SI 221

Bases de la Reconnaissance des Formes

I. Bloch

B. Burtschy

L. Likforman-Sulem

J-M. Nicolas

M. Sigelle

TELECOM ParisTech

édition 2014-2015

AVERTISSEMENT

Le polycopié de l'unité d'enseignement « Bases de la Reconnaissance des Formes » offre aux étudiants un document de base, solide et concis pour une première approche de la reconnaissance des formes statistique et neuronale. Une introduction aux modèles de Markov est également présente.

Le polycopié est organisé de la façon suivante. Les différents modules qui constituent un système de reconnaissance des formes, la distinction entre approches statistique et structurelle, les applications, ainsi que les notions d'espace de représentation et de métrique sont présentés au chapitre 1. Le chapitre 2 concerne les méthodes d'apprentissage supervisé de la décision statistique, qui permettent de connaître les lois de densité de probabilité ou déterminer les surfaces discriminantes entre classes à partir de vecteurs échantillons. On y trouvera aussi la règle de décision Bayésienne ainsi que celle des k-ppv (k plus proches voisins). De plus, on introduit la notion de paramètres et on présente les descripteurs de Fourier. Le chapitre 3 présente les méthodes automatiques d'inspection et de visualisation des nuages de points, de réduction de la dimension de l'espace de représentation apportées par l'analyse de données. Les chapitres 4 et 5 présentent les méthodes d'apprentissage non supervisées, permettant de séparer en classes un nuage de points dont les classes ne sont pas connues. Ces méthodes se divisent en méthodes hiérarchiques et non hiérarchiques (méthodes de k-moyennes). Le chapitre 6 est consacré aux méthodes à base de réseaux neuronaux dont les représentants les plus connus sont le perceptron et les réseaux multi-couches. Les règles d'apprentissage basées sur des algorithmes de descente de gradient calculent les poids des connexions entre les différentes couches du réseau. Dans le chapitre 7, on considère que les formes résultent d'un processus stochastique et chaque forme est représentée par une séquence de vecteurs ou de symboles (observations). L'apprentissage consiste à estimer les paramètres de modèles associés à chaque classe de forme. La méthode de décision consiste à rechercher le modèle le plus vraisemblable au vu de la séquence d'observations, cette vraisemblance étant calculée par un algorithme de programmation dynamique (Viterbi).

Laurence LIKFORMAN-SULEM

coordinatrice de l'unité d'enseignement

TABLE DES MATIERES

1. Introduction à la reconnaissance des formes (L. Likforman)

1. Présentation 9
2. Domaines d'application 10
3. Schéma général 11
4. Système d'acquisition et prétraitements 11
5. Extraction de caractéristiques 12
6. Apprentissage et décision 14
    6.1 Principes de décision 14
    6.2 Apprentissage 15
    6.3 méthodes statistiques et structurelles 15
    6.4 approches stochastiques et neuronales 16

2. Analyse statistique (L. Likforman)

1. Introduction 17
2. Paramètres 18
    2.1 Descripteurs de Fourier 18
    2.1.1 Descripteurs basés sur les contours 18
    2.1.2 Descripteurs pour les courbes polygonales fermées 19
3. Théorie Bayésienne de la décision 21
    3.1 Principes 21
    3.2 Probabilité d'erreur 23
    3.3 Décision Bayésienne avec coûts et rejet 25
    3.4 Erreurs de type I et II 28
4. Estimation paramétrique 30
    4.1 Cas des lois normales 30
    4.2 Surfaces discriminantes 31
5. Discrimination paramétrique non Bayésienne 34
    5.1 Principes de la séparation linéaire 35
    5.2 Règle du perceptron 36
    5.3 Interprétation connexionniste 37
    5.4 Minimisation aux moindres carrés 38
6. Discrimination Bayésienne non paramétrique 38
    6.1 Principes 39
    6.2 Fenêtres de Parzen 41
    6.3 Estimateur des kn plus proches voisins 42
7. Méthodes des k ppv 43
    7.1 Principe 43
    7.2 Probabilité d'erreur 44
    7.3 Variantes rapides des k-ppv 46
        7.3.1 Méthode de Vidal 46
        7.3.2 Méthode de Kittler 48
        7.3.3 Méthode de Friedman 49

7.3.4 Méthode de condensation 50
7.3.5 Algorithme d'édition 51
7.3.6 Conclusion 51
8. Evaluation/amélioration des performances 52
8.1 Intervalles de confiance 52
8.2 Validation croisée 53
8.3 Combinaison de classifieurs 53

3. Analyse des données (B. Burtschy)

1. Introduction 57
2. Définitions et notations 59
3. Les 4 types de variables 60
3.1 Les variables nominales 60
3.2 Les variables ordinales 61
3.3 Les variables intervalles 61
3.4 Les variables métriques 61
3.5 Les variables qualitatives et quantitatives 62
4. Quelques types de tableaux de données 62
4.1 Les tableaux quantitatifs et de notes 64
4.2 Les tableaux de préférences et de rangs 64
4.3 Les tableaux de contingence 67
4.4 Les tableaux de variables qualitatives 67
4.5 Les tableaux de présence-absence et tableaux logiques 68
5. Choisir la bonne méthode... 69
5.1 Méthodes descriptives et explicatives 69
5.2 Forme des résultats 71
5.2.1 les méthodes géométriques 71
5.2.2 La classification hiérarchique 72
5.2.3 La typologie 72
5.2.3 Les équations de comportement 73
5.3 Les méthodes de l'analyse de données 73
6. Les objectifs de l'analyse 74
6.1 L'exploration des données 74
6.2 Résumer, réduire, classer 74
6.3 La découverte des faits nouveaux 74
6.4 Validation d'hypothèse 75
6.5 La prédiction de comportements 75
7. Bibliographie 76
8. Annexe 1 : l'analyse en composantes principales 77
9. Annexe 2 : l'analyse discriminante 82

4. Classification automatique : méthodes non hiérarchiques (I. Bloch, J-M Nicolas)

1. Introduction 87
2. Hypothèses des structures probabilistes connues 88
2.1 Expression des paramètres des lois sous jacentes 88
2.2 Exemple de la loi normale 89

2.3 Exemple d'algorithme itératif 90
2.4 Synthèse 91
3. Méthode simple d'approximation 91
4. Critères de classification 92
4.1 Choix des critères 92
4.2 Exemple d'algorithme 93
5. K-moyennes 94
6. Algorithme ISODATA 96
7. Boules optimisées 97
8. Nuées dynamiques 98
9. Fuzzy C-Means 99
10. Limites 100

5. Méthodes hiérarchiques (I. Bloch, J-M Nicolas)

1. Introduction 103
2. Rappels sur les graphes et les arbres 103
2.1 Graphes 103
2.2 Arbres 104
2.3 X-arbres et dendogrammes 105
3. Classification hiérarchique 105
3.1 Ecart et distance entre objets 106
3.1.1 Définitions 106
3.1.2 Exemple 106
3.2 Ecart et distance entre un objet et un groupe d'objet 107
3.3 Classification ascendante hiérarchique 108
3.3.1 Définition 108
3.3.2 Exemple 108
3.3.3 Les effets du choix des écarts 110
3.4 Hiérarchie et hiérarchie indicée 110
4. Approche par la distance ultramétrique 110
4.1 Distance ultramétrique 111
4.2 Distance de chaîne 111
4.3 Relation d'équivalence et partition à partir d'une ultramétrique 113
4.4 Hiérarchie et ultramétrique 113
4.5 Equivalence entre hiérarchie indicée et ultramétrique 114
5. Conclusion 115

6. Modèles connexionnistes-perceptrons multicouches (J.M Nicolas)

1. Réseau connexionniste et reconnaissance des formes 117
1.1 Historique 118
1.2 Les réseaux neuromimétiques en reconnaissance des formes 118
2. Architecture d'un réseau connexionniste 119
2.1 Structures d'un réseau connexionniste 120
2.2 Les perceptrons 123
3. Apprentissage des perceptrons sans couches cachées 125
3.1 Modification des poids : la règle Delta 125

3.2 Un cas particulier de la règle Delta : la règle du perceptron 128
3.3 Apprentissage des perceptrons mono-couches 129
4. Apprentissage des perceptrons multi-couches 129
4.1 L'algorithme de rétropropagation du gradient 130
4.2 Apprentissage des perceptrons multi-couches 133
5. Mise en œuvre des réseaux neuromimétiques 133
5.1 "Apprendre" la valeur σ_j de la fonction sigmoïde 134
5.2 Classification en c classes 135
5.3 Stratégies d'apprentissage 137
6. Ce qu'il faut retenir des réseaux neuromimétiques 141
6.1 Théorème d'existence 141
6.2 Performances et robustesse des réseaux neuromimétiques 142
6.3 Amélioration des capacités de généralisation 142
6.4 Le rôle des prétraitements 142
6.5 Les réseaux de neurones en pratique 143
6.6 Pratiquer les réseaux de neurones 144

7. Chaînes de Markov et Modèles de Markov Cachés (M. Sigelle)

1. Modèle stochastique 146
2. Chaîne de Markov à temps discret et espace d'états fini 146
2.1 chaîne de Markov d'ordre 1 stationnaire 147
2.2 cas d'une station météo 147
2.3 durée de vie d'un état 147
3. HMM : chaîne de Markov cachées 148
3.1 caractérisation d'un HMM 148
3.2 définitions et notations pour un HMM à temps discret 148
3.3 caractéristiques des modèles HMM 148
3.4 exemple : reconnaissance de l'écriture cursive 149
3.5 exemple : reconnaissance de la parole 149
3.6 simulation d'une HMM 150
4. Apprentissage-reconnaissance 150
4.1 estimation des paramètres en données complètes 151
5. Trois problèmes fondamentaux 153
5.1 introduction aux variables backward-forward 153
5.2 application des variables backward-forward 154
5.3 formule à deux états 154
5.4 calcul des variables backward-forward 155
6. Estimation des paramètres en données incomplètes 156
7. Algorithme EM 157
8. Recherche de la segmentation optimale 159
9. Récapitulation pour les HMM 160
9. 1 apprentissage d'un modèle 160
9. 2 reconnaissance : observation 160

CHAPITRE 1 : INTRODUCTION

Laurence Likforman-Sulem

1. Présentation

La reconnaissance des formes est issue de différentes disciplines qui sont les mathématiques (probabilités et statistiques), les sciences de l'ingénieur, l'informatique et plus récemment l'intelligence artificielle. C'est à partir des années 60 que la reconnaissance des formes est devenue une discipline spécifique. L'extraordinaire développement des machines informatiques ces dernières années a donné un élan à la RdF en permettant des applications temps réel, en particulier dans le domaine des applications visuelles et auditives. Les procédés d'acquisition tels que caméra, scanner sont très accessibles, ainsi que des ordinateurs à la fois puissants et bons marchés. Ils permettent le traitement de nombreuses données en un temps raisonnable comme cela est souvent nécessaire en RdF.

L'objectif de la RdF est de réaliser des systèmes informatisés qui simulent les activités humaines de perception, de reconnaissance et de compréhension : reconnaissance de l'écrit, de la parole, interprétation de scènes, robotique, reconnaissance de signaux médicaux EEG (électroencéphalogramme), ECG (électrocardiogramme). Cela implique aussi une certaine pluridisciplinarité pour comprendre l'aspect physique des capteurs, les aspects mathématiques de la classification, ceux relatifs à l'informatique.

Les systèmes de Reconnaissance des formes intègrent toute la chaîne perception-reconnaissance depuis l'acquisition des données brutes jusqu'à la compréhension élaborée de ces données. Ces dernières ont entre temps subi de nombreuses transformations. De ce fait, la RdF fait appel à des disciplines connexes telles que le traitement du signal et de l'image, et l'Intelligence Artificielle. Prenons l'exemple de la lecture d'images de textes imprimés : l'OCR (Optical Character Recognition). Il ne s'agit pas seulement de reconnaître des caractères mais aussi de segmenter l'image en zones, sélectionner les zones de texte, découper celles ci en lignes, mots et caractères. Ces analyses font appel au traitement du signal. Après la reconnaissance proprement dite, celle ci peut être améliorée en utilisant des modèles de langage.

2. Domaines d'application

Parmi les domaines les plus populaires de la RdF on trouve la reconnaissance de l'écriture et de la parole. En ce qui concerne l'écrit, on pense souvent à l'O.C.R (Optical Character Recognition) qui est la reconnaissance des caractères dans les textes imprimés. On parle de système monofonte, multifonte, omnifonte suivant que le système traite une, plusieurs ou n'importe quelle police de caractères On trouve dans le commerce de bons systèmes O.C.R multifontes qui nécessitent toutefois des documents de bonne qualité pour atteindre les taux de reconnaissance affichés.

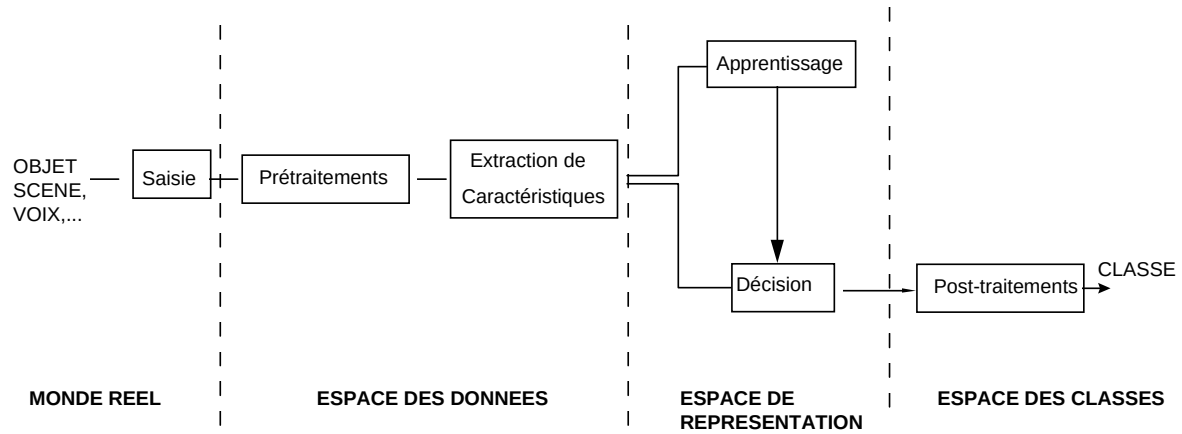


Figure 1 - schéma général d'un système de reconnaissance des formes

La reconnaissance des adresses postales, montants de chèques, formulaires sont des applications industrielles importantes qui traitent l'écriture manuscrite non contrainte. Le vocabulaire doit cependant être limité. La variabilité intra-scripteur et inter-scripteurs de l'écriture sont un défi majeur pour la reconnaissance.

Les systèmes traitant de l'écriture cursive (phrases) intègrent à la fois des techniques relatives à la RdF et à celles de l'IA pour les traitements cognitifs (syntaxe, lexique, sémantique).
L'écriture manuscrite *en ligne* est obtenue à partir d'un stylet adapté et d'une tablette graphique. Des paramètres de vitesse, pression du mouvement de l'écriture sont alors enregistrés pour la reconnaissance des mots ou l'authentification[1] de scripteurs.

Pour le traitement de la parole, on parlera de même de systèmes mono-locuteur si celui est adapté à une personne donnée, multi-locuteurs s'il est adapté à plusieurs personnes connues du système ou omni-locuteurs pour les usages tout public. Ici encore, le degré de difficulté est différent suivant que l'on traite des mots énoncés séparément ou de la parole continue où le défi est justement de segmenter les données en mots. Les systèmes de sécurité utilisent la voix pour identifier ou authentifier des locuteurs. D'autres paramètres biométriques comme

1 On distingue l'identification de l'authentification. Lors d'un processus d'authentification, l'individu clame son identité et le système vérifie que l'individu n'est pas un imposteur.

les empreintes digitales, la signature, les images (2D, 3D) de la main et du visage, sont également utilisés.

Sans pouvoir être exhaustifs, citons d'autres domaines très actifs. L'interprétation d'images aériennes ou satellites conduisant à la surveillance, les prévisions agricoles, celles des images et signaux médicaux pour des tâches de comptage ou de repérage de cellules ou d'événements anormaux, la détection de défauts (pièces industrielles, industrie alimentaire, ...)

3. Schéma Général

Une chaîne de traitement dans un système de reconnaissance comprend plusieurs modules (figure 1) et plusieurs espaces de travail. L'objectif de la reconnaissance des formes va être de définir une suite d'opérations permettant de passer de l'espace des données ou formes, à l'espace des classes où la catégorie de la forme a été estimée ou déterminée. Ces opérations sont en pratique des procédures informatisées.

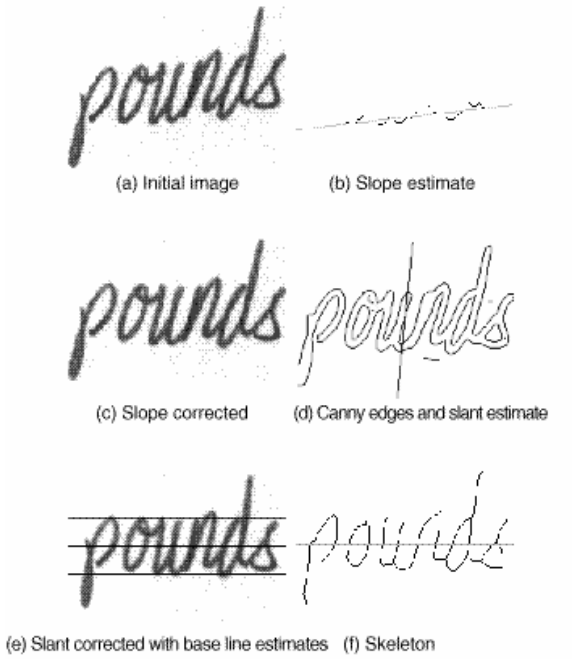


Figure 2- exemple de prétraitements effectués sur des images de mots cursifs : correction de l'inclinaison de la ligne de base et des traits (d'après [Senior and Robinson 98])

4. Système d'acquisition et prétraitements

Suivant la nature du signal, un capteur (caméra, microphone, ...) est nécessaire pour acquérir le signal[2] sous forme numérique ou analogique (il faut alors le convertir en numérique) pour qu'il soit traitable par un système informatisé. On passe ainsi du monde réel au monde des formes ou données.

2 Le terme signal désigne ici aussi bien un signal mono-dimensionnel comme la parole, qu'une image ou un signal multi-dimensionnel comme les images multi-spectrales

Les prétraitements sont spécifiques à un domaine et ont pour utilité de réduire les bruits de capteurs ou inhérents au signal. Ils peuvent servir aussi de préparation aux phases suivantes en réduisant la variabilité du signal. La figure 2 montre le résultat d'une série de prétraitements servant à redresser des mots cursifs.

5. Extraction de caractéristiques

L'espace des formes est généralement trop vaste pour en faire une description exhaustive. Considérons par exemple toutes les façons d'écrire le chiffre 3.

D'autre part, le signal numérique contient beaucoup d'informations qui ne sont pas toutes d'égale importance : une page de texte contient plus de pixels blancs de fond que de pixels noirs d'écriture.
L'objectif de la reconnaissance des formes est de regrouper les formes en catégories à partir de leur caractéristiques communes. On peut par exemple décrire le chiffre 3 par : forme possédant 2 boucles non fermées l'une au dessus de l'autre. Cette description par caractéristiques, ici relative à la structure de la forme, est aussi une réduction notable d'information par rapport à l'image du chiffre elle même. De nombreuses possibilités sont offertes au concepteur du système pour le choix des caractéristiques :

- de type statistique, en extrayant un vecteur de caractéristiques qui consiste en différentes mesures systématiques sur la forme analysée. Ce sont les attributs ou composantes.
- de type structurel : en recherchant à décomposer la forme en constituants élémentaires appelés primitives. La représentation est alors une ordonnée de primitives. Les primitives peuvent être aussi associées sous forme de graphe ou d'arbre.

D'autre part, les caractéristiques doivent permettre de distinguer les différentes classes de formes entre elles.

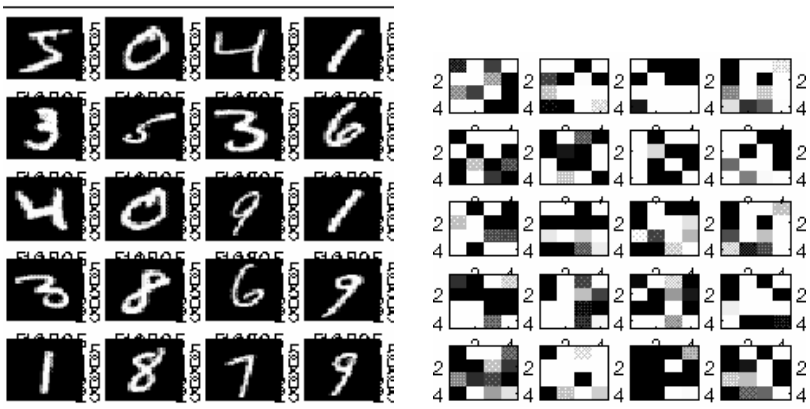


Figure 3 - gauche : exemples de caractères (base NIST), droite : 16 premières composantes principales

Quelques distances (métriques) utilisées dans les méthodes statistiques sur les vecteurs à d composantes sont :

- la distance euclidienne : $d(X , Y) = \sqrt{\sum_{k=1}^{d} |x_k - y_k|^2}$

- la distance L1 : $d(X , Y) = \sum_{k=1}^{d} |x_k - y_k|$

- la distance de Chebychev ou L_∞ ou convergence uniforme : $d(X , Y) = \max_k |x_k - y_k|$

Les méthodes d'analyse de données s'attachent à visualiser et décrire les données dans l'espace R de représentation. Pour réduire la quantité d'information à traiter, notamment la dimension du vecteur de caractéristiques, la projection des formes dans un espace où les caractéristiques sont non corrélées (figure 3) est la méthode d'analyse en composantes principales (cf. chapitre 3)

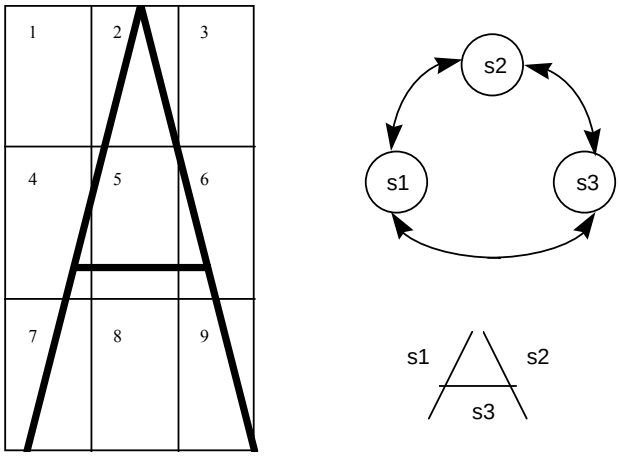


Figure 4 - exemple simplifié d'extraction de caractéristiques sur une image de caractère. Gauche : l'image est découpée en 9 zones. Sur chacune d'elles, on extrait le pourcentage de pixels noirs. L'ensemble des 9 valeurs forme le vecteur de caractéristiques.
Droite : la forme est découpée en segments s1, s2 et s3. L'ensemble de ces segments est agencé sous forme de graphe : s1 est lié à s2, s3 coupe s1 et s2,....

6. Apprentissage et Décision

L'étape suivante consiste à assigner une catégorie ou classe à la représentation extraite. Il s'agit de trouver une fonction de classement[3] d qui permette de passer de la représentation à l'étiquette.

6.1 Principes de décision

Soit Ω, l'espace des formes, i.e. ensemble de tous les formes possibles des objets à analyser .
Soit R, l'espace de représentation, ensemble possible des mesures ou représentations extraites sur les formes
Soit J, l'espace des classes, i.e. ensemble des étiquettes.

Par exemple pour une reconnaissance (statistique) d'image de chiffres isolés,
Ω =ensemble de toutes les images possibles représentant les chiffres de 0 à 9,
J={0,1,2,3,4,5,6,7,8,9}
$R = R^9$ si les mesures représentent des comptages dans 9 zones de l'image

Soit C , la fonction de classement idéale, qui associe à chaque forme sa classe véritable sans jamais se tromper.
C : Ω -----> J
Soit X, la fonction[4] qui à une forme associe sa représentation :
X : Ω -----> R
La fonction de classement d cherchée opère sur la représentation des formes :
d : R -----> J

l'espace Ω peut alors s'écrire sous la forme : $\Omega = \{\omega_1, \omega_2, ... \omega_K\}$
Les ω_i étant des sous parties de Ω constituées des formes de la classe i : $\omega i = C^{-1}$ (i)

La construction de la fonction d va rencontrer les difficultés suivantes : données altérées par le bruit, distorsions de forme, variabilité intra-classe (formes ou tailles différentes dans une même catégorie). Il faut aussi distinguer les classes entre elles, alors qu'elles peuvent être de forme similaire (lettres 'O' et 'Q', chiffre '1' et lettre 'l'). Un apprentissage, à partir d'exemples, est nécessaire pour construire la fonction de décision.

3 les termes de 'décision', 'classement' et 'classification' son équivalents dans ce contexte

4 Les fonctions C et X sont considérées en reconnaissance statistique comme des variables aléatoires. La notation P(ωi | x) équivalant à P(C=i|X=x) (voir chapitre 2)

6.2 Apprentissage

L'apprentissage consiste à apprendre les caractéristiques communes aux classes et à distinguer les différentes classes entre elles. On constitue un ensemble d'apprentissage à partir d'exemples des différentes classes.
Le nombre minimum d'exemples dépend de la dimension de l'espace de représentation et de la méthode utilisée : plus la dimension de R est élevée, plus il faudra d 'exemples. De même, les méthodes nécessitant une estimation ou les méthodes basées sur les k-ppv ont de bonnes propriétés de convergence si elles sont élaborées à partir de grands ensembles d'apprentissage.
Si l'ensemble d'apprentissage sert à construire la procédure ou fonction de décision, il faut aussi que les performances du système restent bonnes pour des exemples ne faisant pas partie de l'ensemble d'apprentissage : c'est la propriété de généralisation.

On distingue 2 types d'apprentissage statistique : supervisé et non supervisé.

Lors d'un apprentissage supervisé, la classe (représentée par son l'étiquette) de chaque objet est fournie au programme d'apprentissage en même temps que les données.

Parfois un apprentissage non supervisé est nécessaire, quand les classes sont mal définies *a priori*. Les classes sont alors déterminées automatiquement en formant des nuages de points dans l'espace de représentation. Ce type d'apprentissage est l'objet des méthodes de classification automatiques (cf chapitre 5).

Les méthodes structurelles utilisent aussi l'apprentissage : inférence grammaticale ou constitution de prototypes.

6.3 Méthodes statistiques et structurelles

Le principe des méthodes statistiques consistent à partitionner, lors de l'apprentissage, l'espace de représentation en régions. Chaque région est affectée à une classe. Lors de la décision, une réalisation sera affectée à la classe de la région à laquelle elle appartient. Cette partition est explicite dans les méthodes de discrimination linéaire ou paramétrique où la frontière entre classes est recherchée. Elle est implicite dans les méthodes bayésiennes ou de k-ppv.

Les méthodes structurelles s'appuient sur la comparaison des représentations structurées avec des représentations prototypes conservées en mémoire : comparaison de graphes, de chaînes et d'arbres. Les grammaires permettent aussi de représenter de manière générique les classes. Une représentation est alors acceptée ou rejetée par la grammaire dans un mécanisme d'analyse syntaxique utilisant des automates.

6.4 Approches stochastiques et neuronales

Les méthodes neuronales sont très utilisées en traitement de la parole et de l'écrit. Elles utilisent une structure de cellules et connexions en réseau. L'apprentissage consiste à déterminer les poids de connexion optimaux. Ces structures permettent de construire implicitement des frontières de n'importe quelle forme et sont donc utiles quand les classes ne sont pas linéairement séparables.

Les méthodes stochastiques sont actuellement en vogue pour leurs bonnes capacité d'apprentissage et de généralisation dans les systèmes de traitement de la parole et plus récemment de l'écriture. Ils évitent la segmentation des mots en unités élémentaires (phonèmes ou lettres) pas toujours nettement présentes dans le signal. Ces méthodes sont utilisables pour un signal, de type stochastique qui se déroule temporellement ou spatialement.

Bibliographie

Duda, Hart, Stork : Pattern Classification and Scene Analysis, 2nd edition - Wiley 2001

Devijver, Kittler : Pattern Recognition. A statistical Approach - Prentice Hall 82

M. **Kunt, G. Coray, G. Granlund, J-P Haton, R. Ingold, M. Kocher** :
Reconnaissance des Formes et Analyse de scènes, traitement de l'Information : Volume 3, presses polytechniques et universitaires romandes, 2000

A. Senior, A. Robinson : An off-line cursive handwriting recognition system, IEEE PAMI, Vol 20, no 3, mars 1998.

Cornuéjols, L. Miclet, Apprentissage artificiel : concepts et algorithmes, Eyrolles, 2002

CHAPITRE 2 : DECISION STATISTIQUE

Laurence Likforman-Sulem

1. Introduction

Les méthodes qui vont être présentées dans ce chapitre appartiennent à la classe des méthodes statistiques à *apprentissage supervisé*. Cette démarche suppose que l'on dispose d'un ensemble de points d'apprentissage appartenant aux différentes classes possibles du problème considéré et que pour chaque point, sa classe soit connue. Ces points d'apprentissage permettent de faire des hypothèses sur la distribution statistique des objets de chaque classe. On supposera généralement que les classes suivent des lois de densité de probabilité normales, et on cherchera à estimer les paramètres de ces lois (vecteur moyenne, matrice de covariance) à partir des échantillons d'apprentissage (section 3). Si l'hypothèse de gaussiannité n'est pas pertinente, on cherchera à estimer les lois de distribution de probabilité caractéristiques de chaque classe directement sur l'ensemble d'apprentissage. C'est le but des méthodes d'estimation des fenêtres de Parzen et celle des kn plus proches voisins (section 4).

Une fois l'apprentissage réalisé, les méthodes de décision se séparent en 2 groupes. D'une part les méthodes dites Bayésiennes (section 3) où on affecte à une forme inconnue la classe à laquelle le point à le plus de chances d'appartenir. La décision Bayésienne est une méthode de classement qui garantit un taux global d'erreur minimal. Le deuxième groupe concerne les méthodes dites non Bayésiennes telles que celles de la section 5, qui recherchent les frontières entre les classes pour positionner le point inconnu par rapport à ces frontières. Les frontières choisies sont le plus souvent des hyperplans, et ce sont les méthodes de séparation linéaires. Une autre approche non Bayésienne est la méthode des k plus proches voisins (section 7) où on classe directement le point inconnu en fonction de la classe de ses plus proches voisin dans l'ensemble d'apprentissage.

Les méthodes statistiques supposent l'extraction d'un vecteur de paramètres de taille fixe pour chaque forme considérée. Le choix de ces paramètres dépend fortement du domaine d'application et fait largement appel à l'expérience et à l'intuition de l'analyste. Nous présentons ici des paramètres classiques pour les objets plans, qui sont les descripteurs de Fourier (section 2).

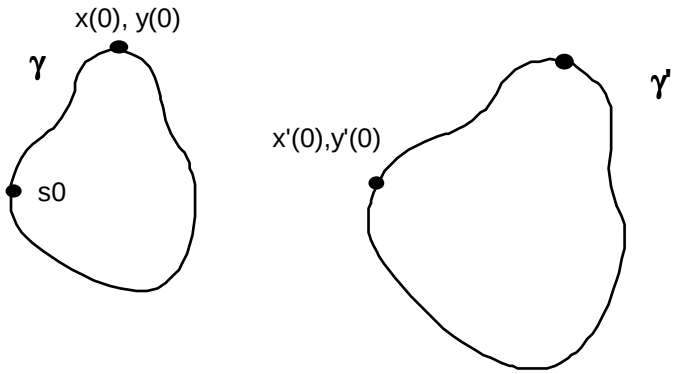
2. Paramètres

Les paramètres sont extraits des formes dans le but de discriminer les classes. A une forme correspond donc un vecteur dont les éléments sont les valeurs des paramètres, propres à la forme considérée. Dans les méthodes de type statistique, les paramètres sont généralement des mesures de type numérique. L'expérience et l'intuition guident souvent l'analyste dans le choix des paramètres dont le nombre est déterminé, en rapport avec la taille de l'ensemble d'apprentissage.
Nous présentons ci dessous des paramètres caractérisant des formes 2-D (images, dessins) appelés descripteurs de Fourier, proches de la théorie du Signal. Ils présentent de bonnes propriétés relatives aux déformations causées par translation, homothétie et rotation.

2.1 Descripteurs de Fourier

2.1.1 Descripteurs basés sur les contours

On suppose que l'on dispose de formes dont on a extrait les contours et que ces contours sont fermés. Une première définition de descripteurs est décrite ci dessous.



On considère la courbe continue γ qui correspond aux points du contour. La courbe est définie par (x(s), y(s)) où s représente l'abscisse curviligne, son point de départ étant (x(0), y(0)). On pose :

u(s) = x(s) + j y(s)

Si P est le périmètre de la courbe, alors u(s) est périodique de période P. u étant périodique, elle se décompose en série de Fourier dont la forme complexe est :

$$u(s) = \sum_{n=-\infty}^{+\infty} a_n e^{\frac{jns2\pi}{P}}$$

Les a_n sont les descripteurs (ici complexes) de Fourier et se calculent par :

$$a_n = \frac{1}{P}\int_0^P u(s) e^{\frac{-j2\pi ns}{P}} ds \quad n=...-1, -2, 0, 1, 2, ...$$

Ces descripteurs sont ceux de la courbe γ considérée. Considérons maintenant quelques propriétés de ces descripteurs.

On fait subir à γ une translation (d'un vecteur Z), une rotation (d'angle ϕ), une dilatation (de facteur R). De plus on modifie le point de départ (x(0), y(0)) par un décalage de longueur s0 dans le sens inverse des aiguilles d'une montre. On appelle γ' la courbe obtenue.
On peut montrer que les descripteurs a'_n de la courbe γ' se déduisent de ceux de la courbe γ par:

$a'_n = e^{jn\tau} R e^{j\phi} a_n$ avec $\tau=2\pi s_0/P$ $(\tau \in [0, 2\pi])$

$a'_0 = a_0 + Z$

Ces propriétés sont intéressantes pour normaliser, puis comparer deux formes.

On s'intéresse maintenant au calcul effectif des descripteurs a_n. Dans le cas présent, la courbe γ est connue par ses points de contours. On suppose que l'on dispose de N points de contours équidistants de δs (ce qui nécessite un ré-échantillonage en distance des points le long de γ).

On a $P=N\delta s$ et l'abscisse curviligne du kième point du contour u(k) est $s = k\,\delta s$.

on en déduit la formule suivante :

$$a_n = \frac{1}{N}\sum_{k=0}^{N-1} u(k) e^{\frac{-j2\pi nk}{N}}$$

Les a_n, dont on ne cherche qu'un nombre limité, se calculent donc par une Transformation de Fourier Discrète.

2.1.2 Descripteurs pour les courbes polygonales fermées

On peut définir d'autres descripteurs en associant au contour une fonction périodique le caractérisant, que l'on développe de la même façon en série de Fourier. La fonction donnant en chaque point le changement net d'orientation depuis le point de départ, en est un exemple (Zahn et Roskies, 1972).

Soit x(l), y(l) un point du contour d'une courbe fermée de périmètre L, orientée dans le sens des aiguilles d'une montre. l est l'abscisse curviligne le long de la courbe.

on définit la fonction ϕ (l) représentant le changement net d'orientation entre le point de départ et le point l par :

ϕ (l) = Θ(l) - Θ (0)
on a : ϕ (0) = 0 et ϕ (L) = -2π
on pose : $t = \frac{2\pi l}{L}$

On définit la fonction normalisée $\phi*$ sur $[0, 2\pi]$, en posant :

$t = t = \frac{2\pi l}{L}$ et $\phi* (t) = \phi [\frac{Lt}{2\pi}] + t$

$\phi*$ est périodique de période 2π et $\phi^* (0) = \phi^* (2\pi) = 0$

Par définition, ϕ^* est invariante pour la translation et la rotation. On peut aussi montrer qu'elle est invariante par homothétie.

$\phi*$ se décompose en série de Fourier, ses coefficients étant caractéristiques de la courbe.

$$\phi* (t) = A_0 + \sum_{k=1}^{+\infty} A_k \cos(kt - \alpha_k)$$

L'étude de quelques propriétés des harmoniques et des phases donne les résultats suivants :

Si deux courbes γ et γ' sont identiques, avec des points de départ différents :

$A'_k = A_k$ et $\alpha'_k = \alpha_k + k\Delta\alpha$ avec $\Delta\alpha = \frac{-2\pi\Delta l}{L}$

Δl étant la longueur le long de la courbe pour passer du point de départ de γ à celui de γ' en parcourant la courbe dans le sens des aiguilles d'une montre.

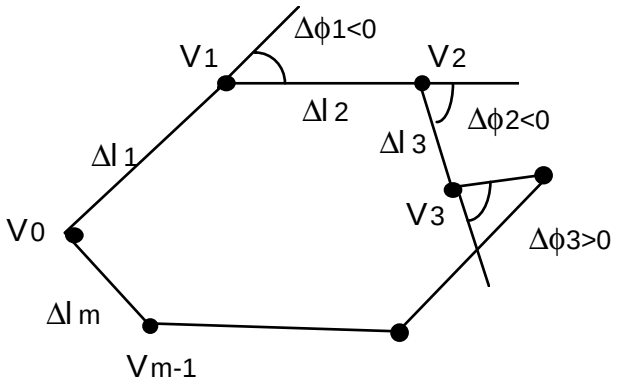
Si deux courbes γ et γ' sont miroir l'une de l'autre, avec des points de départ identiques:

$A'_k = A_k$ et $\alpha'_k + \alpha_k = \pi \pmod{2\pi}$

Pour une courbe γ ayant une symétrie de rotation d'ordre n

$A_k = 0$ pour $k \neq 0 \pmod n$

Les A_k étant invariants par rotation, homothétie, translation, et changement du point de départ, les premières amplitudes sont souvent utilisées pour caractériser les formes. D'autres paramètres utilisant les phases peuvent être définis pour distinguer les formes miroir, ou repérer les formes à symétrie axiale (voir Zahn et Roskies, 1972)



Dans le cas d'une courbe polygonale fermée définie par ses n sommets V0, V1, Vn-1, la fonction ϕ est une fonction en escalier. Ceci permet de trouver les formules analytiques suivantes pour les coefficients :

$\mu_0 = -\pi - \frac{1}{L}\sum_{k=1}^{m} l_k \Delta\phi_k$

$a_n = \frac{-1}{n\pi}\sum_{k=1}^{m} \Delta\phi_k \sin\frac{2\pi n l_k}{L}$

$b_n = \frac{1}{n\pi}\sum_{k=1}^{m} \Delta\phi_k \cos\frac{2\pi n l_k}{L}$

avec Δl_n : longueur du segment $[V_{n-1}\ V_n]$, $l_k = \sum_{i=1}^{k} \Delta l_i$

et $\Delta\phi_n$: changement net de direction au point Vn

3. Théorie Bayésienne de la décision

3.1 Principes

La décision Bayésienne est une approche fondamentale de la reconnaissance des formes statistique. C'est une approche dite optimale dans la mesure où elle minimise le *risque global d'erreur* du système.
Soient P() les probabilités, et p() les fonctions de densité de probabilité.
Soient ω_i, i=1,K les classes possibles pour une observation x inconnue. On a:

$$\sum_{i=1}^{K} P(\omega_i) = 1$$

et $p(x) = \sum_{i=1}^{K} p(x \mid \omega_i)\, P(\omega_i)$

$P(\omega_i)$ est la *probabilité a priori* de la classe ω_i, i.e. la probabilité pour qu'un vecteur x quelconque appartienne à la classe ω_i. p(x) est la fonction de densité de probabilité de l'observation x et $p(x \mid \omega_i)$ est la densité de probabilité de x lorsqu'on sait que x appartient à la classe ω_i.

On rappelle que la fonction de densité de probabilité conditionnelle $p(x|\omega_i)$ permet de calculer la probabilité pour que l'observation x, dont on sait qu'elle appartient à la classe ω_i, appartienne à un sous ensemble A de l'espace de représentation R. On a :

$$P(x \in A \mid \omega_i) = \int_A p(x \mid \omega_i)dx$$

et $\int_R p(x \mid \omega_i)dx = 1$

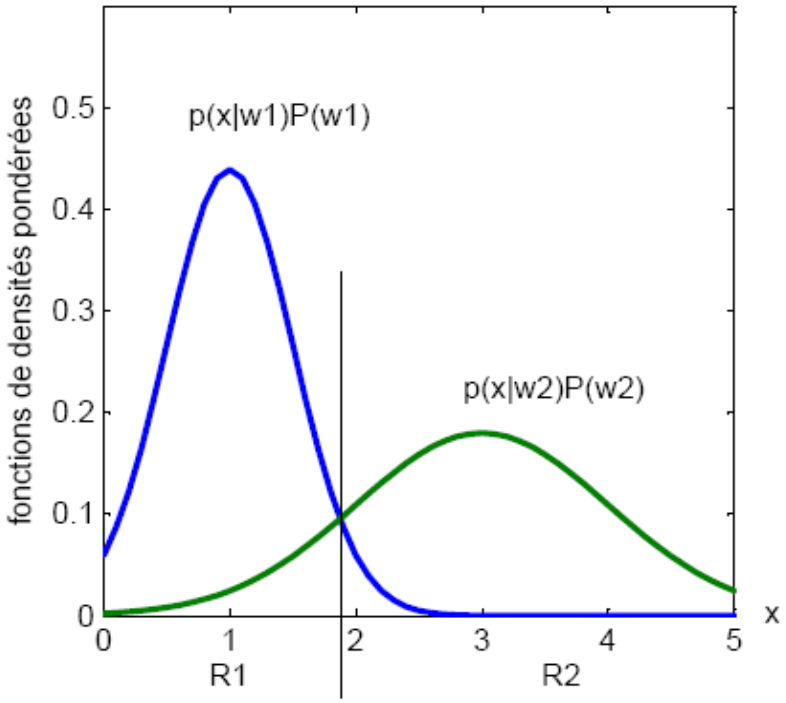


Figure 2 - cas unidimensionnel. Séparation de l'espace de représentation en 2 régions R1 et R2 correspondant respectivement aux classes ω_1 et ω_2

On suppose connues les lois de $P(\omega_i)$ et $p(x \mid \omega_i)$. Pour classer un vecteur x inconnu, on choisit la classe ω_i qui maximise la probabilité *a posteriori* $P(\omega_i \mid x)$, i.e. la probabilité d'être en présence d'une forme de la classe ω_i connaissant son vecteur x :

$$d(x)= i \text{ si } \quad \forall j \quad P(\omega_i \mid x) \geq P(\omega_j \mid x)$$

On choisit pour x la classe qui maximise la probabilité d'appartenance de x à cette classe. En appliquant la règle de Bayès, $P(\omega_i \mid x)$[1] se calcule par :

$$P(\omega_i \mid x) = \frac{p(x|\omega_i)\, P(\omega_i)}{p(x)}$$

Les lois $P(\omega_i)$ et $p(x|\omega_i)$ étant connues, on en déduit $p(x)$ et $P(\omega_i|x)$. En pratique, le terme $p(x)$ est commun à tous les rapports et n'intervient donc pas dans la comparaison. Si les lois $P(\omega_i)$ et $p(x|\omega_i)$ ne sont pas connues, on cherchera à les estimer lors d'un apprentissage supervisé (i.e. les classes des vecteurs d'apprentissage sont connues).

Les lois $P(\omega_i)$, probabilité a priori, ou probabilité d'apparition de la classe ω_i sont soit connues à l'avance, soit considérées comme égales, soit faciles à estimer sur l'ensemble d'apprentissage. Dans le cas de la classification des lettres dans un texte, les classes sont les

[1] Les notations P(i|x) et P(ωi|x) sont équivalentes à P(C=i|X=x)

lettres de l'alphabet et $P(\omega_i)$ sera la fréquence d'apparition d'une lettre donnée dans la langue considérée.

Plus difficile est l'estimation de la loi de densité de probabilité $p(x|\omega_i)$ pour chaque classe ω_i et pour chaque vecteur de représentation x. Plusieurs méthodes d'apprentissage sont possibles suivant les hypothèses plus ou moins fortes que l'on fait sur la loi $p(x|\omega_i)$:

- les méthodes *paramétriques* supposent une forme de loi connue (notamment de forme gaussienne) dont on cherchera à estimer le vecteur moyenne et la matrice de covariance sur les données d'apprentissage.

- les méthodes *non paramétriques* qui ne font pas d'hypothèses sur la forme des lois $p(x|\omega_i)$. Ce sont notamment les méthodes d'estimation par la méthode des noyaux de Parzen, l'estimateur des k plus proches voisins.

3.2 Probabilité d'erreur

Les performances d'un système de reconnaissance des formes peuvent s'évaluer en termes de probabilité d'erreur. Supposons un problème à 2 classes ω_1 et ω_2. Ces classes sont respectivement associées aux régions R_1 et R_2 de l'espace de représentation. Il y a erreur de décision dans les cas suivants :

- x est dans la région R_1 (et est donc classé dans la classe ω_1) mais appartient en fait à la classe ω_2.
- x est dans la région R_2 (et est donc classé dans la classe ω_2) mais appartient en fait à la classe ω_1.

La probabilité d'erreur s'exprime par :

$$E_b = P(x \in R_1, \omega_2) + P(x \in R_2, \omega_1)$$

En appliquant la règle de Bayès relative à la conjonction de 2 événements, on a :

$$E_b = P(x \in R_1 | \omega_2).\ P(\omega_2) + P(x \in R_2 | \omega_1).\ P(\omega_2)$$
$$= \int_{R_1} p(x|\omega_2)\ P(\omega_2)\ dx \quad + \int_{R_2} p(x|\omega_1)\ P(\omega_1)\ dx$$

Dans le cas unidimensionnel (figure 3), la probabilité d'erreur est la surface hachurée. Si on déplace le point frontière de décision x0, la probabilité d'erreur ne sera plus minimale.

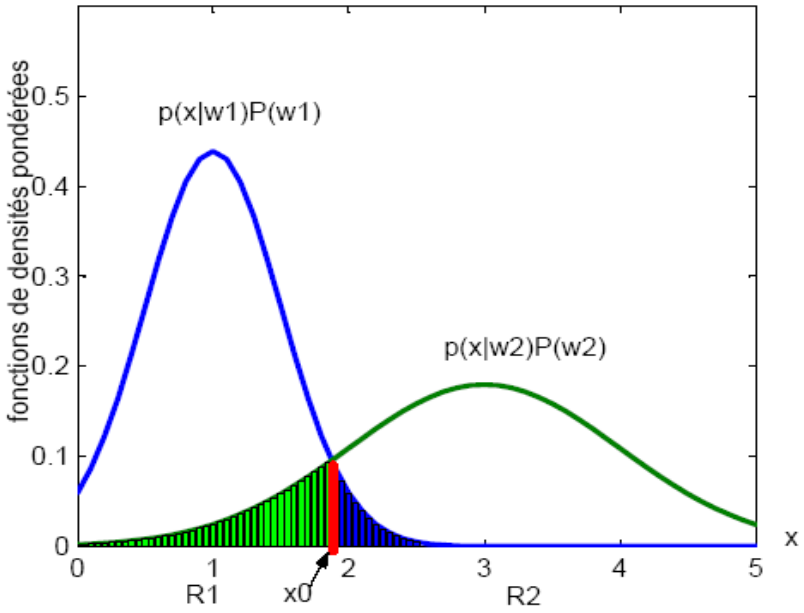


Figure 3 - la règle de décision Bayésienne donne une probabilité d'erreur minimale, égale ici à la somme des 2 surfaces colorées.

Montrons maintenant que la fonction de décision Bayésienne minimise le risque global d'erreur du système de reconnaissance.

