algorithme possède la garantie théorique de récupérer la vraie dimensionnalité ainsi que la structure géométrique d'un ensemble strictement plus grand de variétés (possiblement non-linéaires) que les méthodes précédentes. Ces variétés, comme les exemples des Figures 4.1, 4.2 et 4.3, doivent posséder une géométrie intrinsèque équivalente à une région convexe d'un espace euclidien. Si l'espace n'est pas euclidien, par exemple si la variété repose sur une sphère ou sur un torus dans l'espace latent, le système d'équation de la matrice de Gram serait contradictoire, et la projection résultante serait linéaire.


**Figure 4.2** – *Un champ d'application classique pour la réduction de dimensionnalité est la vision artificielle. Le domaine des données d'entrée est défini sur des variables aléatoires en 4096 dimensions, où chaque dimension représente le ton de gris (une valeur sur 16 bits, entre 0 et 65535 par exemple) d'un pixel d'une image de 64 pixels par 64 pixels. Ces images, au nombre de 698, sont celles d'un visage généré par ordinateur sous différentes poses et conditions d'éclairage. En appliquant k-Isomap avec un voisinage $k = 6$ et une dimensionnalité de variété résultante de $d = 3$ on apprend l'incorporation suivante : les deux premières dimensions découvertes sont sur l'abscisse et l'origine ; elles sont fortement corrélées avec la pose verticale et horizontale. La troisième dimension découverte est l'angle d'éclairage, elle est dénotée avec une case de défilement sous chaque image associée au point encerclé dans le plan. (Tiré de* (TENENBAUM, DE SILVA *et* LANGFORD *2000))*

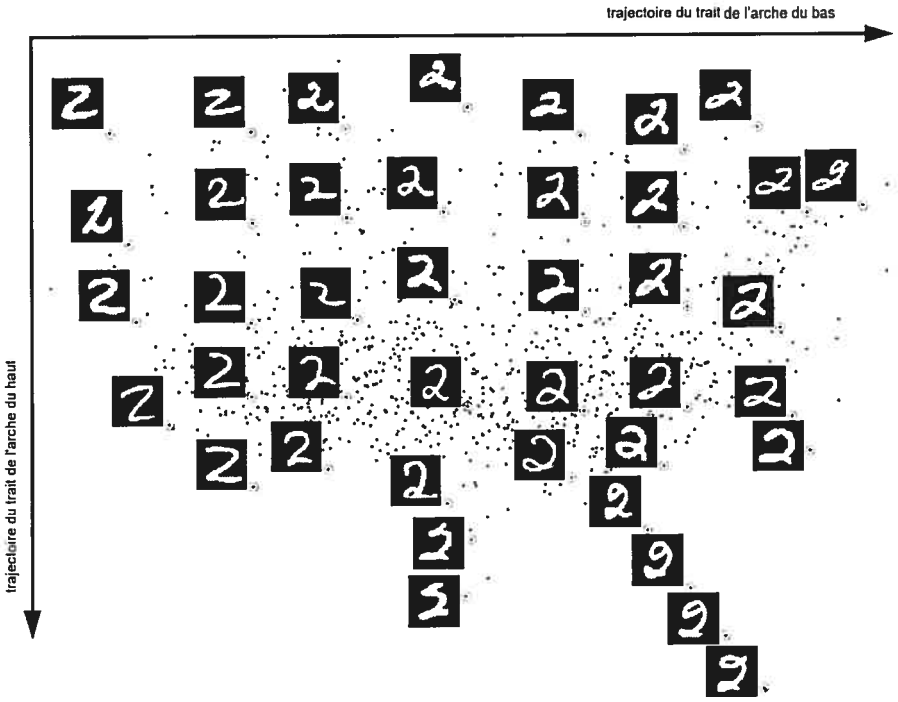



**Figure 4.3** – *L'algorithme Isomap appliqué à 1000 "2" en 4096 dimensions de la base de chiffres manuscrits de MNIST. Les deux dimensionnalités les plus significatives extraites à l'aide de la version $\epsilon$-Isomap avec un $\epsilon = 4.2$ ; elle sont fortement corrélées avec la tracé du crayon pour la boucle d'en haut et celui de la boucle d'en bas. (Tiré de (*TENENBAUM, DE SILVA *et* LANGFORD *2000))*

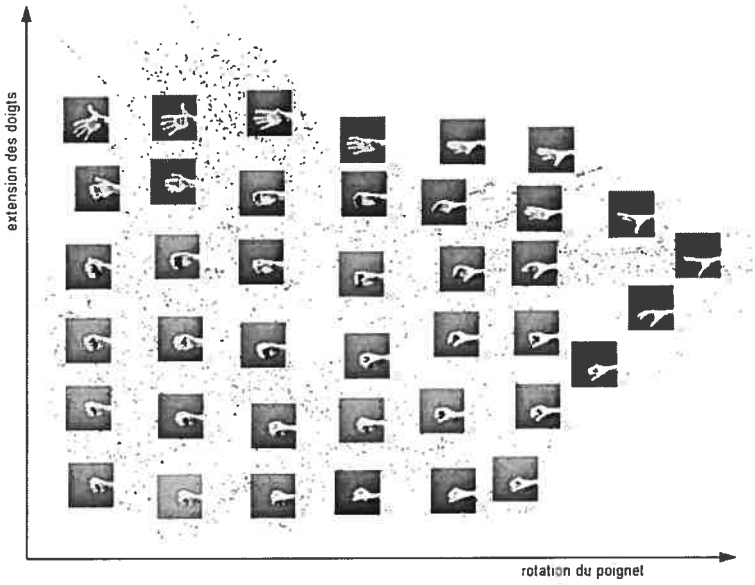



**Figure 4.4** – *L'algorithme Isomap appliqué à des images en 4096 dimensions de mains humaines dans différentes positions. Une des dimensionnalités les plus significatives extraites est le mouvement naturel, même si ce mouvement n'était pas présent dans les données originales; on peut y voir une certaine extraction de temporalité. (Tiré de (*TENENBAUM, DE SILVA *et* LANGFORD *2000))*

La preuve mentionnée dans (TENENBAUM, DE SILVA et LANGFORD 2000) tourne autour de la démonstration que, pour un échantillon assez dense de $\alpha$ points, on peut toujours choisir un voisinage (soit de taille $\epsilon$ ou en nombre de voisins $k$) assez grand pour que le chemin entre tous points $i$ et $j$ sur le graphe $G$ ne soit pas tellement plus grand que la vraie distance géodésique $M_{ij}$ sur la variété, mais assez réduit pour ne pas causer de courts-circuits entre les parties de la vraie géométrie de la variété dans l'espace latent. De façon plus formelle, étant donné des valeurs arbitrairement petites pour $\lambda_1$, $\lambda_2$ et $\mu$, nous pouvons garantir avec une probabilité d'au moins $1 - \mu$ qu'une estimation de la forme :

$$(1 - \lambda_1)M_{ij} \leq A_{ij} \leq (1 + \lambda_2)M_{ij} \tag{4.7}$$

sera valide pour toutes les paires de points $i$ et $j$.

Pour $\epsilon$-Isomap nous avons :

$$\epsilon \leq (\frac{2}{\pi})r_0\sqrt{24\lambda_1} \qquad \epsilon < s_0 \tag{4.8}$$

$$\alpha > \frac{\left(\log\left(\frac{V}{\mu}\eta_d(\frac{\lambda_2\epsilon}{16})^d\right)\right)}{\eta_d(\frac{\lambda_2\epsilon}{8})^d} \tag{4.9}$$

où $r_0$ est le rayon de courbure minimale de la variété $M$ comme incorporé dans l'espace latent $X$, $s_0$ est la distance minimale séparant les replis dans l'espace $X$, $V$ est le volume (en $d$ dimensions) de la variété $M$, et $\eta_d$ est le volume de la boule unitaire dans l'espace euclidien de $d$ dimensions en ignorant les effets de bord. Pour $k$-Isomap, nous permettons à $\epsilon$ de prendre la même valeur que ci-dessus et nous fixons le ratio à :

$$\frac{(k+1)}{\alpha} = \frac{\eta_d(\frac{\epsilon}{2})}{2} \tag{4.10}$$

Nous demandons que :

$$e^{-\frac{(k+1)}{4}} \leq \frac{\mu\eta_d(\frac{\epsilon}{4})^d}{4V} \tag{4.11}$$

$$(\frac{e}{4})^{\frac{(k+1)}{2}} \leq \frac{\mu\eta_d(\frac{\epsilon}{8})^d}{16V} \tag{4.12}$$

$$\alpha > \frac{\left(4\log(\frac{8V}{\mu\eta_d}(\frac{\lambda_2\epsilon}{32\pi})^d)\right)}{\eta(\frac{\lambda_2\epsilon}{16\pi})^d} \tag{4.13}$$

Pour plus de détails sur le contenu exact de ces bornes et sur les hypothèses techniques que les auteurs adoptent pour justifier celles-ci, voir (BERNSTEIN, DE SILVA, LANGFORD et TENENBAUM 2000) .

### 4.1.3 Avantages

L'avantage principal d'Isomap par rapport à d'autres algorithmes effectuant une réduction de dimensionnalité non linéaire comme LLE (voir 4.2) est la stabilité topologique, découlant de la garantie asymptotique de convergence sur un ensemble assez grand de variétés. On peut voir dans la Figure 4.5 le résultat en termes de variance résiduelle et du nombre de dimensionnalité pour les applications mentionnées d'Isomap.

Les variétés "conformallement" équivalentes à des espaces euclidiens, comme le "bol à poisson" (en anglais le *fishbowl*), ne sont pas nécessairement bien traitées par Isomap : dans l'article (TENENBAUM et DE SILVA 2002), on introduit une variante *C-Isomap* qui donne de meilleurs résultats sur ce genre de variétés ; on pondère simplement les plus courts chemins par la moyenne des distances du voisinage avant de calculer la diagonalisation. Cette opération se fait au détriment de la garantie de convergence asymptotique, et pour des variétés qui ne sont pas repliées de façon "conformes", l'incorporation résultante est moins topologiquement précise.

### 4.1.4 Inconvénients

Un inconvénient majeur d'Isomap par rapport à certains autres algorithmes comme LLE est le temps de calcul et l'espace mémoire ; la matrice de Gram

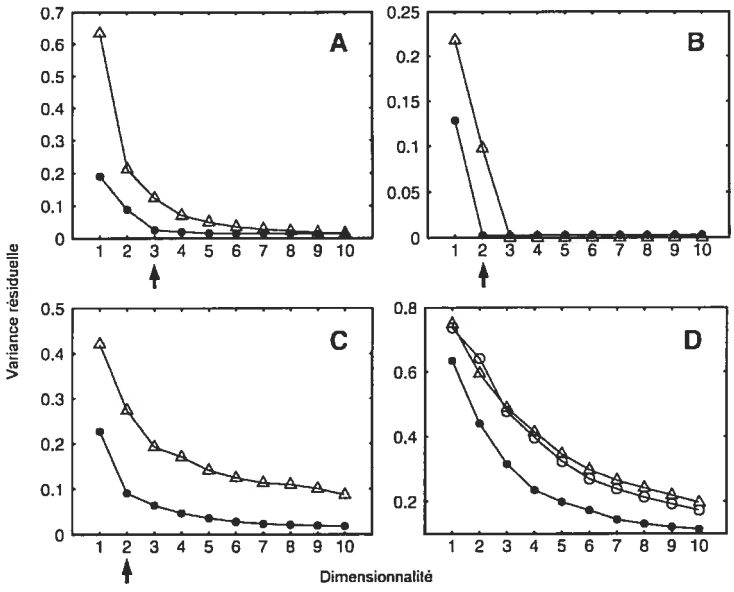



**Figure 4.5** – *La variance résiduelle des données après l'application de l'algorithme Isomap. En A nous avons les visages en B le roulé suisse, en C les images de main, et en D les "2" manuscrits. Pour évaluer la dimensionnalité intrinsèque on cherche le coude dans la courbe. Les triangles vides sont la variance avec l'échelonnement multidimensionnel et les points pleins sont la variance résiduelle pour Isomap (tirée de (*TENENBAUM, DE SILVA *et* LANGFORD *2000))*

est dense et occupe $O(n^2)$. L'étape deux du calcul des plus courts chemins est une borne assez dure; pour l'instant notre implémentation tourne en $O(n(a+n)\log(n))$ où $a$ est le nombre d'arêtes dans le graphe $G$, en utilisant un tas de fibonacci. ( Voir les notes sur Djikstra à la section 6.4 de (BRASSARD et BRATLEY 1996) pour plus de détails.) L'étape finale d'extraction de valeurs et vecteurs propres est aussi très exigeante; la version non optimisée prend $O(n^3)$ avec un assez grand coefficient, nous allons voir quelques optimisations possibles au chapitre 5.

### 4.1.5 Généralisation

Les auteurs de l'algorithme original dans (TENENBAUM et DE SILVA 2003) proposent une méthode pour mitiger la charge computationnelle quand le problème devient grand; ils proposent le "landmark Isomap". Cette version diffère de la première en ce qu'on choisit un sous-ensemble de points, on calcule l'incorporation pour cet ensemble de points, puis on calcule le plongement des points restants étant donné le sous-ensemble de points utilisés par l'algorithme en utilisant ceux-ci comme balises. Évidemment, de cette solution pour réduire le coût computationnel résulte une perte au niveau de la stabilité topologique, mais nettement moindre qu'on pourrait croire; cette perte est fortement reliée à la complexité de la variété dans l'espace latent.

Dans l'article de *Neural Computation* (BENGIO, DELALLEAU, PAIEMENT, VINCENT, OUIMET et ROUX 2004) [2] les auteurs proposent un cadre théorique pour l'extension de ce type d'algorithme (dont Isomap et LLE (4.2) ) à de nouveaux points. Cette généralisation de "landmark Isomap" qu'on retrouve dans l'article (TENENBAUM et DE SILVA 2003) est importante afin de prévoir un usage des méthodes de réduction de dimensionnalité à des fins de généralisation en évitant de recalculer l'algorithme sur le nouvel ensemble composé de tous les points originaux et du nouveau point à prédire.

[2] autres détails dans le rapport technique (?)bengioJFPaiement03)

## 4.2 LLE

L'algorithme LLE, pour "Local Linear Embedding" (ROWEIS et SAUL 2000) est aussi une méthode d'apprentissage statistique qui tient compte de la possibilité que la structure interne de distribution des données soit de dimensionnalité plus petite que celle des données, et que cette distribution soit située sur une variété non linéaire[3] ; elle cherche une approximation de cette variété. Cette méthode est non-supervisée car elle n'inclut aucune information de qualification ou de quantification des points de données ; on peut voir l'effet de l'algorithme dans la figure 4.6.

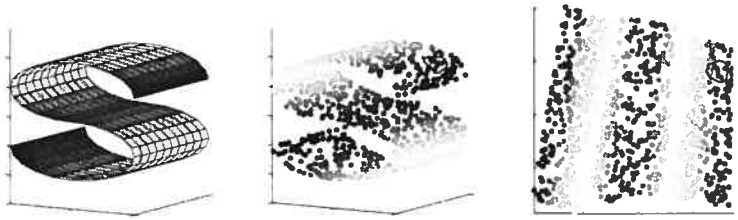


**Figure 4.6** – *Une illustration de LLE : on extrait d'un ensemble de données en 3 dimensions qui a clairement une distribution à dimensionnalité moindre sous-jacente l'approximation de cette distribution en préservant autant que possible la structure locale des données ; dans B et dans C on voit encerclés quelques points qui sont proches, leur relative proximité est préservée (tirée de (*ROWEIS et SAUL *2003))*

LLE est différent en plusieurs points d'Isomap (4.1) en ce qu'il n'utilise pas de concept de plus courts chemins. Intuitivement, la différence est que chaque point de données est plutôt pondéré par son voisinage de taille $k$ de sorte que la somme des pondérations égale 1 ; c'est ce voisinage qui est considéré comme étant linéaire. La table 4.2 est une description de l'algorithme, les concepts mathématiques utiles à sa compréhension suivent.

Formellement, nous avons un vecteur de poids $W_i$ de taille $n$ ($n$ étant le la taille de l'ensemble $S_n$) associé à chacune des variables aléatoires $X_i$. Ce vecteur est soumis à deux contraintes ; premièrement pour que le point $i$ ne

[3] une variété linéaire serait un hyperplan

**Tableau 4.2** – *Algorithme LLE*

| | |
|---|---|
| 1. | Affecter un voisinage à chaque point, par exemple en utilisant les $k$ plus proches voisins. |
| 2. | Calculer la matrice de poids $W_{ij}$ qui reconstruit linéairement le mieux $X_i$ à partir de ses voisins. |
| 3. | Trouver les valeurs propres de $W$ $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ et leurs vecteurs propres respectifs $\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_n}$. |
| 4. | Choisir les $d$ valeurs propres les plus significatives : les coordonnées des points et l'incorporation en $d$ dimensions sont données par leurs $d$ vecteurs propres associés. |

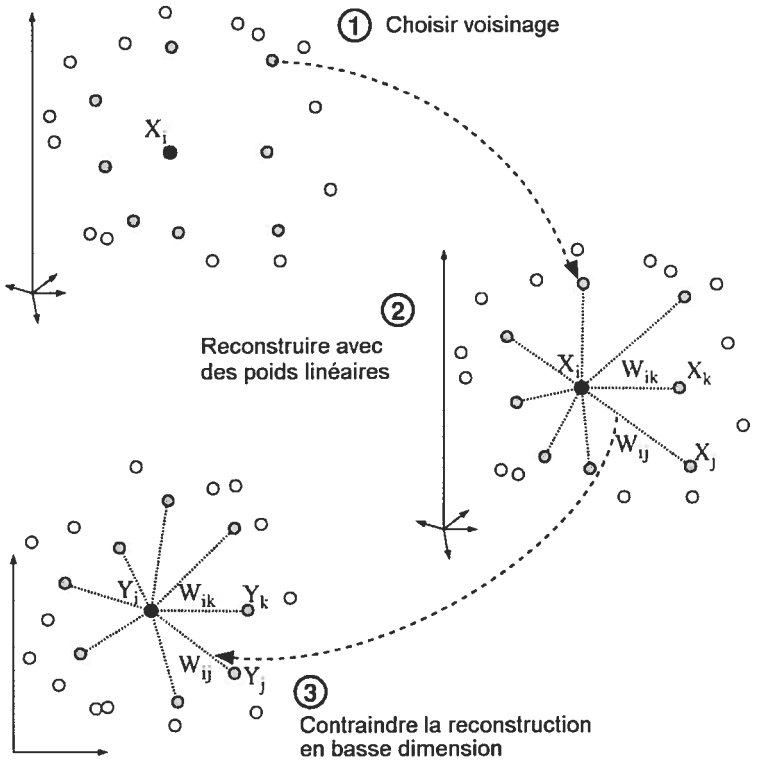



**Figure 4.7** – *Description visuelle de la reconstruction avec linéarité locale. Tiré de (*ROWEIS *et* SAUL *2000)*

soit reconstruit qu'à partir de son voisinage immédiat : nous imposons donc que $W_{ij} = 0$ si $X_j$ ne fait pas partie du voisinage de $X_i$ ; la deuxième contrainte est que le vecteur somme à 1.

$$\sum_j W_{ij} = 1 \quad i \in (1, 2 \ldots, n) \tag{4.14}$$

Nous tentons donc de trouver $W$ qui minimise l'erreur de reconstruction suivante en respectant la contrainte précédente :

$$R(W) = \sum_i \left| X_i - \sum_j W_{ij} X_j \right|^2 \tag{4.15}$$

### 4.2.1 Voisinage de LLE

Pour désigner le voisinage immédiat (voir 4.1.1 pour plus de détails sur le voisinage) nous avons les deux options suivantes : définir un voisinage de taille $k$ fixe(voir la Figure 4.7), ou un voisinage à l'aide d'une boule de rayon $\epsilon$ centrée sur le point en question. Ces deux options impliquent le calcul de la matrice des distances euclidiennes de tout les points à tout les points. La pondération d'un point à l'aide de son voisinage est une opération de moindres carrés, et la matrice $W$ en est le résultat.

On peut voir à la figure 4.8, l'approximation linéaire de $X_i$ à partir de son voisinage en 2 dimensions.

À l'aide de cette matrice $W$, à la dernière étape de l'algorithme on cherche le rendu, les coordonnées $Y_i$ qui représentent la position de $X_i$ dans l'espace global à $d$ dimensions. Cela revient à minimiser la fonction de coût :

$$\Phi(Y) = \sum_i \left| Y_i - \sum_j W_{ij} Y_j \right|^2 \tag{4.16}$$

Cette fonction est similaire à la précédente en ce que'elle aussi est basée sur la minimisation de l'erreur de reconstruction linéaire locale, mais ici on fixe $W_{ij}$ en optimisant $Y_i$. La reconstruction est calculée directement à partir de

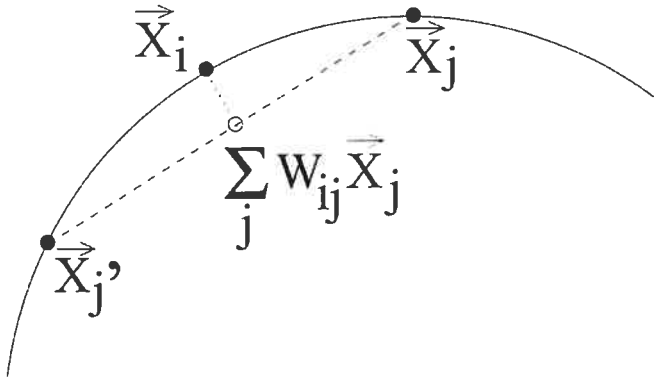



**Figure 4.8** – *Reconstruction de $X_i$ avec linéarité locale à partir de ses voisins (Tiré de (*ROWEIS *et* SAUL *2003))*

la matrice éparse[4] définie et semi positive de poids $W_{ij}$ ; les points $X_i$ ne sont pas utilisés à cette étape de l'algorithme. Le but est donc de trouver des coordonnées $Y_i$ qui ont les mêmes poids $W_{ij}$ que leur points respectifs $X_i$ en haute dimensionnalité.

Cette minimisation est équivalente à calculer les valeurs et les vecteurs propres de la matrice éparse $W$ de dimension $n \times n$ ; les $d$ vecteurs propres correspondant aux $d$ plus grandes valeurs propres sont les coordonnées en basse dimension. On peut changer ce $d$ pour incorporer plus ou moins de dimensions sans recalculer la décomposition de $W$.

Il est important de noter que la linéarité locale est préservée, mais de façon invariante à la translation, à la rotation où à l'échelonnement : pour préserver cette linéarité locale, chacune des "patch" locales peut subir une ou plusieurs des précédentes transformations. L'algorithme n'est pas sujet à l'ordre d'apprentissage, puisqu'il n'y a pas d'ordre dans lequel les points sont traités. À noter au sujet de ces méthodes : il n'y a pas d'entrée aléatoire à l'initialisation, ce qui limite la variabilité des résultats ; ceci est une différence

[4]on dit aussi souvent creuse

majeure de l'approche de réduction avec les réseaux de neurones (les réseaux tronqués, voir (DUDA, HART et STORK 2000)).

### 4.2.2 Avantages et Inconvénients

Un avantage important de LLE par rapport à Isomap est le temps de calcul ; la matrice d'adjacence est éparse. Il existe des méthodes pour extraire les valeur propres et leur vecteurs propres respectifs rapidement en exploitant le fait que cette matrice est éparse et symétrique (Voir l'article (ROWEIS et SAUL 2003) pour plus de détails sur le temps de calcul).

Le principal désavantage de cette méthode est qu'elle ne trouve pas la structure topologique globale de la variété ; la topologie locale de différentes sous-variétés linéaires peut influencer l'ensemble de l'incorporation.

## 4.3 Méthodes à Noyau

Cette section n'offre qu'un bref survol des méthodes à noyaux. Comme introduction à celle-ci, on y aborde les SVM. Ces derniers sont plutôt une famille de méthodes de classification et de régression et non de réduction de dimensionnalité, mais puisque ce sont des méthodes à noyau très simples elle remplissent bien leur fonction d'introduction.

On aborde ici aussi une approche permettant de généraliser toute une classe d'algorithmes de réduction de dimensionnalité à l'apprentissage d'un Noyau, cette généralisation conceptuelle amène à l'introduction d'une méthode qui permet la généralisation au sens pratique à un nouveau point. Nous allons voir ces deux généralisations plus loin, c'est par la seconde qu'il est conséquent d'inclure cette section dans ce chapitre bien qu'elle pourrait sembler hors contexte.

Le récent regain d'intérêt pour les méthodes à noyaux est attribué au succès de méthodes de classification à marges maximales dans un espace de caractéristiques [5] (BOSER, GUYON et VAPNIK 1992). Mais l'introduction du

[5] la famille de méthodes que l'on nomme SVM, c'est un acronyme anglais pour : *Support*

théorème de Mercer dans la communauté de l'apprentissage statistique peut être retracée jusqu'à (AIZERMAN, BRAVERMAN et ROZONOER 1964) ; c'est cependant le premier article mentionné qui présente l'interprétation des noyaux comme des produits scalaires dans un espace de caractéristiques potentiellement infini.

Ces méthodes reposent sur un fondement mathématique dont le développement est dû à J.Mercer(MERCER 1909). Ce développement s'est poursuivi dans les années 1940 avec l'étude des noyaux reproduisants dans des espaces de Hilbert (ARONSZAJN 1950).

L'intérêt algorithmique et théorique des SVM repose essentiellement sur ce qui a été nommé *l'astuce du noyau*[6] : Intuitivement l'idée est la suivante, en effectuant une projection des points originaux dans un espace de caractéristiques potentiellement infini (qui n'est pas nécessairement un sous-espace de $\mathbb{R}^d$) nous pouvons trouver une surface de décision linéaire (SVC[7] dans ce cas) qui sépare les points dans cet espace ; une surface non linéaire dans l'espace original correspond à celle-ci.

Représenter ces points dans cet espace potentiellement infini pourrait se révéler une tâche à toutes fins pratiques impossible : mais grâce à une propriété mathématique distincte des noyaux reproduisants il n'est pas nécessaire de calculer ces projections dans cet espace de Hilbert potentiellement infini. Grâce à celle-ci, on peut effectuer les calculs sur les produits scalaires des points en lieu de leur projection dans l'espace de Hilbert. Cette dernière constatation implique que nous pouvons utiliser la plus grande partie des méthodes linéaires sur ces noyaux, et ce faisant, générer des versions non -linéaires de ces algorithmes.

Il est possible de résumer un noyau noté $k$ comme une mesure de similarité ou de dissimilitude entre deux objets :

$$\begin{aligned} K : \mathcal{X} \times \mathcal{X} &\rightarrow \mathbb{R} \\ (X, X') &\mapsto K(X, X') \end{aligned} \tag{4.17}$$

*Vector Machines*

[6]kernel trick en anglais

[7]support vector classifier

Le noyau $K$ est donc une fonction à deux paramètres qui retourne une valeur réelle. Sauf indication contraire, nous supposons que cette fonction est *symétrique* :

$$K(X, X') = K(X', X) \qquad X, X' \in \mathbb{X}. \tag{4.18}$$

Pour bien saisir les implications de cette formulation, on peut étudier un cas particulier très simple d'une mesure de similarité, le produit scalaire entre deux vecteurs $\boldsymbol{x}$ et $\boldsymbol{x'} \in \mathbb{R}^d$ défini par :

$$\langle \boldsymbol{x}, \boldsymbol{x'} \rangle := \sum_{i=1}^{n} [\boldsymbol{x}]_i [\boldsymbol{x'}]_i \tag{4.19}$$

où $[\boldsymbol{x}]_i$ dénote la $i$-ème composante du vecteur $\boldsymbol{x}$.

Notons que l'interprétation géométrique du calcul du produit scalaire donne le cosinus de l'angle entre les vecteurs $\boldsymbol{x}$ et $\boldsymbol{x'}$ si ceux-ci sont normalisés pour avoir une longueur de 1. Incidemment, la longueur d'un vecteur $\boldsymbol{x}$ notée $\|x\|$ est calculée à partir du produit scalaire par :

$$\|x\| = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle} \tag{4.20}$$

La distance entre deux vecteurs (ces vecteurs peuvent être des observations dans un espace euclidien à $n$ dimensions) est calculée comme la longueur de la différence entre ces deux vecteurs. Nous avons donc :

$$\delta = \sqrt{\langle |\boldsymbol{x} - \boldsymbol{x'}|, |\boldsymbol{x} - \boldsymbol{x'}| \rangle} \tag{4.21}$$

ou $|\boldsymbol{x} - \boldsymbol{x'}|$ est la valeur absolue de la différence.

### 4.3.1 Généralisation à un nouveau point

Cette notion de distance euclidienne (dans un espace de caractéristiques, que nous allons revoir formellement dans un instant) qui représente la similitude entre deux observations peut sembler (à juste raison) connexe au concept de voisinage ; dans l'article de *Neural Computation* (Bengio, Delalleau,

PAIEMENT, VINCENT, OUIMET et ROUX 2004) [8], les auteurs développent le cadre théorique qui permet de dégager une généralisation qui englobe Isomap 4.1, LLE 4.2 et la segmentation spectrale [9] comme des méthodes qui apprennent un noyau à partir des données de l'ensemble d'entraînement : la matrice de *Gram* $K_{ij} = k(X_i, X_j)$ définie par ces algorithmes. Formaliser mathématiquement les opérations algorithmiques qui mènent à cette matrice nous permet de définir un noyau.

L'étape cruciale qui nous permet d'accéder à des variétés non-linéaires est une projection non linéaire notée :

$$\begin{aligned} \Phi : \mathcal{X} &\rightarrow H \\ X &\mapsto \boldsymbol{X} := \Phi(X). \end{aligned} \tag{4.22}$$

On peut voir un exemple volontairement simplifié de cette projection dans la Figure 4.9. Mathématiquement cette dernière équation nous permet d'utiliser une mesure de similarité $k$ qui correspond à un produit scalaire dans l'espace $H$ [10] :

$$K(X, X') = \langle \Phi(X), \Phi(X') \rangle. \tag{4.23}$$

La beauté de la méthode réside dans cette dernière équation : il n'est pas nécessaire(de toute façon dans la plupart des cas ce serait pratiquement impossible) de calculer la projection $\Phi(X)$ de deux observations dans l'espace de Hilbert $H$ afin de calculer le produit scalaire entre celles-ci ; de par la définition du noyau et des caractéristiques de l'espace de Hilbert à noyaux reproduisants, calculer la valeur du noyau nous donne le produit scalaire dans l'espace $H$, la communauté réfère à cette dernière propriété des espaces générés par des noyaux par le terme *l'astuce du noyau.*

[8] autre présentation dans le rapport technique (BENGIO, VINCENT, PAIEMENT, DELALLEAU, OUIMET et ROUX 2003)

[9] Pour une vue d'ensemble, voir (WEISS 1999). Une variante intéressante peut être trouvée dans (NG, JORDAN et WEISS 2002).

[10] Cet espace se nomme un espace de Hilbert à noyaux reproduisants, ou RKHS, l'acronyme de l'anglais

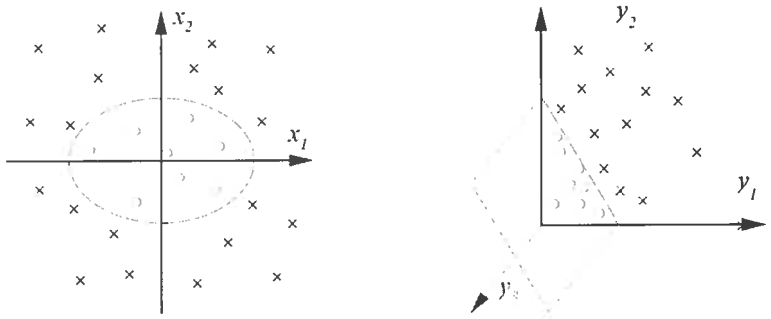



**Figure 4.9** – *Un exemple simple de projection, une tâche de classification binaire dans un espace à 2 dimensions : il n'existe clairement pas de séparation linéaire possible entre les deux classes, mais en utilisant une projection non linéaire avec* $\Phi(X) \mapsto (y_1, y_2, y_3) = [X]_1^2, [X]_2^2, \sqrt{2[x]_1[x]_2}$ *l'ellipse devient un hyperplan ( dans ce cas-ci parallèle à l'axe* $y_3$*, alors tous les points se situent dans le plan* $(z_1, z_2)$ *). On peut comprendre très rapidement l'effet de cette projection parce que les équations d'ellipses peuvent être écrites par des combinaisons linéaires de* $(z_1, z_2, z_3)$*. Avec les noyaux, il est possible de calculer le produit scalaire entre les points (et trouver cet hyperplan de séparation si c'est la tâche à accomplir) sans calculer les projections* $\Phi$*. (Tiré de (*SCHÖLKOPF *et* SMOLA *2002) )*

### 4.3.2 Théorème de Mercer

Nous introduisons ici le théorème de Mercer : Nous assumons que $(\mathcal{X}, \mu)$ est un espace fini de mesure, c'est-à-dire que l'ensemble $\mathcal{X}$ possède un $\sigma$-algèbre et une mesure qui satisfait $\mu(\mathcal{X}) < \infty$. Cette dernière condition implique qu'avec un facteur d'étalement, $\mu$ est une mesure de probabilité. Le terme *"pour presque toutes"* exclut les ensembles de mesure 0.

**Théorème de Mercer 4.3.1** *Etant donné $K \in L_\infty(\mathcal{X})$, une fonction symétrique évaluant à un réel telle que l'opérateur d'intégration*

$$\begin{aligned} T_K &: L_2(\mathcal{X}) \to L_2(\mathcal{X}) \\ (T_K f)(x) &:= \int_{\mathcal{X}} k(x, x') f(x') d\mu(x') \end{aligned} \tag{4.24}$$

*est* défini positif *; ce qui veut dire que pour tout $f \in L_2(\mathcal{X})$, nous avons*

$$\int_{\mathcal{X}^2} K(x, x') f(x) f(x') d\mu(x) d\mu(x') \geq 0. \tag{4.25}$$

*Soit $\psi_j \in L_2(\mathcal{X})$ les fonctions propres normalisées de $T_k$, et $\lambda_j > 0$ ses valeurs propres ordonnées en ordre non croissant. Alors*

$$\begin{aligned} (\lambda_j)_j &\in l_1, \\ K(x, x') &= \sum_{j=1}^{N_H} \lambda_j \psi_j(x) \psi_j(x') \end{aligned} \tag{4.26}$$

*est vrai pour presque tous les couples (x,x'). Soit $N_H \in \mathbb{N}$, où $N_H = \infty$ ; dans ce dernier cas, les séries convergent absolument et uniformément pour presque tous les $(x, x')$.*

De 4.26 nous avons que $K(x, x')$ correspond à un produit scalaire dans $l_2^{N_H}$, puisque $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$ avec :

$$\Phi : \mathcal{X} \to l_2^{N_H} \tag{4.27}$$

$$x \mapsto (\sqrt{\lambda_j} \phi_j(x'))_j = 1, \ldots, N_H \tag{4.28}$$

pour presque tous $x \in \mathcal{X}$.

Il est possible en remplaçant par un noyau le produit scalaire dans l'analyse en composantes principales (voir 3.1 ) de générer une version non linéaire de l'ACP ; l'ACP à noyau (voir (SCHÖLKOPF, MIKA, SMOLA, RÄTSCH et MÜLLER 1998) ).

### 4.3.3 Quelques Noyaux Communs

Voici quelques noyaux couramment utilisés : nous assumons que $\mathcal{X} \subset \mathbb{R}^N$. Le noyau polynomial non homogène qui génère toutes les puissances de $x$ et $x'$ jusqu'à $d \in \mathbb{N}$ :

$$k(x, x') = (\langle x, x' \rangle + C)^d. \tag{4.29}$$

Si $C = 0$ le noyau est polynomial homogène.