Soit d(x) la règle de décision Bayésienne :

$\forall x \in R^d$, $d(x) = j$ si $P(\omega_j \mid x) \geq P(\omega_i \mid x)$ pour j=1,...,K

Soient un vecteur x à classer et d0(x) la classe à laquelle appartient effectivement x. La probabilité d'erreur associée à la décision d(x) prise pour le vecteur x est :

$E_b(x) = \text{Prob}(d(x) \neq d0(x))$

Or la probabilité de faire le bon choix pour x par le règle d est la probabilité que x appartienne effectivement à la classe d(x) choisie par la règle de décision Bayésienne. Ce qui s'écrit :

$\text{Prob}(d(x)=d0(x)) = P(d(x) \mid x)$

d'où :

$E_b(x) = \text{Prob}(d(x) \neq d0(x)) = 1 - P(d(x) \mid x)$

La probabilité globale d'erreur Eb pour le système est l'espérance mathématique de $E_b(x)$:

$E_b = \int_R E_b(x)p(x)dx$

$= \int_R [1 - P(d(x) \mid x)]p(x)dx$

Soit une autre fonction de décision d1 et E_{d1} la probabilité globale d'erreur du système par la règle d1.

On a :

$$Eb - E_{d1} = \int [P(d(x) \mid x) - P(d_1(x) \mid x)]\, p(x)\, dx$$

or la règle de décision Bayésienne impose que :

$$P(d(x) \mid x) \geq P(d_1(x) \mid x)$$

d'où le résultat cherché :

$$E_{d1} \geq Eb$$

Le risque global d'erreur du système est minimum par la règle de décision Bayésienne.

3.3 Décision Bayésienne avec coûts et avec rejet

On veut modifier la décision Bayésienne pour modéliser le fait que certaines décisions coûtent plus cher que d'autres. Par exemple, déclencher une fausse alarme peut être moins grave que la non détection d'un incident. On introduit une matrice de coûts définie par le terme générique $\lambda(\omega_i \mid \omega_j)$[2] qui est le coût de classer un objet de classe ω_j dans la classe ω_i. Les termes diagonaux de la matrice sont nuls.

Soit d une règle de décision quelconque. Pour une représentation x donnée, le coût C (d(X)|X) associé à la décision d(x) est :

$$C(d(X)|X) = \sum_{j=1}^{K} \lambda(d(x) \mid \omega j)\, P(\omega j \mid X)$$

qui représente l'espérance du coût associé à la décision d (X) connaissant X. C (d(x)|X) est le *risque conditionnel* associé à la décision d, connaissant la réalisation X.

Le coût global de décision C_d pour la règle de décision d, appelé *risque moyen* , est l'espérance du risque conditionnel :

$$C_d = E[C(d(x)|X)] = \int_R C(d(x)|X)\; p(X)\, dX)$$

La règle d correspondant à la décision Bayésienne est celle qui minimise le risque moyen. Le risque est alors noté C* et appelé *risque de Bayes*.

---

[2] $\lambda(\omega i | \omega j)$ se note aussi λ_{ij} . Le terme $\lambda(d(x) | \omega j)$ est à interpréter par $\lambda_{d(x)j}$

Pour minimiser le risque moyen , il suffit pour chaque X de minimiser le risque conditionnel, et l'on obtient C*(X) *risque conditionnel minimum*.

Outre les coûts, on peut de plus introduire une classe de rejet, notée ω0, où sont mis les vecteurs X dont la forme est ambiguë. On introduit le coût de rejet Cr qui est égal aux termes $\lambda(\omega_0 \mid \omega j)$.

L'introduction d'une classe de rejet permet de réduire la probabilité globale d'erreur. Une fois mises dans la classe de rejet, on peut leur appliquer d'autres traitements plus fins, et donc plus chers, pour les différencier.

Nous allons illustrer ces principes sur la règle des coûts {0,1}, encore appelée règle à pénalisation symétrique. On a :

$\lambda(\omega_0 \mid \omega_j)=Cr \qquad \forall\ j=1,..., K$
$\lambda(\omega_i \mid \omega_j)=1 \quad si \quad i \neq j \quad \forall\ i, j=1,..., K$
$\lambda(\omega_i \mid \omega_i)=0 \qquad \forall\ i=1,..., K$

On prend pour un vecteur X la décision ωi si $C_i(X)=C(d(X)=i|X)$ est minimum , i.e.:

si $C_i(X) = \sum_{j=1, j\neq i}^{K} P(\omega j \mid X) = 1 - P(\omega i \mid X) \leq C_j(X) \quad \forall\ j=1,...,K$
et si $C_i(X) < C_0(X) = Cr$

d'où d(X)= i si :

$$P(\omega i \mid X) \geq P(\omega j \mid X)\ \forall\ j=1,...,K \quad et \quad P(\omega i \mid X) \geq 1 - Cr$$

On retrouve dans la première condition, la règle classique de décision Bayésienne sans rejet.

On prend la décision de rejeter (d(X)= 0) si $C_0(X) \leq C_i(X)$ pour tout i, i.e. :

$$1 - Cr \geq \max_{j=1,...,K} P(\omega j \mid X)$$

On peut montrer que, pour qu'il y ait possibilité de rejet, il est nécessaire de choisir Cr tel que :

$$0 \leq Cr \leq \frac{K-1}{K}$$

L'espace de représentation R est maintenant partitionné en régions. Les régions Ri correspondent aux observations classées dans la classe ωi, la région R0 aux formes rejetées.

$$Ri = \{ X \in R , \ \max_{j=1,...,K} P(\omega j \mid X) = P(\omega i \mid X) \geq 1 - Cr \}$$

$$R0 = \{ X \in R, \ \max_{j=1,...,K} P(\omega j \mid X) < 1 - Cr \}$$

On pose $RA = \cup_{i=1,...,K} Ri$

Quand le coût de rejet augmente, la probabilité de rejet diminue et la probabilité d'erreur augmente. En effet, soient deux coûts Cr1 et Cr2 tels que Cr1 < Cr2. On a :

$RA(Cr1) \subset RA(Cr2)$. (1)

Car soit $X \in RA(Cr1)$. On a $C(d(X)=i|X) = Ci(X) < Cr1$ d'où $C(d(X)=i|X) < Cr2$

On montre de même que $R0(Cr2) \subset R0(Cr1)$ (2)

Soit P_R, la probabilité (ou taux) de rejet :

$$P_R = \int_{R0} p(X)\, dX \quad (3)$$

D'après (2) et (3), on a bien $P_R(Cr1) > P_R(Cr2)$

Soit P_{Ei} la probabilité d'erreur de classement pour la classe ωi. C'est la probabilité d'erreur quand on classe un échantillon dans la classe ωi :

$$P_{Ei} = \int_{Ri} C^*(X)\, p(X)\, dX = \int_{Ri} e^*(X)\, p(X)\, dX$$

en effet $C^*(X)=\text{Min } Ci(X)= 1-P(d(X) \mid X) = P(X \notin d(X) \mid X) = P(\text{erreur} \mid X) = e^*(X)$.

Le risque conditionnel minimum C*(X) est égal ici à la probabilité conditionnelle d'erreur notée e*(X).

Soit P_E la probabilité (ou taux) d'erreur :

$$P_E = \sum_{i=1}^{K} P_{Ei} = \int_{RA} e^*(X)\, p(X)\, dX \quad (4)$$

D'après (1) et (4) , on a :

$$P_E\ (Cr1) < P_E\ (Cr2)$$

Réciproquement, baisser le coût de rejet, augmente la probabilité de rejet et diminue la probabilité d'erreur.

Le risque de Bayes est $C^* = \int_R C^*(X)\, p(X)\, dX = \int_{RA} e^*(X)\, p(X)\, dX + \int_{R0} Cr\, p(X)\, dX$
d'où $C^* = P_E + Cr\, P_R$

qui est la relation fondamentale reliant le risque Bayésien, et les taux d'erreur et de rejet.

3.4 Erreurs de type I et II

Supposons un problème à 2 classes dans un contexte de détection de cible :
ω1 : absence de la cible
ω2 : présence de la cible

Nous avons vu en 3.2 que la probabilité d'erreur pouvait se décomposer en 2 termes.

$E_b = P\ (d(x)=1, \omega 2) + P\ (d(x)=2, \omega 1)$

Chacun de ces termes correspond à un type d'erreur: erreur de type I ou erreur de type II. Elles sont résumées dans le tableau suivant :

	C=1	C=2
d(x)=1	décision correcte	erreur de type II
d(x)=2	erreur de type I	décision correcte

L'erreur dite de type I correspond au rejet erroné de l'hypothèse nulle (notée H0) de la théorie des tests. Dans un problème de détection, l'hypothèse nulle correspond généralement au fait qu'il y a absence de cible (ou de maladie). L'erreur de type I est ici la probabilité de fausse alarme P_{FA} et l'erreur de type II la probabilité de non détection P_{ND} . Notons que si on modifie l'hypothèse nulle en sa négation, les erreurs de type I et II seront interverties.

Soit x0, le seuil de détection de la règle de décision :
d(x)=2 si x>x0
d(x)=1 sinon

ce seuil dépend de la matrice de coûts et est déterminé par l'équation :

$$\frac{p(x0|\omega 1)}{p(x0|\omega 2)} = \frac{\lambda(\omega 1|\omega 2)P(\omega 2)}{\lambda(\omega 2|\omega 1)P(\omega 1)}$$

et on a :

$P_{FA} = \int_{x0}^{+\infty} p(x|\omega 1)P(\omega 1)dx$

$P_{ND} = \int_{-\infty}^{x0} p(x|\omega 2)P(\omega 2)dx$

$E_b = P_{ND} + P_{FA}$

Le déplacement du seuil x0 induit des modifications pour P_{ND}, P_{FA} et l'augmentation de Eb . En déplaçant la frontière x0 de la figure ci dessous vers la droite, on augmente la probabilité de non détection et on diminue la probabilité de fausse alarme, ce qui est dans ce contexte l'effet recherché. On a l'effet inverse si on déplace la frontière vers la gauche.

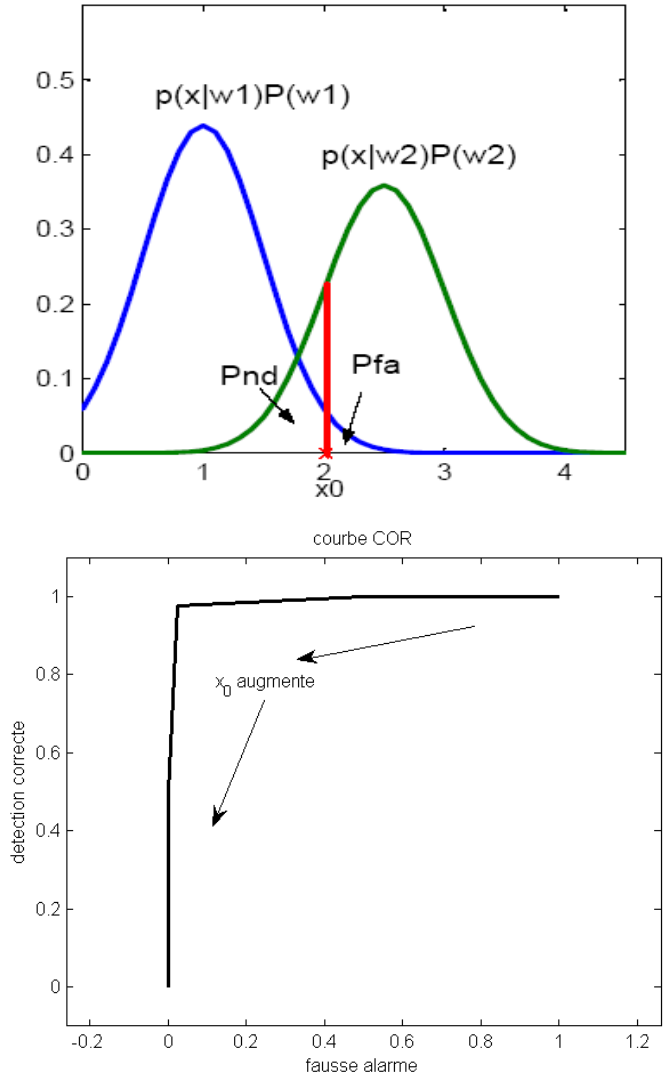


Figure 4 - seuil de décision x0 et courbe COR

Quand les probabilités a priori ne sont pas connues, on utilise les probabilités conditionnelles d'erreur : e_{FA} et e_{ND} qui vérifient : $E_b = e_{ND} P(\omega 2) + e_{FA} P(\omega 1)$

La courbe COR (caractéristique opérationnelle de détection) consiste à tracer la probabilité conditionnelle de détection correcte, $e_{D=} 1\text{-}e_{ND}$, en fonction e_{FA}, probabilité conditionnelle de fausse alarme, en faisant varier x0.

4. Estimation paramétrique

Pour appliquer la règle de décision Bayésienne, il est nécessaire de connaître notamment les lois de densité de probabilité $p(x|\omega_i)$ pour toutes les classes ω_i. Les méthodes paramétriques Bayésiennes cherchent à estimer les paramètres de la loi $p(x|\omega_i)$ supposée de forme connue, à partir de vecteurs d'apprentissage appartenant à la classe ω_i. Dans la plupart des cas, on fera l'hypothèse de distributions gaussiennes.

4.1 Cas des lois normales

Dans le cas gaussien, les fonctions de densité de probabilité conditionnelle multivariables ont une expression connue :

$$p(x|\omega) = \frac{1}{(2\pi)^{d/2}\ |\Sigma|^{1/2}} \exp\left[-\frac{1}{2} (x - M)^t \Sigma^{-1} (x - M)\right]$$

d est la dimension de l'espace de représentation. M est le vecteur moyenne de la loi que suivent les observations appartenant à la classe ω, Σ la matrice de covariance et $|\Sigma|$ son déterminant. M et Σ sont définis par :

$$M = E[X]$$

et

$$\Sigma = E[(X - M)(X - M)^t]$$

La matrice Σ est symétrique, définie positive si aucun des attributs n'a de variance nulle.

Le principe de l'estimation paramétrique de la loi $p(x|\omega)$ supposée normale, est de trouver les paramètres M et Σ constitués de $d + \frac{d(d+1)}{2}$ éléments indépendants. Soit Θ l'ensemble de ces paramètres indépendants.

On considère un ensemble d'apprentissage constitué de n *échantillons* ou *prototypes* de la classe ω. A ces échantillons est associé l'ensemble d'observations :

$$X = \{ x_1, x_2, x_3, ..., x_n\}$$

L'estimation suivant le critère du maximum de vraisemblance permet de trouver les paramètres de la loi qui rendent à l'événement observé (l'ensemble X) la probabilité la plus forte. En d'autres termes, les valeurs estimées $\hat{M}$ et $\hat{\Sigma}$ de M et Σ sont celles pour lesquelles

on a la plus grande probabilité de retrouver l'ensemble X par un tirage aléatoire de n points, les tirages étant indépendants.

On pose* $p(x|\omega) = p(x|\Theta)$. On cherche Θ tel que :

$$p(x1, x2, ...xn \mid \Theta) \text{ soit maximum.}$$

Les échantillons étant indépendants :

$$p(x1, x2, ...xn \mid \Theta) = \prod_{i=1}^{n} p(xi \mid \Theta)$$

La fonction logarithme neperien Log étant croissante, on considère la fonction $L(\Theta)$:

$$L(\Theta) = \text{Log}\,[p(x1, x2, ...xn \mid \Theta)] = \sum_{i=1}^{n} \text{Log}[p(xi \mid \Theta)]$$

Pour que $L(\Theta)$ soit maximum, il faut annuler son gradient :

$$\nabla\,[L(\Theta)] = 0$$

En dérivant par rapport à chacun des paramètres, on obtient le vecteur Θ recherché.

Puisque la fonction de densité de probabilité est une gaussienne :

$$L(\Theta) = -n\,\text{Log}\,(2\pi)^{d/2} - \frac{n}{2}\text{Log}\,|\Sigma| - \frac{1}{2}\sum_{i=1}^{n} (xi - M)^t\,\Sigma^{-1}\,(xi - M)$$

L'estimation au maximum de vraisemblance donne pour estimations de M et de Σ :

$$\hat{M} = \frac{1}{n}\sum_{i=1}^{n} xi$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} (xi - \hat{M})\,(xi - \hat{M})^t$$

Une fois la loi $p(x \mid \omega)$ déterminée sur les échantillons de la classe ω, on fait de même sur les échantillons des autres classes, afin de déterminer toutes les lois $p(x|\omega_i)$ en vue d'appliquer la règle de décision Bayésienne.

4.2 Surfaces discriminantes

Dans le cas gaussien, nous allons rechercher la forme des surfaces séparatrices entre classes.

Soit la fonction $g_i(x)$ correspondant au logarithme népérien de la fonction $p(x|\omega_i)P(\omega_i)$ utilisée lors de la phase de décision. La fonction de densité de probabilité $p(x|\omega_i)$ a M_i pour vecteur moyenne et Σ_i pour matrice de covariance:

$$g_i(x) = \text{Log}[P(\omega_i)] - \frac{1}{2}\text{Log}|\Sigma_i| - \frac{1}{2}(x - M_i)^t\Sigma_i^{-1}\,(x - M_i)$$

* on peut aussi écrire $p(x|\omega) = p(x, \Theta)$ car Θ n'est pas une variable aléatoire mais un paramètre

La fonction $d_i(x) = [(x- M_i)^t \Sigma_i^{-1} (x-M_i)]^{1/2}$ est une distance, appelée distance de Mahalanobis, entre x et le vecteur moyenne M_i de la classe ω_i. Cette distance dépend de la classe ω_i à cause du terme en Σ_i^{-1} .

Les courbes d'équidensité $d_i(x) = Cte$ sont des hyperellipsoïdes de centre M_i. Les directions des axes sont calculées à partir des valeurs propres de la matrice de covariance Σ_i.
On remarquera que le lieu des points où la fonction de densité de probabilité est constante, pour une classe ω_i donnée, sont les hyperellipsoïdes de Mahalanobis. En effet, des observations qui suivent une loi multivariable gaussienne se regroupent autour du vecteur moyenne M_i, la forme du groupement étant déterminée par la matrice de covariance (figure 5). Classer x revient à chercher la classe ω_i telle que $g_i(x)$ est maximum. La frontière entre 2 classes ω_i et ω_j est donnée par l'équation :

$$g_i(x) = g_j(x)$$

Dans le cas où les matrices Σ_i sont égales, diagonales et les classes équiprobables :

Si $P(\omega_i) = P(\omega_j)$
et $\Sigma_i = \sigma^2 I, \quad \forall i$
alors $g_i(x) = Cte - \frac{||x - Mi||^2}{2\sigma^2}$

Les courbes d'équidensité sont des cercles et la frontière de décision est l'hyperplan médiateur des points moyenne M_i et M_j (figure 5, frontière F).

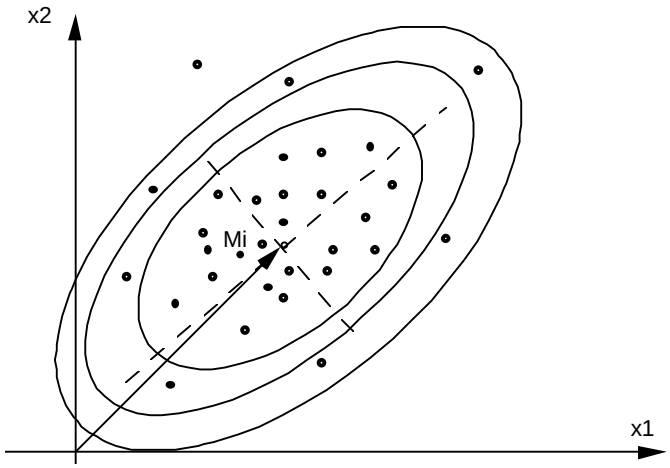


Figure 5 - groupement des observations autour du vecteur moyenne dans le cas gaussien

Si les matrices sont toujours identiques mais les classes ne sont pas équiprobables :

Si $P(\omega_i) \neq P(\omega_j)$
et $\Sigma_i = \sigma^2 I, \quad \forall i$

$$\left|\text{alors } g_i(x) = \text{Log } P(\omega_i) - \frac{1}{2\sigma^2} \left(x^t x - 2\, M_i^t\, x + M_i^t\, M_i \right)\right.$$

La frontière entre 2 classes ω_i et ω_j est régie par l'équation :

$$\frac{1}{\sigma^2} (M_i^t - M_j^t)\, x + \text{Log } \frac{P(\omega_i)}{P(\omega_j)} - \frac{1}{2\sigma^2} \left(M_i^t M_i - M_j^t\, M_j \right) = 0$$

La frontière est un hyperplan orthogonal au vecteur $(M_i - M_j)$ et qui passe par le point :

$$x_0 = \frac{1}{2} (M_i + M_j) - \frac{\sigma^2}{||M_i - M_j||^2} \text{ Log } \frac{P(\omega_i)}{P(\omega_j)} (M_i - M_j)$$

Si les classes sont équiprobables, on retrouve le fait que x_0 est au milieu de la droite M_iM_j, donc sur la médiatrice. Si les classes ne sont pas équiprobables, x_0 est sur la droite reliant les points moyenne M_i et M_j en s'éloignant du point moyenne dont la probabilité a priori de la classe est la plus forte (figure 6, frontière F').

Dans le cas où les matrices Σ_i sont toutes identiques mais quelconques :

$$\left|\begin{array}{l}\text{Si } \Sigma_i = \Sigma\ ,\ \forall\ i \\ \text{alors } g_i(x) = -\frac{1}{2}(x - M_i)^t \Sigma^{-1} (x - M_i) + \text{Log}[P(\omega_i)] + \text{Cte}\end{array}\right.$$

Si les classes sont équiprobables, on classe x en calculant la distance de Mahalonobis définie par Σ^{-1} entre x et les vecteurs moyenne de toutes les classes et en affectant x à la classe donnant la distance minimum.

La frontière de décision entre 2 classes ω_i et ω_j est régie par l'équation :

$(M_i^t - M_j^t)\Sigma^{-1}\, x + C = 0$

qui est l'équation d'un hyperplan orthogonal à $\Sigma^{-1}\ (M_i - M_j)$.

Dans le cas où les matrices Σ_i sont arbitraires , on calcule :

$$g_i(x) = \text{Log}[P(\omega_i)] - \frac{1}{2} \text{Log } |\Sigma_i| - \frac{1}{2} (x - M_i)^t \Sigma_i^{-1} (x - M_i)$$

les frontières sont des hyperquadriques de révolution : hyperplans, hypersphères, hyperellipsoïdes, hyperparaboloïdes, hyperhyperboloïdes.

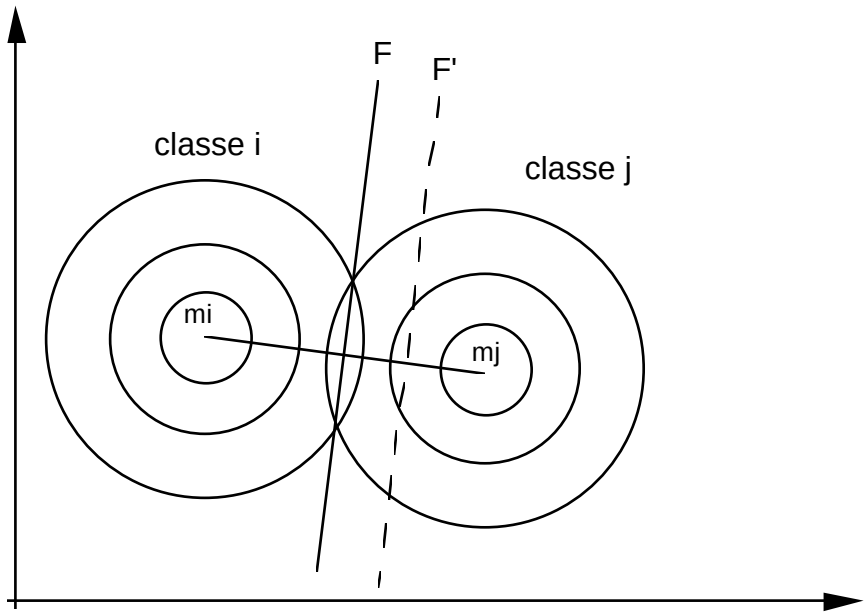


Figure 6 - frontière de décision F dans le cas où les matrices de covariance sont égales, diagonales, et les classes équiprobables. Si la classe i est plus probable que la classe j, la frontière se déplace en F'.

5. Discrimination paramétrique non Bayésienne

Dans l'approche Bayésienne, il est nécessaire de connaître les lois de probabilité (les densités de probabilité) que suivent les différentes classes, en vue de déterminer les surfaces de séparation entre les classes. L'approche paramétrique non Bayésienne est une autre approche qui permet de partitionner l'espace de représentation ,sans pour autant connaître les densités de probabilité des différentes classes. Il s'agit alors d'imposer une famille de surfaces séparatrices et de calculer les paramètres de ces surfaces (en optimisant un critère) qui s'adaptent le mieux aux vecteurs de l'ensemble d'apprentissage.
Une fois la surface séparatrice déterminée, la décision est très simple puisqu'il suffit de se déterminer par rapport aux frontières délimitées par les surfaces.

5.1 Principes de la séparation linéaire

La forme la plus simple de surface séparatrice est l'hyperplan. On suppose que l'on a K classes à discriminer et que les vecteurs de chaque classe forment un nuage convexe dans l'espace de représentation. Suivant la topologie des classes en présence, 2 cas sont possibles. Le premier consiste à séparer les classes 2 à 2 ce qui nécessite la détermination de $C_K^2 = \frac{K(K-1)}{2}$ hyperplans. Le problème de la discrimination à K classes se ramène alors à des problèmes à 2 classes. Dans le cas où chaque classe est linéairement séparable des K-1 autres, K-1 hyperplans suffisent, mais cela crée de larges zones d'indécision dans l'espace de représentation (figure 7).

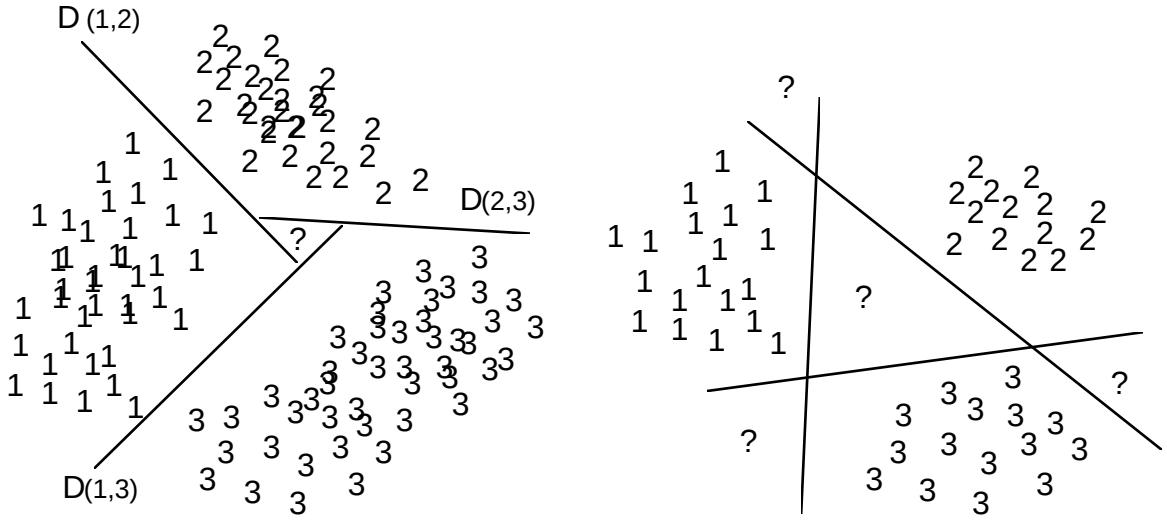


Figure 7 - problème multi-classes. Soit on sépare chaque classe 2 à 2 (à gauche), soit on sépare chaque classe avec toutes les autres (à droite).

On dispose de n vecteurs d'apprentissage dont on connait les classes a priori (apprentissage supervisé). On se place dans le cas binaire où les formes ne peuvent appartenir qu'à deux classes appelées l+ et l-.

Le principe de la séparation linéaire est de trouver l'équation de l'hyperplan H d'équation $a^t x + b = 0$, tel que :

$$\forall x \in l+ \qquad a^t x + b > 0$$
$$\forall x \in l- \qquad a^t x + b < 0$$

avec $a^t = (a1,..., ad)$, représentant le vecteur orthogonal à la direction de H.

Pour simplifier les équations on considère les vecteurs *étendus* :

$$A^t = (a1,..., ad, b) \qquad \text{et} \qquad X^t = (x1, ..., xd, 1)$$

où A est le vecteur a dont la dernière composante vaut b et X le vecteur X dont la dernière composante vaut 1.

Les équations deviennent :

$$\forall X \in L+ \qquad A^t X > 0$$
$$\forall X \in L- \qquad A^t X < 0$$

La règle d'apprentissage du perceptron est un algorithme simple qui permet de trouver un hyperplan séparateur, donc un vecteur A solution, en un nombre fini d'itérations, à partir des données d'apprentissage. Cependant les classes doivent être au départ linéairement

séparables, c'est-à-dire il faut qu'il existe un hyperplan H séparant, sans erreur de classification, les données d'apprentissage (figure 8).

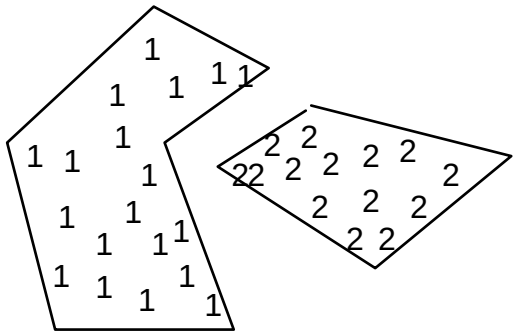


Figure 8 - classes non linéairement séparables

5.2 Règle du Perceptron

Rosenblatt a proposé en 1957 avec l'algorithme du perceptron un modèle très simplifié du fonctionnement du cerveau humain, notamment de ses capacités d'apprentissage et de perception. Avant de présenter l'interprétation connexionniste de cet algorithme, les bases théoriques sont les suivantes. Il s'agit de minimiser une fonction de coût J(A) qui sera minimum ou nulle pour les vecteurs A solutions. Cette minimisation s'établit par une méthode de gradient, c'est à dire qu'on modifie A dans le sens opposé à son gradient $\nabla J(A)$, jusqu'à trouver un minimum de la fonction J(A).

Soit Y(A) l'ensemble des vecteurs X mal classés par l'hyperplan défini par A. On peut prendre comme fonction de coût :

$$J(A) = \sum_{X \in Y(A)} - c(X) A^t X$$

avec $c(X) = 1$ si $X \in L+$, $c(X) = -1$ si $X \in L-$

Cette fonction est nulle si tous les vecteurs de l'ensemble d'apprentissage sont bien classés.

Le gradient est alors $\nabla J(A) = - \sum_{X \in Y(A)} c(X) X$

L'algorithme est itératif peut être le suivant :

A l'étape k+1:
$A(k+1) = A(k) - \alpha_k \nabla J(A(k))$

α_k est un paramètre qui, s'il est trop faible, ralentit la convergence. S'il est trop fort il peut entraîner un dépassement de l'optimum.

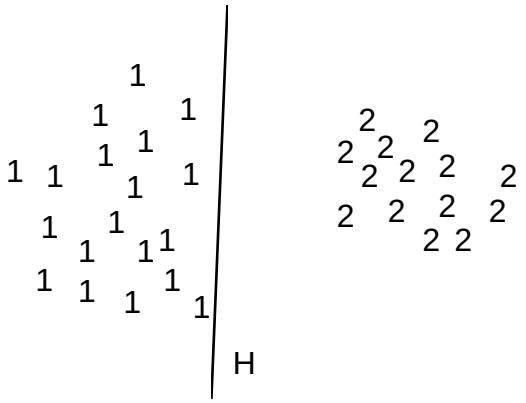


Figure 9 - l'hyperplan trouvé par la règle du perceptron a tendance à passer trop près des données

Au lieu de modifier A en fonction de tous les vecteurs mal classés, on peut modifier A dès que l'on rencontre un vecteur mal classé. C'est la version séquentielle de l'algorithme d'apprentissage, la plus connue :

Règle du perceptron

(1) choisir le vecteur A quelconque
(2) tant qu'il existe un vecteur étendu X mal classé :

- si X appartient à la classe L+ (et est classé par erreur dans la classe L-), faire
A = A + X

- si X appartient à la classe L- (et est classé par erreur dans la classe L+), faire
A = A - X

(3) Arrêter quand tous les exemples présentés sont correctement classés

L'hyperplan H trouvé, s'il existe, n'est pas unique et dépend du vecteur initial A choisi. Dans le cas où les classes ne sont pas linéairement séparables, l'algorithme ne converge pas et oscille indéfiniment. Il n'est d'ailleurs pas capable de détecter la non séparabilité linéaire des exemples. De plus, l'hyperplan trouvé n'est souvent pas optimum car tangent aux classes, ce qui entraîne des problèmes de généralisation pour les données bruitées (figure 8). Dans le cas séparable, l'algorithme converge en un nombre fini d'itérations.

5.3 Interprétation Connexionniste

Le perceptron peut être interprété comme réseau connexionniste à une couche, la couche de sortie. La *couche d'entrée* est constituée des caractéristiques de la forme, i.e. le vecteur X. La *couche de sortie* est constituée d'une cellule dont l'activité est une combinaison linéaire des entrées : on dit alors que l'architecture du réseau est de type *propagation en avant* (figure 10).

$$xs = \sum_{i=1}^{d} ai\ xi\ +\ b$$

Les ai et b sont à la fois les poids des connexions entre les unités et les paramètres de l'hyperplan H cherché.
La sortie de la cellule d+2 (cellule de sortie) n'est pas sont activité mais une fonction de l'activité, la fonction Signe par exemple:

sortie = f(xs), f fonction Signe

L'apprentissage du réseau consiste à déterminer les valeurs des poids ai et b à partir des exemples présentés en entrée de façon à ce que la sortie désirée corresponde à la sortie observée.

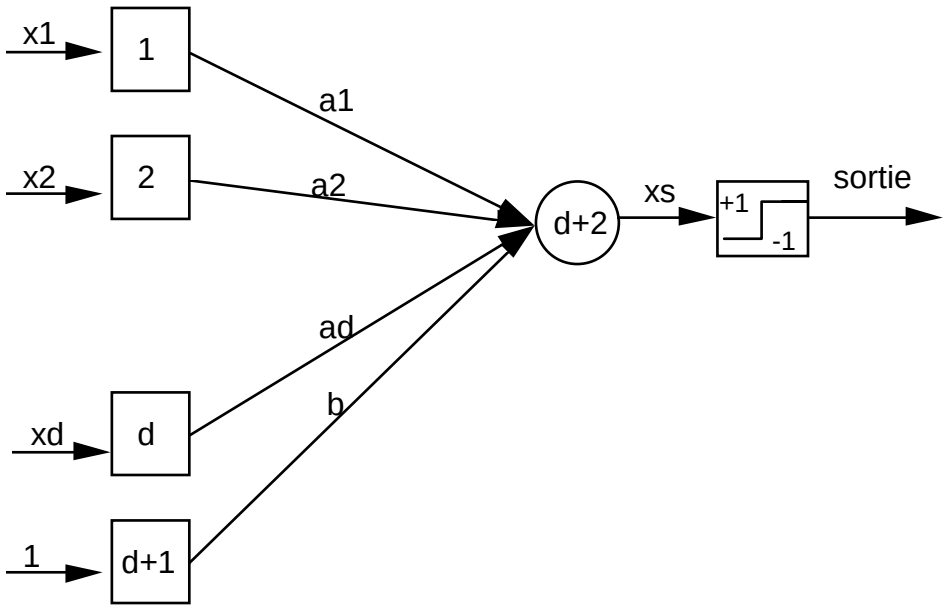


Figure 10 - réseau à une couche de type perceptron

5.4 Minimisation aux moindres carrés

L'algorithme suivant, dit de Widrow-Hoff, est une variante de l'algorithme du perceptron qui s'affranchit des problèmes d'oscillation dans le cas de classes non linéairement séparables et qui est plus robuste (moins tangent aux données). Cet algorithme recherche l'hyperplan H, donc le vecteur A, qui minimise la fonction de coût J(A) aux moindres carrés :

$$J(A) = \sum_{X \in E} (A^t X - b_X)^2$$

J(A) est ici calculée sur tous les vecteurs de l'ensemble d'apprentissage et non plus sur les seuls mal classés. On ne cherche plus, contrairement au perceptron, à annuler J(A) mais à la minimiser. D'autre part, on introduit la marge bx, dépendante de la classe de X, entre les points d'apprentissage et l'hyperplan pour éviter qu'il passe trop près des données (on veut une réponse suffisante $A^t X > b_X$).

On peut prendre pour b_x la fonction c(X) précédente, i.e. c(X)=1si X$\in$ L^+si X, c(X)=-1 sinon.

L'algorithme séquentiel et itératif qui modifie A dans le sens opposé à $\nabla J(A)$, jusqu'à trouver un minimum de la fonction J(A) est :

Règle de Widrow-Hoff

1- A0 quelconque ; k= 0

2- $A(k+1) = A(k) + r_k (b[k] - A(k)^t X[k]) X[k]$

3- si $J(A) < \varepsilon$ alors FIN; sinon k = k+1 et aller en 2

On prend classiquement r_k = r1/k. Et X[k] signifie le vecteur k modulo n, n étant le cardinal de l'ensemble d'apprentissage.

Le lecteur intéressé se reportera à (Milgram, 93) pour la justification théorique de cette règle.

6. Discrimination Bayésienne non paramétrique

La règle de décision Bayésienne suppose la connaissance des lois de probabilité pour chaque classe p(x | ωi) et P(ωi). Pour connaître la loi de densité de probabilité p(x | ωi). Contrairement à l'approche paramétrique (cf section 3), l'approche non paramétrique ne fait pas d'hypothèse sur la nature des distributions : on cherche à les estimer directement à partir des données d'apprentissage.

6.1 Principes

Soit un ensemble d'apprentissage E comprenant n vecteurs d'apprentissage $\in R^d$ appartenant tous à la classe ωi. On cherche à estimer la loi p(X |ωi).à partir de ces n vecteurs échantillons.

Soit A une région de R^d. Soit p, la probabilité qu'un vecteur X appartienne à la région A.
$p = P(X \in A) = \int_A p(X)\, dX$

Soit k_A la variable aléatoire représentant le nombre d'échantillons appartenant à une région A donnée. C'est une loi binômiale de paramètres p et n. On a donc :

prob ($k_A = k$) = $C_n^k p^k (1-p)^{n-k}$
car c'est la probabilité pour que k vecteurs échantillons, parmi n, appartiennent à A, et n-k échantillons n'y appartiennent pas.

La loi faible des grands nombres permet de relier la probabilité de l'événement $X \in A$, de probabilité p, à sa fréquence d'apparition k_A/n obtenue quand l'ensemble d'apprentissage contient une infinité d'échantillons. Cette relation est :

$$p \text{ -----> } k_A/n$$
$$n \text{--> } +\infty$$

On peut prendre comme estimateur non biaisé de p :

$$p_e = k_A/n$$

Soit maintenant un point x0 quelconque de l'espace de représentation et supposons que la fonction de densité de probabilité p(x) ne présente pas de discontinuité autour du point x0. Prenons V_n le volume d'une boule centrée sur x0. Si on fait tendre ce volume vers zéro, on a par définition de la densité de probabilité au point x0 :

$$\frac{\text{Prob}\,(x \in V_n)}{V_n} \text{ ------> } p(x0)$$
$$V_n \text{ ----> } 0$$

Si on dispose d'un nombre infini d'échantillons, on peut estimer p(x0) par :

$$p_e(x0) = \frac{k_n}{nV_n}$$

kn étant le nombre d'échantillons dans le volume V_n

Les 3 conditions de convergence vers p(x0) sont donc :

- $\lim_{n \to +\infty} V_n = 0$
- $\lim_{n \to +\infty} k_n = +\infty$
- $\lim_{n \to +\infty} \frac{k_n}{n} = 0$

Autrement dit, les boules V_n diminuent de volume quand n--> +∞ , suffisamment lentement pour le nombre d'échantillons dans V_n augmente, mais ce nombre d'échantillons augmente moins vite que n. La suite des boules V_n doit aussi évoluer sans ruptures.

Deux approches sont possibles pour l'estimation de p(x0). Soit on se fixe la suite des régions V_n et on compte le nombre de vecteurs échantillons appartenant à la classe ω_i dans V_n. C'est la méthode des fenêtres de Parzen. Soit on se fixe kn et on fait croître V_n jusqu'à obtenir k_n échantillons . C'est l'estimateur des k_n plus proches voisins.

6.2 Fenêtres de Parzen

Dans cette méthode, on fixe les volumes V_n . On peux prendre pour V_n l'hypercube de côté h_n:

$$V_n = h_n^d$$

Soit x0 un point de l'ensemble de représentation. Comme nous l'avons vu précédemment, on peux estimer p(x0) par :

$$p_e(x0) = \frac{k_n}{nh_n^d}$$

On estime $p(x0 \mid \omega_i)$ (on rappelle que l'on nomme ici indifféremment p(x0) et $p(x0 \mid \omega_i)$ puisque les échantillons appartiennent exclusivement à la classe ω_i) par la sommation de fonctions élémentaires :

$$p_e(x0) = \frac{1}{nh_n^d} \sum_{k=1}^{n} \phi(\frac{x0 - xk}{h_n})$$

Les vecteurs de l'ensemble d'apprentissage contribuent de façon pondérée à l'estimation de la densité de probabilité en fonction de leur distance au point x0. Les fonctions ϕ sont des fonctions, appellées noyaux ou fenêtres, centrées en 0 et qui vérifient les conditions :

$$\forall x \quad \phi(x) \geq 0$$

$$\int \phi(x)\, dx = 1$$

On choisit ϕ gaussienne, carré, ou triangle (figure 12). On remarquera que si ϕ est la fonction carré, $\sum_{k=1}^{n} \phi(\frac{x0 - xk}{h_n})$ compte le nombre de vecteurs échantillons à une distance limitée à hn/2 autour d'un point x0 quelconque (figure 11). Si ϕ est une gaussienne ou la fonction triangle, les points d'apprentissage de E seront pris en compte avec un poids d'autant plus important qu'ils sont proches de x0.

Le point délicat de la méthode est de régler le paramètre h_n caractérisant le volume V_n. En pratique, on règle h_n pour qu'un volume de côté h_n centré autour d'un point d'apprentissage englobe aussi d'autres points d'apprentissage, ni trop ni trop peu. Car si h_n est trop petit, la fonction de densité p(x0) présentera des variabilités non significatives et les zones de faible densité auront une estimation nulle, tandis que si h_n est trop grand, on risque d'ignorer des variations fines de la fonction de densité (mauvaise résolution).
On rappelle qu'en vertu des propriétés des volumes Vn,(cf section précédente), hn doit vérifier:

$\lim h_n = 0$
$n \to +\infty$
$\lim n h_n^d = +\infty$
$n \to +\infty$

On pourra prendre : $h_n = n^{-\alpha/d}$ avec $0 < \alpha < 1$

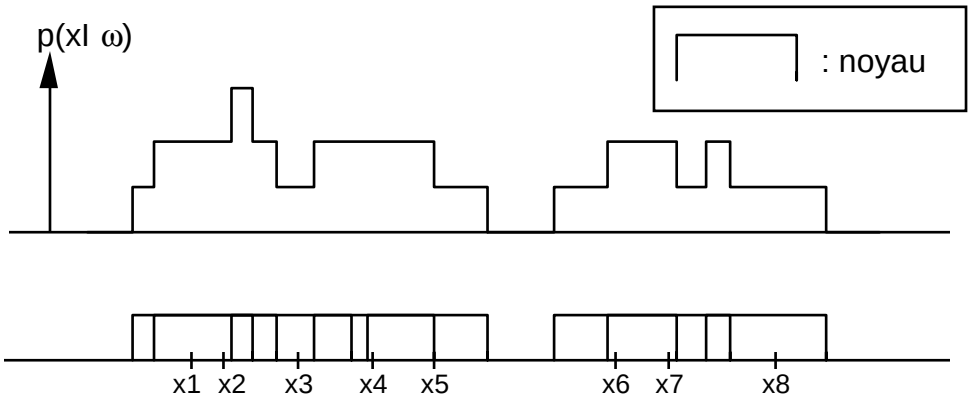


Figure 11 - estimation de la densité de probabilité (en haut) à partir de 8 points échantillons et en utilisant un noyau ϕ carré.

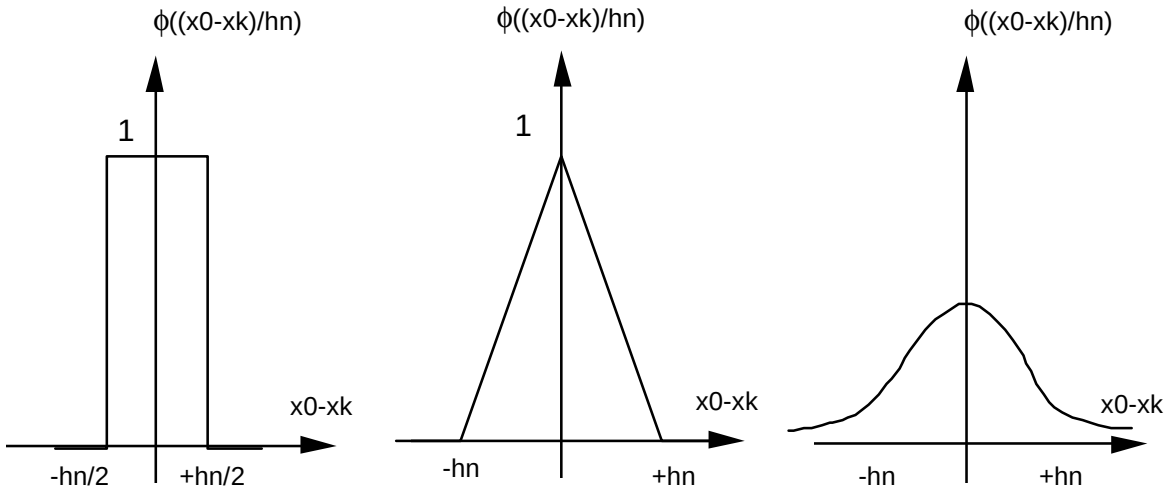


Figure 12: allure dans le cas monovariable de 3 fonctions noyaux couramment utilisées : de gauche à droite, la fonction carré, la fonction triangle et un noyau gaussien. En abscisse, la distance d'un point x0 quelconque au point xk de l'ensemble d'apprentissage.

6.3 Estimateur des kn plus proches voisin

Dans cette approche, pour estimer la densité de probabilité en chaque point x0, on ajuste la taille des volumes Vn en fonction de la densité de points échantillons (vecteurs d'apprentisssage) autour du point x0. On fixe kn et on fait croître le volume Vn centrée en x0 jusqu'à ce qu'il contienne exactement kn échantillons.

Vn étant trouvé, l'estimateur de la densité de probabilité s'exprime par :

$$p_e(x0) = \frac{k_n}{nV_n}$$

Les conditions nécessaires de convergence de l'estimateur sont :

$\lim_{n \to +\infty} k_n = +\infty$

$\lim_{n \to +\infty} \frac{k_n}{n} = 0$

Un choix possible de k_n est $k_n = k1\sqrt{n}$

Le choix du paramètre k1 est cependant délicat et est déterminé par la nature des données. L'avantage de cet estimateur est que la fonction de densité p(x0) n'est jamais nulle.

7. Méthode des k plus proches voisins

La méthode des k plus proches voisin s'inscrit dans le cadre des méthodes non Bayésiennes et non paramétriques. En effet, on n'applique pas de règle de décision de type Bayésien, on ne fait aucune hypothèse sur les lois de densité de probabilité, et on ne cherche pas à estimer ces lois. La classification s'opère directement en comparant le point à classer avec les points préalablement étiquetés de l'ensemble d'apprentissage.

7.1 Principe

Le principe de la règle du 1-ppv est très simple. Soit $x \in R^d$ et soit x_i le plus proche voisin de x, appartenant à l'ensemble d'apprentissage E.

$$\forall x_j \in E \qquad d(x,x_i) \leq d(x,x_j)$$

La règle de décision consiste à attribuer le vecteur x à la classe du vecteur x_i. Le choix de la distance d est important. On peut montrer que les frontières de décision inhérentes à cette règle, sont, si d est la distance euclidienne, des frontières polygonales. Elles sont formées de la réunion des hyperplans médiateurs entre toutes les paires de points échantillons (de l'ensemble d'apprentissage).