Le noyau gaussien à fonction à base radiale [11]

$$K(x, x') = exp(-\frac{\|x - x'\|^2}{2\sigma^2}) \tag{4.30}$$

où $\sigma > 0$. Le noyau sigmoïdal :

$$K(x, x') = \tanh(\kappa \langle x, x' \rangle + \vartheta) \tag{4.31}$$

où $\kappa > 0$ et $\vartheta < 0$. Le noyau gaussien est un cas particulier de la classe des noyaux à fonction de base radiale :

$$K(x, x') = f(d(x, x')), \tag{4.32}$$

où $d$ est une métrique pour $\mathcal{X}$, et $f$ est une fonction sur $\mathbb{R}_0^+$. Un autre exemple de noyau défini par une fonction à base radiale est le noyau $B_n$-spline où $I_X$ dénote la fonction indicatrice(ou caractéristique) sur l'ensemble $X$ et $\otimes$

[11] RBF pour Radial Basis Function en anglais.

l'opération de convolution $(f \otimes g)(x) := \int f(x')g(x' - x)dx'$ :

$$K(x, x') = B_{2p+1}(\|x - x'\|) \quad ou \tag{4.33}$$

$$B_n := \bigotimes_{i=1}^{n} I_{[-\frac{1}{2},\frac{1}{2}]}. \tag{4.34}$$

Ce dernier noyau calcule les $B$-splines d'ordre $2p + 1$ ($p \in \mathbb{N}$), définie comme la $n$-ième convolution sur l'intervalle unitaire $[-\frac{1}{2}, \frac{1}{2}]$.

### 4.3.4 Généralisation impliquant un Noyau

L'article (Bengio, Vincent, Paiement, Delalleau, Ouimet et Roux 2003) isole un noyau pour Isomap(voir 4.1), LLE(voir 4.2) et quelques autres algorithmes de réduction de dimensionnalité : Par exemple, celui d'Isomap y est défini de façon suivante :

$$K(x, x') = \begin{cases} d_{\mathcal{X}}(x, x') & \text{si } x \in D \text{ et } x' \in D \\ \min_{z \in \mathcal{N}(x)}(d_G(x, z) + d_{\mathcal{X}}(z, x')) & \text{si } x \notin D \text{ et } x' \in D \\ \min_{z \in \mathcal{N}(x')}(d_{\mathcal{X}}(x, z) + d_G(z, x')) & \text{si } x \in D \text{ et } x' \notin D \end{cases}$$

où $d_G(x, x')$ est la distance euclidienne entre les observations voisines dans l'espace original comme défini dans l'algorithme, et $d_{\mathcal{X}(x,x')}$ , la longueur géodésique calculée par Isomap. $\mathcal{N}(x)$ est l'ensemble des $k$ plus proches voisins de $x$ dans $D$.

CHAPITRE 5

# Vecteurs et valeurs propres

Comme nous l'avons vu dans les sections et chapitres précédents, la décomposition en valeurs et vecteurs propres d'une matrice carrée semi-définie positive est une opération algébrique très utile en apprentissage statistique. Presque toutes les méthodes de réduction de dimensionnalité linéaire et non linéaire y font appel. Nous allons brièvement survoler quelques optimisations qui permettent d'effectuer cette opération plus rapidement : plusieurs "eigen-packages" de "eigen"-systèmes de décomposition existent et l'optimisation des méthodes de décomposition en fonction de chaque propriété connue ou vérifiable d'une matrice est un domaine de recherche en soi.

Pour plus de détails nous suggérons au lecteur de voir (GOLUB et VAN-LOAN 1996). Pour certaines implémentations[1] voir plutôt (PRESS, FLANNERY, TEUKOLSKY et VETTERLING 1992).

[1]Selon l'Office de la langue française, ceci n'est pas un anglicisme quand il est utilisé dans ce sens.

## 5.1 Diagonalisation

**Valeur et vecteur propre 5.1.1** *Soit $A$, une matrice carrée. On dit que le nombre réel $\lambda$ est une* valeur propre *de $A$ (en anglais* eigenvalue, *de l'allemand eigen : propre) et que le vecteur colonne $V$ est un* vecteur propre *de $A$* associé à $\lambda$ *si :*

$$V \neq 0 \qquad et \qquad AV = \lambda V \tag{5.1}$$

La diagonalisation de matrice consiste à trouver une matrice diagonale $\Lambda$ qui contient les valeur propres et une matrice inversible (et orthogonale si $A$ est symétrique) $V$ de sorte que :

$$\begin{aligned} V^{-1}AV &= \Lambda \\ AP &= V\Lambda \end{aligned} \tag{5.2}$$

Les valeurs propres d'une matrice triangulaire (supérieure, inférieure, ou les deux) sont les éléments de sa diagonale principale.

Lorsqu'on tente de diagonaliser une matrice réelle quelconque, deux difficultés peuvent apparaître. Tout d'abord les valeurs propres ne sont pas nécessairement toutes réelles, et de plus, la dimension de l'espace propre associé à une valeur propre $\lambda$ peut être strictement inférieure à sa multiplicité algébrique.

Dans le cas qui nous intéresse, la matrice $A$ est symétrique puisque $k(x, x') = k(x', x) = K_{ij} = Kji$ : ces difficultés ne se présentent pas.

La multiplication matricielle qui nous donne les valeurs et vecteurs propres

$$V^{-1}AV = diag(\lambda_1, \ldots, \lambda_n)$$

est un cas spécial de la transformée de similarité de la matrice A :

$$A \rightarrow Z^{-1}AZ \tag{5.3}$$

Pour une matrice $Z$ quelconque. Cette transformée est cruciale au calcul des

valeurs propres d'une matrice parce qu'elle laisse ces valeurs inchangées nous permettant donc de modifier graduellement la matrice $A$ pour lui donner une forme plus diagonale sans pour autant affecter ses valeurs propres. Cette propriété de la transformation peut être démontrée par :

$$\begin{aligned} \det(Z^{-1}AZ - \lambda 1) &= \det(Z^{-1}(A - \lambda 1)Z) && (5.4)\\ &= \det(Z)\det(A - \lambda 1)\det(Z^{-1}) && (5.5)\\ &= \det(A - \lambda 1) && (5.6) \end{aligned}$$

Puisque nous nous intéressons seulement aux matrices réelles symétriques, les vecteurs propres $V$ de droite et de gauche de la transformation qui donnent les valeurs propres sont les mêmes à une inversion près. De plus, ils sont réels et orthonormaux, ce qui implique que la matrice de transformation est orthogonale ; la transformée de similarité est donc aussi une *transformée orthogonale* :

$$A \rightarrow Z^T AZ. \tag{5.7}$$

La stratégie globale de presque tous les *eigensystèmes* modernes repose sur la transformation itérative de la matrice $A$ en une matrice diagonale (qui contient les valeurs propres) par une série de transormées de similarité.

$$\begin{aligned} A &\rightarrow P_1^{-1}AP_1 \rightarrow P_2^{-1}P_1^{-1}AP_1P_2 \\ &\rightarrow P_3^{-1}P_2^{-1}P_1^{-1}AP_1P_2P_3 \rightarrow \cdots \end{aligned} \tag{5.8}$$

L'accumulation des matrices de transformation nous donne une matrice dont les colonnes sont les vecteurs propres de la matrice $A$ :

$$V = P_1P_2P_3\ldots. \tag{5.9}$$

Il existe deux familles de méthodes pour effectuer cette stratégie de diagonalisation itérative ; elles ont chacune leurs forces et leurs faiblesses. Une stratégie efficace employée par la pluspart des systèmes modernes est d'utiliser successivement un membre de chacune des deux familles.

## 5.2 Méthodes Itératives

La première famille est composée de techniques qui construisent successivement des matrices $P_1$ explicites effectuant des transformations unitaires dont l'objectif est d'effectuer une tâche précise. Par exemple, pour la transformée de Jacobi, celle ci est d'annuler la valeur d'un indice en particulier. Pour celle de Householder, c'est plutôt une colonne ou une ligne en particulier. Ces méthodes tendent à converger rapidement quand la matrice est très peu diagonale, mais procèdent lentement par la suite.

On pourrait continuer à ajouter des transformations jusqu'à ce que les valeurs qui ne sont pas sur la diagonales soient plus petites qu'une valeur donnée (l'erreur d'approximation de la machine par exemple, voir (PRESS, FLANNERY, TEUKOLSKY et VETTERLING 1992) section 1.3 ), mais bien que simple conceptuellement, cette méthode est très sous-optimale ; l'alternative d'utiliser une méthode du second ensemble (QR par exemple, (voir5.3) ) est plus rapide par un facteur constant de 5 pour $n > 10$.

### 5.2.1 Transformée de Jacobi d'une matrice symétrique

La transformée de Jacobi consiste en une séquence de transformées de similarité orthogonales nommée rotations de Jacobi. Celles-ci consistent en une rotation de plan choisie afin d'annuler la valeur d'un indice qui n'est pas sur la diagonale principale. Une succession de ces transformations aura l'effet de "défaire" les indices qui ont été réduits à zéro dans des itérations précédentes. Néanmoins les éléments hors de la diagonale deviendront de plus en plus petits, jusqu'à ce que la matrice devienne à toutes fins pratiques diagonale. En accumulant par multiplication les matrices de rotation on récupère les vecteurs propres.

Quoique sérieusement plus lente, cette méthode possède une garantie de convergence, et elle est conceptuellement très simple. La matrice de base de rotation de Jacobi $P_{pq}$ est une matrice de la forme suivante.

$$P_{pq} = \begin{pmatrix} 1 & 0 & \cdots & & & & & \\ 0 & 1 & \cdots & & & & & \\ \vdots & \vdots & \ddots & & & & & \\ & & & c & \cdots & s & & \\ & & & \vdots & 1 & \vdots & & \\ & & & -s & \cdots & c & & \\ & & & & & & \ddots & \vdots & \vdots \\ & & & & & & \cdots & 1 & 0 \\ & & & & & & \cdots & 0 & 1 \end{pmatrix}$$

La matrice $P_{pq}$ est la matrice identité sauf pour les éléments $c$ de la diagonale dans les colonnes (et lignes) $p$ et $q$ et les valeurs $s$ et $-s$ aux index $p_{pq}$ et $p_{qp}$. Les valeurs $c$ et $s$ sont les sinus et cosinus d'un angle de rotation $\phi$, ce qui implique que $c^2 + s^2 = 1$.

Une rotation planaire comme décrite dans la matrice $P$ est utilisée pour transformer $A$ en $A'$.

$$A' = P_{pq}^T A P_{pq}$$

Le produit matriciel $P_{pq}^T A$ modifie seulement les lignes $p$ et $q$ de $A$, alors que la multiplication $AP_{pq}$ ne modifie que les colonnes $p$ et $q$. La matrice $A'$ ainsi transformée est la matrice $A$ sauf pour les éléments notés ci-dessous.

$$A' = \begin{pmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{pmatrix}$$

En utilisant la symétrie de $A$ nous avons explicitement :

$$a'_{rp} = ca_{rp} - sa_{rq} \qquad r \neq p, r \neq q \tag{5.10}$$
$$a'_{rq} = ca_{rq} + sa_{rp} \qquad r \neq p, r \neq q \tag{5.11}$$
$$a'_{pp} = c^2 a_{pp} + c^2 a_{qq} + 2sca_{pq} \tag{5.12}$$
$$a'_{qq} = s^2 a_{pp} + c^2 a_{qq} + 2sca_{pq} \tag{5.13}$$
$$a'_{pq} = (c^2 - s^2)a_{pq} + sc(a_{pp} - a_{qq}) \tag{5.14}$$

L'idée de la méthode de Jacobi est donc d'essayer de réduire à zéro les éléments qui ne sont pas sur la diagonale par une suite de rotations ; pour que $a'_{pq} = 0$ l'angle de rotation $\phi$ est :

$$\theta \equiv \cot 2\phi \equiv \frac{c^2 - s^2}{2sc} = \frac{a_{qq} - a_{pp}}{2a_{pq}}. \tag{5.15}$$

Si nous remplaçons $t \equiv \frac{s}{c}$, $\theta$ devient :

$$t^2 + 2t\theta - 1 = 0 \tag{5.16}$$

La plus petite racine de l'équation (5.16) correspond à un angle de rotation plus petit que $\frac{\pi}{4}$, nous avons une réduction stable en choisissant celui-ci à chaque itération. On peut réécrire :

$$t = \frac{sgn(\theta)}{|\theta| + \sqrt{\theta^2 + 1}}$$

Nous avons donc que :

$$c = \frac{1}{\sqrt{t^2 + 1}} \tag{5.17}$$
$$s = tc \tag{5.18}$$

Évidemment, numériquement afin de minimiser l'erreur due à l'imprécision de la machine, certaines des valeurs décrites ci-dessus sont remplacées par des approximations ou par 0 le cas échéant.

On peut constater la convergence de la méthode avec :

$$S = \sum_{r \neq s} |a_{rs}|^2 \tag{5.19}$$

parce que de (5.10) à (5.14) nous avons que :

$$S' = S - 2|a_{pq}|^2. \tag{5.20}$$

La somme des éléments hors diagonale converge donc vers 0.

Les vecteurs propres sont donnés par la matrice identité multipliée par les matrices $P_i$ :

$$V = IP_1P_2P_3 \ldots. \tag{5.21}$$

Ce calcul est aussi sujet à l'erreur de la machine, et les opérations doivent être calculées en conséquence.

L'ordre de sélection des $a_{pq}$ qui sont réduits à zéro doit être fait rapidement ; choisir un élément est un luxe trop coûteux, on procède donc en passant par chaque position $P_{12}, P_{13}, \ldots, P_{1n}$, suivi de $P_{23}, P_{24}, \ldots$. La convergence est généralement quadratique, plusieurs "balayages" de la matrice seront nécessaires.

Empiriquement, il a été vérifié que l'on peut généralement accélérer le processus en modifiant les premiers balayages en effectuant seulement une rotation pour les valeurs plus grandes qu'un $\epsilon$ donné [2] ; par la suite, on peut également considérer après quelques balayages que si $|a_{pq}| \ll |a_{pp}|$ et $|a_{pq}| \ll |a_{qq}|$ on assigne $a_{pq}$ à 0 et on saute la rotation.

L'ordre du temps de calcul pour diagonaliser une matrice $A$ symétrique typique est entre $18n^3$ et $30n^3$ en incluant l'extraction des vecteurs propres.

[2] calculé à partir de propriétés numériques de la matrice

### 5.2.2 Réduction de Householder et de Givens d'une matrice symétrique

La méthode combinée, la technique de pointe de la plus part des *eigen-systèmes* modernes, fait premièrement appel à la réduction de Household-der. Contrairement à la méthode de Jacobi qui tente d'itérer jusqu'à convergence, celle-ci tente plutôt de réduire la matrice symétrique à une forme plus simple ; la forme tridiagonale. La *réduction de Givens* est une modification de la réduction cyclique de Jacobi où, au lieu de choisir une rotation $P_{pq}$ qui annule une valeur $a_{pq}$, on la choisit pour qu'elle annule plutôt $a_{q,(p-1)}$. La séquence des matrice des transformations est donc :

$$P_{23}, P_{24}, \ldots, P_{2n}, P_{34}, \ldots, P_{3n}, \ldots, P_{n-1,n}.$$

Cette méthode assigne donc linéairement les colonnes à 0 en concentrant les valeurs autour de la diagonale ; les $a'_{rp}$ et $a'_{pq}$ avec $r \neq q$ et $r \neq p$ sont des combinaisons linéaires de leur valeurs à l'itération précédente ; si $a_{rp}$ et $a_{pq}$ ont déjà été réduits à 0, ils resteront à zéro aux itérations subséquentes. La tridiagonalisation avec cette méthode coûte $\frac{4n^3}{3}$ sans tenir compte des calculs requis pour conserver les vecteurs propres (généralement une prime de 50% ).

La réduction de Householder est 2 fois plus efficace que celle de Givens ; elle est généralement utilisée comme première étape de l'approche combinée. Elle réduit une matrice symétrique $A$ à une matrice tridiagonale en $n-2$ transformations orthogonales. Chacune de ces transformations annule la partie ayant des valeurs non nulles d'une colonne et d'une ligne. La matrice de Householder est composée à partir d'un vecteur $w$ ayant une norme $|w|^2 = 1$ de façon suivante :

$$P = 1 - 2ww^T \tag{5.22}$$

Les vecteurs (presque) propres sont accumulés avec :

$$Q = P_1 P_2 \cdots P_{n-2}. \tag{5.23}$$

## 5.3 Méthodes de Factorisation

Cette section présente une version d'un algorithme de factorisation : la méthode $QR$. Les eigensystèmes modernes en utilisent différentes variantes comme seconde étape à la méthode combinée. Supposons que la matrice $A$ peut être factorisée en un facteur de gauche $F_L$ et un facteur de droite $F_R$. On pourrait donc écrire :

$$A = F_L F_R \qquad \text{ou de façon équivalente} \quad F_L^{-1} A = F_R \tag{5.24}$$

En multipliant les facteurs ensemble en ordre inverse, en utilisant la seconde équation de (5.24) nous avons :

$$F_R F_L = F_L^{-1} A F_L \tag{5.25}$$

ce qui est facilement identifiable comme un transformation de similarité $F_L$ sur $A$.

L'idée de base derrière celle-ci est donc que toute matrice réelle peut être décomposée de façon suivante :

$$A = QR \tag{5.26}$$

où $Q$ est une matrice orthogonale et R est triangulaire supérieure. Considérons la matrice produite par la multiplication des facteur dans l'ordre inverse : [3]

$$A' = RQ. \tag{5.27}$$

Puisque $Q$ est orthogonale, nous avons de 5.26 que $R = Q^T A$. En remplaçant dans 5.27 nous avons :

$$A' = Q^T A Q \tag{5.28}$$

$A'$ est donc une transformation orthogonale de $A$. Une matrice $L$ qui serait

[3] la multiplication matricielle étant évidemment non commutative

triangulaire inférieure respecte tout aussi bien ce raisonnement, on l'utilise généralement plutôt que la triangulaire supérieure dans les *eigensystèmes* modernes parce que l'ordre d'opérations de la première étape de la méthode combinée produit une erreur d'approximation plus petite pour celle-ci. L'algorithme $QR$ est composé d'une série de transformations comme suit :

$$A_s = Q_s R_s \tag{5.29}$$
$$A_{s+1} = R_s Q_s \tag{5.30}$$
$$( = Q_s^T A_s Q_s) \tag{5.31}$$

Cet algorithme appliqué à une matrice quelconque repose sur un théorème un peu contre-intuitif :

**Convergence des valeurs propres 5.3.1** *Si la matrice $A$ possède des valeurs propres de valeurs absolues $|\lambda_i|$, alors premièrement :*

$$A_s \rightarrow [\textit{triangulaire supérieure}] \quad \textit{quand} \tag{5.32}$$
$$s \rightarrow \infty. \tag{5.33}$$

*De plus, si $A$ possède une valeur propre $|\lambda_i|$ de multiplicité $p$,*

$$A_s \rightarrow [\textit{triangulaire supérieure}] \quad \textit{quand} \tag{5.34}$$
$$s \rightarrow \infty. \tag{5.35}$$

*sauf pour une matrice diagonale en bloc d'ordre $p$ pour lequel ces valeurs singulières tendent vers $\lambda_i$.*

La preuve de ce théorème est longue et fastidieuse. Voir par exemple((STOER et BULIRSCH 1980) ).

Cet algorithme nécessite $O(n^3)$ opérations par itération pour une matrice quelconque, mais seulement $O(n)$ pour une matrice tridiagonale [4], ce qui le

[4] $O(n^2)$ pour une matrice Hessenberg, une matrice triangulaire avec une diagonale additionnelle en retrait de 1 de la diagonale principale.

rend très attrayant comme complément à la méthode de Householder, surtout si on utilise les techniques de *shifting* (voir (PRESS, FLANNERY, TEUKOLSKY et VETTERLING 1992) pour plus détails) pour obtenir une convergence accélérée vers les valeurs propres.

En excluant le calcul des vecteurs propres nous avons un coût de $30n^2$, et un coût beaucoup plus élevé de $3n^3$ si les vecteurs propres sont nécessaires.

### 5.3.1 Méthodes itératives de Lanczos

Certaines des matrices produites par les algorithmes que nous avons vu (comme celle de LLE) sont creuses. Le procédé itératif de Lanczos ( voir (BROWN, CHU, ELLISON et PLEMMONS 1994) ou les deux volumes de (J.CULLUM et WILLOUGHBY 1985) pour plus de détails d'implémentation) est un cas particulier des méthodes de la première famille qui permet d'extraire certaines valeurs propres et, optionnellement, leurs vecteurs propres correspondants beaucoup plus rapidement que les méthodes vu précédemment. Les méthodes de Lanczos ont la particularité de ne pas calculer de matrice intermédiaires complètes ; ils utilisent un vecteur choisi de façon particulière et construisent un sous-espace de Krylov une colonne à la fois. Des valeurs de Ritz sont calculées pour approximer les valeurs propres. Ce processus itératif peut rendre la matrice non-orthogonale, et la ré-orthogonalisation est coûteuse en terme de calcul ; c'est sur cette fine ligne que les développeurs de *eigensystèmes* modernes doivent marcher : ce niveau de complexité dépasse cependant largement le propos de ce document et nous renvoyons le lecteur au références mentionnées.

CHAPITRE 6

# Modèles

Dans ce chapitre nous présentons quelques modèles de réduction de dimensionnalité non linéaire utilisés à des fins d'apprentissage semi-supervisé. Ils sont inspirés des méthodes que nous avons vues au chapitre 4, mais ils insistent sur la stabilité topologique afin de préserver (et tenter d'améliorer) la séparabilité des points dans la variété apprise. Il serait intuitif d'évaluer la stabilité topologique de l'apprentissage d'une variété en utilisant la méthode des moindres carrés sur l'erreur de reconstruction, mais l'objet ici est plutôt de quantifier la séparabilité des classes résultantes, et pour plusieurs problèmes, ces deux mesures ne sont pas nécessairement corrélées. La "classifiabilité" sur une variété est une mesure relativement dure à établir ; nous avons choisi d'utiliser un SVC à noyau RBF. Pour une introduction voir la section 4.3 ou l'article (BOSER, GUYON et VAPNIK 1992) ou encore le guide(HSU, CHANG et LIN 2002). Cet algorithme non linéaire à noyau a relativement peu de paramètres : nous accomplissons une recherche de paramètres limitée choisis par validation croisée. On pourrait aussi utiliser la méthode extrêmement simple des $n$ plus proches voisins pour évaluer la séparabilité des modèles produits.

## 6.1 RISIMAP

La réduction de dimensionnalité non linéaire isomorphique par apprentissage de variété avec élimination de sous graphes disconnecté, ou RISIMAP pour *Reconnected Independant Subgraph Isomorphic MAPping* est une des deux variantes d'Isomap(voir 4.1) que l'auteur de ce mémoire propose. C'est un algorithme d'apprentissage non-supervisé puisqu'il n'utilise pas les étiquettes des points. La différence fondamentale réside dans l'établissement de la matrice d'adjacence.

Voici le raisonnement qui précède et justifie cette modification : pour certains ensembles de données à très haute dimensionnalité (comme par exemple ARCENE) on a beaucoup moins de points que de dimensions, une taille de voisinage (définit comme les $k$ plus proches voisins dans cet exemple) restreinte risque de produire à l'étape 2 de l'algorithme (voir 4.1) une matrice possédant une déficience particulière : il pourrait y avoir présence de sous graphes non connectés. Ceci implique que, puisque la matrice a été initialisée à $\infty$ au début de la procédure, certains couples $A_{ij}$ ont toujours la valeur $\infty$. Cette déficience va rendre erronés les résultats de la diagonalisation à l'étape 4.

Si on ne modifie pas la procédure, il faut palier à cette déficience; en agrandissant le voisinage jusqu'à ce que la matrice $A_{ij}$, résultante de l'étape 2, ne possède plus de sous graphes disconnectés nous lisserions potentiellement de l'information structurante de la variété. Par conséquent nous rendrions la frontière de décision plus *floue*, limitant ainsi la séparabilité dans l'espace de dimensionnalité réduite récupéré par l'algorithme. Rappelons que le cas dégénéré, comme mentionné précédemment dans ( 4.1), est équivalent à un positionnement multidimensionnel, donc à une réduction linéaire, quand $k = n$, où $k$ est le voisinage et $n$ est le nombre de points dans l'ensemble traité.

### 6.1.1 Détection de Sous Graphes Disconnectés

La solution que nous proposons à ce problème dans RISIMAP, semble donner de bons résultats (voir le prochain chapitre) et ce même quand le voisinage restreint choisi donne une matrice de distances géodésiques approximatives

$A_{ij}$ optimales au point de vue de la stabilité topologique, mais comportant des sous-graphes disconnectés. Elle consiste simplement à insérer une étape entre les étapes 2 et 3 de l'algorithme original où on y détecte et élimine chaque sous graphe en ajoutant itérativement à la matrice $A$ la plus petite arête $\delta_{ij}$ entre deux sous-graphes ne fait pas partie du voisinage original. On répète itérativement cette procédure jusqu'à ce que la matrice $A$ ne contienne plus de sous-graphes disconnectés. Le tableau 6.1 donne une description de l'algorithme RISIMAP.

**Tableau 6.1** – *Algorithme RISIMAP*

1. Construire le graphe de voisinage : On définit un graphe $G$ sur l'ensemble des observations, et pour tous les couples $(i, j)$ possibles on y connecte $i$ et $j$ par la distance euclidienne $\delta_{ij}$, soit s'ils sont plus proches que le paramètre de voisinage $\epsilon$ ($\epsilon$-Isomap), soit si $j$ est un des $k$ plus proche voisins de i($k$ étant l'autre paramètre de voisinage possible pour la version $k$-Isomap).
2. Calculer les plus courts chemins : On initialise la matrice d'adjacence $A$ à $\delta_{ij}$ si les points $i$ et $j$ possèdent une arête dans le graphe $G$ et à $\infty$ dans le cas contraire. Pour chaque $k = 1, 2, \ldots, n$, on remplace successivement $A_{ij}$ par $\min\{A_{ij}, A_{i,k} + A_{k,j}\}$ ; la matrice $A$ contient donc les plus courts chemins entre chaque paire de points dans le graphe G étant donné les distances euclidiennes du voisinage données en 1.
3. Pour chaque paire de sous-graphe non connecté on calcule la distance la plus petite à ajouter au voisinage pour que ces sous graphes soit connectés ; on ajoute ces distances à la matrice $A$
4. Comme en 3 dans MDS, on applique l'opérateur $\tau$ à la matrice $A$ et on obtient la matrice $B$.
5. Trouver les valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ de $B$ et leur vecteurs propres respectifs $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}$.
6. Choisir les $d$ valeurs et vecteurs propres les plus significatifs en ordre décroissant : les coordonnées des points $y_i$ de l'incorporation dans l'espace à $d$ dimensions sont données par : $y_i = \sqrt{\lambda_p}\boldsymbol{v_n^i}$ où $\boldsymbol{v_n^i}$ est la $i$-ième composante du $p$-ième vecteur propre.

Définissons un noyau qui décrit la matrice de Gram produite en 3 :

$$k(x,x') = \begin{cases} d_G(x,x') & \text{si } x \in D \text{ et } x' \in D \\ \min_{z \in \mathcal{N}(x)}(d_{\mathcal{X}}(x,z) + d_G(z,x')) & \text{si } x \notin D \text{ et } x' \in D \\ \min_{z \in \mathcal{N}(x')}(d_G(x,z) + d_{\mathcal{X}}(z,x')) & \text{si } x \in D \text{ et } x' \notin D \\ \min_{y \in \mathcal{C}(x), z \in \mathcal{C}(x')}(d_G(y,z)) & \text{si } d_{\mathcal{X}}(x,x') = \infty \end{cases}$$

où $d_G(x,x')$ est la distance euclidienne entre les observations voisines dans l'espace original comme défini dans l'algorithme, et $d_{\mathcal{X}(x,x')}$ est la longueur géodésique calculée par Isomap. $\mathcal{N}(x)$ est l'ensemble des $k$ plus proches voisins de $x$ dans $D$, et $\mathcal{C}(x)$ est l'ensemble des points faisant partie du même sous-graphe que $x$.

### 6.1.2 Garantie Asymptotique de Convergence

Cet ajout à l'algorithme ne modifie pas la garantie de convergence asymptotique que nous avons énoncée dans la section 4.1.2 ; les propriétés de Isomap qui soutiennent cette garantie sont aussi vérifiables pour RISIMAP, puisque l'ajout d'arêtes après la procédure de récupération des chemins de tous les points à tous les points n'impacte pas sur la définition de voisinage telle qu'elle est détaillée dans la section 4.1.1. Cette modification ne peut par conséquent causer de courts-circuits entre différentes parties de la vraie géométrie de la variété, telle qu'elle est incluse dans l'espace latent, puisque avant même de pouvoir causer un court-circuit (qui annulerait la garantie de convergence) il faut que le graphe possède un circuit. C'est cette propriété que RISIMAP vérifie. RISIMAP ne peut donc pas causer de court-circuit puisque son objectif est de causer un circuit quand le réseau possède des composantes non connexes.

## 6.2 Isostretch

Isostrech est une méthode de réduction de dimensionnalité supervisée puisqu'elle utilise les étiquettes des points. Elle possède une extension semi-supervisée naturelle que nous allons aussi évaluer au prochain chapitre. Comme pour RISIMAP, c'est une méthode qui s'inspire d'Isomap, de plus elle tient aussi du discriminant linéaire de Fisher. La version actuelle de l'algorithme (qui se trouve sur le site de l'auteur de ce mémoire au
http ://www.iro.umontreal.ca/~payettf ) dérive maintenant de RISIMAP, parce que l'élimination de sous-graphes semble donner de meilleurs résultats pour Isostretch aussi. Le questionnement à l'origine du développement de cette procédure est le suivant : est-ce possible d'intégrer les étiquettes des points dans la procédure de réduction de dimensionnalité non linéaire afin de produire une résolution en basse dimension qui améliore la séparabilité des points ? Les résultats, nous allons le voir au chapitre suivant, sont mitigés. Cette méthode donne de bien meilleurs résultats que d'autres méthodes traditionnelles, mais pas d'aussi bons que RISIMAP.

### 6.2.1 Étirement de variété

L'objectif de la variante est de distordre la variété qui est découverte en mettant le plus d'emphase possible sur les courtes distances qui séparent des points possédant des étiquettes différentes. Pour ne pas affecter la stabilité topologique, la fonction de distance alternative recherchée ne devrait pas avoir un effet notable sur les points qui sont séparés par une grande distance euclidienne dans l'espace latent ou par une grande distance géodésique sur la variété.

Plusieurs approches sont possibles : on peut étirer les distances entre les points qui ne partagent pas la même étiquette avant le calcul des plus courts chemins, ou ajuster la distance résultante après le calcul des plus courts chemins. Une autre interrogation pouvant donner naissance à plusieurs stratégies potentielles est à savoir quelle est la distance utilisée comme différentiel : est-elle constante pour tous les points ? Ou est-elle fonction de

la distance séparant les points ? Ou de l'ensemble de données ? Ou des deux propositions précédentes ? À partir de justifications purement intuitives et de résultats expérimentaux, quelques fonctions candidates ont été retenues ; nous en présentons une ici qui semble donner de bons résultats sur les données utilisées.

Certains de ces ensembles de données n'étaient que partiellement étiquetés. Quel genre de différentiel devrions-nous donc appliquer à ces points afin que la méthode puisse accomplir une réduction semi-supervisée ? Après quelques essais, comme définir une nouvelle classe pour les points, il a semblé plus optimal de ne pas appliquer le facteur d'éloignement relié à l'étiquette à ces points. La justification en est la stabilité topologique : plus la matrice d'adjacence comporte de modifications, plus l'apprentissage de la variété est difficile, voire faussé.

Nous cherchions donc une fonction qui a un impact relativement important quand les distances entre points possédant des étiquettes différentes sont petites mais qui n'induit pas de modification significative quand la distance entre les points est grande. Après plusieurs tentatives qui ne se sont révélées désastreuses, nous avons abouti sur une fonction quadratique qui implique la distance minimum différente de zéro se trouvant dans la matrice d'adjacence. La définition de noyau de la version supervisée de l'algorithme qui a semblé le plus prometteur est la suivante :

$$k(x,x') = \begin{cases} d_{\mathcal{X}}(x,x') & \text{si } x \text{ et } x' \in D \text{ et } L(x) = L(x') \\ \frac{d_G(x,x')^2+\epsilon^2}{d_{\mathcal{X}}(x,x')} & \text{si } x \text{ et } x' \in D \text{ et } L(x) \neq L(x') \\ \min_{z\in\mathcal{N}(x)}(d_G(x,z)+d_{\mathcal{X}}(z,x')) & \text{si } x \notin D \text{ et } x' \in D \\ \min_{z\in\mathcal{N}(x')}(d_{\mathcal{X}}(x,z)+d_G(z,x')) & \text{si } x \in D \text{ et } x' \notin D \\ \min_{y\in\mathcal{C}(x),z\in\mathcal{C}(x')}(d_{\mathcal{X}}(y,z)) & \text{si } d_{\mathcal{X}}(x,x') = \infty \end{cases}$$

où

$$\epsilon = \min d_G(x,x') \qquad x,x' \in S \, et \, d_G(x,x') \neq 0. \tag{6.1}$$

$\epsilon$ est donc le plus petit $d_G(x,x')$ séparant deux points plus grand que 0. Comme

auparavant, $d_G(x, x')$ est la distance euclidienne entre les observations voisines dans l'espace original comme défini dans l'algorithme, et $d_{\mathcal{X}(x,x')}$ est la longueur géodésique calculée par Isomap. $\mathcal{N}(x)$ est l'ensemble des $k$ plus proches voisins de $x$ dans $D$, et $\mathcal{C}(x)$ est l'ensemble des points faisant partie du même sous-graphe que $x$.

Dans ce noyau, le différentiel de distance ajouté est relatif à la distance entre les points en questions ; $\epsilon$ est aussi déterminé par l'ensemble des points.

Dans la table 6.2 on trouve la définition d'Isostretch.

La variante-semi supervisée de cet algorithme est très similaire ; nous ne faisons qu'ignorer les distances des points qui n'ont pas d'étiquettes. Son noyau est le suivant :

$$k(x,x') = \begin{cases} d_{\mathcal{X}}(x,x') & \text{si } x \text{ et } x' \in D \text{ et } L(x) = L(x') \\ d_{\mathcal{X}}(x,x') & \text{si } x \text{ et } x' \in D \text{ et } L(x|x') = \text{UL} \\ \frac{d_G(x,x')^2+\epsilon^2}{d_{\mathcal{X}}(x,x')} & \text{si } x \text{ et } x' \in D \text{ et } L(x) \neq L(x') \\ \min_{z\in\mathcal{N}(x)}(d_G(x,z) + d_{\mathcal{X}}(z,x')) & \text{si } x \notin D \text{ et } x' \in D \\ \min_{z\in\mathcal{N}(x')}(d_{\mathcal{X}}(x,z) + d_G(z,x')) & \text{si } x \in D \text{ et } x' \notin D \\ \min_{y\in\mathcal{C}(x),z\in\mathcal{C}(x')}(d_{\mathcal{X}}(y,z)) & \text{si } d_{\mathcal{X}}(x,x') = \infty \end{cases}$$

où $UL$ est l'étiquette des points inconnus. La stratégie qui s'est expérimentalement révélée la plus probante est de ne pas modifier la distance entre deux points quand l'un (ou les deux) ne possède pas d'étiquette connue.