Une variante de cette règle consiste à examiner, non plus un seul voisin, mais les k plus proches voisins du point x : les vecteurs x1, x2, .. xk. La règle de décision consiste à attribuer le vecteur x à la classe ωi majoritairement représentée parmi les k voisins.

Une variante de la règle des k ppv consiste à introduire un seuil de rejet m. On n'attribue x à la classe majoritairement représentée ωi que si au moins m voisins parmi k appartiennent à cette classe. Sinon on rejette le point x. Ceci permet de fonder sa décision sur un nombre suffisant de points voisins, surtout dans le cas où on a plus de 2 classes en présence.

On peut déjà remarquer que le choix du paramètre k n'est pas neutre. Si une classe a un effectif plus petit que les autres, on choisira k pas trop grand pour avoir la possibilité

d'affecter des points à cette classe. A l'inverse, si les classes sont mêlées, on prendra k assez grand pour se prémunir du bruit. D'autre part, nous verrons que pour garantir de bonnes performances au classifieur, il faut suffisamment de points d'apprentissage.

Une autre remarque concerne la complexité de la méthode. Si n est le nombre de points d'apprentissage, appartenant aux différentes classes en présence ω_1, ω_2 ω_M, il faut calculer les n distances entre le point x à classer et les points d'apprentissage pour sélectionner les k voisins les plus proches. D'où une complexité en O (card E).

7.2 Probabilité d'erreur

Il est intéressant de comparer la probailité d'erreur avec la règle du 1-ppv avec la probabilité d'erreur de la décision Bayésienne. Dans le cas où le nombre de vecteurs d'apprentissage n, tend vers l'infini, la probabilité d'erreur par la règle du 1-ppv (E_{ppv}) converge vers le risque optimal de Bayès (E_b). On peut montrer que :

$$E_b \leq E_{ppv} \leq 2\, E_b \quad (\text{quand } n \dashrightarrow +\infty)$$

Cette propriété n'est valable qu'à la limite et non sur un nombre limité n d'exemples. Le comportement du classifieur pour un ensemble fini d'échantillons (la situation en pratique) n'est plus prévisible.

Nous allons démontrer les inégalités ci dessus dans un problème à M=2 classes. Considérons x un vecteur à classer et x'_n son plus proche voisin ($x'_n \in E$)
La probabilité de mal classer x est :

$$Ep(x, x'_n) = \text{Prob}\ (\omega(x) \neq \omega(x'_n)) \mid x, x'_n)$$

En sommant sur toutes les classes ω_i possibles pour x'_n et en considérant l'indépendance des points d'apprentissage vis à vis du point x :

$$Ep(x, x'_n) = \sum_{i=1}^{M} P(\omega(x)=\omega_i \mid x)) \bullet P(\omega(x'_n) \neq \omega_i \mid x'_n)$$

$$= \sum_{i=1}^{M} P(\omega_i \mid x)) \bullet [\, 1- P(\omega_i \mid x'_n)]$$

Par intégration sur l'ensemble de représentation R, on obtient l'espérance mathématique de $Ep(x, x'_n)$, i.e. la probabilité d'erreur totale :

$$E_{ppv} = \int_R \left(\sum_{i=1}^{M} P(\omega_i \mid x) \bullet [\, 1- P(\omega_i \mid x'_n)] \right) p(x)\, dx$$

Quand le nombre d'échantillons n --> $+\infty$, alors x'_n converge en probabilité vers x, et $P(\omega_i \mid x'_n)$ converge en probabilité vers $P(\omega_i \mid x)$.

d'où $E_{ppv} = \int_R \left(\sum_{i=1}^{M} P(\omega_i \mid x) \bullet [1 - P(\omega_i \mid x)] \right) p(x)\, dx \quad (n \dashrightarrow +\infty)$

Dans le cas où on a 2 classes en présence. Soit x un vecteur à classer et soit $d_1(x)$ la règle de décision Bayésienne, qui donne par exemple pour le vecteur x considéré $d_1(x) = \omega_1$. Soit d_2 la règle de décision qui attribue à x la classe opposée à d_1, dans ce cas de figure on a donc $d_2(x) = \omega_2$.

On peut maintenant exprimer la probabilité d'erreur E_{ppv} avec l'hypothèse d'un nombre infini d'échantillons d'apprentissage comme :

$E_{ppv} = \int_R P(\omega_1 \mid x)\,[1 - P(\omega_1 \mid x)]p(x)dx + \int_R P(\omega_2 \mid x)\,[1 - P(\omega_2 \mid x)]p(x)dx$

or $P(d_1(x) \mid x) = 1 - P(d_2(x) \mid x)$

d'où $E_{ppv} = 2\int_R P(d_2(x) \mid x)[1 - P(d_2(x) \mid x)]p(x)dx \leq 2\int_R P(d_2(x) \mid x)p(x)dx$

Nous avons vu section 2.2 que la probabilité d'erreur pour la règle de décision Bayésienne est :

$Eb = \int_R (1 - P(d_1(x) \mid x))\, p(x)\, dx$

d'où $Eb = \int_R P(d_2(x) \mid x)\, p(x)\, dx$

On obtient donc :

$$\lim_{n \dashrightarrow +\infty} E_{ppv} \leq 2Eb$$

Comme d'autre part, on ne peut pas descendre en dessous du risque de Bayès, on a :

$$Eb \leq \lim_{n \dashrightarrow +\infty} E_{ppv}$$

On obtient donc le résultat cherché :

$$Eb \leq E_{ppv} \leq 2Eb \qquad (n \dashrightarrow +\infty)$$

Ce résultat indique que l'on fait au plus 2 fois plus d'erreur par la règle du 1-ppv que par la règle de décision Bayésienne. Cette propriété n'est vraie qu'asymptotiquement.

7.3 Variantes rapides des k-ppv

L'inconvénient majeur de la méthode des k-ppv est son temps de décision important qui dépend directement du nombre n de prototypes dans l'ensemble d'apprentissage. A chaque classement d'un point x, n distances doivent être calculées. Les variantes dites accélérées de la méthode des k-ppv effectuent un prétraitement sur l'ensemble d'apprentissage E. Ce prétraitement demande des calculs supplémentaires, mais il n'est fait qu'une fois et le temps de décision sera réduit. Deux approches sont utilisées. Soit on met en évidence lors du prétraitement, une structure sur l'ensemble d'apprentissage pour définir une zone restreinte de l'espace où seront recherchés les k-voisins. Soit on diminue la taille de l'ensemble d'apprentissage en éliminant des prototypes.
Les gains, établis en fonction du nombre de distances calculées, dépendent généralement du cardinal de E, de la dimension de l'espace et de k. D'autre part, il est intéressant de comparer les performances de ces méthodes, en termes de complexité algorithmique.

7.3.1 Méthode de Vidal

L'algorithme AESA (Approximation and Elimination Search Algorithm) permet de trouver le ppv d'un point test en un temps de recherche relativement constant par rapport à la taille de l'ensemble d'apprentissage. Il s'appuie sur les propriétés de la métrique utilisée pour calculer les distances, et sur un prétraitement de l'ensemble d'apprentissage consistant à calculer toutes les distances point à point entre prototypes. Ce traitement est coûteux (en $O(n^2)$) mais s'effectue hors-ligne, avant la phase de décision.

Considérons le cas k=1 (1-ppv). L'algorithme s'effectue en deux phases. Une phase de sélection d'un prototype ancrage, une phase d'élimination de prototypes à examiner. Un prototype ancrage est un point s pour lesquel la distance d(x,s) au point test x est effectivement calculée. Soit U l'ensemble des prototypes ancrage, soit P l'ensemble de tous les prototypes, soit E l'ensemble des prototypes éliminés.

Soit un ancrage s sélectionné et n le ppv courant de x parmi les prototypes ancrage. Il est inutile d'examiner les prototypes p tels que :

règles d'élimination
$d(p,s) \geq d(x,s) + d(x,n)$ ou (1)
$d(p,s) \leq d(x,s) - d(x,n)$

Ces règles utilisent les distances d(p,s) calculées lors du prétraitement.

En effet les prototypes à l'intérieur de la sphère de rayon d(x,s)-d(x,n) ou à l'extérieur de la sphère de rayon d(x,n) + d(x,s) ne peuvent concourir pour être ppv de X.
Par exemple :
$d(x,p2) > d(x,s) - d(p2,s) > d(x,s) - (d(x,s) - d(x,n)) = d(x,n)$ et
$d(x,p1) > d(p1,s) - d(x,s) > d(x,s) + d(x,n) - d(x,s) = d(x,n)$
en utilisant les inégalités triangulaires propres à toute métrique d et les relations (1).

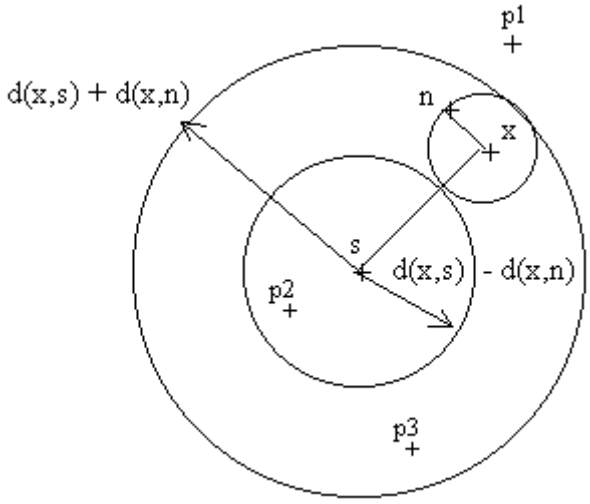


Figure 13 - à partir du prototype ancrage s, du point test X, et du plus proche voisin courant n, les prototypes p1 et p2 sont éliminés.

On élimine d'autant plus de prototypes que l'ancrage s est proche du point test x. Pour sélectionner un tel ancrage, on applique la règle suivante :

règle de sélection :
s ∈ P -{ E } est sélectionné si :
$\sum_{u \in U} |d(s,u) - d(x,u)|$ est minimum

Le prototype s est choisi proche de l'intersection des hypersphères de centre u et de rayon d(x,u).

On peut maintenant établir *l'algorithme AESA* :

début :
    U = { Ø } , E = { Ø }
    sélection : U = { s } (au hasard)
    n = s
    calcul de d(x,s)
    élimination : E = { p | p ∈ P-{s} et d(p,s) ≥ 2 d(x,n) }
tant que E≠ P faire :
    sélection : d'un nouveau prototype s par règle de sélection
    calcul de d(x,s)
    U = U ∪ { s }
    E = E ∪ { s }
    n = ppv des éléments de U
    élimination : E = { p | p ∈ P-E et d(p,s) ≥ d(x,s) + d(x,n) ou
                                                d(p,s) ≤ d(x,s) - d(x,n) }
fin tant que

le ppv est le ppv courant n dans U

Au prix d'un prétraitement important, l'algorithme AESA est une bonne technique d'accélération. En dimension 2, on calcule en moyenne pour chaque point test moins de 4 distances, et ce pour une taille de l'ensemble d'apprentissage allant de 100 à 1000 points.

7.3.2 Méthode de Kittler

Cette méthode consiste à chercher un sous ensemble de l'ensemble d'apprentissage où rechercher les k-ppv. Les voisins trouvés sont des voisins approchés mais constituent une bonne approximation des voisins réels. Une distance plus rapide que la distance euclidienne, la distance de Manhattan d_c, est utilisée pour le calcul entre un point x et les points d'apprentissage. La connaissance de ces distances permet de définir une zone restreinte où seront calculées les distances euclidiennes.

$$d_c(x,y) = \sum_{k=1}^{d} | xk - yk |$$

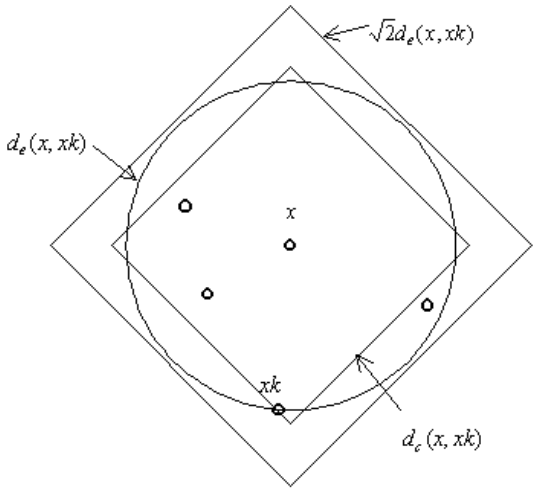


Figure 14 - Exemple en dimension 2. A partir du kième voisin x_k du point x au sens de la distance d_c, on détermine la boule, au sens de dc, de rayon $\sqrt{2}\, d_e(x, x_k)$ où seront recherchés les voisins au sens de d_e.

Soit d_e la distance euclidienne, on a la relation :

$d_e(x,y) \leq d_c(x,y) \leq \sqrt{d}\, d_e(x,y)$

Cela implique qu'une boule centrée en x de rayon $\sqrt{d}$ .r, suivant la distance de Manhattan, contient nécessairement la boule centrée en x de rayon r suivant la distance euclidienne.

L'algorithme proposé par Kittler, est le suivant :

1) Soit x le point dont on cherche les k-ppv au sens de d_e
$\forall$ xi $\in$ E, calculer les $d_c(x, xi)$
2) chercher le point xk , le kième voisin de x au sens de la distance d_c
3) calculer $d_e(x, xk)$
4) déterminer l'ensemble restreint Y où on recherche les k-ppv de x au sens de d_e par :

$$Y = \{ xi \in E, dc(x, xi) \leq \sqrt{d}\ d_e(x,xk)\}$$

5) déterminer les k-ppv au sens de d_e sur Y

7.3.3 Méthode de Friedman

Dans sa version de base, le prétraitement consiste tout d'abord à sélectionner la coordonnée (la mesure) des vecteurs de E qui présente la variance maximale. Puis on trie les vecteurs d'apprentissage suivant les valeurs croissantes de cette coordonnée. Durant la phase de test, l'ensemble des vecteurs d'apprentissage sur lesquels les distances sont calculées, sera restreint. En effet, quand la distance en projection devient supérieure à la distance euclidienne entre X et le kième voisin courant, il est inutile de continuer puisque les distances en projection sont toujours inférieures ou égales aux distances en dimension d.

L'algorithme de Friedman est le suivant :

1) Soit le vecteur inconnu X(a1, a2, ..., ad) à classer

2) on examine les vecteurs de E suivant les valeurs croissantes de leurs distances en projection à X.

3) soit x_i^m la ième coordonnée du vecteur x^m de E

4) si $| a_i - x_i^m | > D_k$ on s'arrête, sinon on retourne en 2) pour prendre le vecteur de E suivant

D_k est la distance (*en dimension d*) de X au kième ppv des points déjà examinés.

La liste des k-ppv trouvés lors de l'arret sont les k-ppv vrais.

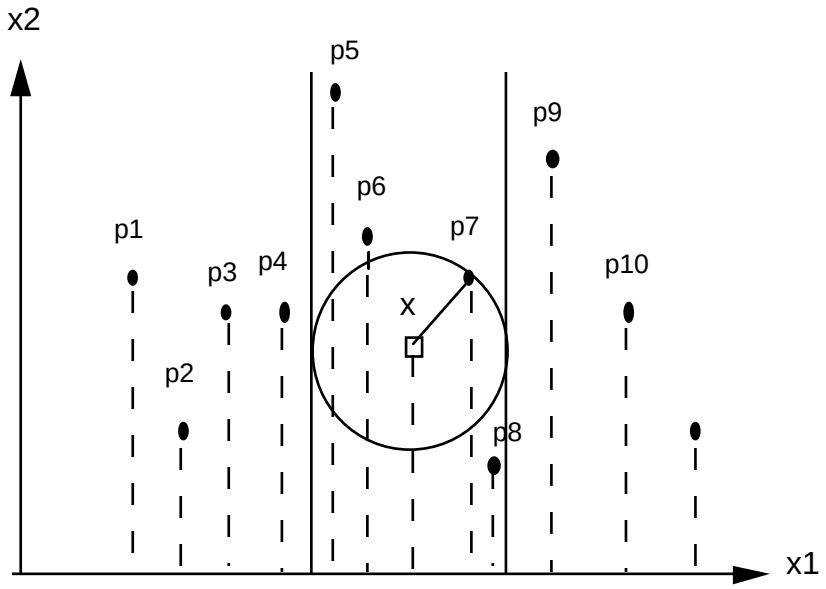


Figure 15 - algorithme de Friedman en dimension 2 avec k=1. Les prototypes sont examinés dans l'ordre de leur projection croissante au point test X, ici p6, p7, p5, p8, p4. Le plus proche voisin est le prototype p7.

La figure 14 est un exemple du déroulement de l'algorithme pour d=2et k= 1 . Les prototypes ont été triés suivant la coordonnée x1.

- on examine p6 . On calcule D_1 = de (X, p6) , de est la distance euclidienne.

Comme $| X^i_6 - a^i | < D_1$, on continue.

- on examine p7 . On a encore :

$| X^i_7 - a^i | < D_1$, donc on continue

mais on met à jour D_1 car d_e (X, p7) < d_e (X, p6) d'où $D_1 = d_e$ (X, p7)

- on examine de même p5 puis p8 , D_1 reste inchangé.

- on examine p4 :

Ici on a $| X^i_4 - a^i | > D_1$ donc on s'arrête et le plus proche voisin est p7.

On peut montrer qu'en dimension deux, le nombre de distances à calculer est en O ($\sqrt{N}$), ce qui représente une bonne économie. Cependant l'efficacité de cet algorithme diminue rapidement avec la dimensionnalité

7.3.4 Méthode de condensation

Cette méthode, due à Hart, consiste à supprimer des prototypes de l'ensemble d'apprentissage pour sélectionner un ensemble de prototypes plus consistant . L'effet de la condensation est de réduire fortement l'ensemble d'apprentissage de E en un ensemble Ec en conservant les points considérés comme les plus pertinents, i.e. ceux proches des frontières entre classes.

La suppression des points se fait par la règle du ppv. On élimine de E les points dont le plus proche voisin dans Ec appartient à la même classe.

L'algorithme de condensation utilise 3 ensembles, l'ensemble d'apprentissage E, l'ensemble d'apprentissage condensé Ec et un ensemble de points rejetés, appelé Poubelle.

Algorithme de condensation

1) i = 1
   Ec = {x1}
   Poubelle = {ø}
2) i = i + 1
3) prendre le prototype xi de E
   si classe (ppv_{Ec} (xi)) = classe (xi) alors Poubelle =Poubelle ∪ {xi}
   sinon Ec = Ec ∪ {xi}
4) aller en 2) jusqu'à ce que i = card (E)
5) repasser la boucle sur Poubelle jusqu'à stabilité

L'ensemble E_c est dit sous-ensemble consistant de E car les points de E ne faisant pas partie de E_c sont classés sans erreur par la règle du ppv sur l'ensemble E_c considéré comme ensemble d'apprentissage. L'ensemble E_c contenant peu de points par rapport à E, la décision consiste à appliquer la règle du 1-ppv sur Ec.

7.3.5 Algorithme d'édition

Le principe de ce prétraitement dû à Wilson consiste à supprimer les prototypes dont l'étiquette n'est pas fiable. A partir de l'ensemble initial E, on construit l'ensemble épuré E_e. L'effet de l'édition est de supprimer les points appartenant à des classes proches, près des frontières.
L'algorithme d'édition opére en deux phases à partir de l'ensemble E : recherche de la classe par la règle des k-ppv puis suppression des points pour construire E_e.

Algorithme d'édition

1) $E_e = E$
2) Pour tout $x \in E$, on applique la règle de décision des k-ppv : $d_{k\text{-ppv}}$
   $d_{k\text{-ppv}}(x \in \omega i) = \omega j$
3) Pour tout x de E, si $\omega j \neq \omega i$ alors $E_e = E_e - \{ x \}$

Comme précédemment, l'ensemble E_e est très réduit par rapport à E et la règle de décision consiste à appliquer la règle du 1-ppv sur l'ensemble E_e.

Cette méthode amène plusieurs remarques. Si on augmente k, alors plus de points seront supprimés. D'autre part, la méthode est sensible à la topologie des classes car les classes dont les nuages sont proches vont subir plus d'épuration que celles dont les nuages sont plus éloignés.

Si l'ensemble E est grand au départ, les performances sont améliorées si on procède à une suite d'éditions (épurations) successives. Si E est vraiment très grand, on peut combiner une édition avec une condensation.

7.3.6 Conclusion

La méthode des plus proches voisins ne fait pas d'hypothèses sur les lois de probabilité sous jacentes et nécessite en contrepartie un grand nombre de points d'apprentissage. De ce fait, elle coûte cher en temps de calcul puisqu'il faut calculer cardE distances. Les variantes proposées pour réduire le temps de décision par un prétraitement sur l'ensemble d'apprentissage sont moins efficaces quand la dimension de l'espace augmente.

8. Evaluation/amélioration des performances

Différentes mesures permettent de caractériser les performances d'un système de reconnaissance des formes : probabilités d'erreur/de bonne classification, erreurs de type I/de type II, TFR/TFA (taux de faux rejet, fausse acceptation),...Ces probabilités ou taux ont été calculés sur un ensemble de test donné et seraient un peu différents pour un autre ensemble de test. Une façon d'aborder ce problème est de moyenner les taux sur plusieurs ensembles de tests (cf. 8.2). Une autre façon de montrer la variabilité statistique de ces taux est de préciser leurs limites par un intervalle de confiance.

Pour améliorer les performances de reconnaissance pour une tâche donnée, et si on dispose de plusieurs classifieurs complémentaires, on peut les combiner par différents mécanismes (cf. 8.3). D'autre part, l'apprentissage lui même peut être amélioré par des techniques de *boosting* et *bagging*. Ce domaine est actuellement en plein développement (cf. brique MASTA).

8.1 Intervalles de confiance

Un classifieur est évalué sur un ensemble de test donné. Soit r le taux de reconnaissance (bonne classification) trouvé. Ce taux de reconnaissance r est une estimation du taux de reconnaissance réel (mais inconnu) p.

Soit X_i, où X_i est une variable aléatoire associée à l'échantillon test i : X_i=1 si l'échantillon est bien classé, X_i=0 sinon. Chaque X_i suit donc une loi de Bernoulli de paramètre p et $\sum_{i=1}^{N} X_i$ suit une loi binomiale B(N,p).

On peut alors définir R, un estimateur (v.a.) de p :

$$R = \frac{\sum_{i=1}^{N} X_i}{N}$$

Les Xi étant des variables iid (indépendantes et identiquement distribuées), N étant supposé grand et d'après le théorème central limite, R suit une loi gaussienne $N(p, \frac{p(1-p)}{N})$ de moyenne p et d'écart type $\sigma = \sqrt{\frac{p(1-p)}{N}}$

L'intervalle de probabilité pour R, i.e. la probabilité que R se trouve entre les 2 valeurs limites $[p - u_{\alpha/2}\sigma \quad p + u_{\alpha/2}\sigma]$, est :

$$P\left(p - u_{\alpha/2}\sigma < R < p + u_{\alpha/2}\sigma\right) = 1 - \alpha$$

1-α est le niveau de confiance, généralement égal à 95% ou 99% , α est le risque. Les termes $u_{\alpha/2}$ sont issus de tables (fonctions de répartition) de la loi normale centrée-réduite.
Pour un niveau de risque α= 5 % (confiance à 95 %), $u_{\alpha/2}$=1.96
Pour un niveau de risque α= 1 % (confiance à 99 %), $u_{\alpha/2}$=2.57

L'intervalle de confiance, qui encadre les valeurs de p, se calcule à partir de l'estimation r trouvée et par la formule approchée suivante :

$$r_{\min} < p < r_{\max} \quad \text{avec} \quad r_{\min} = r - u_{\alpha/2}\sqrt{\frac{r(1-r)}{N}} \;;\; r_{\max} = r + u_{\alpha/2}\sqrt{\frac{r(1-r)}{N}}$$

Cet intervalle dépend donc de r, du nombre d'échantillons testés et du niveau de confiance recherché. Le résultat est donné sous la forme : r [r_{min} r_{max}]

8.2 Validation croisée

Les performances s'évaluent généralement sur un ensemble de test distinct de l'ensemble d'apprentissage. Quand le nombre d'échantillons récoltés est insuffisant pour faire cette séparation, on peut utiliser le mécanisme de la validation croisée (*cross-validation*). Le principe est le suivant : on découpe l'ensemble des échantillons en F sous-ensembles de même taille. On construit F classifieurs : chaque classifieur est entraîné sur F-1 sous-ensembles et est testé sur le sous-ensemble restant. Les F taux de reconnaissance sont ensuite moyennés : on obtient alors le taux de reconnaissance moyen (*cross-validation rate*).
Un cas particulier (*leave-one-out*) correspond à F=N, où N est le nombre total d'échantillons.

La validation croisée permet aussi de rechercher de bons paramètres pour un classifieur (réseau neuronal, machines à vecteurs de support,...). Les paramètres devant être réglés sur l'ensemble d'apprentissage et non sur l'ensemble de test, ceux-ci peuvent être sélectionnés par validation croisée : les paramètres donnant le meilleur taux moyen sont conservés. On peut ensuite re-entraîner un classifieur avec les paramètres pertinents, et sur toute la base d'apprentissage.

8.3 Combinaison de classifieurs

Combiner plusieurs classifieurs permet d'améliorer les performances de reconnaissance sur une tâche donnée. Il y a plusieurs manières de construire les différents classifieurs : on extrait les mêmes caractéristiques sur les formes mais la fonction de décision est différente, ou on extrait des caractéristiques différentes (pour des systèmes combinant différentes modalités issues de différents capteurs : voix-image du visage par exemple).

Les sorties de classifieurs sont de nature variée. Pour une forme donnée, la sortie du classifieur peut être : l'étiquette de la classe estimée, un ensemble de scores indiquant pour chaque classe la probabilité (ou une mesure de confiance) que la forme appartienne aux différentes classes. Ces sorties sont généralement normalisées avant combinaison, classifieur par classifieur.

Quelques exemples de normalisation (s représente le score à normaliser, s_{norm} le score normalisé):

$$s_{norm} = \frac{s}{s_{\max}}$$

$$s_{norm} = \frac{s - s_{\min}}{s_{\max} - s_{\min}} \qquad \text{(Min-max)}$$

$$s_{norm} = \frac{s - \mu_s}{\sigma_s} \qquad \text{(Z-norm)}$$

où μ_s et σ_s sont la moyenne et la variance de la distribution des scores issus du classifieur dont on normalise les scores, et s_{max} et s_{min}, les valeurs maximales et minimales des valeurs des scores.
On peut citer les règles simples de combinaison (règle produit, règle somme, …) qui supposent que les C classifieurs sont indépendants. La règle produit s'écrit par exemple :

$$d(x) = \text{ i si : } i = Arg \max_j Q_j(x) \quad \text{avec} \quad Q_j(x) = \prod_{k=1}^{c} s_{k,j}(x)$$

Les $\{s_{k,j}(x)\}_{1 \le j \le K}$ sont les scores normalisés du classifieur k pour l'échantillon x, et pour chacune des K classes.

Une autre façon de combiner est de construire un nouveau classifieur « superviseur ». La fonction de décision du classifieur superviseur peut être construite de manière classique (fonction de discrimination linéaire,…) à partir des sorties des classifieurs individuels.

Bibliographie

[1] P. Blake, C.L. , Merz, C.J. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[2] R. Duda, P. Hart, D. Stork, Pattern classification, 2nd edition, Wiley, 2001.

[3] B. Dubuisson, Diagnostic et reconnaissance des formes, Hermès, 1990

[4] Alain Faure, Perception et reconnaissance des formes, édiTests, 1985 (épuisé).

[5] J.H Friedman, J.L Bentley, R.A. Finkel, An algorithm for finding nearest neighbors, IEEE Trans. on Computers, Vol. 24, pp. 1000-1006, Juillet 1975.

[6] G. Gaillat, Méthodes statistiques de reconnaissance des formes, Polycopié de l'ENSTA, 1983.

[7] J. Kittler, M. Hatef, R. P. Duin, J. Matas, On combining classifiers, IEEE PAMI, Vol. 20, no 3, 1998, pp. 226-239.

[8] M. Milgram, Reconnaissance des formes, méthodes numériques et connexionnistes, Armand Colin, 1993.

[9] J-G Postaire, De l'image à la décision, Dunod Informatique, 1987.

[10] G. Saporta, Probabilités, analyse des données et statistique, Editions Technip, 1990.

[11] E. Vidal Ruiz, An algorithm for finding nearest neighbours in (approximately) constant average time, Pattern Recognition Letters 4, pp. 145-157, 1986.

[12] C. Zahn, R. Roskies, Fourier descriptors for plane closed curves, IEEE Trans. on Computers, Vol 21, no 3, mars 1972, pp 269-281.

CHAPITRE 3 ANALYSE DES DONNEES

Bernard Burtschy

1. Introduction

La reconnaissance statistique des formes s'appuie sur des données. Classiquement, les données sont représentées par des mesures qu'on a accumulées sur des objets. Ces « mesures » peuvent être de natures diverses. Par exemple, prenons les objets qui sont formés par tout ce qui vole dans le ciel et pouvant être détecté par un radar (avions, hélicoptères, missiles, etc.). Les mesures qui permettent de décrire ces objets et qu'on appelle variables, peuvent être l'altitude, la vitesse, la température et la surface. Cela tombe bien (si on peut dire), car on peut calculer, pour toutes ces variables, une moyenne et d'autres indicateurs statistiques. Ce n'est pas toujours le cas, car les variables pourraient être plus complexes comme la forme de l'objet, en référence à un catalogue. Il est, dans ce cas, hors de question de calculer une moyenne mais on n'éjecte pas pour autant la variable qui peut être particulièrement pertinente.

Indépendamment de la reconnaissance des formes, il existe une discipline qui a pour objet de décrire et d'analyser des données. C'est la statistique et ce n'est pas un hasard si on parle de reconnaissance statistique des formes tout comme on parle de mécanique statistique ou d'économie statistique.

La statistique traditionnelle s'intéresse aux variables provenant d'une population ou d'un échantillon comme, par exemple, les questions d'une enquête à la condition qu'elles ne soient pas trop nombreuses et si possibles continues. Elle en tire des graphiques (histogrammes), des résumés (moyenne, écart-type), des moyens de comparaison entre diverses sous populations (tests) et de mesure d'atypisme d'un individu (normalité). Elle a aussi établi tout un catalogue de lois, comme la loi gaussienne, que de nombreuses mesures ont le bon goût de suivre, grâce au théorème de la limite centrale.

Décrire un phénomène par une ou quelques mesures, c'est généralement trop peu pour une étude même sommaire. Très vite, on se trouve confronté à une dizaine de variables en même temps, voire beaucoup plus; les moyens classiques (histogrammes, tests) sont impuissants devant la complexité exponentielle des interactions possibles. Les méthodes doivent pouvoir maîtriser la réalité **multidimensionnelle** que l'on retrouve, par exemple, au travers des questions nombreuses d'une enquête mais aussi dès qu'on cherche à décrire précisément des objets. Dans l'exemple cité précédemment, si on possède dix variables sur chaque objet volant, on peut dire que chaque objet est décrit dans un espace de dimension 10. Cet espace est de dimension trop élevée pour le commun des mortels: on ne peut pas le visualiser et sa compréhension est problématique. La dimensionnalité élevée représente le premier blocage.

Les variables appréhendées par la statistique traditionnelle sont souvent des mesures que l'on peut résumer facilement par une moyenne. Les phénomènes plus qualitatifs sont, soit franchement éliminés, soit rendus faussement quantitatifs à l'aide d'échelles (j'aime les yaourts pas du tout, un peu, beaucoup, passionnément). L'analyse de **variables de toutes natures**, en

particulier qualitatives ou dont les modalités sont simplement ordonnées, a permis d'étendre très largement le champ d'application de ces méthodes. Hélas ces variables sont difficilement manipulables par les techniques classiques de la statistique. Deuxième blocage sur la nature des variables.

Enfin, l'absence de moyens de traitements informatiques a conduit à privilégier, dans le passé, des modèles simples, aisément calculables. La fameuse loi normale (la courbe en cloche ou loi gaussienne) a ainsi pour immense avantage de permettre, quand elle s'applique, le calcul analytique, c'est-à-dire sur le papier, de nombreuses solutions sans avoir recours à des calculs compliqués. Quand on n'avait pas le choix, on était peu regardant sur les conditions d'application. A notre époque, ce n'est plus un avantage suffisant.

L'apparition des ordinateurs, voire maintenant de simples micro-ordinateurs a permis de **s'affranchir des hypothèses trop encombrantes**. Ceci ne signifie nullement qu'en analyse de données on ne fasse pas d'hypothèses a priori, car on fait toujours des hypothèses. Mais on cherche à en faire le moins possible et, surtout, d'en être pleinement conscient. En tout état de cause, il ne s'agit plus d'adhésion aveugle à des lois vaguement vérifiées. Ce souci de vouloir contrôler a tout moment la validité des hypothèses de travail représente la troisième caractéristique de l'analyse des données.

L'analyse des données est une extension récente de la statistique classique qui vise à contourner les trois obstacles énumérés ci-dessus. L'analyse des données peut donc se résumer (tableau 1) en l'analyse de phénomènes multidimensionnels, quels que soient leurs moyens de description (qualitatifs, quantitatifs), avec des hypothèses clairement explicitées et, si possible, peu nombreuses. Les résultats sont souvent présentés sous forme graphique, en tout état de cause le plus synthétiquement possible.

	Statistique traditionnelle	**Analyse des données**
Nombre d'objets	Illimité	Illimité, mais partitionnable
Nombre de variables	Peu (de l'ordre de 5)	Très grand (quelques centaines ou plus)
Types de variables	Surtout quantitatives	Quelconques
Graphiques	Histogrammes	Géométriques arbres partitions
Hypothèses de travail	Lois statistiques (en général gaussien)	Faibles et toujours contrôlées
Calculs	A la main	Informatique

Tab. 1: Statistique traditionnelle et analyse des données

Ces méthodes sont relativement récentes (années 70). Pourquoi ne pas avoir utilisé les méthodes de l'analyse des données plus tôt tant elles paraissent naturelles? Tout simplement parce que tant que les calculs s'effectuaient « à la main », elles étaient trop complexes. Les méthodes de la statistique traditionnelle avaient pour vertu essentielle de minimiser les calculs en se raccrochant à des lois pré-établies, en espérant que tout se passe bien. C'était souvent le cas, mais de toute façon, il n'y avait pas d'autres moyens.

Maintenant que l'informatique est largement diffusée, le comportement est radicalement différent. On peut tester les hypothèses de travail, s'attaquer à des données plus complexes (multidimensionnelles) sans avoir à faire des hypothèses techniques audacieuses. Seul l'inertie des comportements laisse encore une part trop importante à la statistique traditionnelle qui a, bien entendu, son champ d'utilisation mais dans un cadre relativement étroit. Les conséquences de l'impact de la puissance des ordinateurs sur la statistique en général sont d'ailleurs loin d'être terminées. Pour preuve, avec les méthodes récentes, on peut maintenant tout simuler sans avoir à faire d'hypothèse de lois statistiques. Bref, l'analyse des données est la statistique de notre temps.

Bon nombre des méthodes d'analyse des données ont déjà largement innervé la reconnaissance statistique des formes, au point qu'on n'en connaît plus l'origine, et c'est tant mieux. Cela prouve leurs généralités. L'objet de ce chapitre est de replacer l'ensemble des méthodes statistiques de la reconnaissance des formes dans leur contexte naturel afin d'avoir une vue d'ensemble d'un traitement de données.

2. Définitions et notations

Quelques définitions sont couramment utilisées en analyse des données. Issues de l'histoire de la discipline, elles sont suffisamment générales pour qu'un consensus se soit dégagé. Il est à noter que certaines de ces définitions divergent avec celles communément utilisées en reconnaissance des formes. On lèvera l'ambiguïté chaque fois que possible.

On appelle **individus, ou encore observations, objets ou entités**, les éléments de même nature que l'on veut comparer. Il peut s'agir effectivement d'individus, mais aussi d'entreprises, de noms de marques de cigarettes, de pays ou de minéraux. La définition des objets doit être faite avec précision: une entreprise et un établissement sont deux notions différentes, il en est de même d'un caractère manuscrit ou d'une ligne. Dans le cas où l'étude n'est pas exhaustive, on se préoccupera des propriétés de l'échantillonnage. En reconnaissance des formes, on utilise volontiers le terme d'objet.

On appelle **variables, ou encore critères, attributs ou questions** (dans les enquêtes), les éléments caractéristiques servant de critères à la comparaison. Ces variables peuvent être des éléments quantitatifs (vitesse, altitude) ou qualitatifs (type d'objet). Idéalement, il faut choisir, le plus exhaustivement possible, les variables les plus pertinentes. Le critère de pertinence est moins important que l'exhaustivité car l'analyse des données permet de faire le tri entre de nombreuses variables. Encore convient-il que les variables caractérisant le phénomène ont bien été retenues. L'analyse permet d'éliminer sans peine des variables redondantes, montrer éventuellement qu'on manque d'information pour résoudre le problème, mais elle ne peut pas suppléer une donnée défaillante.

Les **données** sont les valeurs numériques prises par les différentes variables pour les divers individus. Un **tableau de données** (Tab. 2) comporte, assez classiquement les individus en ligne et les variables en colonne, bien que cette disposition soit assez formelle. On notera n le nombre d'individus, p le nombre de variables et $X_{i\,j}$ l'élément du tableau qui se trouve à l'intersection de la ligne i et de la colonne j.

	V_1		V_j	V_p
Objet 1				
Objet 2				
Objet i			$X_{i\,j}$	
Objet n				

Tab. 2: Un tableau-type de l'analyse des données

3. Les quatre types de variables.

Les variables retenues peuvent posséder des caractéristiques diverses. Sur certaines variables, il est possible de calculer une moyenne comme, par exemple, l'altitude moyenne pour la variable altitude; d'autres ont une numérotation purement arbitraire, comme la catégorie type d'objet, où il est hors de propos de se livrer à des opérations arithmétiques sur les codes. On distingue quatre types de variables: les variables nominales, ordinales, intervalles, métriques.

3.1. Les variables nominales.

Les variables nominales sont décrites par des nombres qui ont un rôle d'identificateur des éventualités possibles. Ces éventualités possibles sont souvent nommées **modalités**. Le type d'avion a, par exemple, quatre modalités:

1 Airbus

2 Boeing

3 Mac Douglas

4 Tupolev

Les modalités reflètent l'ensemble des réponses possible. L'absence de modalité se règle très aisément par la création d'une modalité "Non renseigné".

La numérotation des modalités est purement arbitraire, on aurait pu prendre n'importe quelle autre. Calculer une moyenne sur ces modalités n'a aucun sens: additionner un

Airbus et un Mac Douglas ne donne pas un Boeing. Les seules opérations permises sont des comptages comme, par exemple, le comptage du nombre d'objets pour chaque modalité.

3.2. Les variables ordinales.

Les variables ordinales sont équivalentes aux variables nominales, mais les modalités sont ordonnées. On pourrait très bien avoir l'altitude selon les paliers suivants:

1 Moins de 3000 pieds

2 De 3000 à 5000 pieds

3 De 5000 à 10 000 pieds

4 Plus de 10 000 pieds

La numérotation n'est plus purement arbitraire, puisqu'un avion qui appartient à la modalité 2 a une altitude supérieure à un avion de la tranche 1. Mais la tranche 3 n'est pas trois fois supérieure à la tranche 1 et on ne peut calculer de moyenne ou, tout au moins, que très approximativement.

Les opérations permises sont les divers comptages comme pour les variables nominales, le calcul de la médiane ainsi que diverses statistiques sur les ordres.

3.3. Les variables intervalles.

L'échelle d'intervalle est un cas spécial d'une variable ordinale où toutes les tranches sont égales. La température mesurée en degré Celsius en est un bon exemple car la différence entre 20° et 21° est la même qu'entre 21° et 22°. Cependant, le rapport n'a pas de sens car une température de 20° n'est pas deux fois plus grande qu'une température de 10° ce qui est la conséquence de la nature arbitraire du zéro; si l'on avait mesuré la température en degré Kelvin où il existe un zéro absolu, ce rapport aurait un sens.

Les échelles d'intervalle sont beaucoup utilisées dans les études et plus généralement chaque fois que l'on cherche à quantifier un phénomène qualitatif. Si les rapports de variables intervalles n'ont pas de sens, la plupart des calculs statistiques sont autorisés: moyenne, variance, corrélation. Un exemple d'échelle d'intervalle est fourni par la question suivante:

> Notez de 1 (tout à fait d'accord) à 5 (tout à fait en désaccord) votre réponse à la question « L'objet volant non identifié est un avion ».

3.4. Les variables métriques.

Les variables métriques, appelées quelquefois échelles de rapport, sont caractérisées par la possession d'un zéro naturel qui indique l'absence de phénomène. De nombreuses variables manipulées dans la vie pratique sont dans ce cas: vitesse, poids, prix, longueur, quantité. Toutes les opérations sont permises: addition, moyenne, variance, division, etc.

Ces variables sont, grâce à ces caractéristiques, idéales pour l'analyse des données; malheureusement, dans la pratique, l'existence des non-réponses ou de variables non renseignées (panne d'un instrument de mesure, brouillage) interdisent leur utilisation sans

hypothèses supplémentaires. Les caractéristiques essentielles des quatre types de variables sont résumées dans le tableau 3.

Variables	Caractéristiques	Opérations	Exemples
Nominales	Codes arbitraires Modalités	Comptages	Type d'avion
Ordinales	Codes ordonnés	Statistiques sur les rangs	Préférences Altitude en paliers
Intervalles	Zéro arbitraire	Moyenne Ecart-type	Température (°C) Calendrier
Métriques	Zéro naturel	Toutes opérations	Vitesse Altitude Dimensions

Tab. 3: Les quatre types de variables

3.5. Variables qualitatives et quantitatives.

Il est d'usage d'appeler **variables qualitatives**, les variables nominales et ordinales et **variables quantitatives**, les variables intervalles et les variables métriques (Tab. 4). Les variables peuvent, sous certaines conditions (linéarisation de l'échelle par exemple), être considérées comme quantitatives. C'est une pratique courante, mais elle est souvent abusive.

Qualitatives		Quantitatives	
Nominales	Ordinales	Intervalles	Métriques

Tab. 4: Variables quantitatives et qualitatives

4. Quelques types de tableaux de données.

Dans le terme Analyse des Données figure le terme « données ». Les données représentent la matière de base. Elle est d'autant plus privilégiée que l'analyse de données se veut coller aux données pour retrouver leur saveur originelle en réaction à nombre de méthodes qui, à force d'hypothèses et de pseudo-théorisation mathématique, en sont arrivées à oublier, qu'à la base, on traitait un problème concret.

Les concepts d'individus, de variables ont déjà été évoqués; les individus (en ligne), les variables (en colonne) forment un tableau de données rectangulaire appelé matrice. Selon

la nature des variables, on obtiendra divers types de tableaux qui ont des propriétés différentes et sont donc passibles de traitements de données différents.

4.1. Les tableaux quantitatifs et de notes.

Ce tableau est entièrement constitué de variables quantitatives. Soit, par exemple, un groupe de vingt objets décrits par leur taille et leur poids. La matrice a 20 lignes et 2 colonnes. Rien n'empêche de représenter les vingt individus sur un graphique à deux dimensions: on dit qu'on a une représentation dans l'espace R^2 (Fig. 1). En ajoutant une troisième variable, telle que l'age, la représentation se trouve dans l'espace R^3. Ajouterait-on une quatrième caractéristique, la représentation (dans R^4) serait plus problématique à dessiner pour les pauvres moyens dont nous disposons. Nous entrons dans le domaine de l'analyse des données.

	Taille	Poids
Ind 1	1,88	87
Ind 2	1,65	66
Ind 3	1,78	77
Ind 4	1,71	74
Ind 5	1,75	77
Ind 6	1,69	65
Ind 7	1,66	67
Ind 8	1,75	74
Ind 9	1,72	70
Ind 10	1,83	84

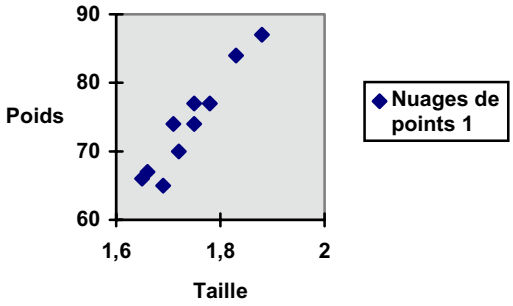


Fig. 1: Un tableau de mesures et sa représentation dans R^2

Notons qu'il est tout à fait possible d'additionner les chiffres d'une colonne et d'en calculer la moyenne: cette moyenne correspond, pour la première colonne, à la taille moyenne des individus. Il n'est cependant pas réaliste d'additionner les caractéristiques d'un même individu, c'est-à-dire les colonnes, car on ne peut ajouter, sans précaution, la taille et le poids. Ce tableau apparaît donc comme dissymétrique car l'opération moyenne est possible dans un sens du tableau et non dans l'autre. Néanmoins, cette double opération a un sens dans le cas de variables quantitatives homogènes telles qu'un **tableau de notes** (Tab. 5) qui représentent un cas particulier des tableaux de mesures; les notes sont généralement pondérées.

Expert	**1**	**2**	**3**	**4**	**5**
Objet 1	17	14	19	16	16
Objet 2	7	11	5	9	8
.	.	.	.	.	.
.	.	.	.	.	.
Objet n	12	9	9	11	13
Moyenne	*11*	*13*	*9*	*10*	*11*
Ecart-type	*3,9*	*2,6*	*2,9*	*2,6*	*2,3*

Tab. 5: un tableau de notes

Si on examine attentivement le tableau 5, on constate que les experts ont des notations moyennes différentes. Faut-il en tenir compte ou bien les remet-on à moyenne égale? Il y a plus pernicieux encore. L'expert 1 et l'expert 5 notent avec la même moyenne. Mais ils n'ont pas le même écart-type. L'écart-type étant un indicateur de dispersion, l'expert 1 a des différences de notes bien plus importantes que l'expert 5. Si on agrège les notes des experts, comme cela se fait classiquement dans l'enseignement pour des correcteurs, les notes de l'expert 1 ont plus d'importance que ceux de l'expert 5, même si a priori, ils ont le même poids. Là encore, faut-il corriger ou ne pas corriger, *that is the question*.

Ce qui est sûr, selon que l'on tiendra compte ou non des disparités des moyennes et des écarts-types, on obtiendra des analyses différentes.

4.2. Les tableaux de préférences et de rangs.

On demande à un individu de classer par ordre de préférence des objets, par exemple quatre parfums (Tab. 6). Si chaque individu fournit un classement complet sans ex aequo, chaque ligne à une somme constante puisque chaque classement possible est représenté; dans le cas de cinq parfums, elle est de 10 (1+2+3+4); en cas d'ex aequo, on prend un codage de telle sorte que cette propriété soit préservée.

On considère souvent que les réponses fournies sous forme de classement sont plus cohérentes que celles données sous forme de notes, même si, en toute rigueur, l'information est moins précise. D'autre part, c'est un moyen d'éviter les réponses multiples.

Parfum	**1**	**2**	**3**	**4**	***Somme***
Individu 1	4	2	1	3	*10*
individu 2	3	4	2	1	*10*
.	.	.	.	.	.
.	.	.	.	.	.
Individu n	3	4	1	2	*10*

Tab. 6: un tableau de préférences

Les **tableaux de rangs** sont proches des tableaux de préférences. Considérons par exemple les 95 départements français en ligne et comme variables la population, le revenu moyen, etc. A l'intersection d'une ligne (département) et d'une colonne, on mettra le rang de classement pour la variable considérée: on notera ainsi, pour la population, 1 le département le plus peuplé, 2 le suivant, etc. Ce tableau, bien que similaire, est différent du tableau des préférences car dans les préférences, la ligne représente la permutation des objets alors que pour un tableau de rangs, il s'agit de la colonne. On passe d'un tableau de préférences à un tableau de rang par transposition.

On notera que tout tableau de mesure peut se transformer en tableau de rang (Tab. 7). On perd de l'information car on ne dispose plus de la véritable donnée de départ, mais uniquement de son rang. Le tableau obtenu est bien plus robuste car il permet de tenir compte de l'erreur de mesure. Tous les outils de la statistique se généralisent à ce cas en définissant, par exemple, un rang moyen, une corrélation par rangs, etc.