### 6.2.2 Garantie Asymptotique de Convergence

L'impact sur la garantie de convergence lorsque le nombre de points tend vers l'infini est probablement affecté par la modification qu'Isostrech apporte à Isomap. L'objectif de la méthode n'est pas de récupérer de façon optimale la variété, mais plutôt de récupérer une variété très similaire qui met de l'emphase sur les régions à l'intérieur desquelles la classification n'est pas facile.

**Tableau 6.2** – *Algorithme Isostretch*

1. Construire le graphe de voisinage : On définit un graphe $G$ sur l'ensemble des observations, et pour tous les couples $(i, j)$ possibles on y connecte $i$ et $j$ par la distance euclidienne $\delta_{ij}$, soit s'ils sont plus proches que le paramètre de voisinage $\epsilon$ ($\epsilon$-Isomap), soit si $j$ est un des $k$ plus proche voisins de i($k$ étant l'autre paramètre de voisinage possible pour la version $k$-Isomap).
2. Distorsionner ce graphe pour que les points ayant des étiquettes différentes soient séparés par une plus grande distance.
3. Calculer les plus courts chemins : On initialise la matrice d'adjacence $A$ à $\delta_{ij}$ si les points $i$ et $j$ possèdent une arête dans le graphe $G$ et à $\infty$ dans le cas contraire. Pour chaque $k = 1, 2, \ldots, n$, on remplace successivement $A_{ij}$ par $\min\{A_{ij}, A_{i,k} + A_{k,j}\}$ ; la matrice $A$ contient donc les plus courts chemins entre chaque paire de points dans le graphe G étant donné les distances euclidiennes du voisinage données en 1.
4. Pour chaque paire de sous-graphe non connecté, on calcule la distance la plus petite que nous devons ajouter au voisinage pour que les sous-graphes soit connectés ; on ajoute ces distances à la matrice $A$.
5. Comme en 3 dans MDS, on applique l'opérateur $\tau$ à la matrice $A$ et on obtient la matrice $B$.
6. Trouver les valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ de $B$ et leur vecteurs propres respectifs $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}$.
7. Choisir les $d$ valeurs et vecteurs propres les plus significatifs en ordre décroissant : les coordonnées des points $y_i$ de l'incorporation dans l'espace à $d$ dimensions sont données par : $y_i = \sqrt{\lambda_p}\boldsymbol{v_n^i}$ où $\boldsymbol{v_n^i}$ est la $i$-ième composante du $p$-ième vecteur propre.

CHAPITRE 7

# Résultats et Analyse

Nous présenterons ici des résultats compilés à partir de l'application des 2 variantes d'Isomap présentées au chapitre précédent en les comparant avec Isomap quand celui-ci donne un résultat et avec d'autres méthodes quand ce sera possible. Il découle des chapitres précédents que ces algorithmes ne comportent d'avantages réels que lorsqu'ils sont appliqués à des données en très haute dimensionnalité. Un des ensembles de données présentés ne comporte pas cette particularité ; dans ce cas, nous allons vérifier qu'il n'est pas possible d'améliorer la "classifiabilité" de façon significative. Dans certains cas, la recherche de paramètres pour le *SVC* (classificateur à vecteur de support) servant à l'évaluation implique une multiplication du temps de calcul qui dépasse ce qui est possible d'effectuer ; dans cette éventualité l'espace de recherche est limité à un seul SVC, celui avec les paramètres par défaut pour un *SVC* avec un RBF ( fonction à base radiale) dans l'implémentation que nous avons utilisée (libsvm en java) : soit un coût $C = 1$ et un $\gamma = \frac{1}{k}$ (où $k$ est le nombre de dimensions). Cette limitation nous empêche d'atteindre l'optimalité, mais les résultats sont quand même éloquents. L'analyse de ces résultats suivra, il y sera présenté les conclusions qu'il est possible de tirer des expériences effectuées.

## 7.1 Présentation des Données

Dans cette section nous présentons les données que nous avons utilisées dans nos expériences. Le code source en java qui implémente les algorithmes présentés ici est entièrement disponible sur le site web de l'auteur au http ://www.iro.umontreal.ca/~payettf.

### 7.1.1 Ionosphère

Cet ensemble de données tire son origine du groupe de physique spatiale du laboratoire de physique appliquée de l'université John Hopkins. Ces données ont été colligées par un système de radars à Goose Bay, Labrador. Ce système consiste en un arrangement de 16 antennes à très haute fréquence avec un total de pouvoir de transmission de l'ordre de 6.4 kilowatts. L'étude concernait les électrons libres dans l'ionosphère. Les bonnes lectures sont celles démontrant une structure dans l'ionosphère, les lectures mauvaises sont celles qui n'en montrent pas ; leur signal est passé à travers l'ionosphère.

Il y a 351 échantillons de données, exprimés en 34 dimensions continues, et la totalité des données sont étiquetées avec la valeur 1 et la valeur 0.

### 7.1.2 ARCENE

L'ensemble ARCENE est composé de données spectrométriques de masse ; l'objectif est de distinguer entre les tissus cancéreux et des tissus normaux, un problème de classification binaire. Cet ensemble de données comporte 10000 dimensions, nous avons 200 exemples étiquettés(100 d'entraînement et 100 de validation) et 700 exemples sans étiquette, ce qui se prête bien au modèle d'apprentissage semi-supervisé que nous proposons ici : soit une réduction de dimensionnalité par RISIMAP couplé à un SVC pour la classification. Cet ensemble de données a été recompilé par Isabelle Guyon pour une compétition de *"'Feature Selection"* relié à NIPS 2003 ; à l'origine, les données ont été publiées par le "National Cancer Institute" et le "Eastern Virginia Medical School". La technique SELDI a été utilisée pour obtenir tous les résultats. Les points de données incluent des patients qui avaient le

cancer des ovaires ou de la prostate, et des patients de contrôle en santé. Pour plus de détails sur cet ensemble de données on peut voir http ://clopinet.com/isabelle/Projects/NIPS2003/Slides/ NIPS2003-Datasets.pdf.

## 7.2 Résultats Expérimentaux

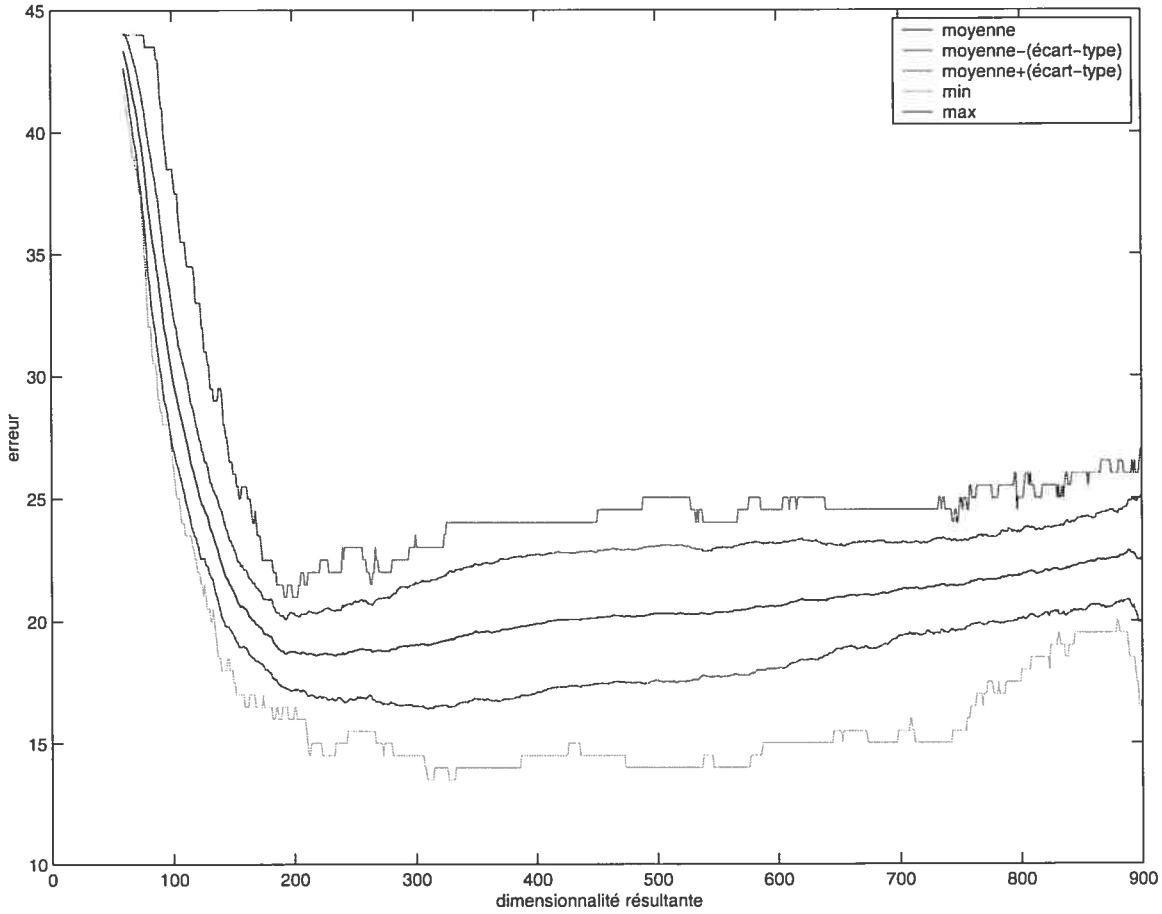
Les expériences que nous avons faites sont exploratoires ; des résultats plus complets pour d'autres ensembles de données prendraient beaucoup de temps puisque les versions des algorithmes que nous utilisons actuellement opèrent en $O(n^3)$, et calculer pour plusieurs paramètres serait extrêmement long pour des ensembles de données comportant plus de 1000 exemples étant donné les technologies actuelles. Par exemple, une application de RISIMAP sur MADELON qui comporte 4400 points de données a pris 3 jours sur une machine. Il y aurait moyen de mitiger cet impact en effectuant un sous-échantillionnement, nous allons aborder cette idée et ces inconvénients dans 7.3.

### 7.2.1 RISIMAP sur ARCENE

Nous avons effectué une recherche exhaustive dans l'espace de paramètres suivant sur la base de données ARCENE : nous avons utilisé la version d'évaluation de proximité qui est définie à l'aide des k plus proches voisins avec un voisinage de taille 6 à 40, et la dimension d'inclusion variant entre 60 et 900.

Dans le graphique 7.4 nous avons un aperçu de la variation de la solution. Il semble y avoir un point ou la moyenne est meilleure autour de 180-190 dimensions ; le maximum et le minimum et l'écart type se resserrent, ce qui indique que la taille du voisinage a un moins grand impact à cet endroit. À partir de 250 le graphique indique une dégradation notoire de l'erreur ; il y a clairement trop de dimensions pour la quantité de données présentes. Il semble que 900 points c'est très peu de données pour bien caractériser un problème qui s'inscrit dans un espace d'approximativement 200-250 dimensions ; malgré tout nous avons une moyenne d'erreur "cross-validée" 40 fois de $\sim$ 13%.

**Figure 7.1** – *L'erreur en fonction de la dimensionnalité résultante : on peut voir qu'il semble y avoir une dimension pour laquelle les résultats semblent meilleurs ; il y a au même endroit un étranglement du maximum, du minimum et de l'écart type avec la meilleure valeur de moyenne.*



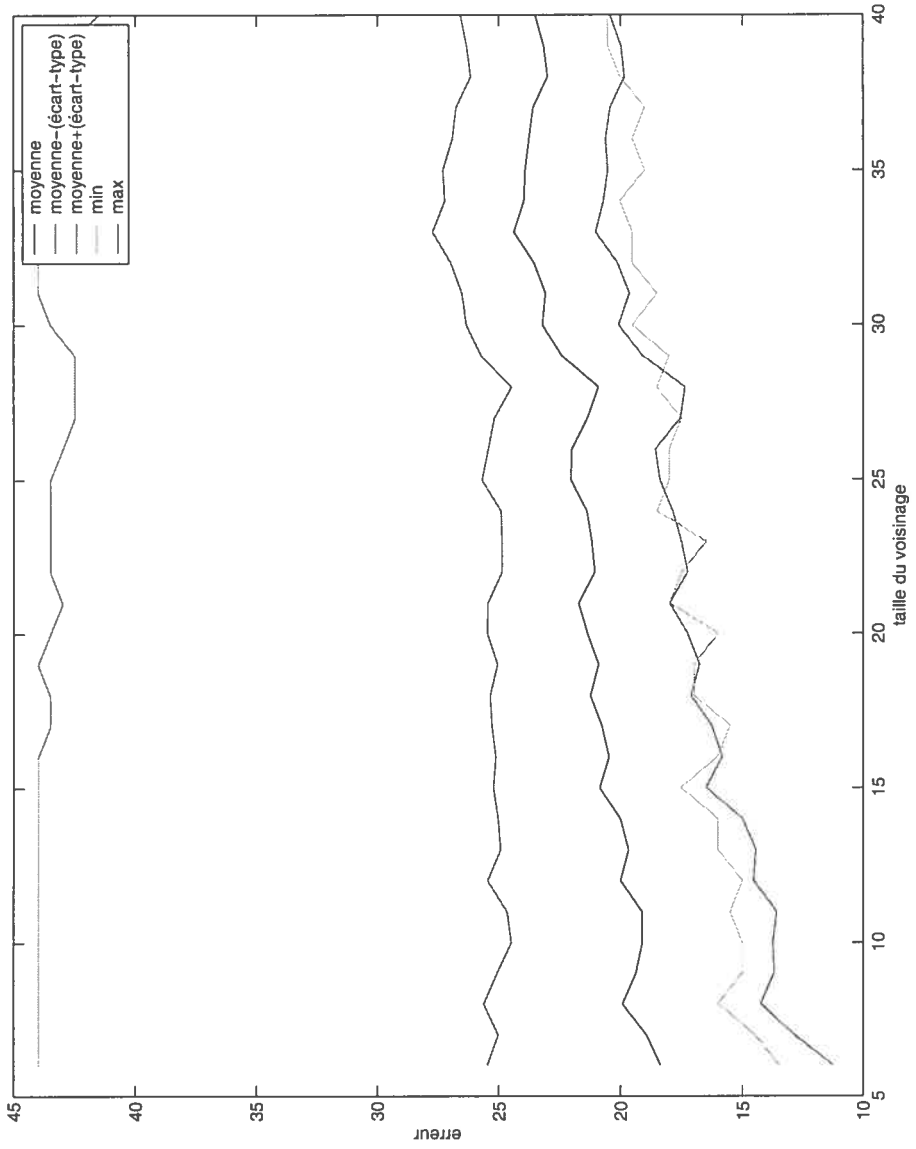
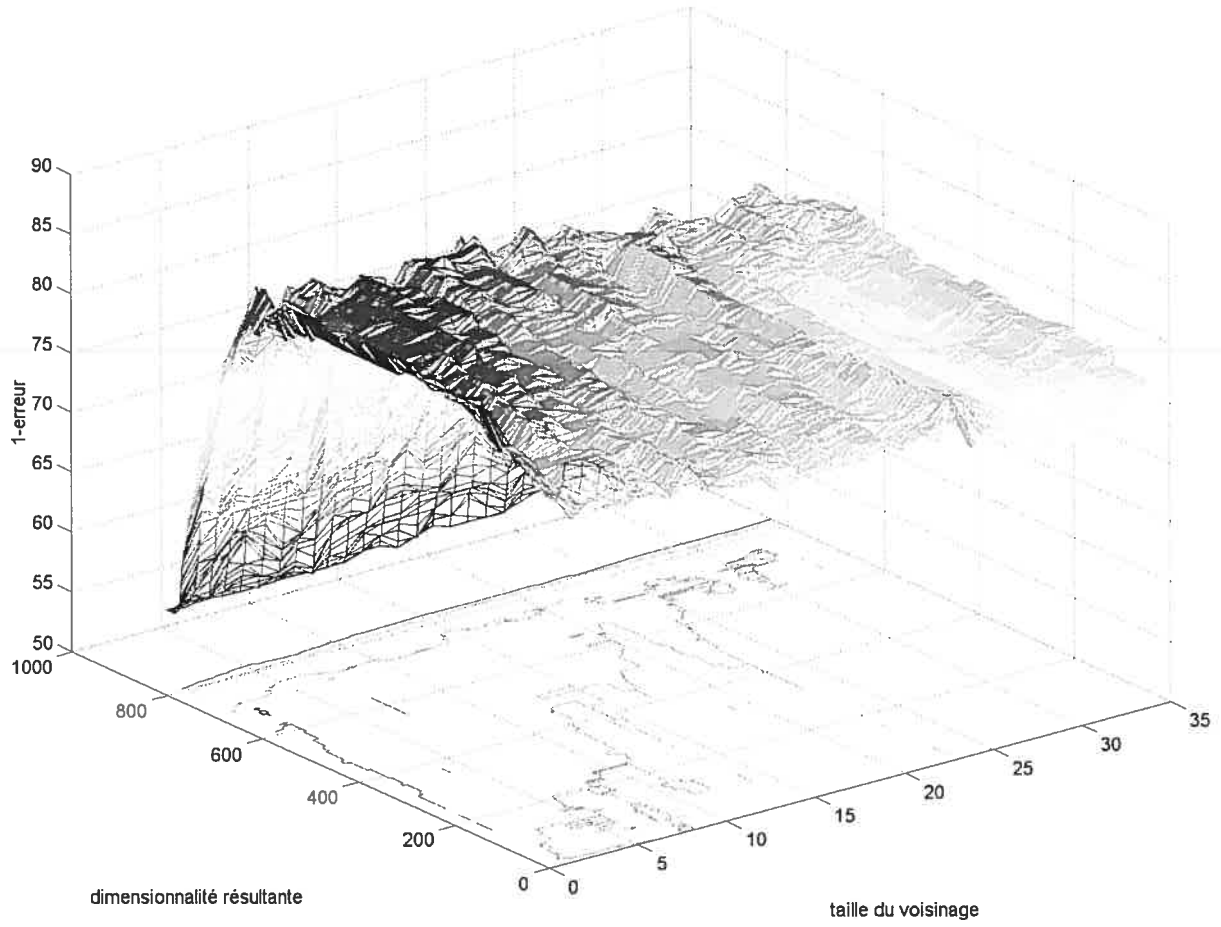



**Figure 7.2** – *L'erreur en fonction de la taille du voisinage.*

**Figure 7.3** – *Rendu 3D de 1−l'erreur de RISIMAP en fonction des 2 paramètres précédents sur ARCENE.*



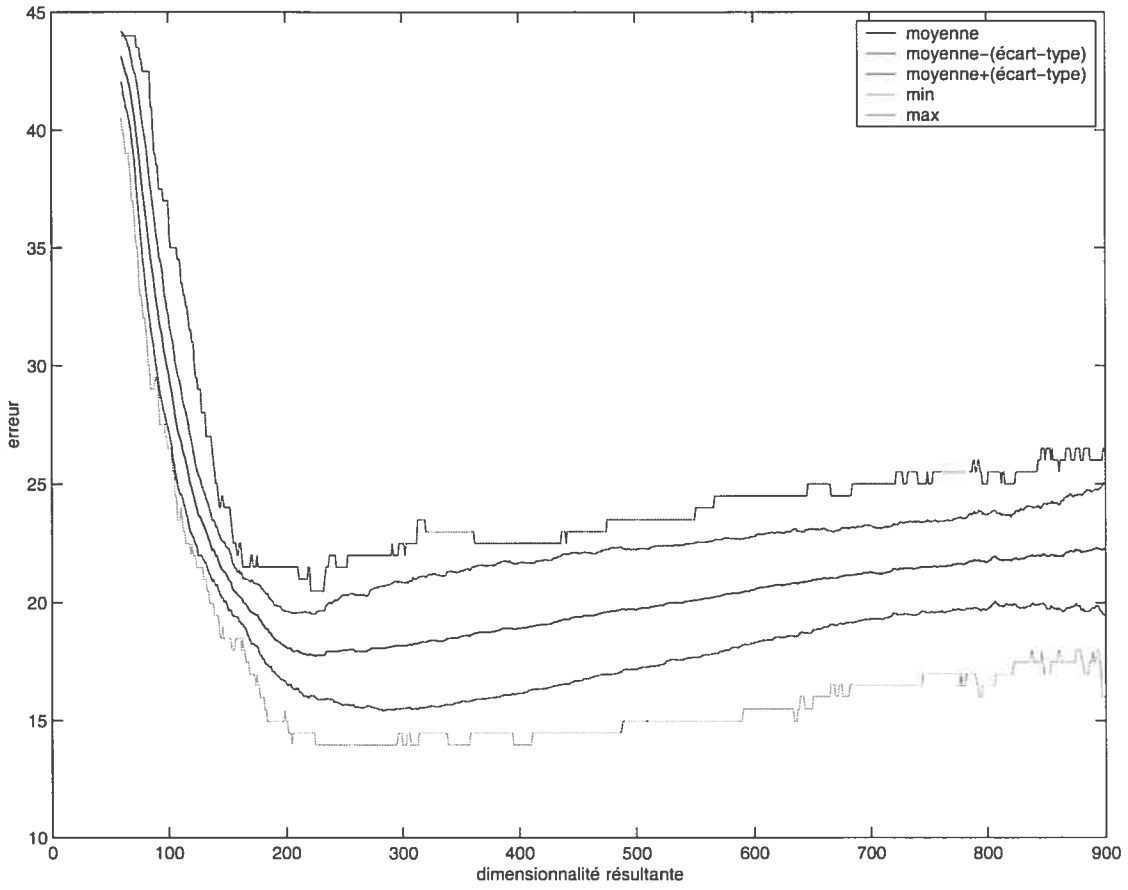
Dans la figure 7.5 nous avons un aperçu de la variation en fonction du voisinage. Il semble que cet aspect a un impact très discret sur le taux de rappel dans l'intervalle observé ; la pire moyenne pour un voisinage de 40 qui comprend presque 5% des points de l'ensemble de données résulte en moyenne à une erreur de 24%, alors que dans le meilleur des cas, avec un voisinage très restreint de taille 5, nous avons une erreur de 19% ; ce qui semble indiquer que la taille du voisinage n'a pas tellement d'impact dans un intervalle assez large autour de la valeur optimale ; nous avons essayé avec des voisinages plus grands (100, 300 ou 500) par exemple, avec des résultats désastreux de lissage : on perdait beaucoup d'information sur la variété ; la projection devient quasi-linéaire, ce qui résultait en des erreurs autour de 50%. Le fait que la courbe semble relativement horizontale sur un assez grand intervalle pourrait indiquer qu'un voisinage constant cause un lissage de la variété ; nous allons revenir a cette dernière possibilité dans la section 7.3.

Dans le graphique 3-D 7.6 nous avons l'ensemble des résultats colligés en un endroit : les paramètres de l'algorithme sur l'axe des $x$ et des $y$ et le pourcentage obtenu de classification correcte($1-erreur$) sur l'axe des $z$. Cette valeur de rappel est une validation croisée validée 40 fois évaluée par un SVM avec des paramètres de $C = 1$ et un $\gamma = \frac{1}{k}$ (où $k = 900 - 900/40$).

### 7.2.2 Isostrech sur ARCENE

Comme on peut le constater dans les graphiques suivants, la variation introduite par l'étirement des distances ne se traduit pas en une augmentation de la "classifiabilité". Tout au plus, elle cause un lissage : l'algorithme est un petit peu moins sensible à ses paramètres.

**Figure 7.4** – *L'erreur en fonction de la dimensionnalité résultante : on peut voir qu'il semble y avoir une dimension pour laquelle les résultats semblent meilleurs ; comparé à RISIMAP les résultats semblent se dégrader plus vite quand la dimensionnalité augmente.*



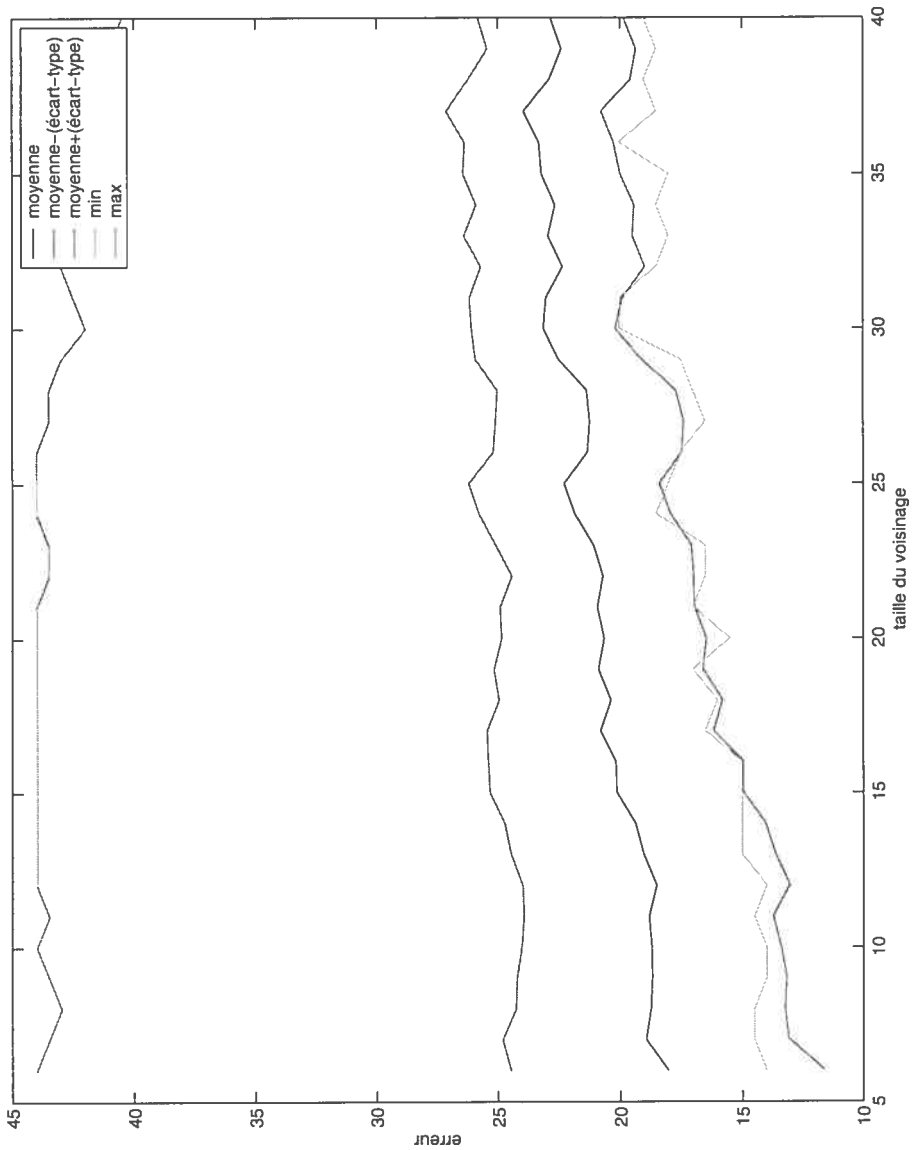
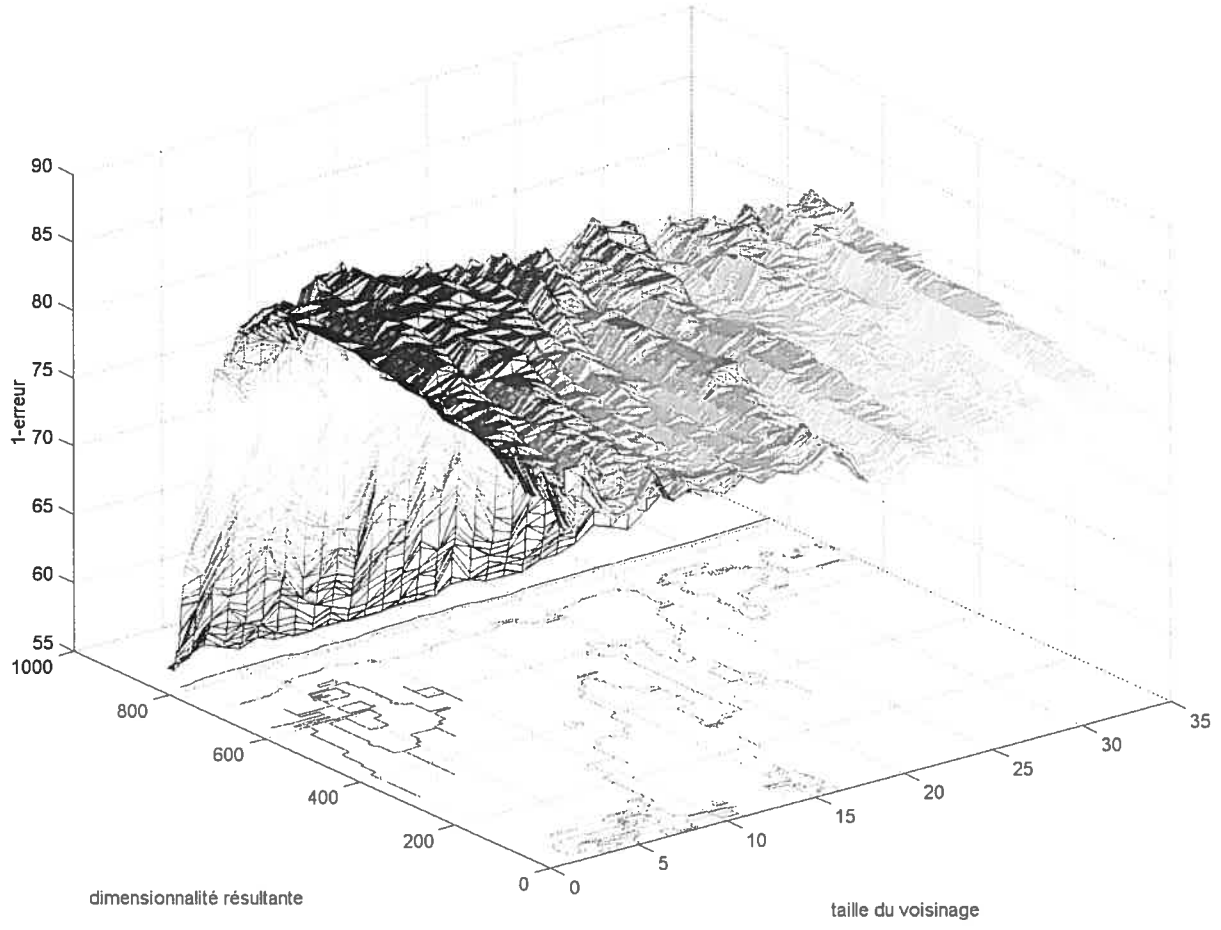



**Figure 7.5** – *L'erreur en fonction de la taille du voisinage.*

**Figure 7.6** – *Rendu 3D des valeurs de* 1−*l'erreur de Isostretch en fonction des 2 paramètres précédents sur ARCENE.*



### 7.2.3 Feature Selection NIPS2003 : ARCENE

La compétition de "Feature Selection" qui fut tenue dans le cadre de NIPS2003 possède un outil en ligne qui permet d'évaluer les méthodes de sélection de caractéristiques ; on peut y soumettre sa classification pour les points non étiquetés des ensembles de données et comparer le résultat à d'autres méthodes. Nous avons fait cette comparaison pour RISIMAP et Isostrech sur ARCENE sans avoir optimisé les paramètres du SVM, en utilisant toujours $C = 1$ et $\gamma = \frac{1}{k}$ et en éliminant les doublons. On peut voir dans le tableau 7.1 que RISIMAP arrive 11ième et Isostretch arrive 13ième. Nous avons choisi un peu heuristiquement 187 dimensions et 19 de voisinage comme paramètres pour RISIMAP et pour Isostretch. Il est presque certain qu'en faisant une recherche sur la combinaison de ces paramètres et ceux du SVM, on aurait des résultats encore meilleurs.

Fait intéressant à noter ; en explorant par validation croisée seulement les paramètres du SVM pour une variété résultante d'un couple de paramètres donnés(187, 19) on arrive à un meilleur taux de rappel validé, soit un taux d'erreur de 13% sur les exemples d'entraînement, mais quand on compare nos résultats pour les exemples de test avec l'outil en ligne, les résultats sont sensiblement les mêmes ; on fait même un peu moins bien qu'avec $C = 1$ et $\gamma = \frac{1}{k}$, ce qui est un signe de surapprentissage.

On peut déduire que l'information structurante extraite extraite par la réduction de dimensionnalité non-linéaire non-supervisé chevauche probablement celle extraite par le classificateur à noyau gaussien (RBF).

**Tableau 7.1** – *Résultats comparés sur ARCENE de NIPS2003; sans recherche de paramètres optimaux pour le SVM RISIMAP et Isostretch se qualifient à 3% du meilleur résultat*

| rang | nom | err. d'entr. | err. de val. | err. de test | auteur |
|---|---|---|---|---|---|
| 1 | final2-2 | 0.0000 | 0.0000 | 0.1073 | Yi-Wei Chen |
| 2 | RF+RLSC | 0.0000 | 0.0000 | 0.1112 | Kari Torkkola Eugene Tuv |
| 3 | CBAMethod3E | 0.0446 | 0.1347 | 0.1112 | CBAGroup |
| 4 | BayesNN-small+v | 0.0000 | 0.0000 | 0.1186 | Radford Neal |
| 5 | Bayesian + SVMs | 0.0268 | 0.0089 | 0.1247 | Chu Wei |
| 6 | RF with feature selection | 0.1258 | 0.0852 | 0.1263 | Vivian Ng; Leo Breiman |
| 7 | Nameless - by Amir and Ran | 0.0179 | 0.0292 | 0.1266 | Ran Bachrach |
| 8 | FS+SVM | 0.0000 | 0.0000 | 0.1276 | Navin Lal |
| 9 | IDEAL | 0.0000 | 0.0000 | 0.1304 | BorisovEruhimovTuv |
| 10 | GhostMiner - Pack 2 | 0.0000 | 0.0000 | 0.1353 | GhostMiner Team |
| 11 | RISIMAP | 0.0471 | 0.0406 | 0.1354 | François Payette |
| 12 | svmrbf | 0.0000 | 0.1737 | 0.1443 | Amir Navot |
| 13 | Isostretch | 0.0471 | 0.0406 | 0.1457 | François Payette |
| 14 | CLOVIS III | 0.0763 | 0.0877 | 0.1462 | P. Gouzien - F. Clérot |
| 15 | New-Bayes-large+v | 0.0000 | 0.0000 | 0.1469 | Radford Neal |
| 16 | testonlgpc | 0.1729 | 0.1583 | 0.1473 | Jovey |

### 7.2.4 RISIMAP sur Ionosphere

Pour l'expérience suivante, nous avons utilisé la version avec les k-plus proches voisins, avec une taille de voisinage variant entre 3 et 30, et une inclusion dans un espace de 2 à 30 dimensions.

Le graphique 7.7 nous laisse voir que l'erreur est minimisée quand les données sont incluses dans un espace de 10 à 15 dimensions. La moyenne de l'erreur semble être à son plus bas pour des voisinages de taille 15-20. Il semble que la taille de voisinage n'a pas beaucoup d'impact quand elle est choisie dans un assez grand intervalle autour de la valeur optimale.

On voit dans le graphique 7.8 que pour un voisinage de taille 5 à 30 il y a peu de variation. Par contre si on choisissait un voisinage de taille 200, (nous avons 351 points dans cet ensemble de données) on forcerait évidement une réduction de dimensionnalité beaucoup plus linéaire, et par le fait même, nous perdrions la structure intrinsèque de la variété.

Il est important de noter que dans cet exemple RISIMAP est équivalent à ISOMAP puisque l'ensemble de données n'est pas en très haute dimension (34), et que même avec les voisinages les plus petits, nous ne trouvons pas de sous-graphes indépendants.

Dans le rendu 3-D présenté dans le graphique 7.9 on peut voir la combinaison des 2 paramètres, la dimensionnalité intrinsèque et la taille du voisinage.