	Taille	**Poids**
Ind 1	1	1
Ind 2	10	9
Ind 3	3	3
Ind 4	7	5
Ind 5	4	4
Ind 6	8	10
Ind 7	9	8
Ind 8	5	6
Ind 9	6	7
Ind 10	2	2
Somme	*55*	*55*

Tab. 7: un tableau de rangs créé à partir d'un tableau de mesures

4.3. Les tableaux de contingence.

Un tableau de contingence donne la répartition d'une population suivant deux critères qualitatifs éclatés chacun en modalités. Prenons comme exemple le cas où l'on teste quatre procédés de moulage de camemberts. En toute rigueur, les formes obtenues devraient être identiques à une forme idéale que nous connaissons tous (du moins quand il est frais). Hélas, les choses ne sont pas aussi simples car les procédés de moulage déforment plus ou moins le célèbre cylindre d'autant que le lait n'est pas de qualité rigoureusement identique. On peut classer les taux de déformation en trois niveaux (nul, faible, moyen).

Bravement, on se lance dans la fabrication d'une centaine de fromages en testant quatre procédés de fabrication puis on procède au décompte par type de procédé et par type de déformation. On obtient le tableau de contingence à quatre lignes (les procédés) et trois colonnes (les déformations). La centaine de fromages est répartie dans ce tableau (Tab. 8).

Procédé	A	B	C	D	*Marge*
Déformation nulle	5	10	6	9	*30*
faible	5	7	4	18	*34*
moyenne	10	13	10	3	*36*
Marge	*20*	*30*	*20*	*30*	*100*

Tab. 8: Tableau de contingence

Le tableau de contingence recense, à l'intersection de la ligne i et de la colonne j, le nombre de fromages qui ont été fabriqué par le procédé i et ont subi une déformation j. Les marges de chaque côté du tableau font les sous-totaux et on doit bien retrouver les 100 camemberts fabriqués. Si on a constitué un tel tableau, c'est qu'on a comme hypothèse de travail que la déformation n'est pas indépendante du procédé de fabrication. Comment en être sûr?

On crée un tableau qui simule l'hypothèse d'indépendance (Tab. 9). Pour calculer une case de ce tableau, par exemple la première, on tient le raisonnement suivant. Par le procédé A, on a fabriqué 20 fromages sur 100. Par ailleurs 35 fromages sur 100 ont subi une déformation nulle. On s'attend donc à ce que le procédé A fournisse 20% des 35 fromages à déformations nulles, soit 6 fromages. On peut appliquer le même type de calcul à toutes les cases du tableau. Le tableau ainsi créé a les mêmes marges que le tableau précédent.

Procédé	1	2	3	4	*Marge*
Déformation nulle	6	9	6	9	*30*
faible	6,8	10,2	6,8	10,2	*34*
moyenne	7,2	10,8	7,2	10,8	*36*
Marge	*20*	*30*	*20*	*30*	*100*

Tab. 9: L'hypothèse d'indépendance

Le tableau 9 est le tableau qu'on aurait obtenu si procédés de fabrication et niveaux de déformations étaient indépendants l'un de l'autre. Plus le tableau réel sera différent de celui simulant l'hypothèse d'indépendance, plus on rejettera l'hypothèse d'indépendance. Il est possible de créer un indice de liaison de deux variables qualitatives à partir de la comparaison de ces deux tableaux. On peut aller plus loin et analyser les différences dans le détail.

Les tableaux de contingence sont fondamentalement symétriques car on peut effectuer des opérations de même type en ligne et en colonne. Rappelons qu'il faut prendre quelques précautions pour la constitution d'un tel tableau. Si les modalités ne sont pas

exclusives, ce qui serait le cas si, hésitant entre une déformation nulle ou faible, on cocherait les deux cases, il se poserait alors un redoutable problème d'échantillon. Le même camembert compterait alors deux fois. Ces problèmes se posent en cas de réponses multiples comme on en trouve dans les (mauvais) questionnaires. Un amateur de yaourts peut très bien aimer les yaourts à la fraise et à l'ananas mais, en fin de compte, il n'y a qu'un seul consommateur. C'était notre rubrique « Prenez garde à l'unité statistique » en cas de réponses multiples.

4.4. Les tableaux de variables qualitatives.

Le tableau de contingence croise fondamentalement deux variables qualitatives. Ainsi le tableau précédent est issu d'un tableau où l'on aurait en ligne les différents camemberts fabriqués et deux colonnes, l'une avec le procédé de fabrication (1 à 4), l'autre avec la déformation subie (Tab. 10). La présentation des données sous cette forme est moins synthétique que le tableau de contingence, mais elle l'avantage de se pouvoir se généraliser. On peut rajouter une troisième colonne, la densité du lait par exemple, notée de 1 à 3.

	Procédé (1 à 4)	**Déformation (1 à 3)**
Camembert 1	2	3
Camembert 2	1	1
.	.	.
.	.	.
Camembert 100	4	2

Tab 10: Tableau de données associé au tableau de contingence

Les codes des modalités des variables qualitatives sont purement arbitraires et n'ont pas de signification propre. Les données mises sous cette forme sont en **codage réduit**. Aucune opération arithmétique n'a de sens sur un tel tableau. On ne peut additionner ni en ligne, ni en colonne.

Ce tableau est souvent transformé en **codage disjonctif complet**. Chaque variable est découpée selon ses modalités. La variable procédé de fabrication est ainsi transformée en quatre variables, chacune représentant une modalité (type 1, type 2, etc.) et on met 1 lorsque la modalité est présente et 0 lorsqu'elle est absente. Le terme disjonctif vient de ce codage en 0/1.

Contrainte importante de ce type de codage, il y a un 1 et un seul par variable, d'où le nom de complet. Ceci signifie qu'il y a obligatoirement une réponse (au besoin on créera une modalité non réponse) et une seule (pas de réponses multiples dont on a déjà démontré le danger). Ce tableau a comme caractéristique d'avoir des lignes de somme constantes et égales au nombre de questions (Tab. 11).

	Procédé (1 à 4)	Déformation (1 à 3)	Type de lait (1 à 3)
Camembert 1	2	3	1
Camembert 2	1	1	2
.	.	.	.
.	.	.	.
Camembert 100	4	2	3

	Procédé (1 à 4)				Déformation (1 à 3)			Type de lait (1 à 3)		
Camembert 1	0	1	0	0	0	0	1	1	0	0
Camembert 2	1	0	0	0	1	0	0	1	0	0
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
Camembert 100	1	0	0	1	0	1	0	1	0	1

Tab 11: Un tableau de données qualitatives et son tableau disjonctif complet associé

Cette technique de mise en disjonctif complet est très utile pour le traitement de **tableaux de données hétérogènes**, c'est à dire comportant des variables de natures diverses. On notera qu'une variable, a priori qualitative, possède toutes les propriétés d'une variable quantitative lorsqu'elle est mise sous forme 0/1.

4.5. Tableaux de présence-absence et tableaux logiques.

Certains tableaux peuvent se présenter sous forme 0/1 de manière naturelle; c'est le cas dès que l'on recense la possession et la non possession d'un bien, ou bien la lecture/non lecture. Ainsi le tableau constitué des réponses à la question « avez-vous lu hier les journaux suivants... » est sous la forme 0/1. Contrairement aux tableaux disjonctifs-complets, il n'y a aucune raison que la somme des lignes soit constante car cette somme correspond au nombre de journaux lus par la personne considérée, ce nombre étant éminemment variable selon les individus.

Dans les tableaux de présence absence, les 0 et 1 ont une signification précise: le 1 représente la lecture ou la possession et le 0 la non-lecture ou la non-possession. Ces tableaux sont un cas particulier des tableaux où les 0/1 n'ont pas cette signification, les **tableaux logiques ou booléens**. Un tableau logique ou booléen est un tableau rempli de 0 et de 1. Le 0 et le 1 n'ont pas forcément une signification particulière. Ainsi, on pourrait coder 0 le sexe masculin et 1 le sexe féminin ou bien l'inverse. C'est pour cette raison que l'on différencie les données de présence/absence des tableaux logiques purs.

5. Choisir la bonne méthode.

Le choix de la bonne méthode revient d'abord à définir les grandes catégories de méthodes, à expliciter leurs caractéristiques et les résultats qu'elles fournissent, à préciser les tableaux de données utilisés. Trois grandes questions sont à se poser préalablement à toute analyse.

- Veut-on décrire ou expliquer?
- Quels sont les objectifs de l'analyse?
- Quelle est la forme souhaitée des résultats?

5.1. Deux grandes catégories: méthodes descriptives et explicatives.

Les méthodes descriptives traitent l'ensemble des données disponibles sans en privilégier aucune variable a priori. Le tableau de données peut être considéré comme amorphe et on cherche à le structurer pour mieux le comprendre.

Les méthodes explicatives, au contraire, distinguent des variables à expliquer et des variables explicatives. Il n'est plus question de comprendre ce qui se passe en général dans les données (encore que cela ne puisse nuire), mais de les regarder par un bout de la lorgnette qui est celui de la variable à expliquer. Les variables explicatives ne sont intéressantes que dans la mesure où elles expliquent effectivement la variable à expliquer.

Si on reprend notre exemple initial sur les objets volants à identifier, sans information a priori, on cherchera à décrire le tableau dans son intégralité (Tab. 12). Toutes les variables sont sur le même plan, on n'en privilégie aucune. En reconnaissance des formes, on appelle cette stratégie, la reconnaissance non supervisée.

	Vitesse	Altitude	Température	Surface	Type
Objet 1	369	937	50	23	M
Objet 2	69	766	5	9	A
Objet 3	270	320	5	11	H
Objet 4	161	270	4.4	8	H
.	.	.	.	.	.
Objet n	373	939	51	20	M

Tab 12: Analyse par une méthode descriptive (ou non supervisée)

On peut aussi vouloir expliquer les différences entre hélicoptères, avions et missiles. Cette variable joue alors un rôle privilégié qu'on appelle la « variable à expliquer » (Tab. 13). Les autres variables ne jouent plus que le rôle de variables explicatives éventuelles. Les méthodes permettant de traiter de tels tableaux sont appelées des méthodes explicatives ou supervisées.

Les méthodes descriptives fournissent une partie de l'explication, mais les méthodes explicatives permettent d'aller bien au-delà. En revanche, les méthodes explicatives perdent la

vue globale. L'idéal est de commencer par des méthodes descriptives puis, éventuellement, utiliser des méthodes explicatives. L'urgence, la connaissance a priori du phénomène, font que cet idéal n'est pas toujours respecté. En tout état de cause, cette distinction entre méthodes descriptives et méthodes explicatives est une des plus fondamentales qui soit.

	variables explicatives				**A expliquer**
	Vitesse	Altitude	Température	Surface	**Type**
Objet 1	369	937	50	23	**M**
Objet 2	69	766	5	9	**A**
Objet 3	270	320	5	11	**H**
Objet 4	161	270	4.4	8	**H**
.	.	.	.	.	**.**
Objet n	373	939	51	20	**M**

Tab 13: Analyse par une méthode explicative (ou supervisée)

Dans ce cas particulier où la variable à expliquer est qualitative, on aurait pu coder les missiles par 1, les avions par 2 et les hélicoptères par 3. On peut réorganiser le tableau 13 en regroupant d'une part tous les avions, puis tous les hélicoptères et enfin tous les missiles. L'analyse des données revient alors à comparer les structures des trois sous matrices (Tab. 14).

	Vitesse	Altitude	Température	Surface	**Type**
Missile 1	369	937	50	23	**M**
.	.	.	.	.	**.**
Miss. n1	373	939	51	20	**M**
Avion 1	69	766	5	9	**A**
.	.	.	.	.	**.**
Avion n2	8	914	6	15	**A**
Hélico 1	270	320	5	11	**H**
.	.	.	.	.	**.**
Hélico n3	161	270	4.4	8	**H**

Tab 14: Réorganisation des données selon une variable explicative qualitative

5.2. Forme des résultats.

Les méthodes d'analyse de données fournissent leurs résultats sous quatre formes différentes:

- représentations géométriques,
- arbres hiérarchiques,
- partitions,
- modélisation (équations).

5.2.1. Les méthodes géométriques

Les représentations géométriques, appelées parfois graphes factoriels selon la méthode qui les utilisent le plus, représentent les points individus ou variables dans un espace, en règle générale un graphique plan (Fig. 2). Les résultats en sont facilement communicables et l'interprétation paraît simple. Ces graphiques ont beaucoup contribué à populariser l'analyse des données.

L'idée de base de ces méthodes est de considérer que les objets étant décrits par p variables, se situent dans un espace de dimension p. On cherchera alors à ajuster un espace de dimension plus faible, par exemple de dimension 2 si c'est possible, sans perdre trop d'information. On appelle ces méthodes géométriques car leur principe est d'ajuster un sous-espace de dimension faible à un espace de dimension élevée.

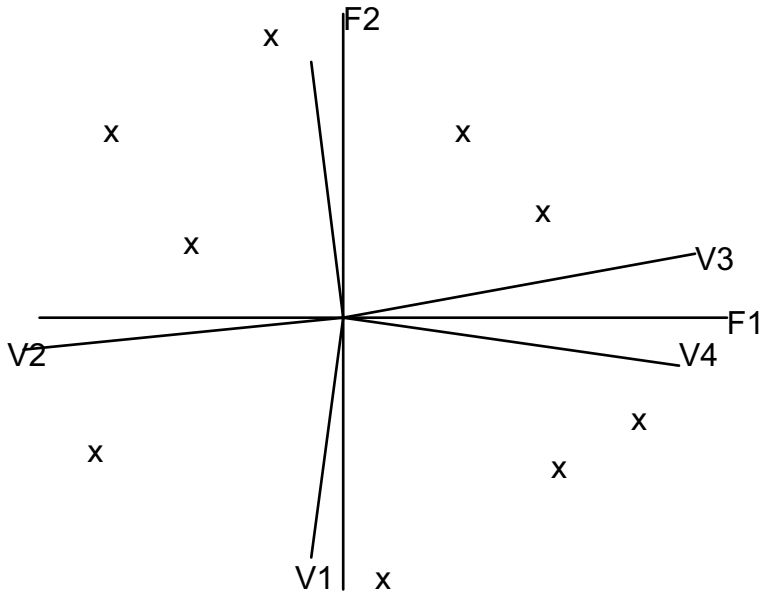


Fig. 2: Représentation géométrique des données
(4 variables V1 à V4, 2 axes de synthèse F1-F2, 9 objets)

5.2.2. La classification hiérarchique

Dans ce type de méthode, on cherche à regrouper les objets seront leur proximité et on le fait dans l'ordre, des plus proches aux moins proches, pas à pas. Il en sort une représentation par arbre. Contrairement aux méthodes précédentes, un utilisera les propriétés algébriques pour regrouper les objets.

Les arbres de classifications hiérarchiques représentent les proximités selon des arbres tels que les organigrammes en râteau que l'on trouve dans les entreprises. La première grande application est sans doute les classifications du règne animal et du règne végétal de Linné. Les groupes obtenus sont emboîtés ce qui permet de passer d'une classification très grossière à une classification plus fine et inversement (Fig. 3).

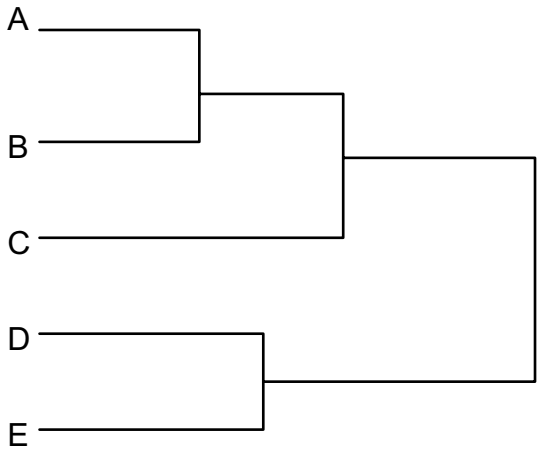


Fig. 3: Représentation des données par arbre (5 objets à classer)

5.2.3. La typologie

Les partitions ou typologies simplifient la réalité par la constitution de groupes composés d'éléments similaires. Les classes obtenues sont souvent disjointes, mais cela n'est pas nécessaire, et aussi homogènes que possible. La figure 4 montre une partition de dix objets en quatre classes. Pour que l'interprétation soit aisée, elles sont en petit nombre. On donne aussi un nom à chaque classe.

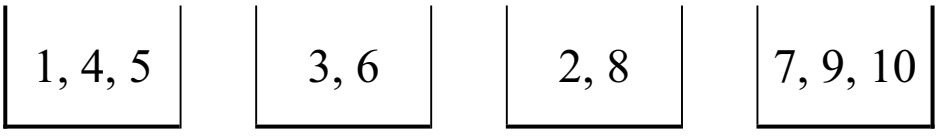


Fig. 4: Représentation des données sous forme de partitions (10 objets, 4 classes)

Comme dans la méthode précédente, on utilise les propriétés algébriques pour regrouper les objets. Mais contrairement à la méthode précédente, on n'impose aucune structure hiérarchique sur les classes obtenues. Ce qui est critique dans cette méthode, c'est la détermination du nombre de classes. L'autre difficulté est purement calculatoire car si on a n objets à mettre en r classes, on cherche à éviter absolument d'évaluer toutes les partitions que

l'on peut ainsi constituer. Il y en a C_r^n, ce qui peut être particulièrement coûteux si n est grand.

5.2.4. Les équations de comportement

Les équations de comportement ou modélisation quantifient sous la forme d'équations les résultats obtenus. Elles sont surtout utilisées dans le cadre des méthodes explicatives comme par exemple la régression classique.

$$Y = a_1 X_1 + a_2 X_2 + ... + a_p X_p$$

Sur le plan mathématique, on peut les considérer comme un cas particulier des méthodes géométriques. Cependant, contrairement aux méthodes géométriques, on cherche à expliciter les équations qu'on peut en tirer. Ces équations sont un bon outil de quantification car on peut mesurer exactement l'impact des variables X_1, X_2, etc. sur la variable à expliquer Y. C'est donc un outil irremplaçable dès que l'on cherche à avoir une idée précise de l'action d'une variable sur une autre.

5.3. Les méthodes de l'analyse des données

Les méthodes de l'analyse des données sont nombreuses. Elles peuvent toutes se décrire selon les deux critères précédents, c'est-à-dire forme des résultats et selon qu'elles sont descriptives ou explicatives (Tab. 15). Si on entre plus profondément dans la technique, le troisième élément de distinction est le type de variables utilisées. Certaines ne permettent d'utiliser que des variables quantitatives, d'autres uniquement quantitatives, d'autres enfin plus rares, permettent de mélanger les deux types de variables.

	Méthodes descriptives Non supervisées	**Méthodes explicatives Supervisées**
Nuages	Méthodes factorielles - composantes principales - correspondances simples - correspondances multiples	Analyse discriminante
Arbre	Classification ascendante hiérarchique	Segmentation
Partition	Nuées dynamiques Plus proches voisins	Segmentation
Equation		Modèle linéaire - Régression - Analyse de variance - Analyse de la covariance

Tab. 15: Les méthodes de l'analyse des données

6. Les objectifs de l'analyse.

Avant de se lancer dans une analyse des données, il est d'autant plus utile d'éclaircir ses objectifs, que ces objectifs président à la constitution des données. Cinq objectifs peuvent être définis:

- explorer les données,
- résumer, classer, réduire les données,
- découvrir des faits nouveaux ou des hypothèses nouvelles,
- valider des hypothèses ou des structures préexistantes,
- prédire des comportements.

6.1. L'exploration des données

Explorer les données est le premier objectif qui vient à l'esprit. "Je veux voir ce qui se passe" est une curiosité bien naturelle. On cherche à décrire objectivement, à voir plus clair dans la masse des données. Cette recherche, contrairement à ce que l'on observe trop souvent, demande de la méthode: vérification de l'homogénéité de l'échantillon et de sa pertinence, détection d'éléments marginaux ou fautes de frappe, validation du recueil des données, qualité de l'information, rien ne doit être laissé au hasard. Cette étape est absolument cruciale pour la suite de l'étude sous peine d'obtenir des résultats incertains.

6.2. Résumer, réduire, classer

Résumer, classer, réduire, forme la trame même de la synthèse des données. Selon les objectifs propres de l'étude, on se focalisera plutôt sur l'un ou sur l'autre des ces trois sous-objectifs. Résumer, c'est mettre à la disposition du plus grand nombre, avec objectivité, l'essentiel de l'information qui est contenue dans les données.

Classer est un pas supplémentaire car il demande de choisir parmi les divers résumés possibles souvent en vue d'une action. Réduire enfin, c'est se restreindre aux caractéristiques les plus essentielles pour, soit effectuer une analyse plus approfondie, soit effectuer une autre collecte de données (sélection d'objets-tests par exemple).

6.3. La découverte de faits nouveaux

Découvrir des faits nouveaux ou des hypothèses nouvelles est l'objectif inavoué de nombreuses études. Mettre en évidence des faits nous ramène au cas précédent. Le Pr Benzecri dirait: « sous la gangue des données, mettre en évidence ce pur joyau qu'est la vérité ». Outre le fait qu'il ne soit pas évident qu'il n'y ait qu'une seule vérité dans les données (la bouteille à moitié vide ou à moitié pleine part du même fait physique, et pourtant l'interprétation est différente), la nouveauté du fait peut poser quelque problème. Il peut être nouveau pour le chargé d'étude et bien connu par ailleurs, ou bien il peut être banal pour quelques personnes déjà dans le secret et inconnu pour les autres, ou bien encore inconnu de tous (dans le cas de nouveaux champs de données), voire carrément contre l'idéologie

dominante. Dans tous les cas de figure, avant d'annoncer par monts et par vaux la bonne nouvelle, il vaut mieux en être sûr.

Plutôt que la découverte d'un fait nouveau, c'est la découverte d'un fait « contre-intuitif » qui est fréquente. Comme le fait est surprenant a priori, on y prête plus d'attention. La vérification de la validité d'un tel fait, passe par la définition d'une véritable hypothèse de travail que l'on testera (cf paragraphe suivant). Si la découverte d'un tel fait ne passe pas inaperçu, il ne faut pas oublier que symétriquement, on risque de passer sous silence des hypothèses couramment admises et qui ne sont jamais vérifiées. Vigilance, donc!

6.4. Validation d'hypothèses

La validation d'hypothèses est la suite logique de la découverte d'un fait, disons peu connu, mais aussi de toutes les hypothèses farfelues ou non qui peuvent passer dans la tête d'une imagination fertile. Il faut une définition claire de l'hypothèse à tester et un contrôle de l'effet des autres variables.

Cette validation passe par le test de la validité et de la robustesse des méthodes et des données. La réponse attendue est, à l'extrême, du type oui/non, avec un certain intervalle de confiance. Un ensemble de faits et d'hypothèses définit la notion de structure. La validation des structures, préexistantes ou non, est nettement plus complexe qu'une simple réponse du type oui/non.

6.5. La prédiction de comportements

Prédire des comportements, c'est vouloir extrapoler à d'autres individus dont on connaît partiellement les caractéristiques, les résultats obtenus sur un groupe d'entre eux. On distingue les éléments de « base » qui ont permis la mise au point des comportements et de les modéliser, les éléments « tests » qui ont servi à valider la modélisation et enfin les individus « anonymes » qui sont les vrais cobayes de l'expérimentation.

Ce sont les méthodes explicatives qui servent essentiellement à ce type de prédiction. Rappelons à tout hasard l'incompatibilité entre l'efficacité d'un modèle sur un échantillon test et sa capacité de généralisation à d'autres bases de données. En gros, un modèle sera d'autant plus efficace sur un échantillon de test qu'il aura de nombreux paramètres et donc qu'il s'ajustera mieux à ces données. Malheureusement, en épousant trop la configuration particulière des données de tests, il sera moins efficace sur d'autres données. Sur des données test, on souhaite un modèle le plus complexe possible. Pour que ce modèle soit généralisable à d'autres données, il vaut mieux qu'il ne reflète pas trop les spécificités des données de test et donc qu'il ne soit pas trop complexe. Il faut rechercher le compromis optimal.

Ces prédictions sont utilisées dans les domaines aussi divers que le crédit-scoring (le crédit aux particuliers), les opérations chirurgicales ou les études de marché.

7. Bibliographie

7.1. Analyse des données en reconnaissance des formes

Devijver P. and Kittler J., *Pattern Recognition; a statistical approach*. Prentice Hall, New-York, 1982.

Fukunuga K., *Statistical Pattern Recognition*. Academic Press, Boston. Second Edition, 1990.

Simon J.C., *La reconnaissance des formes par algorithmes*. Masson, 1984.

7.2. Analyse des données

Benzécri J.P. (dir.), *Pratique de l'analyse des données*, Tome 1, Dunod, Paris, 1980.

Breiman L., Friedman J.H., Ohlsen R.A., Stone C.J., *Classification and Regression Trees*, Wadsworth, 1984.

Caillez F., Pagès J.P., *Introduction à l'analyse des données*. SMASH, Paris, 1976.

Celeux G. (ed.), *Analyse discriminante sur variables continues*, INRIA, Rocquencourt, 1990.

Jambu M., *Exploration informatique et statistique des données*, Dunod, coll. Cnet-Enst, 1989.

Lebart L., Morineau A., Fenelon J.P., *Traitement des données statistiques*, Dunod, Paris, 1979.

Lebart L., Morineau A., Tabard N., *Techniques de la description statistique*, Dunod, Paris, 1977.

Lebart L., Morineau A., Warwick K.W., *Multivariate Descriptive Statistical Analysis. Correspondence Analysis and Related Techniques for Large Matrices*, Wiley, New York, 1984.

Roux M., *Algorithmes de classification*, Masson, Paris, 1985.

Saporta G., *Probabilités, analyse des données et statistique*, Technip, Paris, 1990.

Tomassone R., Danzart M.., Daudin J.-J., Masson J.-P., *Discrimination et classement*, Masson, Paris, 1987.

Tomassone R., Lesquoy E.., Millier C., *La régression: nouveaux regards sur une ancienne méthode statistique*, Masson, Paris, 1983.

Volle M., *Analyse des données*, Economica, 3° ed., 1985.

8. Annexe 1 l'analyse en composantes principales

L'analyse en composantes principales (ACP) fait partie des méthodes descriptives. Ses résultats se présentent sous des formes géométriques. Elle s'applique classiquement à des tableaux de n lignes et p colonnes. Mais un des deux ensembles (généralement les colonnes) doit obligatoirement être composé de données quantitatives. Cette contrainte se traduit par le fait qu'il est pertinent de calculer des moyennes sur cet ensemble.

Historiquement, l'ACP est due à Spearman (1901) qui en a jeté les principes et à Hotelling (1933) qui l'a intégrée à la statistique mathématique. Les méthodes connues sous les vocables « décomposition en valeurs singulières », SVD pour « singular value decomposition » ou encore transformation de Karhunen-Loève, apparues bien postérieurement (plus d'un demi-siècle), sont rigoureusement identiques à l'analyse en composantes principales. Pour éviter toute ambiguïté, il vaut donc mieux abandonner ces dénominations qui n'ont cours, et à tort, que dans quelques domaines restreints.

8.1. Exemples d'applications

1 – Plusieurs centaines d'objets volant dans le ciel sont identifiés par leur signature radar qui se concrétise par quelques dizaines de caractéristiques (altitude, vitesse, surface de l'objet, température, etc.). A partir de ces caractéristiques, il faut identifier totalement l'objet. L'analyse en composantes principales permettra, dans un premier temps, de comprendre très rapidement le tableau de données dans son ensemble.

2 - Sur 18 pays de l'O.C.D.E. ont été mesurées 16 variables quantitatives représentatives de leur économie au sens large. D'unités diverses, les variables sont très hétérogènes : elles peuvent être mesurées soit en valeurs absolues (dollars, kwh, calories) soit dans des ratios (pourcentages). Toutes ces variables sont quantitatives, mais elles ne sont pas homogènes.

3 - Dans une année donnée, ont été recensées certaines caractéristiques des diverses automobiles disponibles sur le marché. Ces caractéristiques techniques sont forcément hétérogènes (chevaux, francs, km/s, litre, mètres). De plus, elles ne semblent même pas être représentatives d'un phénomène particulier, comme la vitesse par exemple. Ces variables représentent, probablement d'une manière lacunaire, les caractéristiques des automobiles prises dans leur ensemble.

Ces trois exemples ont pour objectif de comprendre « ce qui se passe dans le tableau », ce qui signifie bien que l'optique est descriptive. Par ailleurs, les colonnes de ces trois tableaux étaient composées de mesures, donc de variables quantitatives. En revanche, il se peut que ni l'homogénéité de forme, ni l'homogénéité de fond ne soient respectées.

8.2. Types de tableaux

Le tableau de données modèle (Tab. 1) est constitué, en lignes, d'observations (ou d'individus) indépendantes les uns des autres et, en colonne, de variables quantitatives sur lesquelles il est possible de calculer une moyenne. On considérera que les données

quantitatives sont en colonnes et qu'il y a n observations (n lignes) et p variables (p colonnes)..

	$\mathbf{V_1}$		$\mathbf{V_j}$		$\mathbf{V_P}$
Obs. 1					
Obs. I					
.					
Obs n					
Moyenne	m_1		m_j		m_p

Tab 1 Le tableau modèle de l'analyse en composantes principales

8.3. Forme des résultats

L'analyse en composantes principales cherche à extraire des synthèses, si possible indépendantes les une des autres, ce qui facilite l'interprétation. Chacune de ces synthèses est appelée facteur et est notée F. Les facteurs possèdent les deux propriétés suivantes :

- les facteurs sont des combinaisons linéaires des variables initiales :

$$F_k = \sum_{j=1}^{p} \alpha_j \, V_j$$

- les facteurs sont indépendants deux à deux :

$$F_i \perp F_j \quad i \neq j$$

8.4. Principe de la méthode

Chaque ligne (appelée observation) est décrite par p mesures $x_i = (x_{i1},x_{i2},...,x_{ip})$. Cette ligne peut être considérée comme un point de l'espace $\Re^p$. Dans cet espace constitué de n points, on s'intéresse à la proximité des observations. Cette proximité permet de déceler, par exemple, quels sont les groupes d'observations qui ont des mesures voisines les unes des autres. L'ensemble des observations dans $\Re^p$ constitue un nuage.

$$\text{n observations} \in \Re^p$$

Un raisonnement analogue peut être tenu pour les colonnes, qu'on appelle des variables. Chaque variable est décrite par les mesures sur les n observations : $x_j = (x_{1j}, x_{2j},...,x_{nj})$. On est donc confronté à un autre nuage, celui des p variables qui sont dans $\Re^n$.

p variables $\in \Re^n$

Comme le tableau est composé de données quantitatives, il est possible de calculer la moyenne m_j de la variable j sur l'ensemble des observations ainsi que son écart type σ_j.

$$m_j = \frac{1}{n}\sum_{i=1}^{n} X_{i\,j}$$

$$\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n} (X_{i\,j} - m_j)^2$$

8.4.1. Les trois analyses en composantes principales

On peut directement **analyser les données brutes**, telles quelles. L'origine du nuage est définie par l'observation qui a comme coordonnées (0, 0, ..., 0). Sauf cas particulier, cette origine est rarement intéressante, car elle n'a guère de sens physique.

Il est généralement plus judicieux de se ramener au centre de gravité du nuage, c'est-à-dire prendre comme référence l'observation qui a comme coordonnées (m_1, m_2, ..., m_p). L'analyse devient alors une **analyse centrée**. Mathématiquement, cela revient, pour chaque observation i à remplacer ses coordonnées par les écarts à la moyenne.

$X_i \quad \Rightarrow \quad Y_i = (Y_{i1}, Y_{i2}, \ldots Y_{ip}) \quad \text{où} \quad Y_i = X_{ij} - m_j$

En raison des hétérogénéïtés d'échelle, il faut, qu'en moyenne, aucun des termes $(Y_{ij}-Y_{i'j})^2$ ne soit pas prépondérant dans la somme ci-dessus. Plus une coordonnée a un écart-type élevé, plus cette coordonnée aura des chiffres disparates et plus elle influera sur la comparaison. Pour donner le même poids a chaque variable, on procédera à une **analyse normée** en remplaçant les coordonnées de l'observation i de la manière suivante :

$$X_{i\,j} \quad \Rightarrow \quad Y_{i\,j} = \frac{(X_{i\,j} - m_j)}{\sigma_j \sqrt{n}}$$

8.4.2. La distance entre deux observations

Quelle que soit la transformation préalable, la comparaison de deux observations s'effectue en comparant les deux vecteurs correspondants. La distance euclidienne usuelle est souvent utilisée à cet effet. La distance entre deux points i et i', s'exprime par la formule :

$$d^2(i,i') = \sum_{j=1}^{n} (Y_{i\,j} - Y_{i'\,j})^2$$

8.4.3. Liaisons entre les variables

Parallèlement à ces trois analyses, on peut également calculer un indicateur qui mesure la relation entre deux variables, tout comme il existe une distance entre deux

observations. Comme il s'agit de variables quantitatives, il est naturel de considérer le produit scalaire entre deux variables comme un indicateur de liaison.

Données brutes

$$\langle j,j' \rangle = \sum_{i=1}^{n} X_{ij}\ X_{ij'}$$

Données centrées

$$\langle j,j' \rangle = \sum_{i=1}^{n} (X_{ij} - m_j)\ (X_{ij'} - m_{j'}) = n\ Cov(j,j')$$

Dans ce cas le produit scalaire est, au facteur n près, la covariance entre les deux variables.

Données normées

$$\langle j,j' \rangle = \frac{1}{n} \frac{\sqrt{\sum_{i=1,n} (X_{ij} - m_j)\ (X_{i'j} - m_{j'})}}{\sigma_j\ \sigma_{j'}} = n\ Corr\ (j,j')$$

On constate que le produit scalaire entre deux variables normées donne, au facteur n près, la corrélation entre les deux variables. En raison de la généralité de cet indicateur mais surtout grâce aux propriétés de normalisation, l'analyse en composantes principales normée est la plus courante, car la plus adaptée aux données hétérogènes.

8.4.4. La projection dans un sous-espace

Le but de la méthode est de visualiser dans un espace de petite dimension (un plan par exemple), les proximités entre observations, et aussi les corrélations entre variables. On cherche donc à projeter le nuage de R^p (les observations) sur un plan.

Parrallèlement, les variables (nuage de R^n) sont projetées sur un plan. Alors que les espaces de départ $\Re^p$ et $\Re^n$ sont très différents, on peut faire en sorte que les deux espaces de projection soient de même dimension et, si possible, superposables.

Dans $\Re^p$, on cherche le sous espace de dimension 1, donc un axe, qui passe au mieux à travers le nuage ou, plus mathématiquement, tel que la projection de $\Re^p$ soit la moins déformée possible. On procède ainsi ensuite pour le sous espace de dimension 2, puis de dimension 3, etc.

Un critère de déformation minimum de nuage par projection est celui où les distances entre les points projetés sont les plus voisines de celles entre les points initiaux. On montre mathématiquement que la solution de ce problème revient à rechercher les vecteurs propres de la matrice $Y^t\ Y$ où la matrice Y est constituée de (Y_{ij}).

Le premier vecteur u_1 associé à la plus grande valeur propre λ_1 donne l'axe sur lequel, par projection, les distances entre observations sont les mieux respectées.

Le deuxième vecteur propre u_2, orthogonal au premier, fournit avec celui-ci le plan de projection recherché, etc. Plus généralement, le sous-espace de dimension q est défini par les q premiers vecteurs propres. La détermination de la taille exacte du sous espace de projection optimal n'est pas triviale.

8.4.5. Représentation simultanée

De la même manière, on peut s'intéresser à la proximité des variables dans $\Re^n$. La transformation subie par la matrice $X = (X_{i\ j})_{nxp}$ des données initiales, devenues par transformation $(Y_{i\ j})$, a centré et réduit les variables. Conséquence dans l'espace $\Re^n$, les points variables sont tous situés sur la sphère de rayon unité.

Les proximités entre variables ainsi réduites s'expriment en terme de **corrélations.**

$$d^2(j,j') = 2\,(1 - c_{j\ j'})$$

où $c_{j\,j'}$ est la corrélation entre j et j'

Dans ce cas, la matrice dont il faut chercher vecteurs et valeurs propres est $Y\ Y^t$ qui est de de dimension $n_{x}n$. Les valeurs propres sont identiques à celles de $Y^t\ Y$, du moins pour les p premières, les autres étant nulles. Mais il n'est pas nécessaire de calculer vecteurs et valeurs propres de $Y\ Y^t$, car on a la relation suivante :

$$v_k = \frac{1}{\sqrt{\lambda_k}}\, u_k$$

où u_k est le kème vecteur propre de $Y^t\ Y$ et v_k celui de $Y\ Y^t$, tous deux associés à la kème valeur propre.

Le principe de la représentation simultanée revient, une fois projeté le nuage des observations dans un plan factoriel, à représenter les variables sur le même graphique par ses coordonnées sur le plan factoriel correspondant de R. Dans notre cas, celles-ci sont les coefficients de corrélation de la variable avec les deux axes factoriels.

L'intérêt de la représentation simultanée est de fournir une aide à l'interprétation des axes factoriels. Une corrélation très grande d'un axe avec un groupe de variables (c'est-à-dire géométriquement des points représentatifs de celles-ci très éloignés de l'origine), permet de supposer que cet axe « représente » en quelque sorte un résumé de l'information apportée par ces variables. Nous verrons sur les exemples ce que cela signifie exactement.

Il faut cependant remarquer que la proximité entre un point observation (élément de $\Re^p$) et un point variable (élément de $\Re^n$) ne signifie rien. Par contre, la position d'une variable par rapport à l'ensemble des observations ou réciproquement celle d'une observation par rapport à l'ensemble des variables, est significative.

Notons cependant que le centrage et la réduction qui permettent l'interprétation simultanée des variables et des observations n'est pas accessible sur tous les logiciels.

L'utilisateur fera donc bien de voir si les variables s'interprètent bien en termes de corrélations et notamment si le terme σ de l'écart-type est présent dans la réduction. Dans le cas contraire, les projections des points variables pourront avoir des coordonnées supérieures à l.

9. Annexe 2 l'analyse discriminante

L'analyse discriminante s'applique au cas où l'on dispose d'une variable qualitative à expliquer permettant de partitionner les unités statistiques a priori et de variables à expliquer quantitatives. Il s'agit donc d'une méthode explicative. La méthode fait partie des méthodes factorielles et, par conséquent, fournit, à la manière d'une analyse en composantes principales, des composantes synthétiques.

Plus pragmatiquement, l'objectif de la méthode est le suivant : connaissant le résultat des mesures des variables quantitatives effectuées sur une observation, peut-on en déduire à quelle classe de la partition, il appartient ? Cette prédiction s'appuie sur une base de données d'apprentissage pour « entraîner » l'algorithme à prédire et sur une base de tests pour tester son efficacité.

Philosophiquement, la méthode est proche de la régression, mais la variable à expliquer est qualitative (alors qu'elle est quantitative pour la régression). Elle est aussi proche de l'analyse en composantes principales, mais au lieu de rechercher le meilleur sous-espace d'approximation pour les données, elle recherche le meilleur sous-espace d'approximation qui sépare le mieux possible les modalités de la variable à expliquer.

Le vocable analyse discriminante cache un ensemble de techniques dont la paternité revient à Fisher (1936). Ce document se restreint à l'analyse factorielle discriminante, aussi nommée analyse discriminante linéaire, ces deux méthodes étant identiques quoique venant de deux horizons différents, l'une de l'analyse multidimensionnelle, l'autre de la statistique classique.

9.1. Exemples

1- Un individu souffre d'ictères (jaunisse). Ceux-ci se répartissent en deux groupes : les ictères qui nécessitent une intervention chirurgicale et les ictères nécessitant une thérapeutique médicamenteuse. On cherche à savoir, à partir d'un certain nombre d'examens cliniques, biologiques, à quel groupe appartient le patient ainsi que la marge d'erreur de cette prédiction.

2 - Peut-on déterminer le groupe politique auquel appartient un député, en connaissant la fréquence avec laquelle il emploie un certain nombre de mots-clés lors de ses interventions à l'Assemblée. L'exemple a été traité sur le cas des députés de 1881, ce qui est une attitude sereine, mais de nombreux exemples d'analyse de données textuelles ont été traités.

3 - Un industriel s'intéresse au marché du micro-ordinateur. Il voudrait savoir comment des variables telles que l'âge du chef de ménage, son revenu, le nombre d'enfants, la surface de la salle de séjour…, influencent le choix d'achat d'un micro. Il faut donc d'une base de données où figurent ces diverses caractéristiques ainsi que la variable « dispose ou ne dispose pas d'un micro ».

9.2. Le tableau modèle

Supposons que, dans un tableau de p colonnes, la variable à expliquer à q modalités. On peut alors partitionner le tableau de données en q blocs de p colonnes chacun. Notons n_i, le nombre d'observations de la partition i. L'ensemble de blocs comprend n observations, qui est la somme du nombre d'observations de chacun des blocs.

	V_1	V_2	.	V_p	**Type**
I1 1	369	937	50	23	**1**
.	.	.	.	.	**.**
I1 n_1	373	939	51	20	**1**
	69	766	5	9	**.**
.	.	.	.	.	**.**
	8	914	6	15	**.**
Iq 1	270	320	5	11	**q**
.	.	.	.	.	**.**
Iq n_q	161	270	4.4	8	**q**

9.3. Principe

9.3.1. Les trois analyses intermédiaires

A partir de ce tableau, plusieurs analyses peuvent être menées. La première est de considérer qu'on a qu'un seul tableau de p colonnes et de n lignes avec $n = n_1 + n_2 + ... + n_q$. Pour un tel tableau, en notant m_j la moyenne de la variable j, la covariance entre deux variables j et k peut s'écrire :

$$Cov(j, j') = \frac{1}{n} \sum_{i=1}^{n} (X_{ij} - m_j)(X_{i\,j'} - m_{k\,j'})$$

Cette approche est la même que celle de l'analyse en composantes principales du tableau entier. Elle ne tient pas compte du découpage de la variable à expliquer.

L'approche qui vient d'être effectuée globalement, peut s'appliquer à chacun des sous tableaux. Chaque classe k de la partition est une sous population de dimension n_k de la population initiale. Il est donc possible de calculer la covariance entre j et j' sur cette sous population.

$$Cov_k(j, j') = \frac{1}{n_k} \sum_{i=1}^{n_k} (X_{ij} - m_j)(X_{i\,j} - m_{k\,j'})$$

La moyenne de la variable j sur la population k est calculée de la manière traditionnelle :

$$m_{k\,j} = \frac{1}{n_k} \sum_{i=1}^{n_k} X_{i\,j}$$

Le centre de gravité de la classe k est donné par le vecteur :

$$\vec{m}_k = (m_{k\,1}, m_{k\,2}, \ldots, m_{k\,p})$$

Enfin, une troisième analyse peut être menée. L'ensemble des q centres de gravité $\vec{m}_k$, chacun affecté du poids de la classe qu'il représente dans la population totale, constitue une population sur laquelle on peut calculer la covariance de deux variables quantitatives par la formule :

$$Cov_Q(j, j') = \frac{1}{n} \sum_{k=1}^{q} \frac{n_k}{n} \left(m_{k\,j} - m_j\right)\left(m_{k\,j'} - m_{j'}\right)$$

Le théorème de Huygens donne la relation remarquable suivante :

$$Cov(j, j') = \sum_{k=1}^{q} Cov_k(j, j') + Cov_Q(j, j')$$

Cette relation prend la signification suivante :

covariance totale = covariance dans les classes + covariance entre classes

Cette propriété s'écrit aisément matriciellement

$$T = D + E$$

T, D, E sont les matrices des covariances totales, intra-classes, interclasses.

9.4. Méthode

9.4.1. L'approche descriptive

Le principe de l'analyse discriminante revient à chercher parmi les combinaisons linéaires des variables quantitatives, celle qui discrimine le mieux les classes.

- Dans un premier temps, les variables quantitatives sont centrées, de telle sorte que le centre de gravité du nuage des observations soit à l'origine dans $\Re^p$.

- Dès lors, la discrimination des classes est analogue à un problème d'analyse en composantes principales : à partir du nuage des centres de gravité des classes, il faut trouver l'axe factoriel le long duquel les distances entre ces centres de gravités des classes sont les mieux respectées.

- Le problème est légèrement différent, car chaque point du nuage des centres de gravité a une masse propre (liée à l'effectif de sa classe) et on doit raisonner en terme d'inertie, et non pas en termes de distance : trouver l'axe factoriel par rapport auquel l'inertie du nuage des centres de gravité est minimum.

- Mathématiquement on montre que la formulation est la suivante :

Dans $\Re^p$, il faut trouver un vecteur normalisé u tel que les distances des projections des centres de gravité soit maximum, ce qui revient à maximiser :

$$\frac{U' D U}{U' T U}$$

D'après le théorème de Huygens et la relation T = D + E, il revient au même de minimiser la dispersion des points autour des centres de gravité, donc de minimiser la relation suivante :

$$\frac{U' E U}{U' T U}$$

Un compromis simple pour intégrer les relations est de maximiser la relation suivante :

$$U' E^{-1} D U$$

On peut montrer que la solution du problème est le vecteur propre normalisé de la matrice E^{-1} U, associé à sa plus grande valeur propre. En poursuivant la recherche des valeurs et vecteurs propres, on peut extraire un deuxième, puis un troisième vecteur propre etc., exactement comme en analyse en composantes principales.

9.4.2. L'approche décisionnelle

Cependant, par le fait que la variable expliquée est qualitative, la méthode peut être utilisée très simplement à des fins décisionnelles. Soit une observation dont on ne connaît que les caractéristiques quantitatives. A quelle classe faut-il l'affecter ?

Une première réponse à cette question passe par une affection graphique. Il suffit d'introduire, sur le plan factoriel, l'observation dont la classe est inconnue, en élément supplémentaire (à poids nul). Et on lui affecte le numéro de la classe qui le contient dans le plan factoriel.

Cette même affection peut se faire numériquement. Pour cet observation supplémentaire, il faut calculer la distance à chacun les centres de gravité par la formule ci-dessous et l'affecter à la classe dont la distance au centre de gravité est la plus proche. La formule est la suivante :

$$d^2\,(a, g_k) = (a - g_k)'\, E^{-1}\,(a - g_k)$$

Il existe d'autres méthodes d'analyse discriminante à but décisionnel. Citons les approches de Sébestyen (l'analyse discriminante quadratique), les approches bayesiennes, etc.

CHAPITRE 4 : CLASSIFICATION AUTOMATIQUE - METHODES NON HIERARCHIQUES

Isabelle Bloch, Jean Marie Nicolas

1 Introduction

L'objectif de ce chapitre est de donner un aperçu des méthodes automatiques non hiérarchiques de classification.

On trouve sous le terme de méthodes automatiques les méthodes non supervisées (sans professeur), appelées aussi méthodes de regroupement, ou coalescence (clustering en anglais). Deux catégories de méthodes automatiques sont classiquement distinguées.

1. Les méthodes non hiérarchiques seront introduites dans ce chapitre en repartant de méthodes classiques, vues dans les chapitres précédents, où l'on fait l'hypothèse que les probabilités sous-jacentes sont connues (partie 2). Cette hypothèse n'étant pas toujours vérifiée, on aboutit par approximation à des méthodes itératives (partie 3). La majorité des méthodes non hiérarchiques relève de cette classe, les itérations étant conduites de sorte à optimiser des critères de classification (partie 4). Plusieurs algorithmes classiques en découlent : k-moyennes (partie 5), ISODATA (partie 6), boules optimisées (partie 7), nuées dynamiques (partie 8) et Fuzzy C-means (partie 9). Nous conclurons ce chapitre par quelques remarques sur les limites de ces méthodes (partie 10).
2. Les méthodes hiérarchiques seront présentées dans le chapitre 5. Nous y montrerons essentiellement l'équivalence entre hiérarchie indicée et ultra-métrique.

Le manque de connaissance qui préside souvent à l'utilisation des méthodes automatiques conduit à faire des hypothèses fortes sur la structure de l'espace de décision (métrique sur l'espace des paramètres ou espace de représentation, forme des nuages des classes). Les méthodes donneront donc de bons résultats si les hypothèses sont vérifiées, et se dégraderont au fur et à mesure qu'on s'en éloigne. Ces méthodes nécessitent ainsi un bon savoir-faire de l'utilisateur, qui vient en quelque sorte remplacer par son expertise les connaissances requises pour un apprentissage supervisé.

Partant de l'hypothèse où les formes des densités de probabilités sous-jacentes sont connues, les méthodes non hiérarchiques se ramènent à un problème d'estimation de paramètres, très proche de ceux traités dans les chapitres précédents, en particulier de celui de l'apprentissage supervisé. La limite imposée par ce type d'hypothèse conduit à reformuler le problème de manière complètement différente, sous la forme d'un problème de partitionnement ou regroupement des objets à classifier selon certains critères. C'est cette formulation qui conduit à la plupart des algorithmes utilisés. La mise en œuvre de ce type d'algorithme requiert cependant une bonne expertise de la part de l'utilisateur. Nous verrons en particulier que le choix du nombre de classes

est pratiquement incontournable. Nous verrons aussi que les performances sont d'autant meilleures que l'utilisateur a une idée globales des caractéristiques des classes lui permettant d'initialiser efficacement ces algorithmes "automatiques".

2 Hypothèses des structures probabilistes connues

2.1 Expression des paramètres des lois sous jacentes

On suppose ici que l'on connait les formes des lois de probabilités sous jacentes et que les paramètres suivants sont connus :

- nombre de classes c (les classes sont notées ω_i, $i = 1, ..., c$),
- probabilités a priori $P(\omega_i)$ $(i = 1, ..., c)$,
- probabilités conditionnelles $p(x|\omega_i; \theta_i)$, où les θ_i sont les paramètres des lois de probabilités relatives à chaque classe, et x désigne un échantillon quelconque à classer.

Les inconnues sont donc les paramètres θ_i. La loi des probabilités totales s'exprime alors par :

$$p(x; \theta) = \sum_{i=1}^{c} p(x|\omega_i; \theta_i) P(\omega_i), \tag{1}$$

où θ désigne le vecteur de paramètres θ_1, ..., θ_c. Cette équation exprime que la loi $p(x; \theta)$ est une loi de mélange.

La reconnaissance à partir d'un tel modèle suppose que le phénomène soit **identifiable**, c'est-à-dire que pour deux vecteurs de paramètres différents, les probabilités conditionnelles doivent être différentes pour au moins un échantillon x :

$$\theta \neq \theta' \quad \Rightarrow \quad \exists x \text{ tel que } p(x; \theta) \neq p(x; \theta'). \tag{2}$$

L'hypothèse d'identifiabilité est généralement admise.

La solution la plus fréquemment adoptée pour estimer les paramètres consiste à utiliser l'estimateur du maximum de vraisemblance, à partir de n échantillons x_1, ..., x_n, et donc à chercher θ qui maximise l'expression :

$$p(x_1, ..., x_n; \theta). \tag{3}$$

On choisit de plus souvent des échantillons indépendants, ce qui simplifie les calculs puisqu'on a alors :

$$p(x_1, ..., x_n; \theta) = \prod_{k=1}^{n} p(x_k; \theta). \tag{4}$$

La résolution s'effectue classiquement en passant au logarithme (pour supprimer le produit, ce qui allège toujours les expressions des dérivées) et à dériver l'expression, en supposant que $p(x_1, ..., x_n; \theta)$ est différentiable[1] :

$$\frac{\partial \log p(x_1, ..., x_n; \theta)}{\partial \theta_i} = \sum_{k=1}^{n} \frac{\partial \log p(x_k; \theta)}{\partial \theta_i}$$

[1] On reconnait le calcul de la log-vraisemblance

$$= \sum_{k=1}^{n} \frac{1}{p(x_k;\theta)} \frac{\partial}{\partial \theta_i} [\sum_{j=1}^{c} p(x_k|\omega_j;\theta_j) P(\omega_j)]$$
$$= \sum_{k=1}^{n} \frac{1}{p(x_k;\theta)} \frac{\partial}{\partial \theta_i} [p(x_k|\omega_i;\theta_i) P(\omega_i)]$$

En appliquant la règle de Bayes (voir chapitre 2) :

$$P(\omega_i|x_k;\theta) = \frac{P(\omega_i)\, p(x_k|\omega_i;\theta_i)}{p(x_k;\theta)}$$

on obtient :

$$\frac{\partial \log p(x_1,...,x_n;\theta)}{\partial \theta_i} = \sum_{k=1}^{n} \frac{P(\omega_i|x_k;\theta)}{p(x_k|\omega_i,;\theta_i)} \frac{\partial p(x_k|\omega_i;\theta_i)}{\partial \theta_i}$$
$$= \sum_{k=1}^{n} P(\omega_i|x_k;\theta) \frac{\partial \log p(x_k|\omega_i;\theta_i)}{\partial \theta_i}.$$

Ce calcul suppose que θ_i et θ_j sont fonctionnellement indépendants pour $i \neq j$. L'estimation du maximum de vraisemblance revient donc à résoudre, pour chaque θ_i, l'équation :

$$\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\theta}) \frac{\partial \log p(x_k|\omega_i;\hat{\theta}_i)}{\partial \theta_i} = 0. \tag{5}$$

Cette méthode est généralisable au cas où les $P(\omega_i)$ sont inconnues, en effectuant cette fois la maximisation sous les contraintes suivantes :

$$\begin{cases} \forall i,\ P(\omega_i) \geq 0, \\ \sum_{i=1}^{c} P(\omega_i) = 1. \end{cases} \tag{6}$$

On aboutit alors au système suivant (pour $i = 1, ...c$) :

$$\begin{cases} \hat{P}(\omega_i) = \frac{1}{n} \sum_{k=1}^{n} \hat{P}(\omega_i|x_k;\hat{\theta}), \\ \sum_{k=1}^{n} \hat{P}(\omega_i|x_k;\hat{\theta}) \frac{\partial \log p(x_k|\omega_i;\hat{\theta}_i)}{\partial \theta_i} = 0, \end{cases}$$

avec :

$$\hat{p}(\omega_i|x_k;\hat{\theta}) = \frac{p(x_k|\omega_i;\hat{\theta}_i)\hat{P}(\omega_i)}{\sum_{j=1}^{c} p(x_k|\omega_j;\hat{\theta}_j)\hat{P}(\omega_j)}.$$

2.2 Exemple de la loi normale

Prenons comme exemple d'application le cas d'un mélange de lois normales, où les inconnues sont les moyennes des lois. Les probabilités conditionnelles ont la forme :

$$p(x|\omega_i;\theta_i) \stackrel{\mathcal{L}}{\equiv} \mathcal{N}(\mu_i, \Sigma_i), \tag{7}$$

où $\mathcal{N}(\mu_i, \Sigma_i)$ désigne la loi normale de moyenne μ_i (de même dimension que les échantillons) et de matrice de variance-covariance Σ_i (supposée connue). On a (d désignant la dimension de l'espace des échantillons) :

$$p(x|\omega_i;\mu_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)},$$

d'où :

$$\log p(x|\omega_i;\mu_i) = -\log[(2\pi)^{d/2}|\Sigma_i|^{1/2}] - \frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i),$$

et donc :

$$\frac{\partial}{\partial\mu_i}[\log p(x|\omega_i;\mu_i)] = \Sigma_i^{-1}(x-\mu_i).$$

En dérivant le logarithme de l'équation 4, on obtient alors l'équation suivante, où $\hat{\mu} = (\hat{\mu_1}, ...\hat{\mu_c})$ est le vecteur des moyennes estimées pour chaque classe :

$$\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\mu})\Sigma_i^{-1}(x_k - \hat{\mu_i}) = 0, \tag{8}$$

d'où on tire l'expression de $\hat{\mu_i}$:

$$\hat{\mu_i} = \frac{\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\mu})x_k}{\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\mu})}. \tag{9}$$

Cette forme de résultat est très satisfaisante intuitivement, puisqu'on obtient $\hat{\mu_i}$ comme moyenne des échantillons pondérée par les probabilités a posteriori (caractérisant l'appartenance des échantillons à la classe ω_i).

Cependant, cette expression est difficile à calculer (elle fait intervenir les $\hat{\mu_i}$ des deux côtés de l'équation). Si l'on remplace dans cette expression les $P(\omega_i|x_k;\hat{\mu})$ par leur développement selon le théorème de Bayes :

$$P(\omega_i|x_k;\hat{\mu}) = \frac{p(x_k|\omega_i;\hat{\mu_i})P(\omega_i)}{\sum_{j=1}^{c} p(x_k|\omega_j;\hat{\mu_j})P(\omega_j)}, \tag{10}$$

avec $p(x|\omega_i;\hat{\mu_i}) \stackrel{\mathcal{L}}{\equiv} \mathcal{N}(\hat{\mu_i}, \Sigma_i)$, on obtient un système d'équations couplées non linéaires, dont la résolution est compliquée.

2.3 Exemple d'algorithme itératif

Plutôt que d'essayer de résoudre un système d'équations couplées non linéaires, on préfère en général des schémas itératifs où, partant d'une initialisation des paramètres, on effectue l'estimation de ces mêmes paramètres.

Dans l'exemple précédent de la loi normale, on a alors l'algorithme suivant :

- **Etape 1** : initialiser $\hat{\mu_i}(0)$, initialiser $j = 0$.
- **Etape 2** : Effectuer l'estimation des μ_i à l'itération $(j+1)$ en fonction des estimations à l'itération j par :

$$\hat{\mu_i}(j+1) = \frac{\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\mu}(j))x_k}{\sum_{k=1}^{n} P(\omega_i|x_k;\hat{\mu}(j))}. \tag{11}$$

- **Etape 3** : incrémenter la valeur de j. Effectuer un test : par exemple, si cette valeur est inférieure à une valeur donnée, retourner à l'étape 2 ; sinon, fin de l'algorithme.

Ce type de schéma d'optimisation ne garantit que l'obtention d'un optimum local, et nécessite donc qu'on parte d'une bonne initialisation $\hat{\mu_i}(0)$ (proche de la solution) pour obtenir l'optimum global.

2.4 Synthèse

Malgré les difficultés présentées, le cas de la loi normale où seules les moyennes sont inconnues est de loin le plus simple. Si l'on suppose maintenant que les moyennes, les matrices de variance-covariance Σ_i et les probabilités a priori $P(\omega_i)$ sont inconnues, la méthode du maximum de vraisemblance donne des solutions singulières en général (on peut trouver des exemples pour lesquels la vraisemblance est arbitrairement grande). Des exemples peuvent être trouvés dans [DUDA-73]. Les solutions adoptées empiriquement consistent à rechercher de bonnes solutions en se restreignant aux maxima finis les plus grands de la fonction de vraisemblance. Il n'en reste pas moins que les calculs sont très lourds. Ils peuvent être toutefois simplifiés si la matrice de covariance est diagonale.

3 Méthode simple d'approximation

Devant les difficultés présentées dans la partie 2, des méthodes d'approximation ont été développées afin de simplifier les calculs. Remarquons d'abord que la probabilité a posteriori $P(\omega_i|x_k, \hat{\theta})$ est d'autant plus grande que la distance de Mahalanobis

$$(x_k - \hat{\mu_i})^t \hat{\Sigma_i}^{-1} (x_k - \hat{\mu_i})$$

est petite. Cela exprime en effet que l'échantillon x_k a une probabilité grande d'appartenir à la classe ω_i si la distance de x_k au centre de la classe (représenté par la moyenne μ_i), pondérée par la dispersion de la classe (représentée par Σ_i), est petite.

Une première approximation consiste à remplacer la distance de Mahalanobis par la distance euclidienne carrée $\|x_k - \hat{\mu_i}\|^2$, permettant de lever ainsi la difficulté de l'estimation des Σ_i. On cherche alors la moyenne $\hat{\mu_m}$ qui minimise $\|x_k - \hat{\mu_i}\|^2$.

Une deuxième approximation porte sur les probabilités a posteriori :

$$\hat{P}(\omega_i|x_k; \hat{\theta}) = \begin{cases} 1 \text{ si } i = m, \\ 0 \text{ sinon,} \end{cases} \tag{12}$$

revenant à binariser le processus : x_k est considéré comme appartenant de manière certaine à la classe ω_m, et n'appartenant (de façon certaine aussi) à aucune des autres classes.

Ces deux approximations conduisent alors à un schéma itératif très simple, où, à partir d'une estimation initiale des moyennes, on classe les échantillons en fonction de la moyenne la plus proche, puis on recalcule les moyennes selon la formule 11, où les probabilités sont simplifiées selon la deuxième approximation. On itère alors ce processus jusqu'à convergence. Cet algorithme se trouve parfois sous le nom de ISODATA de base.

On voit ici apparaître les méthodes non hiérarchiques du deuxième type, où on optimise un critère (ici, la distance au centre des classes) par des schémas itératifs. Les deux approximations effectuées font complètement disparaître les hypothèses sur les probabilités introduites au début de la partie 2, et permettent donc de faire le lien entre les deux types de méthodes.

4 Critères de classification

4.1 Choix des critères

Cette fois, le problème de la répartition de n échantillons x_k (d'un espace E) dans c classes ω_i s'exprime comme l'optimisation d'un certain critère. Les critères les plus souvent employés sont décrits dans cette partie.

On note m_i la moyenne des échantillons appartenant à la classe i :

$$m_i = \frac{1}{n_i} \sum_{x \in \omega_i} x,$$

avec $n_i = Card\{x | x \in \omega_i\}$. En posant

$$s_i = \frac{1}{n_i^2} \sum_{x \in \omega_i} \sum_{y \in \omega_i} \|x - y\|^2$$

et à l'aide de quelques manipulations simples, on montre alors les relations :

$$J_e = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{x \in \omega_i} \|x - m_i\|^2 \quad (13)$$

$$= \frac{1}{2} \sum_{i=1}^{c} n_i s_i \quad (14)$$

Le critère de l'erreur quadratique consiste à minimiser J_e, qui peut être vu comme l'écart entre les échantillons et les centres des classes auxquelles ils sont affectés.

Plus généralement, on peut remplacer la distance euclidienne carrée entre deux échantillons d'une même classe x et y par une fonction de similarité $s(x, y)$, dont la forme peut être adaptée au problème spécifique à traiter.

Les critères les plus populaires sont peut-être ceux portant sur la dispersion. En appelant toujours m_i la moyenne de la classe i et n_i sa population, la moyenne m des échantillons s'exprime sous la forme :

$$m = \frac{1}{n} \sum_{x \in E} x = \frac{1}{n} \sum_{i=1}^{c} n_i m_i.$$

La matrice de dispersion de la classe i s'écrit :

$$S_i = \sum_{x \in \omega_i} (x - m_i)(x - m_i)^t.$$

D'où la dispersion totale intra-classe :

$$S_W = \sum_{i=1}^{c} S_i$$

et on montre que

$$J_e \;=\; tr(S_W)$$

La dispersion inter-classes est définie par l'expression suivante :

$$S_B = \sum_{i=1}^{c} n_i (m_i - m)(m_i - m)^t, \tag{15}$$

qui exprime une distance entre classes.

La dispersion totale peut s'exprimer de deux manières équivalentes :

$$S_T = \sum_{x \in E} (x - m)(x - m)^t = S_W + S_B. \tag{16}$$

Cette identité (dont la démonstration est très simple et souvent connue sous le nom de théorème de Huyghens) est très importante. En effet, la première expression ne dépend que des données, et pas de la classification qui en est faite, montrant ainsi que la dispersion totale est constante. La deuxième forme dépend de la classification. Il est donc équivalent de minimiser la dispersion intra-classe ou de maximiser la dispersion inter-classes (puisque la somme des deux est constante pour un nombre de classes fixé). Ces deux optimisations expriment de manière mathématique le fait qu'on cherche à regrouper les échantillons en nuages qui soient espacés le plus possible les uns des autres, les points à l'intérieur de chaque nuage étant regroupés le plus possible.

Une fois choisi le critère de classification à optimiser, le problème consiste à trouver la répartition des échantillons en c classes qui satisfasse le mieux le critère. Une recherche exhaustive sur toutes les classifications possibles étant prohibitive, l'optimisation se fait de manière itérative, en vérifiant à chaque itération que le critère est amélioré. La démarche typique est celle qui consiste à déplacer un échantillon d'une classe vers une autre à condition que cette opération améliore le critère.

Prenons par exemple le critère de l'erreur quadratique. Si l'on change un échantillon x de la classe ω_i à la classe ω_j, les erreurs quadratiques J_i et J_j de ces deux classes deviennent :

$$J_j^* = J_j + \frac{n_j}{n_j + 1} \|x - m_j\|^2,$$

$$J_i^* = J_i - \frac{n_i}{n_i - 1} \|x - m_i\|^2.$$

Ainsi, le changement sera intéressant si le total des erreurs J_e a dimininué (si l'on a plus gagné sur J_i que perdu sur J_j), c'est-à-dire si :

$$J_j^* - J_j \leq J_i - J_i^*.$$

4.2 Exemple d'algorithme

Un algorithme possible de classification découlant de cette règle s'écrirait :

– **Etape 1** : Choix d'une partition initiale des n échantillons et calcul de J_e, et des centres de classes m_1, ..., m_c.

– **Etape 2** : Soit un échantillon x de la classe ω_i.
Si $n_i = 1$ aller en 5 (le nombre de classes est fixé donc on interdit à une classe de se vider complètement).
Sinon, calcul des variations ρ_j des J_j correspondant au transfert de x vers la classe ω_j :

$$\rho_j = \begin{cases} \frac{n_j}{n_j+1}\|x - m_j\|^2 \text{ si } j \neq i, \\ \frac{n_i}{n_i-1}\|x - m_i\|^2 \text{ si } j = i. \end{cases}$$

– **Etape 3** : Affecter x à ω_k si $\rho_k = \min_{j=1}^{c} \rho_j$.
– **Etape 4** : Mise à jour de J_e (seulement deux termes ont changé, de quantités calculées à l'étape 2, donc le calcul est très rapide), de m_i et m_k.
– **Etape 5** : Si J_e n'a pas changé en n essais (examen de tous les échantillons), fin. Sinon, retour à l'étape 2.

Cet algorithme a l'avantage d'être optimal à chaque étape. En revanche, l'optimalité globale n'est pas garantie. Le fait de changer un seul point à la fois (et pas forcément celui qui améliore le mieux le critère) rend l'algorithme sensible aux minima locaux. De plus, l'ordre d'examen des points a alors beaucoup d'importance (la convergence peut se faire vers une minimum local différent si on change cet ordre).

5 L'algorithme des K-moyennes

Un des algorithmes les plus utilisés, découlant directement de l'approche précédente est celui des k-moyennes (k-means). Il optimise le critère de l'erreur quadratique de manière itérative par les étapes suivantes (le nombre de classes est ici traditionnellement noté k, d'où le nom de k-moyennes).

– **Etape 1 : Initialisation.** Choix de centres initiaux $m_j(1)$ arbitraires (équidistribués, tirés au hasard, ou encore k échantillons choisis au hasard parmi les n).
– **Etape 2 : Affectation.** À l'itération i, x est affecté à ω_j si :

$$\|x - m_j(i)\| = \min_{l=1}^{k} \|x - m_l(i)\|. \tag{17}$$

Tous les échantillons sont classés selon cette règle (du centre le plus proche).
– **Etape 3 : Mise à jour des centres.** Calcul des nouveaux centres $m_j(i+1)$ pour minimiser l'erreur quadratique :

$$J_j = \sum_{x \in \omega_j} \|x - m_j(i+1)\|^2$$

En annulant la dérivée de cette expression par rapport à m_j, on obtient :

$$\frac{\partial J_j}{m_j} = -2 \sum_{x \in \omega_j} (x - m_j) = 0,$$

d'où la valeur optimale de m_j pour l'itération $(i+1)$:

$$m_j(i+1) = \frac{1}{n_j} \sum_{x \in \omega_j} x. \tag{18}$$

– **Etape 4 : Test de convergence.** Si $\forall j, m_j(i+1) = m_j(i)$, fin.
Sinon, retour à l'étape 2.

Il n'existe pas pour cet algorithme de preuve générale de convergence vers l'optimum global. Les expériences montrent que le choix des centres initiaux a une grande influence sur l'optimum trouvé. Cette fois, contrairement à l'algorithme présenté dans la partie 4, la mise à jour n'est effectuée qu'une fois tous les points examinés.

Le comportement de l'algorithme est influencé, outre par le choix des centres initiaux, par le nombre de classes et par les propriétés géométriques des données. La figure 1 illustre ces influences : une classification en deux classes de l'ensemble de points (représentés dans un espace à deux dimensions) conduit à des résultats différents suivant l'initialisation. En revanche, une classification en trois classes (intuitivement plus naturelle d'après la géométrie des données) ne pose pas de problèmes.

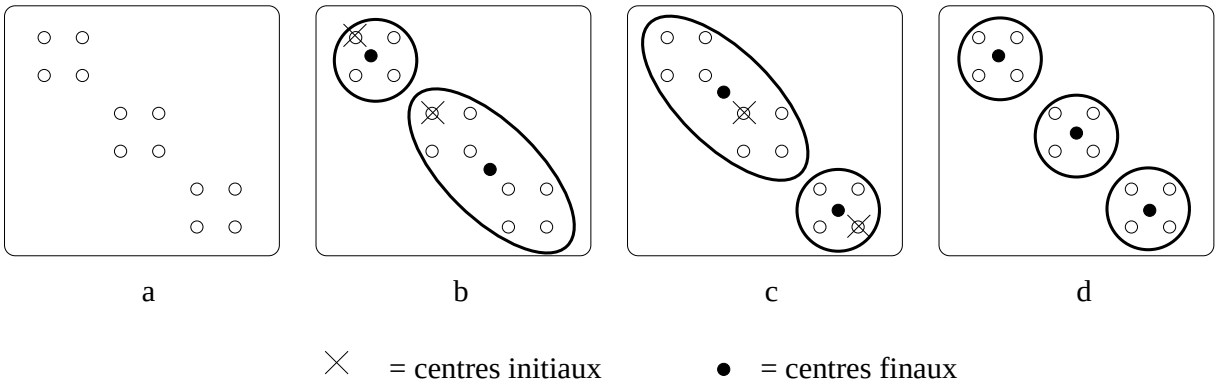


FIG. 1 – Influence des centres initiaux, du nombre de classes et de la géométrie des données dans l'algorithme des k-moyennes. a : ensemble de points à classifier, b et c : deux classifications en 2 classes, avec des centres initiaux différents, d : classification en trois classes.

La version présentée ici peut être rendue plus générale en remplaçant la norme euclidienne par une norme de type Mahalanobis par exemple, le critère à minimiser devenant alors :

$$\min \sum_{j=1}^{k} \sum_{x \in \omega_j} (x - m_j)^t S^{-1} (x - m_j),$$

où S est la matrice de covariance du nuage de points.

Notons que dans le cas des images, si la classification s'effectue sur le niveau de gris (les m_j sont alors les niveaux de gris moyens des classes), l'algorithme peut être exécuté de manière très rapide en travaillant uniquement sur l'histogramme de l'image. Cela permet de n'examiner qu'un nombre réduit d'échantillons : le nombre de niveaux de gris, plutôt que le nombre de points dans l'image. La méthode est souvent utilisée sous cette forme, comme initialisation de méthodes plus sophistiquées, prenant en compte plus de caractéristiques que le seul niveau de gris.

6 Algorithme ISODATA

Optimisant toujours le même critère, l'algorithme ISODATA est une version plus sophistiquée que celui des k-moyennes : il permet de faire varier le nombre de classes en cours d'algorithme par regroupement ou division, ces opérations étant effectuées alternativement, une itération sur deux. Les critères de regroupement et division dépendent eux des dispersions intra-classe et inter-classes. La contre-partie est l'introduction d'un nombre important de variables de contrôle de l'algorithme.

Les itérations suivent les étapes suivantes.

1. Définition des paramètres suivants :
   - nombre de classes voulues c,
   - nombre minimum d'échantillons par classe θ_N,
   - seuil de division des classes θ_S,
   - seuil de regroupement des classes θ_C,
   - nombre maximal de classes pouvant être regroupées L,
   - nombre maximal d'itérations I.

   Initialisation de n_c centres de classes m_1, ..., m_{n_c} (éventuellement, n_c peut être différent du nombre de classes voulues).
2. Classification des échantillons selon la même règle que pour les k-moyennes (équation 17).
3. Suppression des classes de moins de θ_N éléments. Les échantillons correspondants sont reclassés selon la règle 17 parmi les classes restantes.

   Mise à jour de n_c.
4. Calcul des nouveaux centres, comme moyenne de chaque classe (selon l'équation 18).
5. Calcul de la distance moyenne des échantillons ω_j au centre :
$$\bar{D_j} = \frac{1}{n_j} \sum_{x \in \omega_j} \|x - m_j\|.$$
6. Calcul de
$$\bar{D} = \frac{1}{n} \sum_{j=1}^{n_c} n_j \bar{D_j}.$$
7. Si $n_c \leq c/2$, aller à l'étape 8.

   Si l'itération courante est paire ou si $n_c \geq c/2$, aller à l'étape 11 (regroupement).

   Sinon, aller à l'étape 8 (division).
8. En appelant d la dimension de l'espace E des échantillons, calcul de
$$\sigma_j = (\sigma_{1j}, .., \sigma_{dj}),$$
   avec :
$$\sigma_{ij} = \sqrt{\frac{1}{n_j} \sum_{x_k \in \omega_j} (x_{ik} - m_{ij})^2},$$
   pour $i = 1, ..., d$ et $j = 1, ..., n_c$.

9. Calcul de la direction dans laquelle la variance est maximale :

$$\forall j = 1, ..., n_c, \quad \sigma_{mj} = \max_{i=1}^{d} \sigma_{ij}.$$

10. Si $\sigma_{mj} > \theta_S$ et :

$$\begin{cases} \bar{D}_j > \bar{D} \text{ et } n_j > 2(\theta_N + 1) \\ \text{ ou } n_c \leq c/2 \end{cases}$$

alors, division de ω_j en deux classes de centres m_j^+ et m_j^- tels que :

$$\begin{array}{rcl} m_{mj}^+ & = & m_{mj} + \lambda\sigma_{mj} \\ m_{mj}^- & = & m_{mj} - \lambda\sigma_{mj} \\ m_{ij}^+ & = & m_{ij}^- = m_{ij} \text{ pour } i \neq m, \end{array}$$

avec $0 < \lambda \leq 1$.

Si la séparation a lieu, aller à l'étape 2.

Sinon, aller à l'étape 11.

11. Calcul, pour tous i et j tels que $i < j$, de l'écart entre les centres des classes ω_i et ω_j :

$$D_{ij} = \|m_i - m_j\|.$$

12. Comparaison des D_{ij} à θ_C pour sélectionner les classes à regrouper.
13. Regroupement, par paires, des classes telles que $D_{ij} \leq \theta_C$ en commençant par les classes les plus proches l'une de l'autre. Limiter le nombre de regroupements à L.
14. Si la dernière itération est atteinte, fin.

    Sinon, aller à l'étape 1 si l'utilisateur veut modifier des paramètres, à l'étape 2 sinon.

    Incrémenter le nombre d'itérations.

7 L'algorithme des Boules optimisées

L'algorithme des boules optimisées est proche de celui des k-moyennes. Le critère de classification est identique (centre de classe le plus proche). La différence réside dans l'introduction d'une contrainte supplémentaire sur la forme des classes : chaque classe doit être incluse dans une boule de rayon R fixé. Le nombre de classes n'est en revanche pas toujours imposé.

Voici une version de l'algorithme où seul le rayon maximal R est imposé.

- **Etape 1** : Initialisation du nombre de classes $n_c = 1$ et du centre m_1 (on prend en général un des échantillons).
- **Etape 2** : L'affectation d'un échantillon x à la classe ω_i de centre m_i suit la règle 17, augmentée d'une contrainte sur la distance au centre :

$$\begin{cases} d(x, m_i) & = & \min_{j=1}^{n_c} d(x, m_j), \\ d(x, m_i) & \leq & R, \end{cases}$$

où d est une distance sur l'espace E des échantillons (le plus souvent, la distance euclidienne).
Si la condition sur le rayon n'est pas satisfaite, on crée une nouvelle classe ω_{n_c+1} de centre $m_{n_c+1} = x$.

- **Etape 3** : Mise à jour des centres des classes, selon l'équation 18.
- **Etape 4** : Test de convergence : si la partition ne change plus, fin.
  Sinon, aller à l'étape 2.

Cet algorithme est donc équivalent à celui des k-moyennes avec rejet , c'est-à-dire où la décision d'appartenance à une classe est rejetée si le point est trop loin du centre de la classe, ce point donnant alors lieu à la création d'une nouvelle classe.

Des variantes de l'algorithme imposent un nombre maximum d'itérations, ou encore un nombre maximum de classes, les points ne répondant pas aux contraintes étant alors rejetés (sans reclassification).

8 L'algorithme des Nuées dynamiques

La méthode des nuées dynamiques généralise les algorithmes précédents en remplaçant la mesure de distance entre un échantillon x et le centre m_i d'une classe ω_i par une mesure plus générale de dissemblance entre l'échantillon et la classe, notée dans la suite $f(x, \omega_i)$.

Le nombre de classes c étant fixé, une partition $P = \{\omega_1, ..., \omega_c\}$ est la meilleure classification de l'ensemble des échantillons en c classes si le critère suivant est minimal :

$$\sum_{i=1}^{c} \sum_{x \in \omega_i} f(x, \omega_i), \tag{19}$$

qui est bien une généralisation des critères vus dans la partie 4, en particulier de celui utilisé dans les k-moyennes.

Ici encore, le minimum global est souvent difficile à trouver et les algorithmes itératifs utilisés recherchent des minima locaux. Ces algorithmes suivent exactement le même schéma que celui des k-moyennes où seule la règle d'affectation 17 est remplacée par :

$$x \in \omega_i(k+1) \quad \Leftrightarrow \quad f(x, \omega_i(k)) = \min_{j=1}^{c} f(x, \omega_j(k)), \tag{20}$$

où $\omega_i(k)$ désigne la classe i à l'itération k. Nous ne détaillerons donc pas plus ces algorithmes. Notons que leur convergence (vers un minimum local) n'est démontrable que pour certaines fonctions f.

Toute la difficulté de la méthode réside dans le choix d'une fonction de dissemblance f adaptée au problème. Nous décrivons ici la méthode de Diday. Elle consiste à représenter une classe par un noyau , généralisant la notion de centre utilisée dans les parties précédentes. La fonction f est alors une mesure de dissemblance entre l'échantillon et le noyau de la classe. Donnons quelques exemples.

1. Le cas le plus simple est celui où le noyau est simplement le centre de gravité de la classe, donc exactement le centre utilisé précédemment. La fonction de dissem-

blance est alors la distance entre l'échantillon et le centre. Ce cas particulier des nuées dynamiques coïncide donc exactement avec les k-moyennes, et on retrouve les mêmes propriétés de convergence vers un minimum local.

2. Le noyau d'une classe ω peut être constitué de plusieurs points $(y_1, ..., y_p)$ au lieu d'un seul. Ils sont par exemple choisis de telle sorte que la fonction suivante soit minimale :

$$\sum_{x \in \omega} \inf_{k=1}^{p} d(x, y_k). \tag{21}$$

La fonction de dissemblance est alors la moyenne des distances de x à chacun des points du noyau de la classe :

$$f(x, \omega) = \frac{1}{p} \sum_{k=1}^{p} d(x, y_k). \tag{22}$$

Avec un seul centre, on obtient toujours des classes compactes où les points sont regroupés autour du centre. En revanche, ici une classe est représentée par plusieurs points, et la méthode est donc mieux adaptée aux formes complexes. On peut montrer la convergence de l'algorithme vers un minimum local.

3. Au lieu de prendre des points pour constituer le noyau, on peut prendre des objets géométriques plus compliqués. Par exemple, pour la reconnaissance de caractères majuscules, une bonne représentation peut être obtenue en prenant pour noyau d'une classe son axe principal d'inertie Δ_ω. La fonction de dissemblance s'exprime alors sous la forme :

$$f(x, \omega) = d(x, \Delta_\omega) = \inf_{y \in \Delta_\omega} d(x, y). \tag{23}$$

9 L'algorithme des "Fuzzy C-Means"

L'idée directrice de l'algorithme des "Fuzzy C-Means" est de considérer que la notion d'appartenance à une seule et unique classe parmi C classes peut s'avérer trop restrictif : la binarisation effectuée au paragraphe 3 et le système d'équations 17 peuvent en effet apparaître trop sélectifs. Plutôt que de fournir des appartenance en "tout ou rien" (on appartient à une classe ou on n'y appartient pas), on recherche des μ_{ik} représentant le degré d'appartenance de l'individu i à chaque classe k et pour cela on s'appuie sur le formalisme de la logique floue, dont on rappelle ici les bases.

Une grandeur μ_k est un degré d'appartenance si :
- $\mu_k \in [0, 1] \ \forall k$
- $\sum_{k=1}^{C} \mu_k = 1$.

Dans l'algorithme des Fuzzy C-Means, on prend la distance euclidienne entre une forme et les prototypes des classes et on définit la fonctionelle J avec un paramètre supplémentaire m :

$$J_m(B, U, X) \ = \ \sum_{i=1}^{C} \sum_{k=1}^{n} \mu_{ik}^m \, ||x_k - b_i||^2$$

Ce paramètre $m \in]1, \infty[$ est le "facteur de flou".

La fonction d'appartenance est alors donnée par l'expression :

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{C} \left(\frac{||x_k - b_i||}{||x_k - b_j||} \right)^{\frac{2}{m-1}}}$$

Le prototype est alors donné par la relation :

$$b_j = \frac{\sum_{i=1}^{N} \mu_{ij}^m \, x_i}{\sum_{i=1}^{N} \mu_{ij}^m}$$

La matrice de dispersion de la classe j s'écrit :

$$M_j = \sum_{i=1}^{N} \mu_{ij} \, (x_i - b_j)(x_i - b_j)^t$$

ce qui permet d'écrire la matrice globale M

$$M = \sum_{j=1}^{c} M_j$$

Il est facile de vérifier que le critère J_m est la trace de cette matrice.

L'algorithme des Fuzzy C-Means présente néanmoins quelques difficultés. En effet, le degré de dépendance d'un point à une classe dépend des autres classes et les performances de l'algorithme seront variables selon que les individus sont déjà naturellement "regroupés" dans leur classe, ou qu'il existe des individus dont on ne sait rien a priori et qui correspondent de facto à du bruit : dans ce dernier cas, ils influenceront notablement l'estimation du prototype.

Aussi d'autres méthodes, de type possibiliste, ont été proposées pour prendre en compte la représentativité effective d'un point vis à vis de sa classe sans subir outre mesure l'influence des autres classes

10 Limites

Si les méthodes présentées dans cette partie ont l'avantage d'être non supervisées, simples, rapides, elles souffrent d'un certain nombre d'inconvénients qui sont rappelés succinctement ici.

Elles nécessitent d'abord une connaissance sur les données et la classification que l'on souhaite en obtenir. Cette connaissance est, pour la plupart des méthodes, le nombre de classes. S'il est raisonnable dans beaucoup d'applications de supposer que ce nombre est connu, cette hypothèse ne permet pas en revanche de traiter de manière satisfaisante et automatique des problèmes pour lesquels des échantillons pathologiques sont susceptibles d'apparaître, et qui formeraient des classes supplémentaires.

Les méthodes n'utilisant pas le nombre de classes connu a priori remplacent cette connaissance par une autre, par exemple sur la géométrie des données. C'est le cas du rayon dans la méthode des boules optimisées. C'est aussi celui des paramètres de contrôle dans l'algorithme ISODATA, nécessitant d'avoir une idée des dispersions inter-classes et intra-classe.

Tous les algorithmes présentés sont itératifs et nécessitent une initialisation. Le problème, classique en optimisation, des minima locaux se pose ici de manière cruciale. Plusieurs solutions sont envisageables :

- on peut trouver facilement une initialisation proche de l'optimum (c'est le cas par exemple de la classification d'une image scanner à partir des niveaux de gris, où l'on connaît a priori le niveau de gris des os, celui des tissus mous, etc.) ;
- on exécute l'algorithme à partir de plusieurs positions de départ et on choisit la meilleure solution ;
- on exécute l'algorithme un grand nombre de fois, pour des initialisations tirées au hasard, et on recherche les solutions stables.

L'espace E dans lequel sont définis les échantillons est un espace de caractéristiques qui peuvent être de natures et de dimensions très différentes. Si l'on cherche par exemple à classer un certain nombre d'individus et que les caractéristiques retenues sont la taille de l'appartement qu'ils occupent, leur consommation mensuelle de lait et leur nombre d'enfants, l'espace E, tridimensionnel ici, a des axes qui représentent des variables très hétérogènes. Il est alors délicat de définir une métrique sur un tel espace. La solution classiquement adoptée consiste à utiliser une distance de Mahalanobis, permettant de tenir compte des échelles très différentes sur chacun des axes. Cependant, l'idée même de construire une métrique dans de telles conditions est criticable. La solution qui consisterait à traiter chaque dimension séparément puis à les combiner fait appel à des notions de fusion de données qui sortent du cadre de ce cours.

Références

[DUBU-90] **B. Dubuisson :** *Diagnostic et reconnaissance de formes,* Hermès, Paris, 1990.

[DUDA-73] **R. Duda, P. Hart :** *Pattern Classification and Scene Analysis,* New-York,Wiley, 1973 (Chapitre 6).

[GAIL-83] **G. Gaillat :** *Méthodes statistiques de reconnaissance de formes,* Coll. ENSTA, Paris, 1983.

CHAPITRE 5 : LIENS AVEC LES METHODES HIERARCHIQUES

Isabelle Bloch, Jean-Marie Nicolas

1 Introduction

Après quelques brefs rappels sur les graphes et les arbres, nous introduirons dans ce chapitre la notion de distance entre objets et ensemble, puis celle d'ultra-métrique, enfin celle de distance de chaîne comme cas particulier particulièrement intéressant pour la classification.

L'objectif de ce chapitre est de montrer comment une partition (donc une classification) peut être obtenue à partir d'une ultra-métrique (ou de manière équivalente, par une structure d'"arbre de longueur minimale" ou de "hiérarchie indicée"). Cette approche est donc diamétralement opposée à celle du chapitre précédent où on introduisait des critères globaux (qu'il fallait optimiser) pour classifier les individus dans des classes de préférence "compactes". Il faut bien souligner que les méthodes hiérarchiques ont comme point de départ les individus et visent à des regroupements faisant essentiellement intervenir des ressemblances entre individus. C'est cette démarche qui conduira à des partitions dont le nombre (c'est à dire le nombre de classes du chapitre précédent) n'est absolument pas un prérequis.

2 Rappels sur les graphes et les arbres

2.1 Graphes

Définition 1 *On dit que $G = [X, U]$ est un graphe s'il est déterminé par :*

- *un ensemble X de sommets : les* **nœuds**
- *un ensemble U dont les éléments sont des couples ordonnés : les* **arcs**.

Les arcs sont définis par leurs nœuds d'extrémité : $u = (i, j) \in U$ est un arc orienté d'extrémité initiale i et d'extrémité finale j, $i \in X$, $j \in X$. On dira aussi que i est le prédécesseur de j et que j est le successeur de i.

On peut alors poser les définitions suivantes :

- un **p-graphe** est un graphe tel qu'il y ait au plus p arcs de la forme (i, j) entre sommets.
- on peut souhaiter ignorer l'orientation des arcs : on aura alors un **graphe non orienté**, et l'on parlera alors d'**arêtes** (plutôt que d'arcs non orientés).

- un arc $u = (i, i)$, i.e. dont les extrémités coïncident, s'appelle une **boucle**.

Une **chaîne de longueur** q est une séquence L de q arcs : $L = \{u_1, \ldots, u_q\}$ telle qu'une extrémité de l'arc $u_i, i \in [1, q-1]$ coïncide avec une extrémité de l'arc u_{i+1}.

Un **cycle** est une chaîne dont les extrémités coïncident

- Un graphe non orienté est un **graphe simple** si c'est un 1-graphe sans boucle.

degré du sommet i $d(i)$: c'est le nombre d'arcs ayant le sommet i comme extrémité (initiale ou finale).

2.2 Arbre

Définition 2 *Un arbre $T = [S, A]$ est un graphe non orienté répondant à l'une quelconque des définitions suivantes :*
- il est connexe, mais perd cette propriété dès qu'on lui enlève un arc quelconque
- il est connexe et possède n *sommets et* (n-1) *arcs*
- il existe un chemin et un seul entre tout couple de sommets différents
- il possède n *sommets,* (n-1) *arcs et aucun cycle.*

Notons que cette définition des arbres ne nécessite aucune notion de filiation et que, sans information supplémentaire, on ne peut désigner une quelconque relation d'ordre.

Définition 3 *Un arbre planté est un ensemble fini de nœuds vérifiant les deux propriétés suivantes :*
- il y a un nœud privilégié : la racine
- les nœuds restants définissent eux-mêmes des arbres plantés, et chaque sommet de ces nouveaux arbres est relié au sommet initial.

On peut alors donner les définitions suivantes :

- Un nœud F est le **fils** d'un nœud P lorsque F est la racine d'un sous-arbre de P.
- Si F est le fils d'un nœud P, on dira aussi que P est **père** de F
- Si un nœud est au **niveau** n, ses fils sont au niveau $n + 1$
- Par définition, le niveau de la racine d'un arbre est égal à 1
- Des nœuds sont **frères** s'ils sont fils du même père.
- Un arbre $T = [S, A]$ est **étiqueté** s'il existe une application f de X, ensemble fini, dans S. On notera cet arbre étiqueté (T, X, f).
- Un arbre peut être **valué**. Dans ce cas, on définit sa longueur L comme la somme des longueurs des arêtes qui le composent.
- L'**ordre d'ainesse**, $<$, s'applique entre les nœuds s et s' : $s < s'$ si et seulement si s' se trouve sur le chemin entre s et la racine.

- Une **feuille** est un nœud (non racine) de degré 1. Une feuille n'a pas de fils.

2.3 X-arbres et dendogrammes

On voit donc que les étiquettes de tout arbre planté –hormis la racine– correspondent soit à des feuilles, appelées aussi sommets "réels", soit à des sommets "latents" qui sont soit de degré 2 (on parlera alors de sommet "inutile", bien que ce concept de sommet latent de degré 2 ait son importance en classification), soit de degré ≥ 3 et correspondant à des branchements. Ces considérations conduisent à la notion de X-arbre :

Définition 4 *Un X-arbre est un arbre étiqueté* $([S, A], X, f)$ *tel que pour tout* $v \in S - f(X)$, $d(v) \geq 3$.

On voit donc qu'un X-arbre n'a pas d'étiquettes sur les branchements. On parlera d'**X-arbre libre** si seules les feuilles sont étiquetées. Un X-arbre est **séparé** s'il n'y a pas d'étiquetage multiple.

Définition 5 *Un X-arbre hiérarchique est un X-arbre planté libre (donc sans sommet "inutile").*

Schématiquement, un X-arbre hiérarchique se décrit par une racine, des feuilles (confondues avec leurs étiquettes) et des nœuds de branchement (i.e. de degré ≥ 3).

Nous arrivons maintenant à une définition importante pour la suite de ce chapitre.

Définition 6 *Un dendogramme est un arbre hiérarchique valué tel que tous les chemins des feuilles à la racine ont même longueur.*

Cette notion de dendogramme a d'autant plus d'importance que nous admettrons qu'elle est équivalente à d'autres notions que nous verrons dans la suite de ce chapitre : la notion d'ultramétrique et celle de hiérarchie indicée. Même si les options choisies au départ sont différentes (prendre le problème selon la théorie des graphes, ou celle des distances ultramétriques), les résultats finaux, qui sont utilisés en classification, seront identiques. Plus de détails sur ces équivalences peuvent se trouver dans [BART-88].

3 Classification hiérarchique

Nous avons vu au chapitre précédent qu'il est possible de regrouper automatiquement n individus en classes sous certaines hypothèses : avoir une idée du nombre de classes, définir un critère pour qualifier une classe vis à vis des autres classes. Une toute autre approche consiste à rechercher un partionnement des individus conduisant à un cheminement hiérarchique : on a au début chaque individu "isolé" (on a donc n partitions de l'espace) , et à la fin, une structure englobant tous les individus (*i.e.* une partition unique regroupant tous les individus).

Les problèmes levés par cette approche reposent sur les difficultés suivantes :

- Il est nécessaire de définir une "mesure" (qui n'est pas nécessairement une distance) entre individus. Cette mesure peut s'obtenir soit à partir de notions objectives (comme des données en sortie de capteur,...), soit éventuellement sur des bases subjectives (notes données par des opérateurs humains).
- Il est nécessaire de se doter d'une mesure entre un individu et une partition.

Une fois ces mesures définies, il est alors possible de définir une méthode de classification ascendante hiérarchique, conduisant donc à une partition des individus selon une **hiérarchie**, terme que nous redéfinirons, de manière plus stricte, ultérieurement (paragraphe 3.4).

3.1 Ecart et distance entre objets

3.1.1 Définitions

Soit x et y deux individus. On souhaite avoir une grandeur (une note, un indicateur, ...) dans $\mathbb{R}^+$ permettant de savoir si ces individus se ressemblent ou non. Intuitivement, on souhaite que cette grandeur soit nulle si les deux individus sont indiscernables, et soit très grande si les individus sont fortement dissemblables.

C'est dans cet objectif que l'on définit la notion d'écart.

Définition 7 *On dit que la grandeur $d(x, y)$ entre deux objets est un écart si :*

$$\begin{aligned} d(x,y) &= d(y,x) \\ d(x,y) = 0 &\Leftrightarrow x \equiv y \end{aligned}$$

On passe éventuellement de la notion d'écart à la notion de distance en introduisant l'inégalité triangulaire :
L'écart $d(x, y)$ est une distance si il vérifie :

$$d(x,y) \leq d(x,z) + d(z,y)$$

3.1.2 Exemple

Un exemple célèbre (tiré de [BART-88]) consiste à définir la matrice d'écart entre un certain nombre de mammifères. La matrice ainsi fournie donne les écarts $d(i, j)$ entre les mammifères i et j (matrice symétrique par définition : on n'en remplit que la moitié).

Pour obtenir cette matrice d'écart, un certain nombre de personnes ont tenté de construire cette matrice selon leurs critères propres : un simple moyennage a donné la matrice fournie en exemple.

	B	C	D	E	F	G	H	I	J	K	L
A : Ours	47	28	40	50	19	29	23	29	21	20	16
B : Chat		31	56	2	29	25	24	25	43	41	47
C : Vache			44	30	11	8	25	34	17	28	8
D : Daim				51	45	43	45	40	41	20	53
E : Chien					17	24	27	27	45	40	47
F : Chèvre						7	23	40	20	22	2
G : Cheval							29	33	26	30	15
H : Lion								33	29	33	35
I : Souris									35	23	52
J : Cochon										26	20
K : Lapin											33
L : Mouton											

3.2 Ecart et distance entre un objet et un groupe d'objets

Si, avec n individus isolés, une matrice d'écart se construit sans problème (autre que celui de sa construction et qui peut bien entendu être remis en question), il n'en est pas de même dès lors que l'on regroupe deux individus en un seul, et que l'on souhaite alors avoir une nouvelle matrice d'écart avec $n-1$ individus. En effet, si la majorité des valeurs restent les mêmes, on ne sait comment attribuer une valeur d'écart entre les individus restant et le nouvel individu construit à partir du regroupement de deux d'entre eux.

Aussi plusieurs approches existent, puisqu'il suffit que la définition d'un écart soit vérifiée (ce ne serait pas la même chose dans le cas des distances, puisqu'il faudrait conserver l'inégalité triangulaire).

Soit x un objet, et $h = (y, z)$ un regroupement de deux individus. L'écart d_g à définir peut être choisi parmi les techniques suivantes :

le saut minimal $d_g(h, x) = Min\,(d(x, z), d(x, y))$

le saut maximal $d_g(h, x) = Max\,(d(x, z), d(x, y))$

la distance moyenne $d_g(h, x) = \frac{d(x,z)+d(x,y)}{2}$

Notons que la distance moyenne revient à créer un nouvel individu, moyenne des individus x et y. Remarquons aussi que cette liste n'est en rien limitative et que l'on peut tout à fait introduire une distance (ou un écart) sur des critères spécifiques à l'application.

Pour être complet, il faut se poser le problème de la définition de l'écart entre groupe d'objets : en fait, il faut les voir comme des individus (*i.e.* on ne différencie pas dans la matrice d'écart individus et regroupement d'individus) et la méthode utilisée pour les regroupements d'individus s'applique.

Grâce à cette approche, on dispose d'une méthode pour calculer une matrice d'écart avec, indifféremment, un écart entre individus, un écart entre individu et groupe, et un écart entre groupes d'individus.