Nous n'avons pas présenté ici la recherche de paramètre effectuée sur le SVM par souci de consistance avec les résultats pour l'ensemble de données ARCENE puisque que pour ce dernier ensemble cette recherche de paramètre nécessiterait beaucoup trop de temps : Il suffit de mentionner qu'en explorant brièvement les paramètres du SVM nous avons obtenu un taux d'erreur $< 1\%$ validé 15 fois.

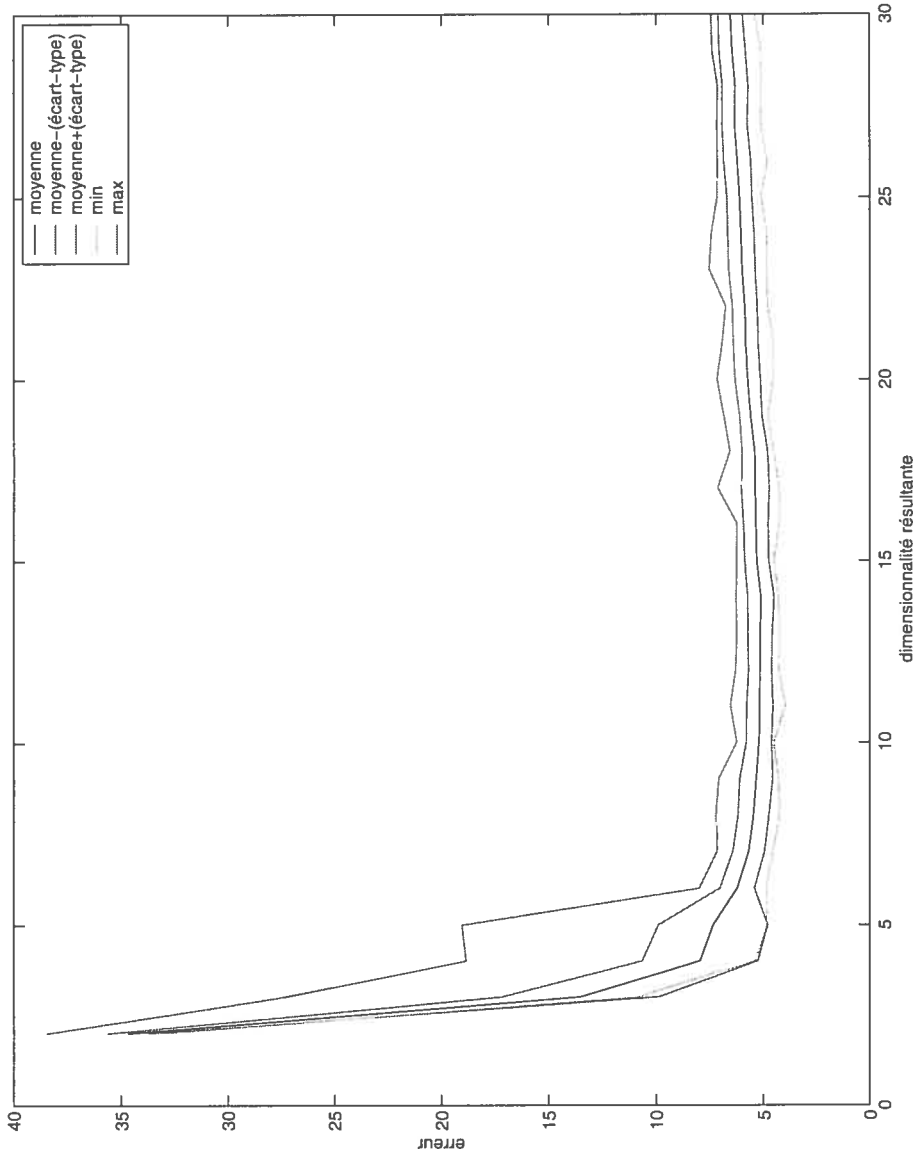



**Figure 7.7** – *Erreur de RISIMAP en fonction de la dimensionnalité sur Ionosphere*

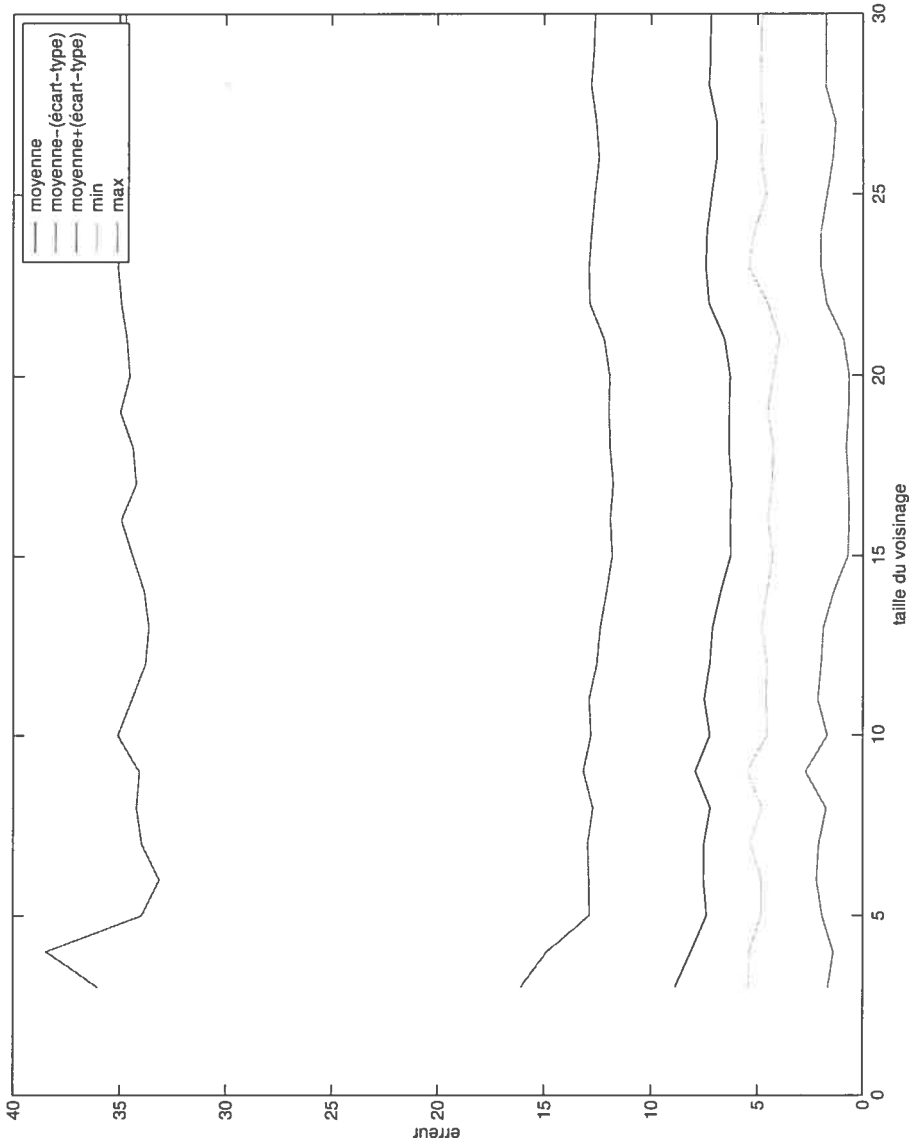
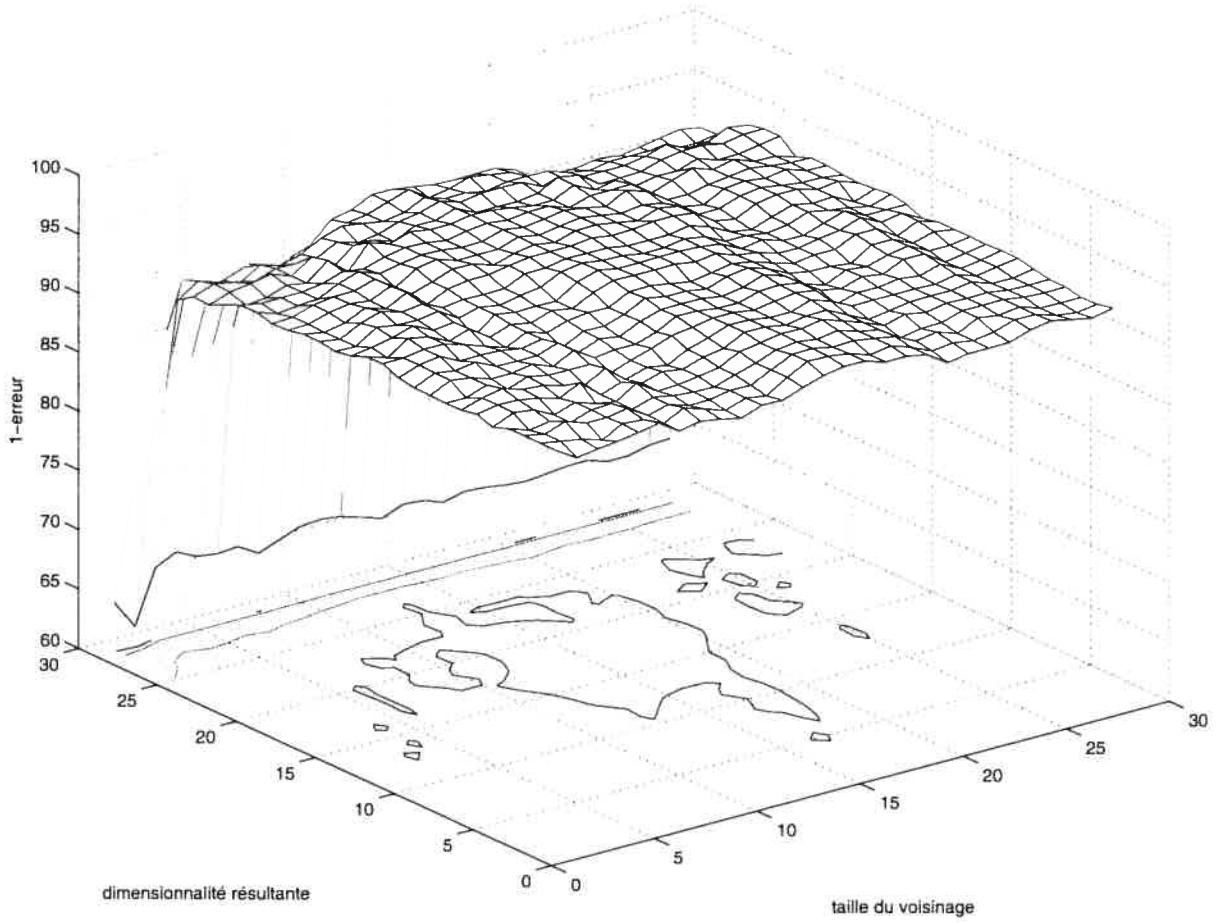



**Figure 7.8** – *Erreur de RISMAP en fonction de la taille du voisinage sur Ionosphere*

**Figure 7.9** – *Rendu 3D des valeurs de rappel de RISIMAP en fonction des 2 paramètres précédents sur Ionosphere*



### 7.2.5 Isostretch sur Ionoshpere

Pour l'expérience suivante, par souci de conformité avec l'expérience précédente, nous avons aussi utilisé la version avec les k-PPV, avec une taille de voisinage variant entre 3 et 30, et l'inclusion dans un espace de variant de 2 à 30 dimensions.

On constate qu'en comparaison à RISIMAP, Isostretch cause un lissage de l'erreur par rapport aux paramètres de taille de voisinage et de dimensionnalité d'inclusion ainsi que l'introduction d'une petite erreur additionnelle. En aucun cas Isostretch ne donne de meilleurs résultats que RISIMAP ; ce qui est décevant, mais relativement prévisible.

En variant les paramètres du SVM qui effectue la classification à partir de la variété perturbée apprise par Isostretch nous avons réussi à obtenir un taux d'erreur relativement bas ; très près mais un peu plus élevé que celui obtenu dans pareilles circonstances pour RISIMAP. Nous ne présentons pas ces résultats ici sur Ionosphère pour la même raison que pour RISIMAP, par souci de consistance.

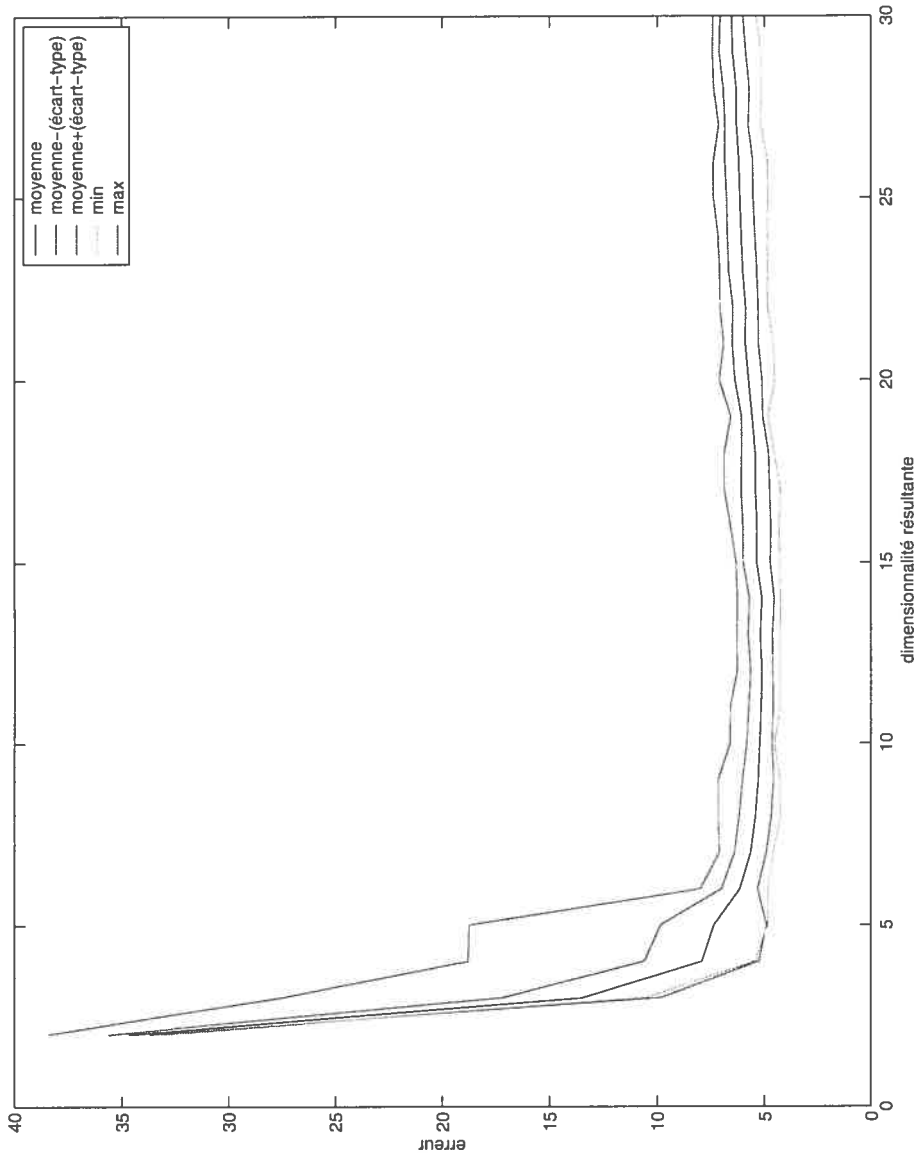



**Figure 7.10** – *Erreur de Isostretch en fonction de la dimensionnalité sur Ionosphere*

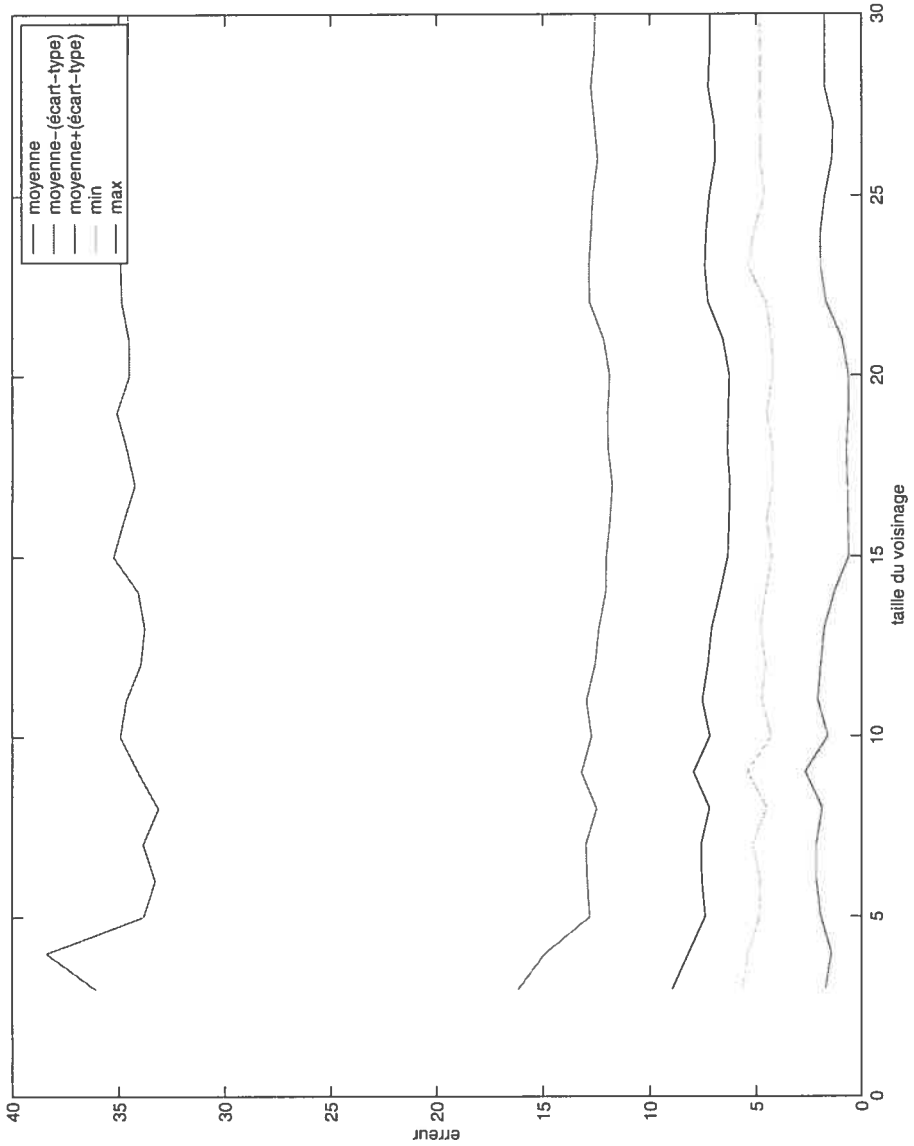
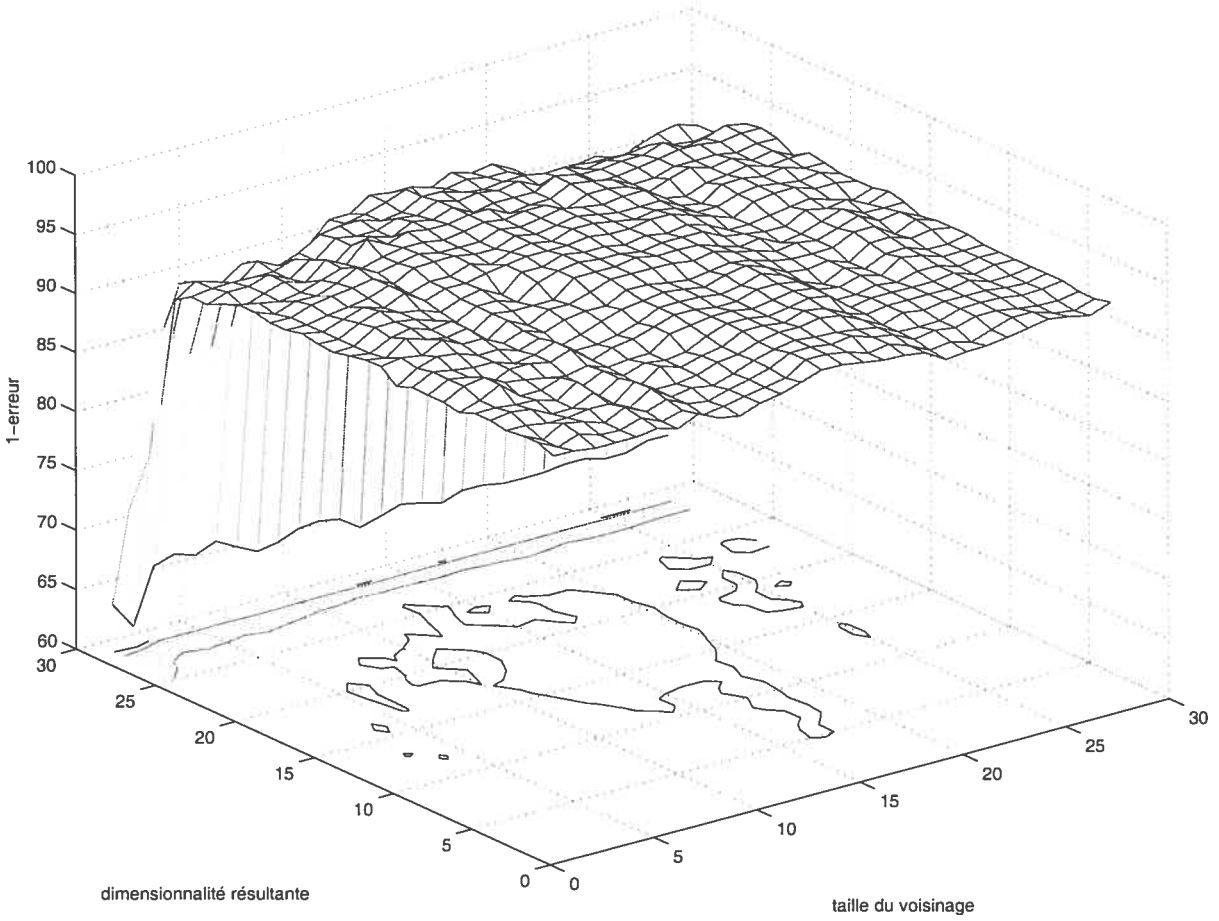



**Figure 7.11** – *Erreur de Isostretch en fonction de la taille du voisinage sur Ionosphere*

**Figure 7.12** – *Rendu 3D des valeurs de rappel de Isostretch en fonction des 2 paramètres précédents sur Ionosphere*



## 7.3 Analyse

### 7.3.1 RISIMAP

L'élimination de sous-graphes indépendants dans les ensembles de données à très haute dimensionnalité semble donner de bons résultats ; il n'est pas vraiment possible de comparer ces résultats à Isomap où on aurait pas éliminé ces sous-graphes puisque l'extraction de valeur singulières et des vecteurs propres correspondants ne serait pas possible pour des voisinages de taille restreinte. Pour ARCENE, par exemple, la décomposition d'ISOMAP ne converge pas à cause de la présence de sous-graphes à partir de voisinages de taille plus petite que 500 points de données. Appliquer Isomap avec un voisinage de 500 est à toutes fins pratiques équivalent à faire une projection linéaire, qui sur cet ensemble de données cause un aplatissement rendant tout apprentissage significatif impossible.

Le principal désavantage de l'utilisation des méthodes de réduction de dimensionnalité dérivé d'Isomap à des fins de classification est le temps de calcul requis afin d'obtenir la variété intrinsèque. Nous avons vu au chapitre 5 que pour diagonaliser une matrice de Gram non creuse, les méthodes modernes nécessitent un temps de calcul dans l'ordre de $O(30n^3)$ opérations.

Pour éviter cette charge computationnelle qui peut rendre incalculable une application à un ensemble comprenant plus de 10 000 points, il est possible d'utiliser quelques trucs, mais non sans introduire une perte. On pourrait par exemple sous-échantillonner l'ensemble de points, faire la réduction de dimensionnalité non linéaire, et projeter sur cette variété les points qui ont été exclus de l'échantillon ; mais cette opération résulterait certainement en une perte d'information et de précision de la géométrie réelle de la variété, puisqu'on éliminerait de l'information possiblement importante à la caractérisation du bruit.

Le recours à la décomposition en valeurs et vecteurs propres de la matrice de Gram est une borne assez dure pour l'utilisation des algorithmes de réduction de dimensionnalité linéaire et non linéaires sur des tâches d'apprentissage dont l'ensemble d'entraînement comporte un nombre relativement

élevé de points, puisque cette matrice est de taille $O(n^2)$ où $n$ est le nombre de points.

L'intérêt est donc grand pour les méthodes de réduction de dimensionnalité comme LLE qui mettent en scène des matrices creuses [1] : Dans ces derniers cas, l'isolation des valeurs et vecteurs propres pourrait potentiellement se faire en un ordre de grandeur plus rapidement, rendant accessible(en termes de temps de calcul d'ordinateurs actuels) à la réduction de dimensionnalité non linéaire des ensembles de donnés confinés jusqu'ici à des algorithmes plus rapides, moins sophistiqués, ou plus heuristiques.

### 7.3.2 Isostretch

Les résultats empiriques semblent démontrer que la distorsion de la distance entre les points en utilisant les étiquettes qu'effectue Isostretch ne donne pas une inclusion meilleure pour des fins de classification. Elle insensibilise l'erreur de classification de l'algorithme aux variations de ses paramètres ; la moyenne de l'erreur mesurée par validation croisée est un peu plus lisse avec l'utilisation cette variation de l'algorithme, ce qui est une bien mince consolation. La variété récupérée n'est donc pas plus utile à la tâche de classification, contrairement à ce que nous avions supposé au départ.

On peut conclure de cette observation qu'en étirant la distance entre les points possédant des étiquettes différentes, on insère un bruit qui nuit à la récupération d'une variété optimale, celle sur laquelle la tâche de classification donnes les meilleurs résultats.

Il semble donc que l'apprentissage semi-supervisé qu'Isostretch effectue est sous-optimal par rapport à celui que RISIMAP effectue ; les perturbations induites par notre fonction de distance variable à la matrice de Gram ne sont pas bénéfiques à l'obtention d'un minimum local meilleur pour l'erreur.

[1] ou possédant d'autres propriétés désirables

CHAPITRE 8

# Conclusion

## 8.1 Contributions théoriques

Ce travail est, à notre connaissance, la première étude de l'impact de l'utilisation à des fins de classification de méthodes de réduction de dimensionnalité non-linéaire comportant une garantie asymptotique de stabilité topologique. L'apprentissage semi-supervisé qui en résulte est intéressant et mérite certainement plus d'études afin de bien comprendre les tenants et aboutissants de cette approche.

Nous avons y avons présenté deux propositions de notre cru : RISIMAP, une variante de l'algorithme Isomap qui permet l'usage de celui-ci avec des données en très haute dimensionnalité en éliminant les sous-graphes indépendants dans le graphe d'adjacence résultant de Djikstra, et Isostretch, une variante du même algorithme proposant une distorsion de la matrice de Gram afin de chercher une variété plus propice à la classification. Cette dernière proposition se révèle moins intéressante que prévue ; l'introduction de perturbation affecte définitivement la fidélité de la variété récupérée.

## 8.2 Pistes futures

La réduction de dimensionnalité non linéaire est définitivement un outil utile à l'apprentissage semi-supervisé. Cette approche va certainement continuer d'évoluer par des investissements de recherche dans les années à venir. Voici quelques idées inexplorées qui semblent prometteuses.

Pour réduire l'impact accru du bruit sur le sous-échantillonnage comme proposé en (7.3.1), on pourrait calculer la moyenne locale d'un nombre donné de "clusters" (la taille maximale que notre temps de calcul alloué nous permet, par exemple), et utiliser ces moyennes pour trouver la variété, pour ensuite trouver la position de chaque point sur cette variété à l'aide des valeurs et vecteurs propres trouvés sur la matrice de Gram correspondante, cette piste semble bonne puisqu'elle serait probablement moins sensible au bruit local et permettrait de récupérer rapidement la structure globale.

Une autre piste intéressante s'inscrivant dans la réduction de dimensionnalité à des fins d'apprentissage semi-supervisé est de ne pas choisir les valeurs propres les plus grandes, mais plutôt celles qui classifient mieux les données selon les étiquettes des points, évaluées soit avec un SVM ou tout simplement avec les PPV.

En effet, il n'y a pas de garantie implicite que les informations nécessaires à une bonne classification se retrouvent dans les vecteurs propres dont les valeurs propres associées ont les plus grandes valeurs : Tout ce que cette valeur signifie c'est que ces directions sont celles de plus grande variance dans les données. Certains ensembles de données produisent certainement des matrices de Gram pour lesquelles les vecteurs et valeurs propres autres que ceux correspondant aux plus grandes valeurs propres contiennent plus d'information utile à la classification, et, en choisissant à l'aide des plus grandes valeurs propres, nous passons outre cette information structurante.

Une piste qui a été brièvement explorée lors de ce travail de recherche (mais non-mentionné ici parce que les résultats préliminaires n'étaient pas probants) est la suivante : appliquer un algorithme de réduction de dimensionnalité aux points possédant une des étiquettes puis au sous-ensemble possédant l'autre étiquette. On combine ensuite les deux sous-espaces pour créer un nouvel

espace dans lequel les points non étiquetés sont projetés étant donné leur position dans les 2 sous-variétés ; ce positionnement dans ce nouvel espace est utilisé pour entraîner le SVM.

D'autres avenues plus générales et intéressantes d'apprentissage semi-supervisé à l'aide de réduction de dimensionnalité sont des définitions alternatives de voisinage, des sous-échantillionnements en fonction des étiquettes, et l'étude des relations plus profondes entre les valeurs et vecteurs propres et le positionnement résultant.

Nous concluons sur une note philosophique par une citation de Henri Poincaré qui résume bien le cheminement parcouru par l'auteur lors de l'exercice menant à ce document et en général ce qu'est la recherche.

> *Un scientifique doit savoir organiser. On construit une science avec des faits de la même façon que l'on construit une maison avec des pierres ; mais une accumulation de faits ne constitue pas plus une science qu'un tas de pierres une maison.*

# Références


AIZERMAN, M., E. BRAVERMAN et L. ROZONOER (1964), « Theoretical foundations of the potential function method in pattern recognition learning », *Automation and Remote Control*, p. 821–837.

ARONSZAJN, N. (1950), « Theory of reproducing kernels », *Transactions of the American Mathematical Society*, p. 337–404.

BELLMAN, R. (1961), *Adaptive Control Processes : A Guided Tour*, Princeton : Princeton University Press.

BENGIO, Y., O. DELALLEAU, J.-F. PAIEMENT, P. VINCENT, M. OUIMET et N. L. ROUX (2004), « Learning Eigenfunctions Links Spectral Embedding and Kernel PCA », *Neural Computation 16*, p. 2197–2219.

BENGIO, Y., P. VINCENT, J.-F. PAIEMENT, O. DELALLEAU, M. OUIMET et N. L. ROUX (2003), « Spectral Clustering and Kernel PCA are Learning Eigenfunctions », Rapport technique 1239, Département d'informatique et recherche opérationnelle, Université de Montréal.

BERNSTEIN, M., V. DE SILVA, J. LANGFORD et J. TENENBAUM (2000), « Graph approximations to geodesics on embedded manifolds », Rapport technique 2000.

BOSER, B. E., I. M. GUYON et V. N. VAPNIK (1992), « A training Algorithm for Optimal Margin Classifiers », *Proceedings of the fifth annual workshop on Computational learning theory*, p. 144–152.

BRASSARD, G. et P. BRATLEY (1996), *Fundamentals of Algorithmics*, New Jersey, USA : Prentice Hall.

BROWN, J., M. CHU, D. ELLISON et R. PLEMMONS (1994), « Proceedings of the Cornelius Lanczos International Centenary Conference », *Proceedings of the Cornelius Lanczos International Centenary Conference*, Philadelphia, USA, SIAM Publications,

CHATFIELD, C. et A. COLLINS (1980), *Introduction to Multivariate Analysis*, London UK : Chapman & Hall.

COX et COX (2001), *Multidimensional Scaling (2nd ed.)*, Boca Raton USA : Chapman & Hall/CRC.

DUDA, HART et STORK (2000), *Pattern Classification (2nd ed.)*, New York USA : Wiley Interscience.

FISHER, R. (1936), « The use of multiple measurements in taxonomic problems », *Annals of Eugenics 7*, p. 179–188.

GOLUB, G. et C. VANLOAN (1996), *Matrix Computations (3rd Edition)*, Baltimore, USA : Johns Hopkins University Press.

GOWER, J. (1966), « Some distance properties of latent root and vector methods in multivariate analysis », *Biometrika 53*, p. 325–338.

HOTELLING, H. (1933), « Analysis of a Complex of Statistical Variables into Principal Components », *Journal of Educational Psychology 24*, p. 339–354.

HSU, C.-W., C.-C. CHANG et C.-J. LIN (2002), « A practical guide to support vector classification », Rapport technique 12, Department of Computer Science and Information Engineering, National Taiwan University.

J.CULLUM et R. WILLOUGHBY (1985), *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, Boston, USA : Birkhauser.

KUNG et DIAMANTARAS (1996), *Principal Components Neural Networks : Theory and Applications*, New York USA : John Wiley.

MARDIA, K., J. KENT, et J. BIBBY (1979), *Multivariate analysis*, London UK : Academic Press.

MERCER, J. (1909), « Functions of positive and negative type and their connection with the theory of integral equations », *Philos. Trans. Roy. Soc. London*, p. 415–446.

NG, A. Y., M. I. JORDAN et Y. WEISS (2002), « On spectral clustering : analysis and an algorithm », *Advances in Neural Information Processing Systems*, Volume 14. Cambridge, MA : The MIT Press,

PRESS, W. H., B. P. FLANNERY, S. A. TEUKOLSKY et W. T. VETTERLING (1992), *Numerical Recipes in C (2nd Edition)*, Cambridge, UK : Cambridge University Press.

ROWEIS, S. et L. SAUL (2000), « Nonlinear Dimensionality Reduction by Locally Linear Embedding », *Science 290*, p. 2323–2326.

ROWEIS, S. et L. SAUL (2003), « Think Globally, Fit Locally : Unsupervised Learning of Low Dimensional Manifolds », *Journal of Machine Learning Research 4*, p. 119–155.

SCHÖLKOPF, B., S. MIKA, A. SMOLA, G. RÄTSCH et K.-R. MÜLLER (1998), « Kernel PCA pattern reconstruction via approximate pre-images », *Proceedings of the 8th International Conference on Artificial Neural Networks*, Berlin, De, Springer Verlag, p. 147–152.

SCHÖLKOPF, B. et A. SMOLA (2002), *Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond*, Cambridge MA : MIT Press.

STOER, J. et R. BULIRSCH (1980), *Introduction to Numerical Analysis*, New York, USA : Springer-Verlag.

TENENBAUM, J. et V. DE SILVA (2002), « Unsupervised learning of curved manifolds », *Nonlinear Estimation and Classification*. New York : Springer-Verlag, p. 453–466.

TENENBAUM, J. et V. DE SILVA (2003), « Local versus global methods for nonlinear dimensionality reduction », *Advances in Neural Information Processing Systems*, Volume 15. Cambridge, MA : The MIT Press, p. 705712.

TENENBAUM, J., V. DE SILVA et J. LANGFORD (2000), « A global geometric framework for nonlinear dimensionality reduction », *Science 290*, p. 2319–2323.

TORGERSON, W. (1952), « Multidimensional scaling : 1. Theory and method », *Psychometrika 17*, p. 401–419.

WEISS, Y. (1999), « Segmentation using eigenvectors : a unifying view », *Proceedings IEEE International Conference on Computer Vision*, p. 975–982.

YOUNG et HOUSEHOLDER (1938), « Discussion of a set of points in terms of their mutual distances », *Psychometrika 41*, p. 505–529.