3.3 Classification ascendante hiérarchique

3.3.1 Définition

Soit n individus. Lorsque l'on connait une matrice d'écart (construite à partir de l'écart d) et une définition pour calculer l'écart d_g entre un regroupement d'individus et un individu, on peut alors effectuer une Classification ascendante hiérarchique selon l'algorithme suivant en $n-1$ étapes :

Etape 1 On cherche les deux éléments d'écart le plus faible

Etape 2 On les agrège

Etape 3 On recalcule la matrice d'écart (utilisation de d et d_g)

Etape 4 Si l'on dispose toujours de $i > 1$ individus, on retourne à l'étape 1

On voit qu'ainsi on dispose de n possibilités de classification (en terme de regroupement entre individus ayant un écart le plus faible possible), et que ces possibilités contiennent entre n classes (chaque individu pris séparément est une classe en soi) et 1 classe (regroupement de tous les individus).

3.3.2 Exemple

On va ne retenir dans cet exemple que 5 individus de l'exemple vu au paragraphe 3.1.2.

	A	B	C	D	E
A : Ours	0	47	28	40	50
B : Chat		0	31	56	2
C : Vache			0	44	30
D : Daim				0	51
E : Chien					0

Pour la méthode de regroupement, on prend dans un premier temps le saut minimal.

	A	B+E	C	D
A : Ours	0	47	28	40
B+E : Chat+Chien		0	30	51
C : Vache			0	44
D : Daim				0

	A+C	B+E	D
A+C : Ours+Vache	0	30	40
B+E : Chat+Chien		0	51
D : Daim			0

	A+B+C+E	D
A+B+C+E : Ours+Chat+Vache+Chien	0	40
D : Daim		0

On prend maintenant le saut maximal.

	A	B+E	C	D
A : Ours	0	50	<u>28</u>	40
B+E : Chat+Chien		0	31	56
C : Vache			0	44
D : Daim				0

	A+C	B+E	D
A+C : Ours+Vache	0	50	<u>44</u>
B+E : Chat+Chien		0	56
D : Daim			0

	A+C+D	B+E
A+C+D : Ours+Vache+Daim	0	56
B+C+E : Chat+Chien		0

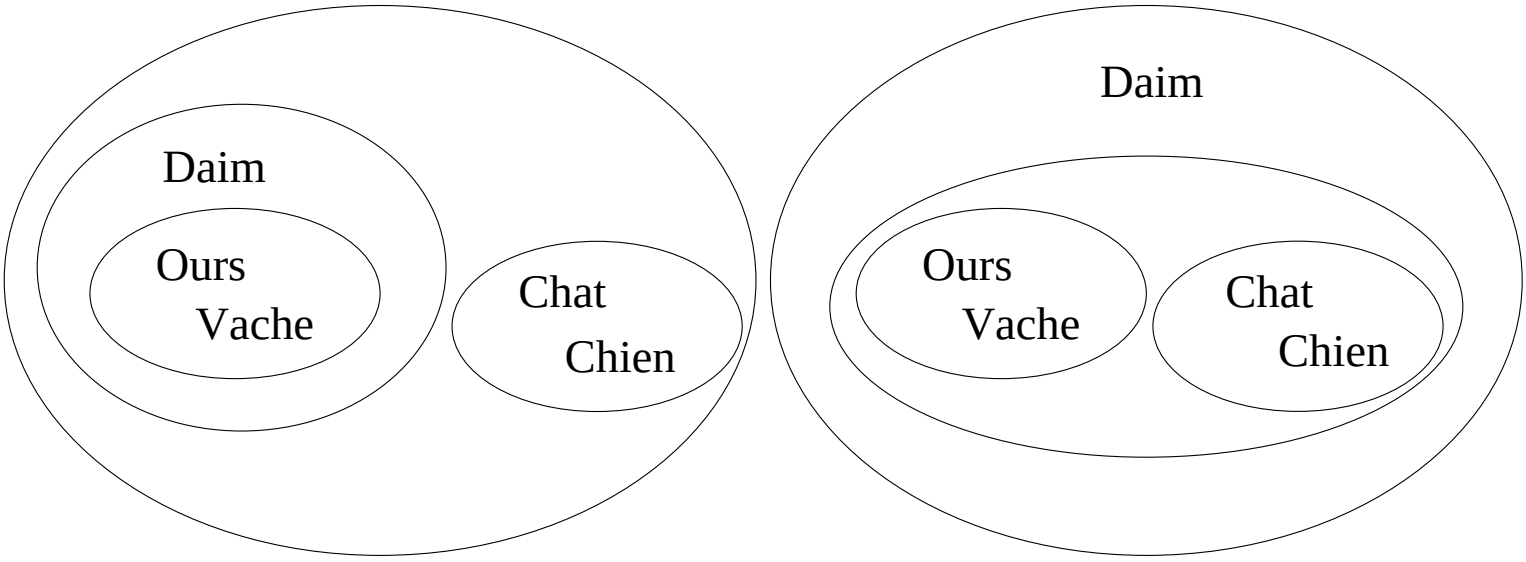


FIG. 1 – Classification hiérarchique : par saut maximal (à gauche) et par saut minimal (à droite)

On peut remarquer qu'à chaque regroupement correspond une valeur, croissant d'une étape à l'autre : nous verrons que cette observation permet d'affecter un indice à la hiérarchie ainsi construite. Nous allons maintenant mieux formaliser cette observation en introduisant de manière plus formelle les notions de hiérarchie et de hiérarchie indicée.

3.3.3 Les effets du choix des écarts

Dans cet algorithme, un des éléments essentiels est le choix de la technique de regroupement. On peut observer les conséquences suivantes de ce choix :

- si l'on choisit la technique du saut minimum, on risque de voir apparaître un effet de chaînage. Un individu en particulier aura donc des "voisins" qui lui ressemblent, mais, de proche en proche, la disparité peut s'accroître et conduire à des regroupements peu convaincants.
- si l'on choisit la technique du saut maximum, les regroupements obtenus seront compacts : on retrouvera alors une classification proche de celles obtenues par les méthodes automatiques non hiérarchiques du chapitre précédent.

3.4 Hiérarchie et hiérarchie indicée

On se place toujours dans le cas où E est fini, et on note $\mathcal{P}(E)$ l'ensemble des parties de E. On appelle hiérarchie sur E un sous-ensemble $\mathcal{H}$ de $\mathcal{P}(E)$ qui vérifie les propriétés suivantes :

$$E \in \mathcal{H}, \tag{1}$$

$$\forall x \in E, \quad \{x\} \in \mathcal{H}, \tag{2}$$

$$\forall (h, h') \in \mathcal{H}^2, \quad \begin{cases} h \cap h' = \emptyset, \\ \text{ou } h \subset h', \\ \text{ou } h' \subset h. \end{cases} \tag{3}$$

On peut remarquer que les partitions introduites dans le paragraphe précédent sont des hiérarchies.

Introduisons maintenant la notion de hiérarchie indicée : c'est un couple $(\mathcal{H}, f)$ tel que $\mathcal{H}$ est une hiérarchie et f est une application de $\mathcal{H}$ dans $\mathbb{R}^+$ telle que :

$$\forall x \in E, \quad f(\{x\}) = 0, \tag{4}$$

$$\forall (h, h') \in \mathcal{H}^2, h \subset h', \quad h \neq h' \quad \Rightarrow \quad f(h) < f(h'). \tag{5}$$

Les partitions introduites dans le paragraphe précédent sont des hiérarchies indicées : à chaque étape, on peut prendre comme indice la valeur minimale de l'écart ayant permis le regroupement des individus. Sur ces arbres de classification, les étiquettes sont les noms des individus initiaux.

4 Approche par la distance ultramétrique

Les méthodes hiérarchiques consistent donc à effectuer plusieurs classifications emboîtées les unes dans les autres, permettant de choisir ensuite celle qui correspond au niveau de finesse désiré. L'application la plus évidente en est la taxinomie arborescente.

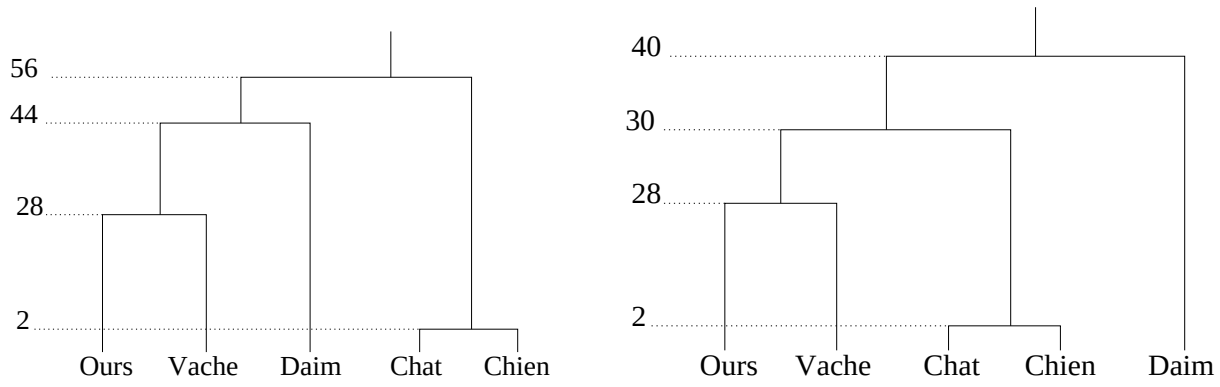


FIG. 2 – Classification hiérarchique : à gauche par saut maximal (cf figure 1 gauche), à droite par saut minimal(cf figure 1 droite).

En fait, ces méthodes reposent aussi sur l'équivalence entre une hiérarchie indicée sur E (espace des échantillons) et une distance ultra-métrique sur E. Les parties suivantes sont consacrées à l'introduction de ce concept d'ultramétrique.

4.1 Distance ultra-métrique

Une distance ultra-métrique sur E est une application δ de $E \times E$ dans $\mathbb{R}^+$ telle que :

$$\forall (x_1, x_2) \in E^2, \quad \delta(x_1, x_2) = 0 \Leftrightarrow x_1 = x_2, \tag{6}$$

$$\forall (x_1, x_2) \in E^2, \quad \delta(x_1, x_2) = \delta(x_2, x_1), \tag{7}$$

$$\forall (x_1, x_2, x_3) \in E^3, \quad \delta(x_1, x_2) \leq \sup[\delta(x_1, x_3), \delta(x_3, x_2)]. \tag{8}$$

Les conditions 6 et 7 sont équivalentes aux conditions de séparation et de symétrie respectivement d'une distance classique. En revanche, la troisième condition (équation 8) est plus forte que l'inégalité triangulaire classique qu'elle entraîne. Ainsi, toute ultra-métrique est une distance.

Une des propriétés des ultra-métriques est que tout triangle est isocèle (la démonstration est immédiate à partir de l'équation 8 exprimée pour toutes les arêtes du triangle).

4.2 Distance de chaîne

La distance de chaîne permet de définir une distance entre échantillons en termes de chemin. Elle correspond à l'idée intuitive de distance entre deux cailloux dans une rivière, où l'on cherche à aller d'un cailloux à l'autre (sans mettre le pied dans l'eau) en minimisant le pas le plus grand que l'on doit faire entre deux cailloux successifs. De manière mathématique, quelques définitions sont nécessaires.

Soit une distance d sur E (toujours l'ensemble des échantillons). On appelle chemin de x_1 à x_2 la suite d'échantillons $c = (c_1, c_2, ..., c_n)$ avec :

$$\forall i, \quad c_i \in E,$$

$$c_1 = x_1,$$

$$c_n = x_2.$$

Le pas du chemin est défini comme la quantité :

$$P(c) = \sup_{i=1}^{n-1} d(c_i, c_{i+1}). \tag{9}$$

La distance de chaîne est alors définie sur $E \times E$ par :

$$\delta(x_1, x_2) = \inf_{c \in CH(x_1,x_2)} P(c), \tag{10}$$

où $CH(x_1, x_2)$ représente l'ensemble de tous les chemins de x_1 à x_2 dans E.

On a alors le résultat important suivant : la distance de chaîne δ est une ultra-métrique.

La démonstration des conditions 6 et 7 est immédiate. Pour la condition 8, on a :

$$\begin{array}{rcl}
\delta(x_1, x_2) & = & \inf_{c \in CH(x_1,x_2)} P(c) \\
& \leq & \inf_{c \text{ passant par } x_3} P(c) \\
& \leq & \inf_{c' \in CH(x_1,x_3)} \inf_{c" \in CH(x_3,x_2)} \sup[P(c'), P(c")] \\
& \leq & \inf_{c' \in CH(x_1,x_3)} [\sup[P(c'), \inf_{c" \in CH(x_3,x_2)} P(c")]] \\
& \leq & \sup[\inf_{c' \in CH(x_1,x_3)} P(c'), \inf_{c" \in CH(x_3,x_2)} P(c")] \\
& \leq & \sup[\delta(x_1, x_3), \delta(x_3, x_2)].
\end{array}$$

Définissons un ordre sur les distances de la manière suivante : pour deux distances d_1 et d_2 sur E,

$$d_1 \leq d_2 \quad \Leftrightarrow \quad \forall (x, y) \in E^2,\ d_1(x, y) \leq d_2(x, y). \tag{11}$$

On peut montrer, pour cet ordre, que pour toute ultra-métrique Δ, on a la relation suivante avec la distance initiale sur E et la distance de chaîne δ (on a $\delta \leq d$) :

$$\Delta \leq d \quad \Rightarrow \quad \Delta \leq \delta. \tag{12}$$

Cette propriété exprime que la distance de chaîne est l'ultra-métrique ¡¡ sous-dominante ¿¿, c'est-à-dire qu'il n'existe pas d'ultra-métrique inférieure à d plus proche de d que la distance de chaîne. Notons qu'il n'existe pas de notion équivalente d'ultra-métrique sur-dominante. C'est en grande partie pour cette propriété, et bien sûr pour son interprétation concrète (voir l'exemple des cailloux), que la distance de chaîne est l'ultra-métrique la plus utilisée.

Pour la démonstration de cette propriété, considérons le chemin c sur lequel est atteint la distance de chaîne (minimum dans l'équation 10) :

$$\delta(x, y) = P(c) = \sup_{i=1}^{n-1} d(c_i, c_{i+1}),$$

avec $c = (c_1, ..., c_n)$, $c_1 = x$, $c_n = y$.

On a alors pour une ultra-métrique Δ quelconque :

$$\begin{array}{rcl} \Delta \leq d & \Rightarrow & \forall i,\ \Delta(c_i, c_{i+1}) \leq d(c_i, c_{i+1}) \\ & \Rightarrow & \sup_{i=1}^{n-1} \Delta(c_i, c_{i+1}) \leq \sup_{i=1}^{n-1} d(c_i, c_{i+1}) = \delta(x, y). \end{array}$$

Comme Δ est une ultra-métrique, on a :

$$\Delta(x, y) = \Delta(c_1, c_n) \leq \sup_{i=1}^{n-1} \Delta(c_i, c_{i+1}),$$

et donc $\Delta(x, y) \leq \delta(x, y)$, ce qui complète la démonstration.

4.3 Relation d'équivalence et partition à partir d'une ultra-métrique

Soit δ une ultra-métrique. La relation

$$\forall \delta_0 \in \mathbb{R}^+,\ \ x R_{\delta_0} y \ \ \Leftrightarrow \ \ \delta(x, y) \leq \delta_0 \tag{13}$$

définit une relation d'équivalence (la démonstration est immédiate à partir de la définition d'une ultra-métrique). La partition formée des classes d'équivalence de R_{δ_0} est notée $P(\delta_0)$.

Il découle de cette propriété l'application à la classification d'un ensemble d'échantillons à partir de la distance de chaîne et d'un seuil δ_0 ; $P(\delta_0)$ est alors simplement l'ensemble des classes obtenues. Une telle classification a les propriétés suivantes :

1. on peut passer d'un point à un autre d'une même classe, en restant dans la classe et en ne faisant jamais de pas de longueur supérieure à δ_0 ; cette propriété est proche de la notion de dispersion intra-classe présentée dans le chapitre 4 (la différence essentielle est que l'on peut avoir ici des classes très allongées à condition qu'il y ait des échantillons tout le long de la classe) ;
2. pour passer d'un point d'une classe à un point d'une autre classe, il faut faire au moins un saut de longueur supérieure à δ_0 ; cette propriété est donc proche de la notion de dispersion inter-classes.

4.4 Hiérarchie et ultra-métrique

La partition introduite dans la partie 4.3 à partir d'une ultra-métrique induit une hiérarchie sur E : $\mathcal{H} = \bigcup_{\delta_0 \in \mathbb{R}^+} P(\delta_0)$ est une hiérarchie.

La démonstration de la propriété 1 découle du fait que E est fini et donc la plus grande distance entre deux éléments existe. La partition associée à cette distance est réduite à la classe E. La propriété 2 est vérifiée : $\{x\} = P(0)$. Pour 3, considérons h et h' deux éléments de $\mathcal{H}$. Alors :

$$\exists (\delta_0, \delta_0'),\ \ \text{tel que}\ \ h \in P(\delta_0), h' \in P(\delta_0').$$

Si $h \cap h' = \emptyset$, la propriété est vérifiée. Sinon, soit $x \in h \cap h'$. h est la classe de x dans la partition $P(\delta_0)$ et h' celle de x dans la partition $P(\delta_0')$. h et h' peuvent donc s'écrire :

$$h = \{y \ / \ \delta(x, y) \leq \delta_0\},$$

$$h' = \{y \ / \ \delta(x, y) \leq \delta'_0\}.$$

Par conséquent, si $\delta_0 \leq \delta'_0$, alors $h \subset h'$. Sinon, $h' \subset h$, ce qui achève la démonstration.

4.5 Equivalence entre hiérarchie indicée et ultra-métrique

Reprenons l'exemple de la hiérarchie $\mathcal{H} = \bigcup_{\delta_0 \in \mathbb{R}^+} P(\delta_0)$. Le couple $(\mathcal{H}, f)$ est une hiérarchie indicée pour f définie par :

$$\forall h \in \mathcal{H}, \quad f(h) = \min\{\delta_0 \ / \ h \in P(\delta_0)\}. \tag{14}$$

Comme, pour tout échantillon x, le singleton $\{x\}$ appartient à $P(0)$, on a $f(\{x\}) = 0$, et la propriété 4 est vérifiée. Soit h et h' dans $\mathcal{H}$, tels que $h \subset h'$ et $h \neq h'$, et soit x dans h. On a alors :

$$\{\delta'_0 \ / \ h' \in P(\delta'_0)\} \subset \{\delta'_0 \ / \ h \in P(\delta'_0)\},$$

et donc :

$$\min\{\delta'_0 \ / \ h' \in P(\delta'_0)\} \geq \min\{\delta'_0 \ / \ h \in P(\delta'_0)\},$$

donc $f(h) \leq f(h')$. Comme de plus, $h \neq h'$, il existe un y dans h' qui ne soit pas dans h. Soit x dans $h \cap h'$. On a :

$$f(h) < \delta(x, y) \leq f(h')$$

et donc $f(h) < f(h')$ (avec une inégalité stricte), ce qui démontre la propriété 5.

Réciproquement, soit $(\mathcal{H}, f)$ une hiérarchie indicée, et soit δ l'application de $E \times E$ dans $\mathbb{R}^+$ telle que :

$$\delta(x, y) = \min_{h \in \mathcal{H}}\{f(h) \ / \ (x, y) \in h^2\}.$$

Alors δ est une ultra-métrique.

Tout d'abord, remarquons que δ est toujours définie puisque E est fini.

Démontrons la propriété 6. $\delta(x, x) = 0$ car $f(\{x\}) = 0$ par définition (propriété 2). Si $\delta(x, y) = 0$, alors :

$$\exists h \in \mathcal{H} \text{ tel que } x \in \mathcal{H}, y \in \mathcal{H} \text{ et } f(h) = 0.$$

Si $h \neq (\{x\})$, $f(h) > f(\{x\}) = 0$, ce qui est impossible. Donc $x = y$.

La démonstration de la propriété 7 est triviale.

Soient h tel que $\delta(x, y) = f(h)$, h' tel que $\delta(x, z) = f(h')$, h" tel que $\delta(y, z) = f(h")$. Puisque $z \in h' \cap h"$, $h' \cap h" \neq \emptyset$. Donc $h' \subset h"$ puisque $\mathcal{H}$ est une hiérarchie (propriété 3), ou le contraire mais le raisonnement serait analogue. Donc x, qui est dans h', appartient à $h"$. y appartient également à $h"$. Or :

$$f(h) = \min\{f(h) \ / \ (x, y) \in h^2\}.$$

Donc $h \subset h$". D'où :

$$f(h) \leq f(h") \leq \sup[f(h'), f(h")],$$

ce qui montre la propriété 8.

La figure 3 illustre l'équivalence entre hiérarchie indicée et ultra-métrique : l'emboîtement des classes d'équivalence quand δ_0 varie correspond aux différents étages de la hiérarchie. Pour aller d'un échantillon x à un autre y dans la hiérarchie, il faut ¡¡ remonter ¿¿ jusqu'à un niveau minimum qui est exactement $\delta(x, y)$.

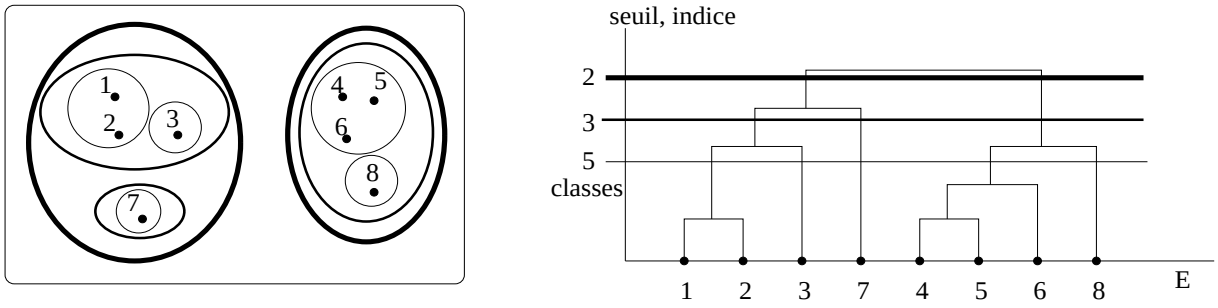


FIG. 3 – Equivalence entre ultra-métrique (à gauche, les partitions induites pour différentes valeurs de δ_0, en 2, 3 et 5 classes respectivement) et hiérarchie indicée (à droite).

5 Conclusion

Les méthodes hiérarchiques ouvrent une piste complémentaire aux méthodes construites sur des critères (paramétriques, k-moyennes,...), et, à ce titre, sont d'une grande utilité en classification automatique. Il est intéressant de noter qu'elles peuvent s'introduire de trois manières différentes (arbre, hiérarchie, ultramétrique), mais néanmoins équivalentes puisque l'on peut s'appuyer sur le résultat suivant :

une ultramétrique $\Leftrightarrow$ un dendogramme $\Leftrightarrow$ une hiérarchie indicée

La notion d'écart peut prendre une dimension proche des utilisateurs, ce qui permet de construire simplement des arborescences et ce qui donne des pistes de classification.

Parmi les limites des méthodes hiérarchiques, on retrouve le problème de la définition de la métrique de base qu'il faut définir sur E. Les méthodes sont très sensibles au choix de cette distance.

Enfin, ces méthodes sont également très sensibles aux positions relatives des échantillons. En effet, par exemple, la distance de chaîne conduit à des résultats très différents si un seul point est mal placé : il peut regrouper deux classes ou au contraire en diviser. Cette grande sensibilité, appelée aussi instabilité structurelle, est à prendre en compte lors de l'analyse des résultats obtenus et dans l'estimation de la robustesse d'un classifieur ainsi construit.

Références

[MIC-84] **L. Miclet :** *Méthodes structurelles pour la reconnaissance des formes*, Eyrolles,1984

[BART-88] **J.P. Bartélemy, A.Guénoche :** *Les arbres et les représentations de proximités*, Masson, 1988

CHAPITRE 6 : MODELES CONNEXIONNISTES-PERCEPTRONS MULTICOUCHES

Jean Marie Nicolas

1 Réseau connexionniste et reconnaissance des formes

1.1 Historique

L'apparition des réseaux de neurones dans le monde scientifique a eu une génèse assez curieuse qu'un peu d'histoire permet de mieux comprendre [1].

L'idée d'utiliser un réseau d'opérateurs élémentaires connectés entre eux est une résurgence d'un courant qui a marqué les travaux en Intelligence Artificielle et en Psychologie des années 60, dont la réalisation la plus marquante a été le Perceptron [Rosenblatt 62]. Elle est fondée sur le comportement de réseaux de processeurs élémentaires interconnectés. Chaque processeur peut être considéré comme une simplification extrême d'un neurone, ce que Mc Culloch et Pitts (1943) ont appelé un "neurone formel".

La première réalisation majeure : Perceptron [Rosenblatt 62] fut sujette à des critiques, en particulier de Minsky et Papert qui, irrités essentiellement par le succès du Perceptron, s'attachèrent à mettre en évidence des limites théoriques et pratiques de celui-ci (problèmes non modélisables par cette approche, difficulté de généralisation à une structure comportant plusieurs niveaux, explosion combinatoire). Suite à cette critique, d'une scientificité étriquée dans la mesure où elle n'envisageait pas des extensions possibles de l'approche, elle fut quasiment abandonnée à la fin des années 60.

Au début des années 80, on assiste à une renaissance de l'approche connexionniste, que l'on peut expliquer par l'intérêt marqué pour le parallélisme d'une part, l'apprentissage de l'autre, ainsi que par le développement technologique qui rend possible la simulation voire la réalisation de machines correspondantes, avec des capacités respectables. Les travaux menés ont permis aux réseaux connectionnistes d'avoir de solides fondements théoriques ainsi que des réalisations informatiques disponibles pour tous : en particulier il existe dans Matlab une *toolbox* consacrée aux réseaux de neurones, démocratisant une fois pour toute ce type d'approche.

L'approche connexioniste a donc eu une génèse longue et tortueuse. Les terminologies sont donc variées : on parle de réseaux de neurones, de réseaux neuromimétiques, de réseaux connexionistes. Tout ceci cache en réalité une discipline aux frontières de domaines aussi différents que la physique théorique, la biologie, les sciences informatiques. C'est probablement cet aspect interdisciplinaire qui en fait sa force.

1. Ce paragraphe a été écrit par Alain Grumbach

1.2 Les réseaux neuromimétiques en reconnaissance des formes

L'approche connexioniste a très rapidement joué un rôle essentiel en reconnaissance des formes. Le cadre est celui de l'apprentissage supervisé dans lequel on dispose d'une base de données étiquetées (qui deviendra une base servant à l'apprentissage), c'est à dire telle qu'il existe un certain nombre de classes C et qu'à chaque individu de la base est associée sa classe. En reprenant le choix de notations du chapitre 4 "Classification automatique-Méthodes non hiérarchiques", on a :

- un nombre de classes connues : C,
- un espace d'état de dimension N,
- un nombre d'individus donné R,
- chaque individu p, $p \in [1, R]$, est décrit par un vecteur d'état X_p
- un vecteur d'état X_p est décrit par ses composantes $X_{i,p}$, $i \in [1, N]$,
- à chaque individu décrit par X_p est associée une étiquette $d_p, p \in [1, R]$. Cette étiquette est tout simplement le numéro de la classe, c'est à dire une valeur entière entre 1 et C.

L'objectif d'un outil de classification est donc d'associer à un individu étiqueté une grandeur qui est, dans le cas idéal, l'étiquette fournie par la base. De ce fait, pour avoir une première opinion sur les performances d'un outil de classification, il suffit d'analyser les sorties de cet outil sur la totalité de la base et de compter les individus bien classés et ceux qui sont mal classés. On définit ainsi une erreur de classification ε définie par

$$\varepsilon = \frac{\text{Nombre d'individus mal classés}}{R} \tag{1}$$

On définit de même le taux de bonnes classifications :

$$\tau = \frac{\text{Nombre d'individus bien classés}}{R} \tag{2}$$

On a bien évidemment :

$$\varepsilon + \tau = 1$$

Ce sont cette erreur et ce taux qui servent en général à analyser et à comparer les performances de divers outils de classification.

Un outil de classification est donc censé donner une étiquette à une entrée quelconque. Comme toute méthode de classification supervisée, on attend d'un réseau neuromimétique d'être capable de généraliser au mieux ses caractéristiques d'apprentissage. Deux points méritent d'être dès maintenant soulignés :

- La représentativité de la base étiquetée est un élément crucial pour obtenir un outil de classification suffisament général. Il sera illusoire d'obtenir la classification d'un individu trop différent des individus ayant permis l'apprentissage du réseau. Eventuellement on pourrait ajouter une classe supplémentaire, dite classe de rejet qui serait attribuée à tout individu manifestement trop différent de ceux qui ont été appris.
- Le taux de bonne classification est un indicateur insuffisant pour qualifier les performances d'un outil de classification. Prenons le cas de l'apprentissage d'une

courbe dans le plan passant par R points : on sait qu'il existe un polynôme de degré R passant exactement par les R points initiaux, mais on sait aussi que son allure fortement oscillatoire ne permet aucune généralisation pour une valeur qui ne serait pas exactement l'une des R valeurs initiales. Classiquement, on recherche plutôt une courbe plus simple (souvent une droite en physique) passant "à peu près" au voisinage des R points initiaux. L'indicateur n'est pas alors seulement le taux de bonne classification (d'autant que le plus souvent la courbe ne passe effectivement sur aucun des points initiaux) : on lui associe aussi un terme d'erreur (le plus souvent une erreur quadratique, dont les propriétés de dérivabilité sont essentielles pour un calcul du type "minimisation au sens des moindres carrés"). Nous verrons donc que l'analyse et la comparaison de méthodes (ou de résultats) utilisent aussi ce type d'erreur.

2 Architecture d'un réseau connexionniste

Un réseau connexioniste associe donc une entrée (vecteur d'état) à une sortie (étiquette scalaire ou vecteur étiquette). Entre ces deux états, le réseau opère un certain nombre d'opérations calculant l'étiquette à partir du vecteur d'entrée, au même titre par exemple qu'un système de décision bayésien associe une étiquette à une entrée.

Ce qui change fondamentalement dans un réseau neuromimétique peut se focaliser sur les deux points suivants :

- Il existe un élément de base, le neurone formel, qui sert de brique élémentaire au système construit entre le vecteur d'entrée et la sortie. Ce neurone formel est capable d'exécuter des opérations simples (suffisament simples pour avoir été implémentées sur des systèmes analogiques dans les années 60).
- Le réseau est un assemblage particulier de neurones formels. Une fois une architecture choisie, on spécifie les neurones formels dans la phase dite d'apprentissage durant laquelle on optimise les neurones pour que la sortie soit la plus proche de la sortie désirée. C'est cette même architecture qui est utilisée en phase d'exploitation (étape de reconnaissance). Aucune hypothèse n'est à faire pour définir les neurones, à la différence d'un système de classification bayésienne qui aura besoin des lois de probabilités a priori des classes pour être construit : c'est simplement à partir de la base de données étiquetées que l'on optimise chaque neurone.

L'objectif de ce paragraphe se focalise sur la définition des neurones et sur l'architecture que l'on peut définir autour de ces neurones. La combinaison de neurones élémentaires peut bien entendu s'effectuer d'une multitude de manières. Dans ce cours nous nous limiterons à un type de réseau : les réseaux en couches (layered perceptrons), qui sont bien adaptés à la reconnaissance des formes. Les paragraphes suivants (3 et 4) seront dédiés à l'étape d'apprentissage, c'est à dire comment optimiser les neurones pour que le réseau donne la sortie souhaitée.

2.1 Structure d'un réseau connexionniste

2.1.1 Le neurone formel individuel

Un neurone formel, c'est à dire une unité du réseau, est doté d'une sortie unique et de plusieurs entrées. Il peut être vu comme un processeur élémentaire calculant sa sortie en fonction de ses entrées.

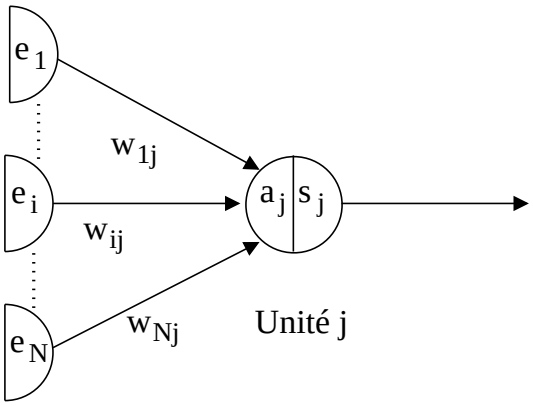


FIGURE 1 – Neurone formel (unité j).

Soit le neurone formel j. Il peut être décrit par les éléments suivants :
- les N entrées e_i de l'unité j.
- a_j l'activité de l'unité j qui décrit comment le neurone opère sur les entrées :

$$a_j = \sum_{i=1}^{N} w_{ij}\, e_i$$

L'activité d'un neurone formel est simplement l'expression d'une somme pondérée (que l'on peut aussi identifier à un filtre linéaire). Le neurone formel j est donc décrit essentiellement par ses poids w_{ij}.
- s_j la sortie de l'unité j : c'est le résultat de l'application d'une fonction f_j, appelée fonction de transition, sur l'activité de l'unité j :

$$\begin{aligned} s_j &= f_j\left(a_j\right) \\ &= f_j\left(\sum_{i=1}^{N} w_{ij}\, e_i\right) \end{aligned} \tag{3}$$

La fonction de transition f_j joue un rôle important dans la génèse des réseaux neuromimétiques. En effet, différents choix peuvent être pris pour cette fonction, conduisant à des réseaux structurellement différents :
- la fonction identité. Dans ce cas la sortie s'exprime simplement comme l'activité du neurone formel :

$$s_j = \sum_{i=1}^{N} w_{ij}\, e_i$$

La valeur en sortie appartient à un intervalle sur $\mathbb{R}$. Le choix de cette fonction conduit aux perceptrons "classiques" de Rosenblatt du chapitre 2.

– la fonction seuil, définie par une variable σ_j représentant le seuil :

$$\left\{ \begin{array}{rclcrcl} s_j & = & f_{seuil}(a_j) & = & -1 & \text{si} & a_j < \sigma_j \\ s_j & = & f_{seuil}(a_j) & = & +1 & \text{si} & a_j \geq \sigma_j \end{array} \right.$$

On voit que les valeurs accessibles se limitent à deux valeurs, ici +1 et -1. C'est en pratique cette fonction qui est utilisée sur un perceptron "à la Rosenblatt" une fois que l'apprentissage est terminé. La classification est dite binaire.

– une fonction sigmoïde, c'est à dire une fonction continue dérivable qui ressemble à une fonction seuil et qui est décrite par un paramètre K_j positif et un seuil σ_j. On peut choisir comme fonction sigmoïde la fonction $f_{sigmoide}$ suivante :

$$f_{sigmoide}\left[K_j, \sigma_j\right](s_j) \quad = \quad \frac{\mathrm{e}^{K_j(a_j-\sigma_j)} - 1}{\mathrm{e}^{K_j(a_j-\sigma_j)} + 1} \tag{4}$$

La fonction sigmoïde décrite ici a ses valeurs dans $]-1;+1[$. On montre aisément qu'en faisant tendre K_j vers l'infini, on retrouve la fonction seuil. De plus sa dérivée existe sur $\mathbb{R}$:

$$f'_{sigmoide}\left[K_j, \sigma_j\right](s_j) \quad = \quad 2\frac{K_j\,\mathrm{e}^{K_j(a_j-\sigma_j)}}{\left(\mathrm{e}^{K_j(a_j-\sigma_j)} + 1\right)^2} \tag{5}$$

La figure 2 illustre cette fonction sigmoïde ainsi que sa dérivée.
Nous verrons que l'on utilise aussi une autre fonction sigmoïde ayant ses valeurs dans $]0;+1[$ et définie par :

$$f_{sigmoide,]0,1[}\left[K_j, \sigma_j\right](s_j) \quad = \quad \frac{\mathrm{e}^{K_j(a_j-\sigma_j)}}{\mathrm{e}^{K_j(a_j-\sigma_j)} + 1}$$

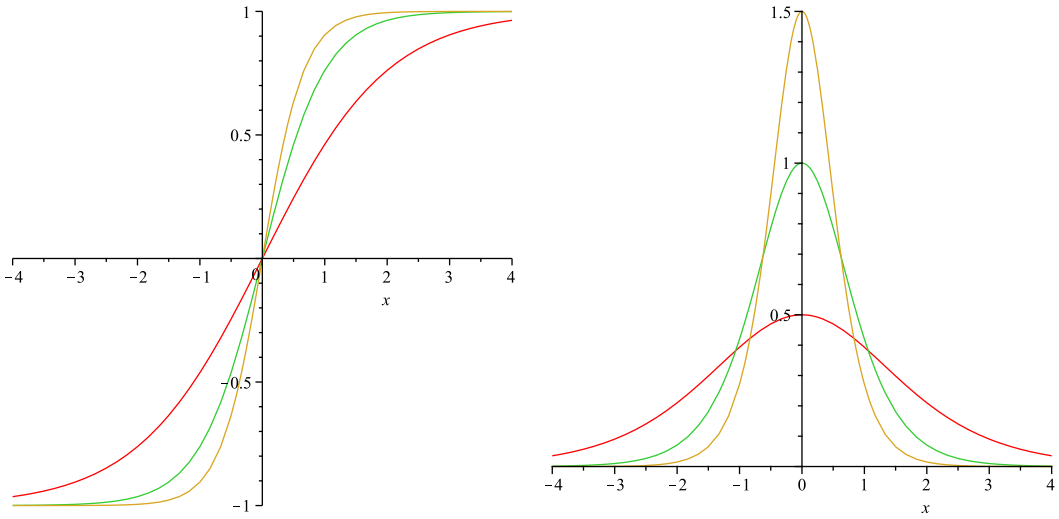


FIGURE 2 – A gauche : fonction sigmoïde (équation 4) pour $K_j = 1, 2$ et 3. A droite : dérivée de la fonction sigmoïde (équation 5). On a ici $\sigma_j = 0$.

2.1.2 Les neurones en réseau

En raccordant des neurones formels entre eux, on construit forment alors un réseau neuromimétique. La manière dont les neurones se connectent peut se faire de différentes manières. On peut en effet envisager les cas suivants :

- Chaque neurone a pour entrées les sorties de tous les autres neurones. On dit alors que le réseau est totalement interconnecté. On parle alors de réseau de Kohonen. L'étude de ce type de réseaux est en fait du ressort de la physique théorique du fait de grandes similitudes avec les aspects statistiques de la thermodynamique.
- Chaque neurone a un petit nombre d'entrées et ne semble pas suivre de règle apparente en terme d'interconnection avec les autres neurones. Les neurones biologiques (comme ceux du cerveau) entrent dans cette catégorie.
- Les neurones sont regroupés en couches. Les entrées d'un neurone d'une couche donnée m sont les sorties des neurones de la couche $m-1$. La sortie d'un neurone de la couche m est aussi une entrée pour les neurones de la couche $m+1$. On parle alors de réseaux multicouches et ce sont ceux là qui entrent dans le cadre de ce chapitre. Nous avons déjà rencontré un tel réseau au chapitre 2 : c'est le perceptron, qui peut être interprété comme un réseau monocouche. Notons que ce type de réseau n'a plus qu'un rapport lointain avec la réalité biologique.

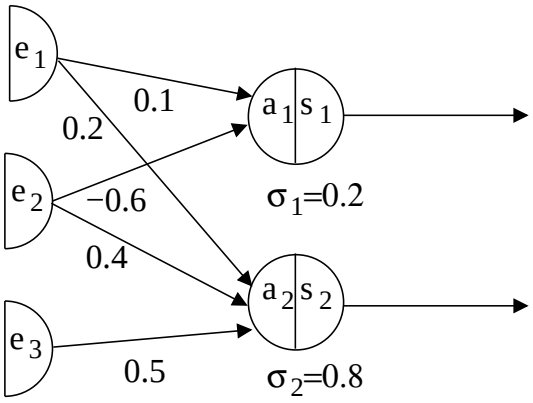


FIGURE 3 – Fonctionnement d'une couche. Noter que ce réseau n'est pas totalement connecté entre les entrées et les sorties (formellement, cela revient à prendre $w_{31} = 0$).

La figure 3 montre une couche d'un réseau de 3 entrées et de 2 sorties. Analysé en terme d'information, on peut décrire le fonctionnement de cette couche comme si chaque neurone était un processeur élémentaire synchronisé avec les autres neurones de la couche :

- L'information en entrée est décrite par les valeurs (e_1, e_2, e_3).
- Elle transite sur les connexions. Chaque unité calcule son activité (somme pondérée), puis sa sortie (fonction sigmoïde associée au seuil σ). Pour cela, à chaque liaison entre entrée et neurone est associé un poids w_{ij} dont la valeur est indiquée sur cette figure.
- Les sorties de cette couche sont les valeurs (s_1, s_2).

Parmi les neurones d'un réseau neuromimétique, se trouvent en général des unités particulières, qui réalisent l'interface avec l'environnement : ce sont les unités d'entrée

du réseau et les unités de sortie du réseau. Les unités d'entrée du réseau ont leurs entrées raccordées au "monde extérieur", par exemple, l'espace des données (décrites par leurs vecteurs d'état). Les unités de sortie du réseau ont leurs sorties raccordées au "monde extérieur", par exemple à l'espace des classes. Les autres unités, si elles existent, ne sont pas visibles du monde extérieur : on parle alors de couche cachée.

2.2 Les perceptrons

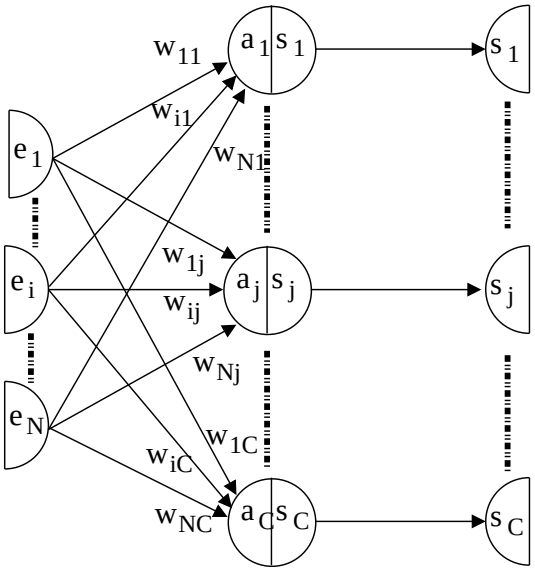


FIGURE 4 – Perceptron monocouche. Il y a N entrées et C sorties. Le réseau est totalement interconnecté, c'est à dire que chaque entrée est connectée à chaque sortie (plus précisément, l'activité de chaque unité de sortie dépend des valeurs de chaque entrée).

Depuis les travaux sur les réseaux neuromimétiques des années 80, la famille des perceptrons (au sens large) est associée à celle des réseaux multicouches. Deux grandes familles s'en dégagent :

- Le perceptron simple, figure 4 (perceptron de Rosenblatt déjà rencontré au chapitre 2). C'est un réseau caractérisé par l'identification des unités d'entrée du réseau aux unités d'entrée de l'unique couche et des unités de sortie du réseau aux unités de sortie de l'unique couche : la couche de neurones est donc totalement "visible" du monde extérieur Dans ce schéma historique, la fonction de transition est la fonction seuil. Les limites d'une telle architecture ont été démontrées dans les années 60 : elle n'est adaptée qu'à des séparations linéaires dans l'espace des données. La détermination des poids s'effectue par la célèbre règle du perceptron décrite au chapitre 2.
- les réseaux à plusieurs couches d'unités que l'on appelle Perceptrons multicouches. Comme le perceptron simple, il communique avec le monde extérieur par les unités d'entrée du réseau et les unités de sortie du réseau. Entre l'entrée et la sortie, il y a donc une ou plusieurs couches dont les sorties ne sont pas connues

du monde extérieur et qui sont les couches cachées. Chaque couche (indicée m) est composée de k_{m-1} entrées $e_j^{(m)}$, de k_m unités d'activités $a_j^{(m)}$ et de k_m sorties $s_j^{(m)}$. Pour un réseau à q couches, il y a donc $k_0 = N$ entrées du réseau et $k_q = C$ sorties du réseau.

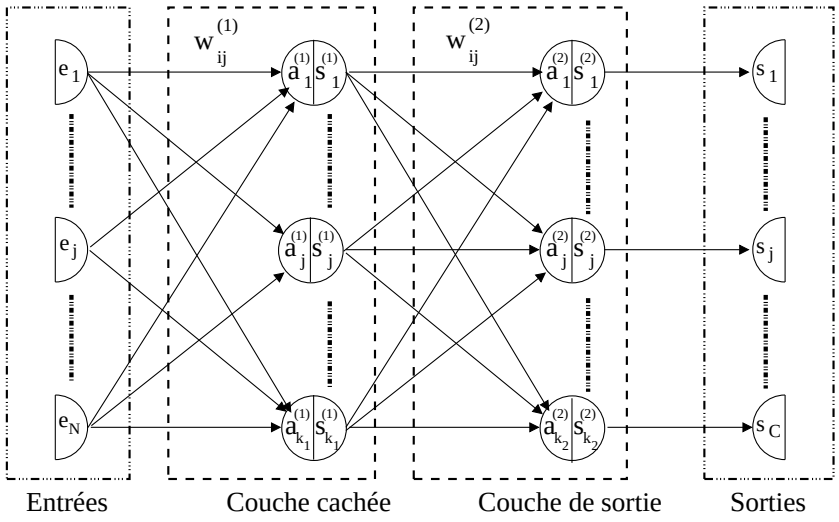


FIGURE 5 – Exemple d'un perceptron à 2 couches, donc à une couche cachée. Le système complet a N entrées et C sorties.

Considérons la couche m d'un perceptron multicouches : soit k_m le nombre de neurones de cette couche. Pour tout neurone j de cette couche ($j \in [1, k_m]$), on peut calculer son activité $a_j^{(m)}$:

$$a_j^{(m)} = \sum_{i=1}^{k_{m-1}} w_{ij}^{(m)} e_i^{(m)} = \sum_{i=1}^{k_{m-1}} w_{ij}^{(m)} s_i^{(m-1)}$$

Connaissant sa fonction de transition $f_{m,j}$, on calcule sa sortie $s_j^{(m)}$:

$$s_j^{(m)} = f_{m,j}\left(a_j^{(m)}\right) = f_{m,j}\left(\sum_{i=1}^{k_{m-1}} w_{ij}^{(m)} s_i^{(m-1)}\right)$$

Par simplification, on prend souvent une même fonction d'activation f pour tous les neurones (ce point sera détaillé au paragraphe 5.1).

On voit dans cette expression que le calcul des sorties des neurones de la couche m nécessite d'avoir les sorties de tous les neurones de la couche $m-1$: on parle alors de propagation.

Le perceptron simple (monocouche) a donc été longuement étudié dans les années 60 : on peut en définir les poids grâce, par exemple, à la règle du perceptron. Pour les perceptrons multicouches, des techniques spécifiques d'apprentissage, seulement découvertes dans les années 80, doivent être mises en œuvre (voir paragraphe 4).

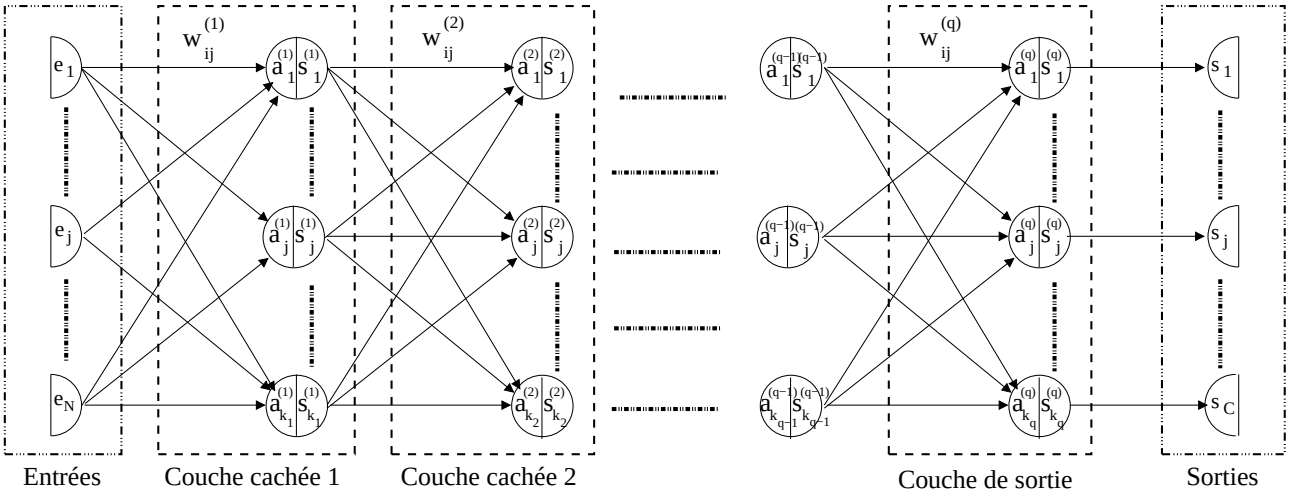


FIGURE 6 – Exemple d'un perceptron à q couches, donc à $q-1$ couches cachées.

3 Apprentissage des perceptrons sans couches cachées

Une fois l'architecture d'un réseau choisie (ici, dans ce paragraphe, un réseau sans couche cachée), il reste à choisir les valeurs des poids de chaque neurone : c'est la phase d'apprentissage, qui nécessite une base servant à l'apprentissage [2] composée de R individus décrits dans un espace d'état de dimension N. Chaque individu de cette base est décrit par son vecteur d'état $X_p = (X_{i,p})$, $i \in [1, N]$. Cette base est étiquetée, c'est à dire qu'à chaque vecteur d'état $X_{i,p}$ on associe une étiquette scalaire d_p qui permet de désigner la classe à laquelle appartient l'échantillon p.

Dans le cadre des réseaux neuromimétiques, on a donc R individus dans la base servant à l'apprentissage. Pour une entrée $(e_{i,p}) = (X_{i,p})$ ($i \in [1, N]$ puisque le vecteur d'état est de dimension N), la sortie désirée du réseau est alors $s_{1,p} = d_p$ (un scalaire en sortie).

Par simplification, nous étudions dans ce paragraphe un cas assez simple : celui de la **classification en deux classes**. La sortie du réseau est alors de dimension 1 : on attribue par exemple la valeur +1 comme sortie souhaitée pour les individus de la classe 1 et la valeur -1 comme sortie souhaitée pour les individus de la classe 2. . Nous verrons que l'on peut, de près ou de loin, se ramener à ce cas dans le cas de la classification en C classes (paragraphe 5.2).

3.1 Modification des poids : la règle Delta

Pour apprendre les poids d'un perceptron monocouche, une première approche est d'évaluer la qualité du résultat de classification en analysant les performances globales sur la totalité de la base servant à l'apprentissage. Cette analyse de performances va devoir être plus complète de celle menée au paragraphe 1.2 où l'erreur se limitait à

2. La notion exacte de "base d'apprentissage" spécifique aux méthodes neuromimétiques sera introduite ultérieurement.

analyser le nombre d'individus bien classés : une fonction de coût (de type "erreur quadratique") va être requise pour permettre une modification des poids débouchant sur de meilleures performances en classification.

Dans ce paragraphe, nous considérons donc le cas où il n'y a que deux classes. On peut sans perte de généralité considérer alors une sortie unique dont les valeurs souhaitées sont les suivantes :

– +1 si l'individu appartient à la classe 1

– -1 si l'individu appartient à la classe 2

Si la fonction de transition est un simple seuil, on aura alors effectivement en sortie les deux valeurs +1 et -1. Puisque la différence entre sortie obtenue et étiquette ne prend que 3 valeurs, -2, 0 et 2, le taux de mauvaise classification est alors aisé à écrire :

$$\varepsilon \;=\; \frac{\sum_{p=1}^{R} |s_{1,p} \;-\; d_p|}{2R}$$

A priori, on ne peut rien déduire de cette expression sur ce qu'il serait possible de faire sur les poids du perceptron pour améliorer ce taux.

Considérons maintenant la sortie d'un réseau dont la fonction de transition peut prendre toutes les valeurs possibles sur un intervalle fixé dans le choix de la fonction de transition, c'est à dire ici l'intervalle $]-1;1[$ (c'est le cas de la fonction sigmoïde du paragraphe 2.1.1). Si l'on compare les valeurs de sortie du réseau à la valeur de l'étiquette, la différence est alors une valeur continue : cette différence peut être décrite par une fonction dérivable si la fonction de transition est dérivable. Introduisons maintenant une fonction de coût J. Il y a bien évidemment un choix très grand pour cette définition. Le choix le plus simple à manipuler est celui de l'erreur quadratique moyenne[3], c'est à dire en considérant la somme des erreurs quadratiques entre étiquette et sortie du réseau pour tous les individus de la base servant à l'apprentissage. On obtient ainsi la fonction de coût J telle que :

$$J \;=\; \sum_{p=1}^{R} (s_{1,p} \;-\; d_p)^2 \tag{6}$$

On retrouve une expression connue en classification automatique puisque c'est celle de la méthode des k-moyennes.

En écrivant la sortie s_p en fonction du vecteur d'état (équation 3), et en choisissant une fonction de transition unique pour tous les neurones, on a :

$$J \;=\; \sum_{p=1}^{R} \left(f\left(\sum_{i=1}^{N} w_{i1}\, e_{i,p} \right) \;-\; d_p \right)^2$$

Puisque la sortie est un scalaire (un seul neurone de sortie), on va simplifier les écritures en remplaçant dans ce paragraphe w_{i1} par w_i et $s_{1,p}$ par s_p. On a alors

$$J \;=\; \sum_{p=1}^{R} (s_p \;-\; d_p)^2 \;=\; \sum_{p=1}^{R} \left(f\left(\sum_{i=1}^{N} w_i\, e_{i,p} \right) \;-\; d_p \right)^2 \tag{7}$$

3. Pour alléger les expressions, on va omettre le terme de division par le nombre d'individus R

(l'écriture w_{i1} a été ici simplifiée en w_i puisqu'il n'y a qu'une seule sortie).

On voit que cette dernière expression donne une erreur, que l'on cherche a priori à minimiser en jouant sur les paramètres w_i. Si f est une fonction de transition dérivable (ce qui n'est pas le cas de la fonction seuil) et puisque l'erreur quadratique est dérivable, on peut rechercher à minimiser J. Ce minimum sera atteint si

$$\begin{aligned}
\frac{\partial J}{\partial w_i} &= 0 \quad \forall i \in [1, N] \\
&= 2 \sum_{p=1}^{R} \frac{\partial f \left(\sum_{i'=1}^{N} w_{i'} \, e_{i',p} \right)}{\partial w_i} \left(f \left(\sum_{i'=1}^{N} w_{i'} \, e_{i',p} \right) - d_p \right) \\
&= 2 \sum_{p=1}^{R} \frac{\partial f \left(\sum_{i'=1}^{N} w_{i'} \, e_{i',p} \right)}{\partial w_i} \left(s_p - d_p \right) \\
&= 2 \sum_{p=1}^{R} e_{i,p} \, \left(s_p - d_p \right) \, \left. \frac{\partial f}{\partial x} \right|_{x = \sum_{i'=1}^{N} w_{i'} \, e_{i',p}} \qquad (8)
\end{aligned}$$

Cette dernière expression conduit donc à un système de N équations à N inconnues : les N poids w_i. Deux cas peuvent se présenter :

- la fonction de transition possède une expression assez simple pour que le système 8 puisse s'inverser analytiquement. On a alors directement la valeur des poids. C'est le cas par exemple où la fonction de transition est la fonction identité. On a alors un système linéaire de N équations à N inconnues aisément inversable s'il est bien conditionné, ce qui est rarement le cas. Cette solution est donc très rarement utilisée, voire totalement illusoire.
- la fonction de transition est plus complexe et le système 8 ne peut être inversé simplement de manière analytique. On a bien un système de N équations à N inconnues, mais c'est un système implicite qui doit être résolu de manière numérique. En général, c'est l'algorithme de descente de gradient qui est appliquée. En désignant le gradient selon le poids w_i par la notation $\nabla_i(J)$, on a alors une expression analytique de ce gradient :

$$\begin{aligned}
\nabla_i(J) &= 2 \sum_{p=1}^{R} e_{i,p} \, \left(s_p - d_p \right) \, \left. \frac{\partial f}{\partial x} \right|_{x = \sum_{i'=1}^{N} w_{i'} \, e_{i',p}} \\
&= 2 \sum_{p=1}^{R} e_{i,p} \, \left(s_p - d_p \right) \, \left. \frac{\partial f}{\partial x} \right|_{x = a_{1,p}} \qquad (9)
\end{aligned}$$

  cette dernière expression mettant en jeu l'activité $a_{1,p}$ du neurone de sortie.
  Il est intéressant de remarquer que le gradient dépend de trois termes :
  - $(s_p - d_p)$: plus la sortie diffère de la sortie désirée, plus le gradient est grand.
  - $e_{i,p}$: le gradient est d'autant plus important que l'entrée est grande.
  - $\left. \frac{\partial f}{\partial x} \right|_{x = a_{1,p}}$: si la valeur de l'activité du neurone $a_{1,p}$ est autour de la valeur nulle, alors l'effet sera grand sur le gradient puisque la dérivée de la fonction de transition joue un rôle uniquement autour de la valeur nulle.

Puisque l'on a une forme analytique du gradient, il suffit de modifier itérativement les poids en prenant en compte la direction du gradient, c'est à dire en appliquant pour le poids w_i la règle suivante :

$$w_i \ \rightarrow \ w'_i \ = \ w_i \ - \ \eta \nabla_i(J) \tag{10}$$

le coefficient η devant être choisi de sorte à éviter les pièges classiques de tout algorithme de descente de gradient (s'il est trop grand, on va être "éjecté" du minimum absolu ; s'il est trop petit, on risque d'être piégé dans un minimum secondaire).
Appliqué à la fonction de transition identité (dont la dérivée est égale à 1), la relation 9 devient :

$$\nabla_i(J) \ = \ 2 \sum_{p=1}^{R} e_{i,p} \ (s_p \ - \ d_p)$$

Si on ne considére qu'un seul échantillon p, on retrouve la célèbre règle Delta :

$$w'_i \ = \ w_i \ - \ \eta \, e_{i,p} \ (s_p \ - \ d_p) \tag{11}$$

Cette règle est simplissime puisque n'interviennent que la valeur d'entrée, la valeur de sortie calculée et la valeur de sortie désirée. On la trouve aussi dans la littérature sous les noms de règle de Widrow-Hoff, règle adaline ou règle LMS (Least Mean Square).
Dans le cas général, il faut effectivement avoir l'expression analytique de la dérivée de la fonction de transition (voilà pourquoi le choix d'une fonction de transition dérivable a été introduit dans l'introduction du neurone formel au paragraphe 2.1.1). Ensuite il faut en calculer la valeur pour la donnée d'entrée e_p.

3.2 Un cas particulier de la règle Delta : la règle du perceptron

Nous avons déjà vu la règle du perceptron (chapitre 2), que nous allons retrouver ici dans le cadre neuromimétique. En effet, considérons la règle Delta (équation 11, cas d'une sortie scalaire) :

$$w'_i \ = \ w_i \ - \ \eta \, e_{i,p} \ (s_p \ - \ d_p)$$

Prenons $\eta = 1$. On a alors :

$$w'_i \ = \ w_i \ - \ e_{i,p} \ (s_p \ - \ d_p)$$

Or dans ce cadre de classification en deux classes (étiquettes +1 ou -1), et si l'on choisit maintenant la fonction seuil (initialement interdite puisqu'il fallait avoir une fonction de transition dérivable), on peut remarquer que le terme $s_p \ - \ d_p$ peut prendre 3 valeurs :

- 0 si l'élément est bien classé. Dans ce cas, on ne modifie pas les poids.
- +2 ou -2 si l'élément est mal classé. On a alors :

– +2 si $s_p = 1$ et $d_p = -1$. On a alors

$$w'_i \;=\; w_i \;-\; 2e_{i,p}$$

– -2 si $s_p = -1$ et $d_p = +1$. On a alors

$$w'_i \;=\; w_i \;+\; 2e_{i,p}$$

Réécrivons maintenant la règle du perceptron (chapitre 2) qui s'applique dès lors que l'échantillon est mal classé :

$$w'_i \;=\; w_i \stackrel{+}{-} e_{i,p}$$

le signe indiqué $\stackrel{+}{-}$ rappelant qu'il faut prendre le signe + si l'individu est étiqueté en classe L^+ et classé en L^-, ou le signe - si l'individu est étiqueté en classe L^- et classé en L^+. On voit qu'à un facteur 2 près, on obtient la même règle. On a bien retrouvé la célèbre règle du perceptron.

Cette règle du perceptron a aussi une autre dimension pratique : c'est en présentant chaque individu de la base servant à l'apprentissage que les poids sont modifiés.

3.3 Apprentissage des perceptrons monocouches

Un des points essentiels des techniques neuromimétiques réside dans le fait que, comme pour la règle du perceptron, on ne calcule pas la fonction de coût sur la totalité de la base servant à l'apprentissage. En pratique, on calcule la fonction de coût pour chaque individu de la base servant à l'apprentissage. Dans ce cas, si on présente l'individu p, le gradient (requis pour la mise en œuvre d'un algorithme de descente de gradient) s'écrit (équation 9) :

$$\nabla_{i,p}(J) \;=\; 2e_{i,p}\;(s_p \;-\; d_p)\;\left.\frac{\partial f}{\partial x}\right|_{x=a_{i,p}} \tag{12}$$

Cet aspect essentiel de l'apprentissage sera abordé au paragraphe 5

4 Apprentissage des perceptrons multi-couches

Face aux limitations aux cas linéairement séparables du perceptron monocouche, l'idée de rajouter des couches au perceptron monocouche et donc d'utiliser un perceptron multicouches semble être une piste prometteuse. Cependant la règle du perceptron n'est pas applicable aux poids d'un perceptron multicouches. Aussi les années 60 ont donc enterré le perceptron faute de pouvoir se pencher dans le calme sur un terme d'erreur quadratique dans un cadre multicouches. C'est cette piste, évidente dès lors que quelqu'un la propose[4], qui n'a été trouvé qu'au milieu des années 80 que nous allons détailler maintenant.

4. Il y a plusieurs chercheurs qui ont indépendamment proposé cette approche : Yann Le Cun, Rumelhart, Hinton, ...

4.1 L'algorithme de rétropropagation du gradient

Considérons un perceptron à 2 couches (donc une couche cachée de k_1 poids, c'est le perceptron de la figure 5), avec N entrées et C sorties. Par simplification, nous prendrons la même forme analytique pour la fonction d'activation f que nous noterons f_1 pour la couche cachée et f_2 pour la couche de sortie (l'objectif de cette notation explicite permettra de bien comprendre les rôles respectifs des couches lors du calcul du gradient).

Détaillons d'abord les activités et les sorties de chaque couche.

Pour la première couche, c'est à dire la couche cachée, on a :

$$a_j^{(1)} = \sum_{i=1}^{N} w_{ij}^{(1)} e_i$$

(avec, par simplification $e_i^{(1)} = e_i$ puisque c'est l'entrée du réseau complet). La sortie des k_1 neurones de la couche cachée $s_j^{(1)}$, $j \in [1, k_1]$, s'écrit :

$$s_j^{(1)} = f\left(a_j^{(1)}\right) = f_1\left(\sum_{i=1}^{N} w_{ij}^{(1)} e_i\right)$$

Propageons ces valeurs dans la seconde couche. On a alors l'activité du neurone j de la seconde couche :

$$\begin{aligned} a_j^{(2)} &= \sum_{q=1}^{k_1} w_{qj}^{(2)} e_q^{(2)} \\ &= \sum_{q=1}^{k_1} w_{qj}^{(2)} s_q^{(1)} \\ &= \sum_{q=1}^{k_1} w_{qj}^{(2)} \left(f_1\left(\sum_{i=1}^{N} w_{iq}^{(1)} e_i\right)\right) \end{aligned}$$

d'où les sorties du réseau s_j :

$$\begin{aligned} s_j &= f_2\left(a_j^{(2)}\right) \\ &= f_2\left(\sum_{q=1}^{k_1} w_{qj}^{(2)} e_q^{(2)}\right) \\ &= f_2\left(\sum_{q=1}^{k_1} w_{qj}^{(2)} \left(f_1\left(\sum_{i=1}^{N} w_{iq}^{(1)} e_i\right)\right)\right) \end{aligned} \tag{13}$$

On voit dans cette dernière expression comment les valeurs d'entrée se propagent à la première couche (la couche cachée), puis de la première couche à la couche de sortie.

Le terme d'erreur quadratique s'écrit donc (cf équation 6 : on prend en compte la dimension C du vecteur de sortie) :

$$J = \sum_{p=1}^{R} \sum_{j=1}^{C} (s_{j,p} - d_{j,p})^2$$

$$= \sum_{p=1}^{R}\sum_{j=1}^{C}\left(f_2\left(\sum_{q=1}^{k_1} w_{qj}^{(2)} e_{q,p}^{(2)}\right) - d_{j,p}\right)^2 \quad (14)$$

Il faut donc modifier cette expression en remplaçant les valeurs de sortie de la couche cachée par le résultat de l'action de la couche cachée sur les entrées du réseau, ce qui donne :

$$\begin{aligned} J &= \sum_{p=1}^{R}\sum_{j=1}^{C}(s_{j,p} - d_{j,p})^2 \\ &= \sum_{p=1}^{R}\sum_{j=1}^{C}\left(f_2\left(\sum_{q=1}^{k_1} w_{qj}^{(2)} e_{q,p}^{(2)}\right) - d_{j,p}\right)^2 \\ &= \sum_{p=1}^{R}\sum_{j=1}^{C}\left(f_2\left(\sum_{q=1}^{k_1} w_{qj}^{(2)}\left(f_1\left(\sum_{i=1}^{N} w_{iq}^{(1)} e_{i,p}\right)\right)\right) - d_{j,p}\right)^2 \end{aligned} \quad (15)$$

En quelque sorte, on a propagé de droite à gauche l'erreur dans le réseau : on parle alors de rétropropagation.

Comme dans le cas du perceptron simple, on cherche les poids du réseau minimisant l'erreur quadratique. Le réseau ayant N entrées et C sorties, et la couche cachée ayant k_1 sorties, il y a donc

- $N \times k_1$ poids $w_{iq}^{(1)}$ entre entrée et couche cachée,
- $k_1 \times C$ poids $w_{qj}^{(2)}$ entre couche cachée et sortie.

Pour trouver le minimum de l'erreur quadratique, il suffit de calculer les dérivées de cette erreur vis à vis de chaque poids et de choisir les valeurs de poids visant à annuler cette dérivée. Bien évidemment, ce n'est pas cette approche qui est utilisée : on va plutôt calculer le gradient de l'erreur quadratique et appliquer une descente de gradient sur chaque poids.

Pour ce perceptron à une couche cachée, deux types de calcul sont à mener :

- l'optimisation des poids $w_{qj}^{(2)}$ entre couche cachée et couche de sortie. L'erreur quadratique s'écrit tout simplement (relation 14) :

$$J = \sum_{p=1}^{R}\sum_{j'=1}^{C}\left(f_2\left(\sum_{q'=1}^{k_1} w_{q'j'}^{(2)} e_{q',p}^{(2)}\right) - d_{j',p}\right)^2$$

  C'est une expression parfaitement similaire à celle trouvée dans le perceptron monocouche à sortie unique (relation 7). Dériver cette expression en fonction du poids $w_{qj}^{(2)}$ donne le terme requis par l'algorithme de descente de gradient (équation 9) :

$$\begin{aligned} \nabla_{qj}^{(2)}(J) &= \frac{\partial J}{\partial w_{qj}^{(2)}} \\ &= 2\sum_{p=1}^{R}\sum_{j'=1}^{C}(s_{j',p} - d_{j',p})\,\frac{\partial f_2\left(\sum_{q'=1}^{k_1} w_{q'j'}^{(2)}\, e_{q',p}^{(2)}\right)}{\partial w_{qj}^{(2)}} \end{aligned}$$

$$
\begin{aligned}
&= 2\sum_{p=1}^{R} e_{q,p}\ (s_{j,p}\ -\ d_{j,p})\ \left.\frac{\partial f_2}{\partial x}\right|_{x=\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\, e_{q',p}^{(2)}} \\
&= 2\sum_{p=1}^{R} e_{q,p}\ (s_{j,p}\ -\ d_{j,p})\ \left.\frac{\partial f_2}{\partial x}\right|_{x=a_{j,p}^{(2)}} \qquad (16)
\end{aligned}
$$

(on utilise $a_{j,p}^{(2)}$, l'activité du neurone q de la couche de sortie, pour alléger l'expression). On retrouve l'expression menant à la règle Delta (équation 10) : pour la couche de sortie, la minimisation de l'erreur quadratique passe donc par une modification des poids prenant en compte la direction du gradient, c'est à dire

$$
w_{qj}^{\prime(2)} \ = \ w_{qj}^{(2)} \ - \ \eta \nabla_{qj}^{(2)}(J)
$$

– l'optimisation des poids $w_{iq}^{(1)}$ entre couche d'entrée et couche cachée. L'erreur quadratique s'écrit (relation 15) :

$$
J \ = \ \sum_{p=1}^{R}\sum_{j=1}^{C}\left(f_2\left(\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\left(f_1\left(\sum_{i'=1}^{N} w_{i'q'}^{(1)} e_{i',p}\right)\right)\right)\ -\ d_{j,p}\right)^2
$$

expression qu'il faut dériver par rapport aux poids $w_{iq}^{(1)}$. En appliquant les techniques de dérivation des fonctions composées [5], on peut écrire :

$$
\begin{aligned}
\nabla_{iq}^{(1)}(J) \ &= \ \frac{\partial J}{\partial w_{iq}^{(1)}} \\
&= \ \sum_{p=1}^{R}\sum_{j=1}^{C} \frac{\partial\left(f_2\left(\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\left(f_1\left(\sum_{i'=1}^{N} w_{i'q'}^{(1)} e_{i',p}\right)\right)\right)\ -\ d_{j,p}\right)^2}{\partial w_{iq}^{(1)}} \\
&= \ 2\sum_{p=1}^{R}\sum_{j=1}^{C} \frac{\partial\left(f_2\left(\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\left(f_1\left(\sum_{i'=1}^{N} w_{i'q'}^{(1)} e_{i',p}\right)\right)\right)\ -\ d_{j,p}\right)}{\partial w_{iq}^{(1)}}\,(s_{j,p}\ -\ d_{j,p}) \\
&= \ 2\sum_{p=1}^{R}\sum_{j=1}^{C} \left.\frac{\partial f_2}{\partial x}\right|_{x=\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\, e_{q',p}^{(2)}} \frac{\partial}{\partial w_{iq}^{(1)}}\left(\sum_{q'=1}^{k_1} w_{q'j}^{(2)}\left(f_1\left(\sum_{i'=1}^{N} w_{i'q'}^{(1)} e_{i',p}\right)\right)\right)\ (s_{j,p}\ -\ d_{j,p}) \\
&= \ 2\sum_{p=1}^{R}\sum_{j=1}^{C} \left.\frac{\partial f_2}{\partial x}\right|_{x=a_{j,p}^{(2)}} \sum_{q'=1}^{k_1}\left(w_{q'j}^{(2)}\,\frac{\partial f_1\left(\sum_{i'=1}^{N} w_{i'q'}^{(1)}\, e_{i',p}\right)}{\partial w_{iq}^{(1)}}\right)\ (s_{j,p}\ -\ d_{j,p}) \\
&= \ 2\sum_{p=1}^{R}\ e_{i,p}\ \left.\frac{\partial f_1}{\partial x}\right|_{x=a_{q,p}^{(1)}}\ \left\{\sum_{j=1}^{C} w_{qj}^{(2)}\ (s_{j,p}\ -\ d_{j,p})\ \left.\frac{\partial f_2}{\partial x}\right|_{x=a_{j,p}^{(2)}}\right\} \qquad (17)
\end{aligned}
$$

Cette expression assez lourde n'a finalement rien de bien compliqué. D'une certaine manière, on voit que l'on propage le gradient de la couche de sortie vers

5. Il est curieux de voir que la communauté scientifique soit restée presque 20 ans sans voir que c'était le seul théorème requis pour passer des perceptrons monocouches aux perceptrons multicouches. Mais il fallait aussi utiliser une fonction sigmoïde, ce qui peut expliquer un aveuglement aussi long.

la couche d'entrée, d'où le nom de l'algorithme : rétropropagation du gradient. Le seul élément un peu technique est le fait que modifier un poids de la couche cachée modifie la sortie d'un neurone de la couche cachée : ce neurone étant en entrée de tous les neurones de la couche de sortie, toutes les sorties sont donc modifiées. Voilà pourquoi il demeure une somme sur tous les éléments de la couche de sortie ($\sum_{j=1}^{C}$), mais les termes de cette somme ont une forte ressemblance avec le gradient calculé pour la couche de sortie (expression 16).

Il serait maintenant assez aisé de généraliser ce calcul à des perceptrons avec Q couches : le seul véritable piège serait celui des notations, qui deviennent alors très lourdes à gérer.

4.2 Apprentissage des perceptrons multicouches

Comme dans le cas monocouche, les algorithmes d'apprentissage considèrent individuellement chaque individu de la base servant à l'apprentissage. En pratique, on calcule la fonction de coût pour chaque individu p de la base servant à l'apprentissage et le gradient (requis pour la mise en œuvre d'un algorithme de descente de gradient) s'écrit :

– pour un poids de la couche de sortie (équation 16) :

$$\nabla_{qj}^{(2)}(J) \;=\; 2e_{q,p}\;(s_{j,p}\;-\;d_{j,p})\;\left.\frac{\partial f_2}{\partial x}\right|_{x=a_q^{(2)}} \tag{18}$$

– pour un poids de la couche cachée (équation 17) :

$$\nabla_{iq}^{(1)}(J) \;=\; 2\sum_{p=1}^{R}\;e_{i,p}\;\left.\frac{\partial f_1}{\partial x}\right|_{x=a_{q,p}^{(1)}}\;\left\{\sum_{j=1}^{C} w_{qj}^{(2)}\;(s_{j,p}\;-\;d_{j,p})\;\left.\frac{\partial f_2}{\partial x}\right|_{x=a_{j,p}^{(2)}}\right\} \tag{19}$$

Cet aspect essentiel de l'apprentissage sera abordé au paragraphe 5

5 Mise en œuvre des réseaux neuromimétiques

Dans ce paragraphe, nous allons maintenant aborder rapidement la mise en œuvre d'un réseau neuromimétique (monocouche ou multicouche) plus spécifiquement dans un contexte de classification. Nous nous plaçons dans le cadre assez général où la fonction de transition est la sigmoïde de paramètres K_j et σ_j et prenant ses valeurs dans l'intervalle $]0;1[$:

$$f(x) \;=\; \frac{\mathrm{e}^{K_j(x-\sigma_j)}}{\mathrm{e}^{K_j(x-\sigma_j)}+1}$$

Le paramètre K_j, dont l'inverse peut être amalgamé au concept de température[6], permet de "jouer" sur la raideur de la sigmoïde. En particulier, pour $K_j \to \infty$, on a :

$$\left\{\begin{array}{lcll} f(x) & = & 0 & \text{si} \quad x < \sigma_j \\ f(x) & = & 1 & \text{si} \quad x \geq \sigma_j \end{array}\right.$$

6. Là aussi la thermodynamique statistique peut aider à la compréhension.

On retrouve donc la fonction seuil pour $K_j \to \infty$ (à la température nulle, il n'y a que deux états possibles : 0 ou 1).

5.1 "Apprendre" la valeur σ_j de la fonction sigmoïde

On voit qu'à chaque neurone est associée une fonction de transition a priori différente puisque dépendant d'un paramètre σ_j. Reprenons l'expression spécifique de la fonction sigmoïde :

$$\begin{aligned} x - \sigma_j &= \sum_{i=1}^{N} w_{ij}\, e_i \; - \; \sigma_j \\ &= \sum_{i=1}^{N} w_{ij}\, e_i \; + \; (-\sigma_j) \times (1) \end{aligned}$$

Dans cette dernière expression, tout se passe comme si on avait ajouté à la somme pondérée initiale une entrée d'indice $N+1$, de valeur $e_{N+1} = 1$ et de poids $-\sigma_j$. On a alors :

$$x - \sigma_j \;=\; \sum_{i=1}^{N+1} w_{ij}\, e_i$$

et il faut trouver $N+1$ poids w_{ij} au lieu de N poids et une valeur σ_j (on a donc exactement le même nombre d'inconnues). La figure 7 illustre cette équivalence sur l'exemple de la figure 3.

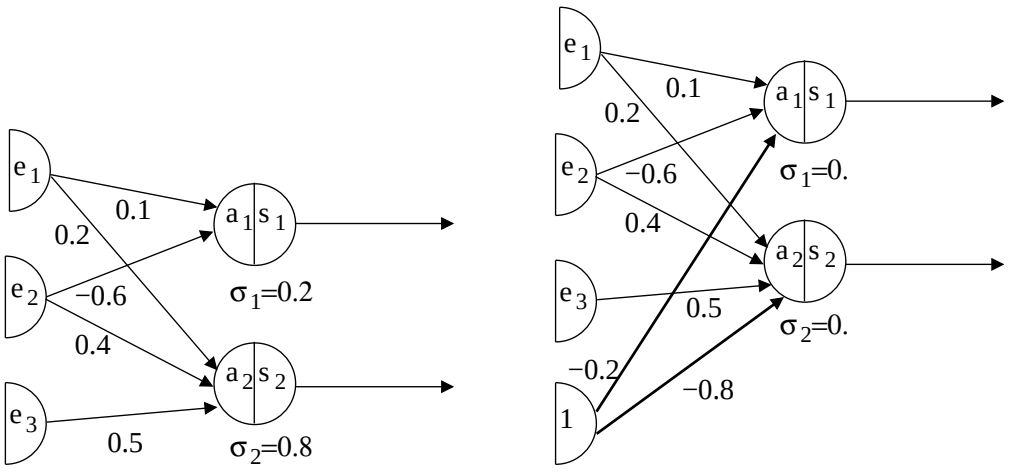


FIGURE 7 – A gauche couche de la figure 3 : la fonction sigmoïde est différente pour chaque sortie. A droite, réseau équivalent doté d'une entrée supplémentaire ayant la valeur 1 et tel que la fonction d'activation ait la même valeur nulle pour σ.

Par simplification, nous oublierons cet aspect dans la suite de ce chapitre et nous considérerons les fonctions sigmoïde avec $\sigma = 0$ en supposant que le concepteur du réseau a bien ajouté une entrée unité pour chaque couche (donc pour chaque neurone).

$$f(x) \;=\; \frac{\mathrm{e}^{Kx}}{\mathrm{e}^{Kx} + 1}$$

On notera au passage que le paramètre K est le même pour tous les neurones du réseau, ce qui est un choix bien pratique et généralement admis.

5.2 Classification en C classes

La présentation des réseaux neuromimétique a été principalement faite dans les précédents paragraphes dans le cadre d'une classification en 2 classes. Le passage à C classes ne pose aucun problème a priori. On a alors une première architecture de sortie pour un réseau de classification (figure 8) : la valeur de la sortie est alors un entier correspondant au numéro de la classe (valeur entière entre 1 et C).

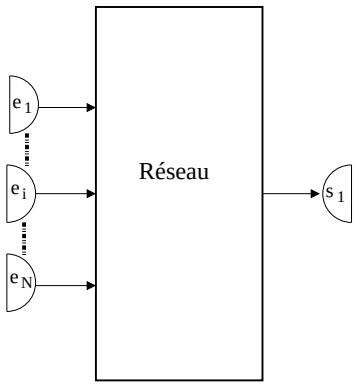


FIGURE 8 – Réseau pour la classification avec N entrées et une sortie prenant la valeur de la classe (valeur entière entre 1 et C).

Cependant, il est d'usage d'utiliser plutôt un vecteur de sortie de dimension C. Pour tout individu de la classe q, la sortie désirée (étiquette) est le vecteur $\mathbf{d}$:

$$\mathbf{d} \;=\; (d_i) \;=\; (\delta_{iq}) \tag{20}$$

avec δ_{iq} symbole de kronecker (c'est à dire $\delta_{qq} = 1$ et $\delta_{iq} = 0 \;\; \forall i \neq q$).

On a donc, par exemple, les C vecteurs suivants comme étiquettes :

$$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

Ceci veut donc dire que la couche de sortie du réseau comporte C neurones : s_1 à s_C. L'apprentissage du réseau vise donc à ce que le réseau donne en sortie la valeur suivante pour un échantillon étiquetté en classe q :

$$\begin{cases} s_q = 1 \\ s_r = 0 \quad \forall r \neq q \end{cases}$$

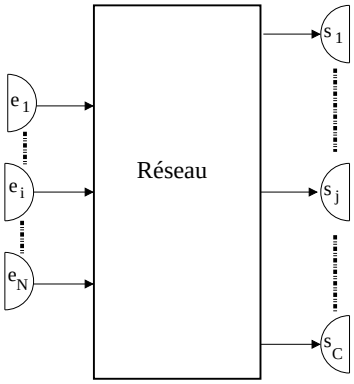


FIGURE 9 – Réseau pour la classification en C classes avec N entrées et C sorties.

et ce réseau requiert l'utilisation de la sigmoïde $f_{sigmoide,]0,1[}$. On a alors une seconde architecture de sortie pour un réseau de classification (figure 9).

Dans la réalité, le réseau ne fournira jamais une telle sortie "parfaite" dans la mesure où le paramètre k de la sigmoïde est fini : on a donc des valeurs réelles entre 0 et 1 pour tous les neurones de sortie.

$$\begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_C \end{pmatrix}$$

Il faut prévoir une méthode pour gérer la couche de sortie et décider à quelle classe appartient l'individu. Deux approches peuvent être envisagées :

- Appliquer une valeur infinie au paramètre k de la sigmoïde, c'est à dire prendre la fonction seuil. Dans ce cas le vecteur de sortie sera composé de 0 et de 1. Trois cas peuvent alors se présenter :
  - un seul neurone de sortie est à la valeur 1. Dans ce cas, on connaît la classe de sortie désignée par le réseau.
  - aucun neurone de sortie n'a la valeur 1 : le réseau ne peut classer l'échantillon.
  - plusieurs neurones de sortie ont la valeur 1 : le réseau ne sait pas classer l'échantillon.

  On voit donc poindre la notion d'échantillon non classé, c'est à dire que naturellement apparaît une classe de rejet.
- garder une valeur finie au paramètre k. Dans ce cas les neurones de sortie ont des valeurs quelconques entre 0 et 1 et il faut choisir une méthode pour décider comment associer ces valeurs à une classe. Le plus simple est de choisir l'étiquette la plus proche en calculant la distance euclidienne des sorties avec les étiquettes et en choisissant la distance la plus petite. Vue la définition des étiquettes (relation 20), cela revient à choisir la sortie s_r la plus proche de 1. Là aussi on peut analyser plusieurs cas :
  - une seule valeur s_r est proche de 1 et les autres valeurs sont proches de 0. Il n'y a alors aucun doute possible sur le choix de la classe.

– les valeurs prenent des valeurs quelconques entre 0 et 1. On peut décider de conserver effectivement la classe r à condition que les autres valeurs s_q soient vraiment plus petites que s_r. On introduit alors un critère de décision γ qui revient à choisir la classe r si

$$s_r \geq s_q + \gamma \quad \forall q \neq r$$

Si ce test n'est pas vérifié, on classe l'échantillon dans la classe de rejet.
On a ainsi rajouté une couche de décision donnant le numéro de la classe et permettant naturellement l'introduction d'une classe de rejet(figure 10).

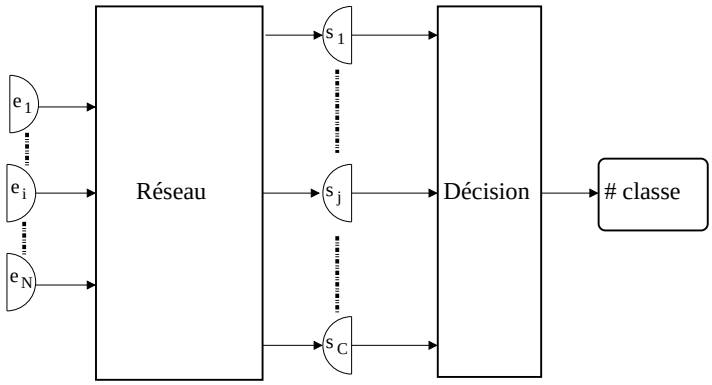


FIGURE 10 – Réseau pour la classification avec N entrées et une sortie prenant soit la valeur de la classe (valeur entière entre 1 et C), soit une valeur indiquant que c'est la classe de rejet qui doit être choisie.

5.3 Stratégies d'apprentissage

5.3.1 Rappel sur les termes d'erreur

Un utilisateur d'outil de classification est en général bien informé des performances intrinsèques de cet outil par le taux de bonne classification (équation 2) : cette valeur résume assez bien certaines qualités d'un classificateur. Bien évidemment, ce serait la matrice de confusion qui serait le meilleur indicateur de la qualité d'un classificateur.

Les réseaux neuromimétiques ont intrinsèquement besoin d'un autre indicateur : le coût, qui est en général l'erreur quadratique moyenne. C'est ce coût qui permet de piloter la phase d'apprentissage des poids du réseau. C'est ce même coût qui permet aussi d'avoir une idée sur le potentiel de généralisation d'un réseau.

A partir de ces deux grandeurs, lors de la mise en œuvre d'un algorithme neuromimétique, il est possible de définir un critère d'arrêt dépendant de l'un ou de l'autre de ces grandeurs, voire des deux.

– L'utilisateur peut fixer un objectif au taux de bonne classification. Si cette valeur est atteinte, alors l'algorithme s'arrête.
– l'utilisateur peut fixer un seuil au coût. Si la valeur du coût est inférieure à ce seuil, alors l'algorithme s'arrête.

5.3.2 Modalités opératoires en phase d'apprentissage

Un des aspects fondamentaux des réseaux neuromimétiques est de requérir une base servant à l'apprentissage, c'est à dire de disposer de R individus dotés de leur étiquette.

Utiliser la base étiquetée en phase d'apprentissage peut se concevoir de deux manières différentes :

- Une modalité d'apprentissage global. Dans ce cas, on évalue l'erreur quadratique moyenne sur la totalité de la base, selon l'équation 6

$$J \;=\; \sum_{p=1}^{R} (s_p \;-\; d_p)^2$$

  On analyse alors à chaque étape tous les échantillons de la base, puis on calcule l'erreur quadratique moyenne et on corrige les poids selon la règle Delta. Cette méthode peut être utilisée, mais elle ne donne pas toujours de bon résultats car elle débouche souvent sur des minima locaux pour la fonction coût.
- Une modalité d'apprentissage échantillon par échantillon. On choisit un échantillon p dans la base servant à l'apprentissage et on évalue son erreur quadratique.

$$J \;=\; (s_p \;-\; d_p)^2$$

  Pour les perceptrons monocouches, on peut alors en calculer le gradient (équation 12) :

$$\nabla_{i,p}(J) \;=\; 2e_{i,p}\;(s_p \;-\; d_p)\;\left.\frac{\partial f}{\partial x}\right|_{x=a_{i,p}}$$

  On modifie alors itérativement les poids selon la règle Delta, c'est à dire, dans le cas monocouche, en appliquant la règle (relation 10) :

$$w_{i'} \;=\; w_i \;-\; \eta\nabla_{i,p}(J)$$

  Les formules pour le perceptron à une couche cachée sont données par les relations 18 et 19.

C'est cette dernière méthode qui est appliquée en général dans les réseaux neuromimétiques : on utilise tous les échantillons de la base servant à l'apprentissage un par un et on modifie les poids pour diminuer l'erreur quadratique.

La mise en œuvre de cette technique requiert quelques précautions :

- Pour utiliser tous les échantillons de la base servant à l'apprentissage, on "visite" cette base en accédant aux échantillons de manière aléatoire (cela devrait éviter de présenter consécutivement au réseau deux individus de la même classe). Pour chaque échantillon on modifie les poids en calculant le gradient pour chaque poids.
- On appelle époque, l'étape complète de visite de la base servant à l'apprentissage. Un apprentissage complet se déroulera donc sur un grand nombre d'époques, ce qui correspond à un certain nombre de "balayages" de la base.

On voit que dans cette démarche, les poids du réseau seront progressivement modifiés et devraient petit à petit tendre vers des valeurs assurant de bons résultats finaux. D'une certaine manière, on peut dire que cette méthode stochastique effectue une sorte de recuit simulé en ce sens que l'on fait tout ce qui semble possible pour éviter de tomber dans des optima locaux.

On peut envisager la modification de la valeur du paramètre k de la fonction sigmoïde tout au long de l'apprentissage. En effet, au début, ce paramètre peut être petit (c'est à dire correspondant à une grande température, ce qui permet des variations fortes des poids si besoin) : ceci permet d'éviter d'être piégé par des minima locaux. En fin d'apprentissage, on peut donner au paramètre k une valeur plus forte, voire tendant vers l'infini (on s'approche alors de la fonction seuil qui ne permet pas d'apprentissage puisque non dérivable).

Le paramètre η lié à l'algorithme de descente de gradient peut lui aussi être modifié tout au long de l'apprentissage.

Dans toute phase d'apprentissage, il faut définir un critère d'arrêt : il n'y a en effet peu de chance d'atteindre une erreur nulle. Plusieurs critères peuvent être associés :

- on fixe le nombre maximal d'époques (ce qui revient à limiter le temps d'apprentissage).
- on fixe une valeur τ de bonne reconnaissance à atteindre.
- on fixe une erreur quadratique minimale à atteindre.

Cependant, cette démarche n'est pas correcte car on peut toujours trouver une architecture permettant d'avoir une valeur de τ quasiment égale à 1 pour une base donnée. Intuitivement, on peut se dire que pour une base étiquetée de R échantillons, un réseau doté de R poids devrait amener un résultat presque parfait (problème classique d'un système de R équations à R inconnues). Donc la qualité d'un apprentissage ne peut se mesurer à l'aune de la base qui sert à construire le réseau : il faut donc avoir aussi une seconde base étiquetée sur laquelle on aura une information crédible sur le réseau obtenu.

5.3.3 Base d'apprentissage et base de reconnaissance

Il est donc indispensable, avant tout apprentissage d'un réseau neuromimétique, d'avoir non seulement une base servant à l'apprentissage (c'est à dire utilisée pour trouver les poids optimaux dans le réseau), mais aussi une base servant à valider et à quantifier les résultats. En pratique, si l'on dispose d'une base étiquetée de $\tilde{R}$ individus, on la découpera en deux bases :

- une base d'apprentissage de R individus
- une base de reconnaissance de R' individus (avec $R + R' = \tilde{R}$).

La base d'apprentissage doit être, en général, mieux fournie que la base de reconnaissance, et l'expérience montre que l'on peut prendre par exemple $R/R' = 4$.

Ainsi découpée, la base étiquetée a donc deux finalités :

- ajuster les poids du réseau par la règle Delta grâce aux échantillons de la base d'apprentissage.
- Evaluer la qualité de la classification grâce aux échantillons de la base de recon-

naissance. C'est sur cette base que l'on calcule le taux de bonne classification et l'erreur quadratique moyenne.

5.3.4 Un exemple classique d'algorithme neuromimétique

On a maintenant le déroulement final de la phase d'apprentissage d'un réseau de neurones, qui peut se dérouler selon plusieurs étapes :

- **Etape 1** : Initialiser l'époque à la valeur $t = 0$. Initialiser les poids $w_{ij}^{(q)}$ de manière aléatoire. Choisir une valeur pour η (paramètre de la descente de gradient).
- **Etape 2** : Effectuer successivement les opérations suivantes sur tous les échantillons de la base d'apprentissage
  - Calculer pour chaque individu tous les gradients $\nabla_{ij}^{(q)}$ correspondant à tous les poids $w_{ij}^{(q)}$ du réseau.
  - Modifier tous les poids selon la règle :

$$w_{ij}^{(q)} \quad \rightarrow \quad w_{ij}^{(q)} \; - \; \eta \nabla_{ij}^{(q)}$$

- **Etape 3** : Calculer le coût sur la base d'apprentissage. Si ce coût est inférieur à un seuil donné, passer à l'étape 5.
- **Etape 4** : Incrémenter l'époque : $t \rightarrow t+1$. Si t est inférieur à une valeur donnée (nombre maximal d'époques), retourner à l'étape 2.
- **Etape 5** : Afficher le coût sur la base d'apprentissage **et** sur la base de reconnaissance. Afficher le taux de bonne classification pour la base d'apprentissage **et** pour la base de reconnaissance.

En général, on s'attend à ce que le taux de bonne classification soit meilleur pour la base d'apprentissage que pour la base de reconnaissance. De même on s'attend à ce que le coût moyen calculé sur la base d'apprentissage soit inférieur à celui calculé sur la base de reconnaissance.

Ce protocole d'apprentissage a toutefois ses limites car un phénomène peu prédictible peut apparaître, celui du sur-apprentissage. En effet, dans certains cas, le réseau peut tellement bien apprendre la base d'apprentissage que cela s'effectue au détriment de ses capacités à généraliser cet apprentissage : en pratique, on observe alors une diminution de l'erreur quadratique sur la base d'apprentissage et en même temps l'apparition d'une augmentation de cette erreur quadratique sur la base de reconnaissance. C'est pour cela que l'on interrompt toujours une phase d'apprentissage même si l'erreur quadratique est loin d'être nulle.

5.3.5 Utilisation d'une base de test (ou base d'évaluation)

Un dernier artefact peut apparaître dans une étape d'apprentissage de réseaux neuromimétiques : celui où la base de reconnaissance "colle" trop bien à certains échantillons de la base d'apprentissage. Dans ce cas, tout se passe comme si on effectuait les tests sur la base d'apprentissage, ce qui, comme nous l'avons déjà vu, n'a pas de sens.

Aussi, si la base étiquetée initiale est assez grande, on peut envisager de la découper en trois bases : la base d'apprentissage (qui sert à trouver les bons poids), la base de reconnaissance (qui sert à fixer le test d'arrêt et à donner les performances du réseau) et enfin la base de test sur laquelle on vérifie si les performances affichées sont réellement celles que l'on peut attendre du réseau obtenu.

Cette base de test peut s'avérer très utile lors d'une phase de recherche de l'architecture idéale d'un réseau, mais ne doit en aucun cas être utilisée pour optimiser l'architecture [7].

6 Ce qu'il faut retenir des réseaux neuromimétiques

6.1 Théorème d'existence

L'échec relatif du perceptron monocouche a été scellé dans les années 60 dans les travaux de Minsky et Pappert [2] où il a été montré que seuls les problèmes linéairement séparables pouvaient être résolus par un perceptron. A cette époque, bon nombre de chercheurs ont abandonné cette piste et il a fallu donc attendre les années 80 pour que soient enfin traités les perceptrons multicouches dont le potentiel est très grand puisqu'ils semblent pouvoir résoudre n'importe quel problème de classification réaliste.

La raison du succès des perceptrons multicouche provient d'un théorème mathématique : le théorème de Kolmogorov (1957) qui ne semble avoir aucun lien avec les perceptrons. Ce théorème [8] montre qu'il est possible d'approximer n'importe quelle fonction de plusieurs variables par un ensemble de fonctions à une variable. Appliqué au réseaux neuromimétiques, ce théorème permet de dire que tout classificateur non linéaire associant des entrées à une sortie peut se modéliser par des fonctions de chaque entrée.

La mise en œuvre du théorème de Kolmogorov n'est pas du tout simple (voir par exemple [1]), voire même totalement déconseillée (les fonctions de transition peuvent n'être plus du tout douces). Cependant, il n'en demeure pas moins que la piste théorique existe et, grâce à elle, on démontre qu'un réseau doté de deux couches cachées peut traiter tous les problèmes. Ceci démontre l'existence du réseau, mais ne donne aucun moyen de savoir à quoi il ressemble.

En pratique, on peut donc attendre d'excellentes performances d'un réseau neuromimétique. Cependant, si le découpage en classes semble a priori compliqué, donc fortement non linéaire, il faut s'attendre à devoir choisir une architecture "lourde" : il n'est pas dit que l'algorithme d'apprentissage converge alors vers un minimum suffisament général pour assurer une bonne classification de la base de reconnaissance.

C'est donc pour cela qu'il ne faut pas hésiter à choisir une architecture simple (une couche cachée, globalement un nombre de neurones cachés du même ordre de grandeur que ceux des entrée et des sorties) et, au vu des résultats, complexifier l'architecture

7. Un industriel à la recherche du meilleur classificateur peut fournir une base étiquettée à un fournisseur de classificateur, mais peut se garder une base de test pour comparer divers fournisseurs.

8. qui n'a rien à voir avec les approches neuromimétiques puisqu'il visait à résoudre la trentième conjecture d'Hilbert !!

en ajoutant éventuellement une seconde couche cachée ou en jouant sur le nombre de neurones cachés.

6.2 Performances et robustesse des réseaux neuromimétiques

Une fois choisie l'architecture, l'utilisation d'un réseau neuromimétique requiert une base étiquetée découpée en une base d'apprentissage et en une base de reconnaissance (voire aussi en une base de test). Ce découpage pourrait apparemment pénaliser des classes ayant peu de représentants dans la base initiale.

Or, à la différence des techniques bayésiennes, un réseau de neurones peut tout à fait s'adapter à la sous représentation d'une classe. En effet, si on analyse bien l'algorithme de rétropropagation du gradient, disposer d'un grand nombre d'individus d'une classe revient à présenter plus souvent un échantillon de cette classe : le réseau aura donc effectivement tendance à adapter ses poids pour bien classer les individus de cette classe. Néanmoins, les autres représentants des autres classes auront leur rôle à jouer dans la modification des poids (moins souvent, mais tout aussi efficacement). Au final, il n'est donc pas si surprenant que cela d'avoir un taux de bonne classification et une matrice de confusion "corrects".

6.3 Amélioration des capacités de généralisation

En pratique, on attend surtout d'un réseau neuromimétique d'être capable de généraliser son apprentissage, c'est à dire d'être capable d'attribuer une classe à un individu dont la ressemblance avec les individus de la base d'apprentissage n'est guère convaincante. Globalement, les réseaux ont cette propriété. De plus des techniques existent pour améliorer cette capacité de généralisation. Deux méthodes semblent être les plus performantes :

- Ajouter à la fonction de coût un terme portant sur les poids pour éviter que certaisn d'entre eux prennent des valeurs beaucoup trop grandes. Ce processus porte le nom de régularisation.
- Ajouter des individus dans la base d'apprentissage en bruitant les données des individus initiaux de la base. Cette approche est aisée à mettre en œuvre. De plus, l'utilisateur peut avoir une idée sur le niveau de bruit qu'il peut injecter dans cette étape.

On trouvera dans [1] l'équivalence de ces deux approches.

6.4 Le rôle des prétraitements

Dans les années 80, des résultats spectaculaires ont été obtenus visant à prouver l'universalité des méthodes neuromimétiques. Par exemple, pour apprendre à classifier des sinusoïdes pures, un réseau à l'architecture savamment choisie pouvait retrouver la transformée de Fourier : à la fin de l'étape d'apprentissage, les poids du réseau approchaient les vrais coefficients de Fourier requis pour ce travail de classification. Si l'on analyse objectivement cette étonnante expérimentation, on peut dire que c'est

beaucoup demander à un réseau neuromimétique que de redécouvrir par l'expérience la théorie de Fourier. A contrario, un connaisseur des méthodes utilisées en traitement du signal pourra "aider" la convergence de son réseau en introduisant des prétraitements avant l'entrée du réseau.

Comme autre exemple de prétraitement "évident", prenons le plan, un cercle centré à l'origine et une base d'apprentissage constitués par deux classes : les points situés à l'intérieur du cercle et les points situés à l'extérieur du cercle. On trouvera toujous un réseau neuromimétique capable d'effectuer correctement cette classification [9]. Mais, si au lieu d'entrer les coordonnées (x, y) des points, on utilise la valeur $d = \sqrt{x^2 + y^2}$, un classificateur linéaire donnera aussi bien le résultat.

Il n'y a aucune règle permettant de définir les meilleurs prétraitements à utiliser : on peut penser que c'est l'expert, qui connaît bien ses données, qui saura trouver les meilleures méthodes de prétraitement.

6.5 Les réseaux de neurones en pratique

Les applications des réseaux neuromimétiques sont à l'heure actuelle multiples et variées. Les questions qui se poseront toujours lors de la mise en œuvre d'une telle méthode se focalisent bien évidemment sur la pertinence d'un algorithme "aveugle" dans le problème posé. Il est néanmoins instructif de se poser les questions suivantes :

- Quelle est la représentativité de ma base de données ?
  - nombre des échantillons et de classes
  - bruit sur les vecteurs d'état
  - qualité de l'étiquetage des données
- Quelle est la dimension de mon vecteur d'état ?
  - trop grande ⇒ nécessité de la diminuer.
  - faible ⇒ possibilité de l'augmenter en prétraitant les données.
- Quelle est la dimension de ma sortie (nombre de classes, ...)
- Quelle est la connaissance a priori dont je dispose sur le système

Aucun ouvrage ne permettra de répondre directement à ces question en terme de méthode à employer (chaque problème détient ses réponses à travers des prétraitements spécifiques), mais on peut quand même noter les évidences suivantes :

- plus les degrés de liberté d'un système sont élevés, plus il peut "diverger" ce qui se traduira par une faible capacité de généralisation. Il est donc recommandé d'éviter un trop grand nombre de poids.
- plus les degrés de liberté sont faibles, plus le système est contraint dans une représentation trop restritive et non satisfaisante (à commencer sur la base d'apprentissage elle même).

Ne reste alors que la sagesse du concepteur pour assurer un résultat crédible selon les multiples critères qu'il s'est lui même dicté.

9. On laisse le soin au lecteur de trouver l'architecture requise pour une telle classification

6.6 La pratique des réseaux de neurones

Pour ceux qui voudraient utiliser des réseaux neuromimétiques, il n'est heureusement plus requis à l'heure actuelle d'écrire soi-même son propre réseau. Un certain nombre de plateformes parfaitement opérationnelles existent :

- la "Neural Network Toolbox" de MATLAB : une partie de l'outil neuromimétique est dédié à des problèmes de reconnaissance.
- SNNS (Stuttgart Neural Network Simulator) une plateforme "recherche" téléchargeable sur le net (linux, windows, java).
  http ://www.ra.cs.uni-tuebingen.de/SNNS/
- pour les élèves de Télécom ParisTech, l'outil RNSat, développé en 2001 par un groupe d'élèves (en java) et qui permet de tester tous les paramètres décrits dans ce document avec des exemples pris dans le domaine de l'imagerie satellitaire..

Références

[1] C.M. Bishop *Neural Networks for Pattern Recognition* Oxford University Press, 1994

[2] M.L. Minsky, S.A. Pappert *Perceptron* Cambridge, 1969

CHAPITRE 7 : CHAÎNES DE MARKOV ET MODÈLES DE MARKOV CACHÉS

Marc Sigelle

Motivation et Plan

Les chaînes de Markov forment depuis plusieurs dizaines d'années un sujet de choix aussi bien au niveau des investigations mathématiques [Meyn and Tweedie(1993)] que des applications à base de chaînes de Markov cachées en Reconnaissance des Formes [Cornuéjols and Miclet(2002)] : Traitement de la Parole, Traitement de l'Écriture, modèles de langages etc.. Au niveau théorique on considère souvent des modèles à temps discret ou continu (dans ce cas on parle de processus stochastiques de Markov) avec des espaces d'états finis ou infinis aussi bien discrets que continus. Nous nous limiterons ici au cas du **temps discret** et à un espace d'états **fini.** Nous avons vu au Chapitre 2 que la reconnaissance consiste à comparer un échantillon de test (par exemple en effectuant une programmation dynamique) :

- avec un ensemble de référence
- par une "distance" appropriée.

Ce processus devient coûteux lorsqu'il y a beaucoup de références, par exemple dans les grandes bases de données actuelles. De plus il s'agit d'estimer ou de modéliser les distances élémentaires, ce qui n'est pas trivial. On trouve donc ici l'intérêt d'une l'approche stochastique, où :

- un "modèle " remplace l'ensemble de références.
- les probabilités sont calculées par apprentissage (ce qui remplace les distances).

Le plan de ce chapitre sera le suivant :

1) introduction : processus stochastique(s)
2) chaînes de Markov - exemples
3) chaînes de Markov cachées (HMMs) - exemples
4) apprentissage avec les HMMs - exemples

1 Modèle stochastique

Un processus aléatoire est lié à une variable d'état pouvant changer de valeur au hasard aux instants $t = 1, 2 \dots T$. On définit alors :

- la variable aléatoire (v.a.) associée : $X(t) =$ état observé au temps t, notée aussi Q_t .
- avec les notations, qui sont équivalentes : $X(t) = i$ ou $Q_t = q_t$.
- l'évolution du système est donc décrite par une suite de transitions depuis l'état initial q_1.
- $\Rightarrow$ problème : connaître les chaînes de transition $q_1 \to q_2 \to \dots q_t \quad \forall t \leq T$.

On peut pour cela calculer la loi d'évolution du système *ie.* la probabilité jointe d'une séquence d'états en utilisant la formule générale et fondamentale [1] : $P(A,\ B) = P(B\ /\ A)\ P(A)$

$$P(q_1,\ q_2,\ \dots q_T) = P(q_T\ /\ q_1 \dots q_{T-1})\ \underbrace{P(q_1 \dots q_{T-1})}$$

$$\text{''} \qquad\qquad P(q_{T-1}\ /\ q_1 \dots q_{T-2})\ P(q_1 \dots q_{T-2})$$

$$= \quad P(q_1)\ P(q_2\ /\ q_1)\ P(q_3\ /\ q_1\ q_2) \dots P(q_T\ /\ q_1 \dots q_{T-1}) \qquad (1)$$

En résumé pour calculer cette loi jointe $P(q_1,\ q_2,\ \dots q_T)$ il faut donc connaître :

- la probabilité initiale $P(q_1)$.
- les probabilités conditionnelles $P(q_t\ /\ q_1 \dots q_{t-1}) \quad \forall t = 2 \dots T$.

2 Chaîne de Markov à temps discret et espace d'états fini

On considère un espace d'états $\{1, 2 \dots M\}$ ensemble fini.

- la propriété de Markov d'ordre k dit une dépendance limitée aux k instants précédents :

$$P(q_t\ /\ q_1 \dots q_{t-1}) = P(q_t\ /\ \underline{q_{t-k} \dots q_{t-1}})$$

- en général l'ordre d'une chaîne de Markov est 1 ou 2. Dans le cas usuel $k = 1$ on a :

$$P(q_t\ /\ q_1 \dots q_{t-1}) = P(q_t\ /\ q_{t-1}) \quad \forall t$$

$$\Rightarrow P(q_1,\ q_2,\ \dots q_T) \quad = \quad P(q_1)\ P(q_2\ /\ q_1)\ P(q_3\ /\ q_2) \dots P(q_T\ /\ q_{T-1}) \quad \forall T \qquad (2)$$

[1] nous omettrons souvent dans la suite la variable aléatoire dans la probabilité que celle-ci ait une valeur donnée, par exemple : $P(Q_t = q_t) \to P(q_t)$.

2.1 chaîne de Markov (d'ordre 1) stationnaire

Le système est décrit par des probabilités de transition $i \to j$ qui ne dépendent pas du temps :

$$P(Q_t = j \ / \ Q_{t-1} = i) = P(Q_{t+k} = j \ / \ Q_{t+k-1} = i) = a_{ij}$$

On définit alors :

$A = [\ a_{ij}\ ]$	matrice des probabilités de transition	$M \times M$
$\Pi = [\ \pi_i\ ]$	probabilités initiales (d'avoir une valeur d'état donnée)	M
$\pi_i = P(Q_1 = i)$		

Les formules suivantes ont bien sur lieu (normalisation) :

$$\forall i \in \{1 \dots M\} \quad 0 \leq \pi_i \leq 1 \quad \text{et} \ \sum_{i=1}^{M} \pi_i = 1$$

$$\forall i,\ j \in \{1 \dots M\} \quad 0 \leq a_{ij} \leq 1 \quad \text{et} \ \sum_{j=1}^{M} a_{ij} = 1$$

2.2 cas d'une station météo, étude de l'évolution du temps [Rabiner(1989)]

3 états : 1 = pluie, 2 = nuages , 3 = soleil . Le modèle est représenté par les matrices A et π :

$$A = \begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{pmatrix} \quad \text{on observe 3 = soleil à } t = 1 = \text{lundi}$$

$\to$ quelle est la probabilité P que le temps au cours du reste de la semaine soit

soleil	soleil	soleil	pluie	pluie	soleil	nuages ?
3	3	3	1	1	3	2

$$P(Q_1 = 3,\ Q_2 = 3,\ Q_3 = 3,\ Q_4 = 1,\ Q_5 = 1,\ Q_6 = 3,\ Q_7 = 2 \ / \ \text{modèle})$$

$$= \Pi_3 \ P(Q_2 = 3 \ / \ Q_1 = 3) \ P(Q_3 = 3 \ / \ Q_2 = 3) \ P(Q_4 = 1 \ / \ Q_3 = 3) \ \times$$

$$P(Q_5 = 1 \ / \ Q_4 = 1) \ P(Q_6 = 3 \ / \ Q_5 = 1) \ P(Q_7 = 2 \ / \ Q_6 = 3)$$

$$= 1 \ (a_{33})^2 \ a_{31} \ a_{11} \ a_{13} \ a_{32} = 7.68.10^{-4}$$

2.3 "durée de vie d'un état" [Rabiner(1989)]

Sachant que le système est dans l'état $i \to$ quelle est la probabilité qu'il y reste la durée d ?

$$P_i(d) = P(Q_1 = i,\ Q_2 = i \ \dots Q_{d-1} = i,\ Q_d = i,\ Q_{d+1} \neq i)$$

$$= P(Q_1 = i) \ P(Q_2 = i \ / \ Q_1 = i) \dots P(Q_d = i \ / \ Q_{d-1} = i) \ P(Q_{d+1} \neq i \ / \ Q_d = i)$$

$$= (a_{ii})^{d-1} \ (1 - a_{ii})$$

On obtient là un modèle exponentiel caractéristique. La durée moyenne de vie de l'état i est :

$$\mathbb{E}[D] = \sum_{d=1}^{+\infty} d \ P_i(d) = \sum_{d=1}^{+\infty} d \ (a_{ii})^{d-1} \ (1 - a_{ii}) = \frac{1}{1 - a_{ii}}$$

3 HMM hidden Markov models - chaînes de Markov cachées

Il s'agit de deux processus stochastiques dont l'un est caché *ie.* :

- une suite de v.a. cachée = suite d'**états** $q_1,\ q_2 \dots q_t, \dots q_T$
- l'autre, observable = suite de **symboles** (observations) produits en chacun de ces états $o_1,\ o_2 \dots o_t, \dots o_T$

La différence fondamentale avec les modèles classiques est que l'on on n'observe pas directement les états mais seulement les symboles produits par ces états. De plus on "code" l'évolution temporelle dans la séquence d'états (cachés), dont l'espace des valeurs possibles est en général de cardinal beaucoup plus faible que celui des observations.

3.1 caractérisation d'un HMM (λ) : deux hypothèses fondamentales

- Indépendance conditionnelle des observations :

$$P(o \ / \ q, \lambda) = P(o_1 \dots o_T \ / \ q_1 \dots q_T, \lambda) = \prod_{t=1}^{T} \underbrace{P(o_t \ / \ q_t, \lambda)}_{b_j(o_t)} \qquad \swarrow (q_t = j)$$

- La loi a priori sur la séquence d'états q est une chaîne de Markov stationnaire d'ordre 1 :

$$P(q \ / \ \lambda) = \underbrace{P(q_1 \ / \ \lambda)}_{\pi_i} \prod_{t=1}^{T-1} \underbrace{P(q_{t+1} \ / \ q_t)}_{a_{ij}} \qquad \swarrow (q_t = i \ , \ q_{t+1} = j)$$

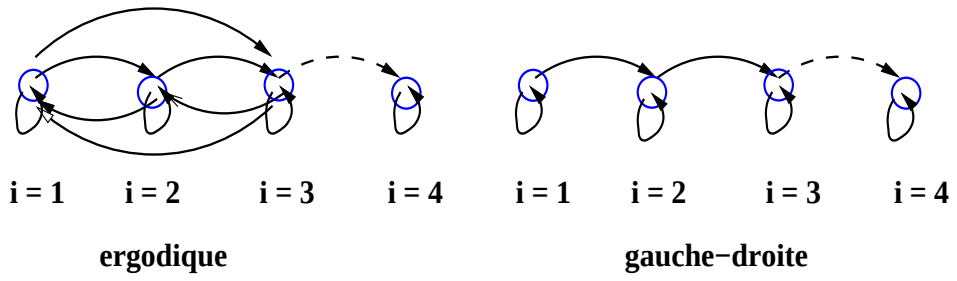
3.2 définition et notations pour un HMM à temps discret

un modèle HMM $\lambda = (A, B, \pi)$ est alors défini par :

S	=	$\{1 \dots M\}$ ensemble des états du modèle M états	
V	=	$\{V_1 \dots V_N\}$ ensemble des N symboles observables dans chaque état	$(N >> M)$
A	=	$[\ a_{ij}\ ]$ matrice des probas de transition (stationnaire)	$M \times M$
Π	=	$[\ \pi_i\ ]$ probas initiales	M
B	=	$[\ b_j(o_t)\ ]$ matrice des probas des symboles émis dans chaque état j	
avec		$b_j(o_t) = P(O_t = o_t \ / \ q_t = j)$ (stationnaire)	$M \times N$

3.3 caractéristiques des modèles HMM

La topologie désigne le graphe des transitions possibles entre états :



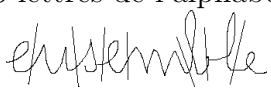
ergodique **gauche−droite**

$a_{ij} > 0 \quad \forall i, j \in \{1..M\}$ $a_{ij} > 0$ si $j = i$ ou $j = i + 1$

3.4 1er exemple : reconnaissance de l'écriture cursive (minuscules)

Les hypothèses sont les suivantes :

- il y a $M = 26$ états cachés correspondant aux 26 lettres de l'alphabet.

  Un mot est supposé déja segmenté en lettres : 

- il y a N "formes" observables, après quantification des observations. Ainsi la forme observée : '***e***' peut provenir de la lettre 'e' ou 'l' : il y a donc des confusions possibles. C'est le rôle du modèle de Markov sous-jacent de lever la confusion par la connaissance des probabilités de transitions permises entre lettres consécutives ou à l'aide d'un dictionnaire. Ainsi ***ensemble*** aura une probabilité faible d'être reconnu comme : 'enslble' .

- les formes dépendent du style et de l'algorithme de reconnaissance.

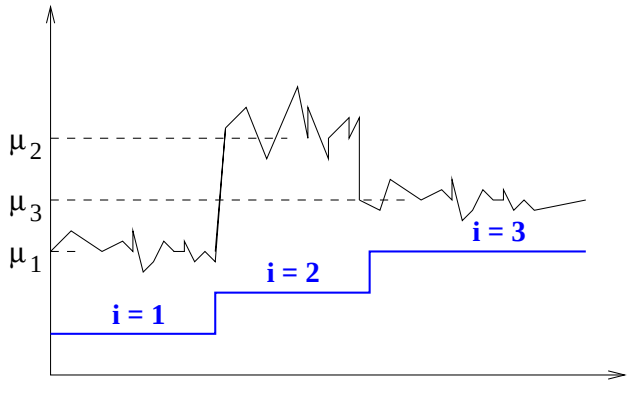
$\Rightarrow$ c'est donc un HMM avec :

- $A = \{a_{ij}\}$ probabilités de transition entre lettres : modèle de langage.
- $\Pi = \{\pi_i\}$ probabilités initiales des lettres.
- $B = \{b_j(k)\}$ probabilités des observations dans chaque état :
  cela dépend du système de reconnaissance OCR $\rightarrow$ apprentissage par comptage.

$\Rightarrow$ **but** : trouver la séquence d'états "optimale" (mot optimal).

3.5 2ème exemple : reconnaissance de la parole

Les observations sont supposées au départ continues et scalaires, pouvant être décrites en première approximation par des modèles gaussiens :



$$P(o_t \ / \ q_t = i, \ \lambda) = \frac{1}{\sqrt{2\pi}\sigma_i} \ \exp - \frac{(o_t - \mu_i)^2}{2{\sigma_i}^2}$$

En fait les descripteurs extraits à partir des observations (depuis des bancs de filtre par exemple) sont souvent vectoriels de dimension d et suivent des lois gaussiennes multi-variées :

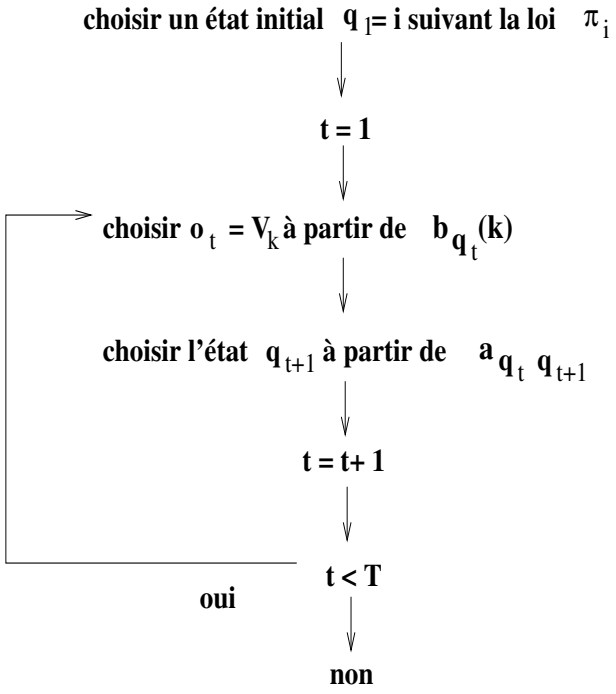
$$P(\bar{o}_t \ / \ q_t = i, \ \lambda) = \frac{1}{(2\pi)^{d/2}} \ \frac{1}{\sqrt{|\mathbf{\Sigma_i}|}} \ \exp - \frac{1}{2} \ (\bar{o}_t - \bar{\mu}_i)^t \ {\mathbf{\Sigma_i}}^{-1} \ (\bar{o}_t - \bar{\mu}_i)$$

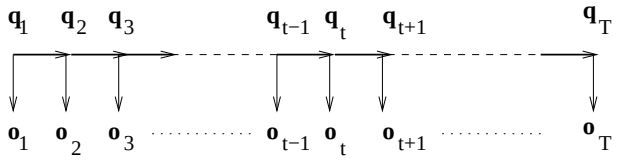
$\bar{\mu}_i$ est la moyenne de l'observation pour l'état i et $\mathbf{\Sigma_i}$ la matrice de variance-covariance associée. Cependant on affine souvent la modélisation en employant des mélanges de gaussiennes :

$$P(o_t \ / \ q_t = i, \ \lambda) = \sum_p w_p \ \mathcal{N}_p(o_t) \rightarrow \ \text{idem pour des lois multi-variées.}$$

3.6 simulation d'une HMM dont le modèle est connu

On veut synthétiser une suite d'observations suivant un modèle connu. Une séquence d'observations $o = o_1 \dots o_t \dots o_T$ est alors produite par :





4 Etapes fondamentales : apprentissage - reconnaissance

a) apprentissage

• on dispose de plusieurs modèles HMM : λ_u, en vue de leur confrontation ultérieure. Exemple : mots isolés en parole (dictionnaire), caractères isolés en écriture (A-Z,0-9).

• le nombre d'états peut dépendre du modèle, et est supposé connu par avance. Il sera toujours noté M, lorsqu'aucune confusion n'est à craindre. Exemple : chaque phonème est de durée plus ou moins longue (et possède donc plus ou moins d'états constitutifs), chaque caractère manuscrit est plus ou moins large. [2]

• on dispose pour chaque modèle d'une base d'apprentissage $\{o^{(l)}\}_{l=1..L}$, de taille L (nombre d'exemples d'apprentissage) pouvant dépendre là aussi du modèle considéré. De plus la durée de chacune des séquences peut également être variable à l'intérieur d'un même modèle.

• apprentissage = estimation des paramètres de chaque modèle λ_u : $\pi_i,\ a_{ij}\ , b_j(o_t)$

b) reconnaissance

• on dispose d'une base de test $\{o^{(r)}\}$: signaux éventuellement "dégradés".

• chaque signal de test est confronté aux divers modèles λ_u appris.

• pour un signal de test $o^{(r)}$ donné → on sélectionne le modèle le plus "vraisemblable".

• on évalue les performances de la reconnaissance globale sur l'ensemble de la base de test.

[2] L'estimation du nombre d'états (optimal) associé un modèle donné est toujours un problème ouvert.

4.1 estimation des paramètres en "données complètes"

La base de tous les raisonnements futurs est l'étude du cas (simple) où une séquence d'observation et la séquence d'états associés sont connues [3] : $(o,\ q)$. La loi **jointe** observations-états s'écrit :

$$\begin{aligned} P(o,\ q\ /\ \lambda) &= P(o\ /\ q,\ \lambda)\ P(q\ /\ \lambda) \\ &= \underbrace{P(q_1\ /\ \lambda)}_{\pi_i}\ \prod_{t=1}^{T-1}\ \underbrace{P(q_{t+1}\ /\ q_t,\ \lambda)}_{a_{ij}}\ \prod_{t=1}^{T}\ \underbrace{P(o_t\ /\ q_t,\ \lambda)}_{b_j(o_t)} \end{aligned}$$

Si on a affaire à plusieurs séquences d'observations indépendantes (*ie.* dans la phase d'apprentissage) avec leurs séquences d'états associés connues $o^{(l)}, q^{(l)}\ , l = 1..L$, alors :

$$P(o,\ q\ /\ \lambda) = \prod_{l=1}^{L} P(o^{(l)}, q^{(l)}\ /\ \lambda) = \prod_{i=1}^{M} {\pi_i}^{N_i^{(1)}}\ \ \prod_{i,j=1}^{M} {a_{ij}}^{N_{ij}}\ \ \prod_{j,k} b_j(o_t = k)^{N_{jk}} \qquad (3)$$

où :

- $N_i^{(1)} = \sum_{l=1}^{L} \mathbb{1}_{q_1^{(l)}=i}$ est le nombre de séquences d'apprentissage dont l'état initial est i .
- $N_{ij} = \sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} \mathbb{1}_{q_t^{(l)}=i,\ q_{t+1}^{(l)}=j}$ est le nombre de fois dans l'ensemble des séquences d'apprentissage où l'on rencontre le couple d'états (consécutifs) : $q_t^{(l)} = i,\ q_{t+1}^{(l)} = j$.
- $N_{jk} = \sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \mathbb{1}_{q_t^{(l)}=j,\ o_t^{(l)}=k}$ est le nombre de fois dans l'ensemble des séquences d'apprentissage où l'on rencontre le couple simultané état-observation : $q_t^{(l)} = j,\ o_t^{(l)} = k$.

On peut alors estimer les paramètres de la chaîne de Markov sous-jacente au maximum de vraisemblance, *ie.* en maximisant la probabilité jointe $P(o,\ q\ /\ \lambda)$ par rapport aux paramètres du modèle sous contraintes. En effet, ces paramètres étant des probabilités, donc normalisés à 1, l'emploi d'autant de multiplicateurs de Lagrange associés $\{\nu\}$ permet d'obtenir les estimateurs suivants.

- Ainsi, pour les probabilités initiales, définissons le lagrangien suivant :

$$\mathcal{L} = \log P(o,\ q\ /\ \lambda) - \nu \left(\sum_{i=1}^{M} \pi_i - 1 \right)$$

De (3) nous déduisons :

$$\frac{\partial \mathcal{L}}{\partial \pi_i} = \frac{\partial \log P(o,\ q\ /\ \lambda)}{\partial \pi_i} - \nu \quad = \quad \frac{N_i^{(1)}}{\pi_i} - \nu = 0$$

$$\Rightarrow \frac{N_i^{(1)}}{\pi_i} = \nu = \frac{\sum_{i=1}^{M} N_i^{(1)}}{\sum_{i=1}^{M} \pi_i}\ \text{(proportions !)} = L \ \ \Rightarrow \ \ \widehat{\pi}_i = \frac{N_i^{(1)}}{L} = \frac{\sum_{l=1}^{L} \mathbb{1}_{q_1^{(l)}=i}}{L} \qquad (4)$$

[3] on peut considérer q comme une segmentation, supposée connue, du signal observé o.

On retrouve bien pour l'état initial i la probabilité empirique en tant que quotient du nombre de séquences qui commencent par cet état sur le nombre de séquences d'apprentissage total.

• De même, pour les probabilités de transitions, définissons le lagrangien suivant :

$$\mathcal{L} = \log P(o,\ q\ /\ \lambda) - \sum_{i=1}^{M} \nu_i \left(\sum_{j=1}^{M} a_{ij} - 1 \right)$$

De (3) nous déduisons :

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = \frac{\partial \log P(o,\ q\ /\ \lambda)}{\partial a_{ij}} - \nu_i \quad = \quad \frac{N_{ij}}{a_{ij}} - \nu_i = 0$$

$$\Rightarrow \frac{N_{ij}}{a_{ij}} = \nu_i = \frac{\sum_{j=1}^{M} N_{ij}}{\sum_{j=1}^{M} a_{ij}} \text{ (proportions !)} = N_i^* \ \Rightarrow \ \widehat{a_{ij}} = \frac{N_{ij}}{N_i^*} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} \mathbb{1}_{q_t^{(l)}=i,\ q_{t+1}^{(l)}=j}}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} \mathbb{1}_{q_t^{(l)}=i}} \quad (5)$$

On trouve le quotient du nombre total de couples d'instants consécutifs où l'on rencontre les états successifs i et j sur le nombre d'instants (excepté l'instant final) où l'on est sur l'état i.

• En ce qui concerne les lois d'observation on pourrait obtenir des formules similaires pour un ensemble d'observations discret fini [4], que nous laissons au lecteur à titre d'exercice. Comme on l'a vu précédemment, les observations sont souvent continues et modélisées, dans le cas le plus simple, par des gaussiennes pures. La composante de la log-vraisemblance dépendant de ces paramètres peut s'écrire assez facilement à l'aide des fonctions indicatrices d'état :

$$LL = \sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \sum_{i=1}^{M} \mathbb{1}_{q_t^{(l)}=i} \ \left(-\frac{(o_t - \mu_i)^2}{2\sigma_i^2} - \log \sigma_i \right)$$

Il est alors assez facile de montrer en dérivant la log-vraisemblance par rapport aux divers paramètres des lois gaussiennes (moyenne et variance dans chaque état) que l'on on obtient les moyennes et variances empiriques suivantes :

$$\widehat{\mu_i} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} o_t^{(l)}\ \mathbb{1}_{q_t^{(l)}=i}}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \mathbb{1}_{q_t^{(l)}=i}} \qquad \widehat{(\sigma_i)^2} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \left(o_t^{(l)} - \widehat{\mu_i}\right)^2\ \mathbb{1}_{q_t^{(l)}=i}}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \mathbb{1}_{q_t^{(l)}=i}} \quad (6)$$

On laisse le soin au lecteur de trouver les formules équivalentes dans le cas des lois gaussiennes multi-variées.

[4] par exemple quantifié à l'aide d'un codebook.

5 Trois problèmes fondamentaux liés aux HMM [Rabiner(1989)]

Etant donnés : un modèle $\lambda = (A, B, \pi)$ (exemple : mot ou caractère isolé) et une séquence d'observations $o = o_1 \dots o_t \dots o_T$:

1) $\rightarrow$ calculer $P(o \ / \ \lambda)$.

2) $\rightarrow$ trouver une séquence d'états "optimale" a posteriori $\widehat{q} = q_1 \dots q_t \dots q_T = \arg\max_q P(q \ / \ o, \ \lambda)$

3) $\rightarrow$ étant données plusieurs séquences d'observations $o^{(l)}$ (de longueur fixe ou variable) et un modèle λ courant , réestimer les paramètres du modèle $\lambda = (A, B, \pi)$ pour maximiser (en fait augmenter) la "vraisemblance" des observations $P(\dots o^{(l)} \dots \ / \ \lambda)$.

Ce programme en trois points peut être réalisé à partir des variables backward-forward que nous allons maintenant introduire.

5.1 introduction aux variables backward-forward

On va dans ce qui suit utiliser l'aspect "**graphique**" des HMM. On rappelle ce qui connu :

- une séquence d'observations o de durée T.
- un modèle λ

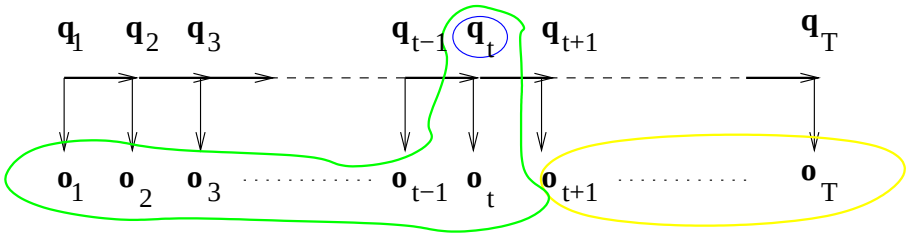


FIG. 1: aspect graphique des HMM

Essayons d'analyser l'expression suivante (Fig. 1) :

$$P(o_{t+1} \dots o_T \ / \ o_1 \dots o_t, \ q_t)$$

Si nous pensons en termes de simulation (cf. paragraphe 3.6), on voit que pour simuler la séquence $(o_{t+1} \dots o_T)$ connaissant $(o_1 \dots o_t, \ q_t)$, il suffit de connaître la valeur de q_t [5] puisque celle-ci permettra de générer les valeurs de q_{t+1}, o_{t+1} etc.. Cette remarque permet alors de définir les expressions suivantes pour chaque instant $1 \leq t \leq T$:

$$P(o_{t+1} \dots o_T \ / \ o_1 \dots o_t, \ q_t) = \boxed{P(o_{t+1} \dots o_T \ / \ \mathbf{q_t}) = \beta_t(i)} \quad \leftarrow \text{variable backward}$$

$$\boxed{P(o_1 \dots o_t, \ \mathbf{q_t}) = \alpha_t(i)} \quad \leftarrow \text{variable forward}$$

[5] qui est le noeud "entrant" vers $(o_{t+1} \dots o_T)$.

5.2 application des variables backward-forward

On a pour un instant donné t :

$$\begin{aligned} P(o,\ q_t = i\ /\ \lambda) &= P(o_1 \dots o_t,\ q_t,\ o_{t+1} \dots o_T) = P(o_{t+1} \dots o_T\ /\ o_1 \dots o_t,\ q_t)\ P(o_1 \dots o_t,\ q_t) \\ &= \underbrace{P(o_{t+1} \dots o_T\ /\ q_t)}_{\beta_t(i)}\ \underbrace{P(o_1 \dots o_t,\ q_t)}_{\alpha_t(i)} \end{aligned}$$

Donc

$$P(o,\ q_t = i\ /\ \lambda) = \beta_t(i)\ \alpha_t(i) \tag{7}$$

$$\Rightarrow P(o\ /\ \lambda) = \sum_{i=1}^{M} P(o,\ q_t = i\ /\ \lambda) = \sum_{i=1}^{M} \beta_t(i)\ \alpha_t(i) \tag{8}$$

$$\Rightarrow P(q_t = i\ /\ o,\ \lambda) = \frac{P(o,\ q_t = i\ /\ \lambda)}{P(o\ /\ \lambda)} = \frac{\alpha_t(i)\ \beta_t(i)}{\sum_{i=1}^{M} \alpha_t(i)\ \beta_t(i)} \tag{9}$$

5.3 formules à deux états

On étudie de la même façon que précédemment les probabilités suivantes faisant intervenir les états associés à deux instants consécutifs :

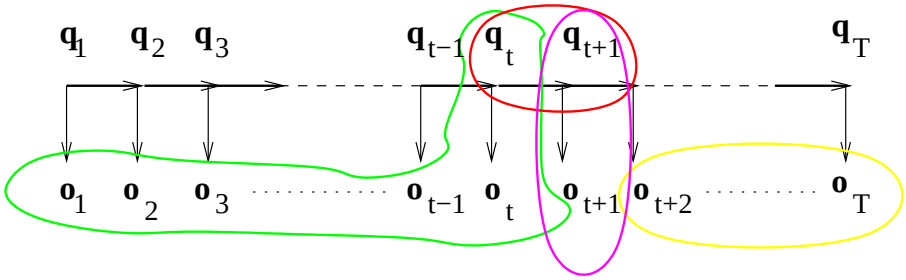


FIG. 2: aspect graphique des HMM : formules à 2 états

Décomposons la probabilité suivante :

$$P(o,\ q_t = i,\ q_{t+1} = j)$$

On a :

$$\begin{aligned} P(o,\ q_t = i,\ q_{t+1} = j) &= P(o_{t+2} \dots o_T\ /\ o_1 \dots o_{t+1},\ q_t,\ q_{t+1})\ P(o_1 \dots o_{t+1},\ q_t,\ q_{t+1}) \\ &= P(o_{t+2} \dots o_T\ /\ q_{t+1})\ P(o_{t+1}\ /\ q_{t+1})\ P(q_{t+1}\ /\ q_t)\ P(o_1 \dots o_t,\ q_t) \\ &= \beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij}\ \alpha_t(i) \end{aligned}$$

Donc

$$P(o,\ q_t = i,\ q_{t+1} = j\ /\ \lambda) \ = \ \beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij}\ \alpha_t(i) \tag{10}$$

$$P(q_t = i,\ q_{t+1} = j\ /\ o, \lambda) \ = \ \frac{P(o,\ q_t = i,\ q_{t+1} = j\ /\ \lambda)}{P(o\ /\ \lambda)} = \frac{\beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij}\ \alpha_t(i)}{\sum_{i=1}^{M} \alpha_t(i)\ \beta_t(i)} \tag{11}$$

• dans (10) "sommons" sur toutes les valeurs possibles de j :

$$P(o,\ q_t = i\ /\ \lambda) \ = \ \alpha_t(i)\ \left[\sum_{j=1}^{M} \beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij}\right] = \alpha_t(i)\ \beta_t(i) \quad \text{(d'après (7)) !}$$

$$\Rightarrow \beta_t(i) \ = \ \sum_{j=1}^{M} \beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij}$$

• dans (10) on "somme" maintenant sur toutes les valeurs possibles de i :

$$P(o,\ q_{t+1} = j\ /\ \lambda) \ = \ \beta_{t+1}(j)\ b_j(o_{t+1})\ \left[\sum_{i=1}^{M} \alpha_t(i)\ a_{ij}\right] = \alpha_{t+1}(j)\ \beta_{t+1}(j) \quad \text{(d'après (7)) !}$$

$$\Rightarrow \alpha_{t+1}(j) \ = \ b_j(o_{t+1})\ \left[\sum_{i=1}^{M} \alpha_t(i)\ a_{ij}\right]$$

5.4 calcul des variables backward-forward

En récapitulant les résultats précédents on obtient donc les formules de récurrence :

$$\alpha_{t+1}(j) \ = \ b_j(o_{t+1})\ \left[\sum_{i=1}^{M} \alpha_t(i)\ a_{ij}\right] \tag{12}$$

$$\beta_t(i) \ = \ \sum_{j=1}^{M} \beta_{t+1}(j)\ b_j(o_{t+1})\ a_{ij} \tag{13}$$

L'initialisation de ces variables se fait avec les remarques simples suivantes :

$$\alpha_T(i) = P(o, q_T = i) = \alpha_T(i)\ \beta_T(i) \Rightarrow$$

$$\begin{aligned} \beta_T(i) \ &= \ 1 \\ \alpha_1(i) \ &= \ P(o_1,\ q_1 = i) = P(o_1\ /\ q_1 = i)\ P(q_1 = i) = \pi_i\ b_i(o_1) \end{aligned}$$

6 Estimation des paramètres en "données incomplètes"

Il s'agit du problème le plus difficile de ce chapitre. On ne connaît maintenant que les séquences d'observations (pas les états). Or on veut aussi estimer les paramètres du modèle HMM, et de façon optimale ! Pour utiliser les résultats vus en données complètes, il va falloir effectuer une "statistique" combinatoire sur l'ensemble des séquences d'états cachés possibles (chemins) associés à chaque observation (séquence temporelle). On verra que c'est la loi de l'ensemble des séquences d'états $P(q \ / \ o, \lambda)$ **a posteriori** *ie.* connaissant les séquences d'observations, qui va s'imposer d'un point de vue statistique dans ce qui suit.
Pour toute v.a. U, nous noterons alors son espérance a posteriori par : $\tilde{\mathbb{E}}[U] = \mathbb{E}[U \ / \ o, \lambda]$.
Annonçons maintenant de suite les résultats :

• les paramètres de la chaine de Markov (probabilités) vérifient les équations suivantes :

$$\widehat{\pi_i} = \frac{\tilde{\mathbb{E}}[N_i^{(1)}]}{L} = \frac{\sum_{l=1}^{L} \tilde{\mathbb{E}}[\mathbb{1}_{q_1^{(l)}=i}]}{L} = \frac{\sum_{l=1}^{L} P(q_1^{(l)} = i \ / \ o^{(l)}, \widehat{\lambda})}{L} \tag{14}$$

$$\widehat{a_{ij}} = \frac{\tilde{\mathbb{E}}[N_{ij}]}{\tilde{\mathbb{E}}[N_i{}^*]} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} \tilde{\mathbb{E}}[\mathbb{1}_{q_t^{(l)}=i,\ q_{t+1}^{(l)}=j}]}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} \tilde{\mathbb{E}}[\ \mathbb{1}_{q_t^{(l)}=i}]} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)} = i,\ q_{t+1}^{(l)} = j \ / \ o^{(l)}, \widehat{\lambda})}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)} = i \ / \ o^{(l)}, \widehat{\lambda})} \tag{15}$$

D'où cela vient-il ? Prenons le cas des probabilités de transition. On a vu que

$$\frac{\partial \log P(o,\ q \ / \ \lambda)}{\partial a_{ij}} = \frac{1}{a_{ij}} \ N_{ij}(q)$$

Ici on indique bien que la quantité N_{ij} dépend de la séquence d'états courante q. Or

$$\begin{aligned} P(o \ / \ \lambda) &= \sum_q P(o,\ q \ / \ \lambda) \\ \Rightarrow \frac{\partial P(o \ / \ \lambda)}{\partial a_{ij}} &= \frac{1}{a_{ij}} \ \sum_q N_{ij}(q) \ P(o,\ q \ / \ \lambda) \\ \Rightarrow \frac{\partial \log P(o \ / \ \lambda)}{\partial a_{ij}} &= \frac{1}{a_{ij}} \ \sum_q N_{ij}(q) \ \frac{P(o,\ q \ / \ \lambda)}{P(o \ / \ \lambda)} = \frac{1}{a_{ij}} \ \tilde{\mathbb{E}}[N_{ij}] \end{aligned}$$

En reprenant les conditions de normalisation des probabilités $\sum_{j=1}^{M} a_{ij} = 1 \ \ \forall i$ sous la forme des multiplicateurs de Lagrange précédents (paragraphe 4.1) on arrive facilement à :

$$\widehat{a_{ij}} = \frac{\tilde{\mathbb{E}}[N_{ij}]}{\sum_{j=1}^{M} \tilde{\mathbb{E}}[N_{ij}]} = \frac{\tilde{\mathbb{E}}[N_{ij}]}{\tilde{\mathbb{E}}[N_i{}^*]}$$

On voit (et ceci est très général) que d'un point de vue "mnémotechnique", il suffit de remplacer au numérateur et au dénominateur les quantités déterministes par leurs espérances a posteriori.

• les paramètres des lois d'observations gaussiennes vérifient les équations suivantes :

$$\widehat{\mu_i} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} o_t^{(l)}\ \tilde{\mathbb{E}}[\mathbb{1}_{q_t^{(l)}=i}]}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \tilde{\mathbb{E}}[\mathbb{1}_{q_t^{(l)}=i}]} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} o_t^{(l)}\ P(q_t^{(l)}=i\ /\ o^{(l)},\widehat{\lambda})}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} P(q_t^{(l)}=i\ /\ o^{(l)},\widehat{\lambda})} \tag{16}$$

$$\widehat{(\sigma_i)^2} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} (o_t^{(l)}-\widehat{\mu_i})^2\ \tilde{\mathbb{E}}[\mathbb{1}_{q_t^{(l)}=i}]}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \tilde{\mathbb{E}}[\mathbb{1}_{q_t^{(l)}=i}]} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} (o_t^{(l)}-\widehat{\mu_i})^2\ P(q_t^{(l)}=i\ /\ o^{(l)},\widehat{\lambda})}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} P(q_t^{(l)}=i\ /\ o^{(l)},\widehat{\lambda})} \tag{17}$$

On obtient donc un ensemble d'équations (14) (15) et (16) (17) auto-cohérentes puisque les paramètres figurent des deux cotés des équations, et non solubles analytiquement. Une méthode itérative extrêmement puissante pour estimer ces paramètres va maintenant être abordée.

7 Algorithme EM

Cet algorithme a pour base les résultats fondamentaux obtenus en données complètes. Pour les lecteurs intéressés, on trouvera un exposé complet de l'algorithme EM dans [Redner and Walker(1984)], avec en particulier l'application importante à l'estimation de mélanges (de gaussiennes). Son principe général en est le suivant. Chaque étape de l'algorithme comprend deux phases :

- phase E (Expectation) : estimation des statistiques a posteriori courantes $\tilde{\mathbb{E}}^{(\mathbf{n})}[\ U\ ]$
- phase M (Maximisation) : remise à jour des paramètres du modèle suivant le schéma :

$$\lambda^{(\mathbf{n+1})} = \arg\max_{\lambda} \tilde{\mathbb{E}}^{(\mathbf{n})}[\ \log P(q,\ o\ /\ \lambda)\ ] \tag{18}$$

Le résultat fondamental est que la vraisemblance des paramètres croît au cours des itérations. Dans notre cas l'EM va transformer le système d'équations précédentes exactes en un système itératif. Annonçons là aussi de suite les résultats de la phase M (Maximization) :

• en ce qui concerne les paramètres de la chaîne de Markov cachés :

$$\pi_i{}^{(\mathbf{n+1})} = \frac{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_i^{(1)}]}{L} = \frac{\sum_{l=1}^{L} P(q_1^{(l)}=i\ /\ o^{(l)},\lambda^{(\mathbf{n})})}{L} \tag{19}$$

$$a_{ij}{}^{(\mathbf{n+1})} = \frac{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_{ij}]}{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_i{}^*]} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)}=i,\ q_{t+1}^{(l)}=j\ /\ o^{(l)},\lambda^{(\mathbf{n})})}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)}=i\ /\ o^{(l)},\lambda^{(\mathbf{n})})} \tag{20}$$

D'où cela vient-il ? Prenons ici encore à titre d'exemple le cas des probabilités de transition.

On forme le lagrangien : $\mathcal{L} = \tilde{\mathbb{E}}^{(\mathbf{n})}[\ \log P(q,\ o\ /\ \lambda)\ ] - \sum_{i=1}^{M} \nu_i \left(\sum_{j=1}^{M} a_{ij} - 1 \right)$.

Il faut bien noter ici que l'espérance a posteriori est prise pour les valeurs de paramètres courants, donc fixés. Comme on sait maintenant presque par coeur (!) que

$$\frac{\partial \log P(o,\ q\ /\ \lambda)}{\partial a_{ij}} = \frac{1}{a_{ij}}\ N_{ij}(q)\ , \quad \text{on en déduit}$$

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = \tilde{\mathbb{E}}^{(\mathbf{n})}[\ \frac{\partial \log P(o,\ q\ /\ \lambda)}{\partial a_{ij}}\ ] - \nu_i = \frac{1}{a_{ij}}\ \tilde{\mathbb{E}}^{(\mathbf{n})}[N_{ij}] - \nu_i \quad = \quad 0$$

$$\Rightarrow \frac{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_{ij}]}{a_{ij}} = \nu_i = \frac{\sum_{j=1}^{M} \tilde{\mathbb{E}}^{(\mathbf{n})}[N_{ij}]}{\sum_{j=1}^{M} a_{ij}}\ \text{(proportions !)} = \tilde{\mathbb{E}}^{(\mathbf{n})}[N_i^*] \ \Rightarrow\ \widehat{a_{ij}} = \frac{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_{ij}]}{\tilde{\mathbb{E}}^{(\mathbf{n})}[N_i^*]}$$

• pour les paramètres des lois gaussiennes :

$$\mu_i{}^{(\mathbf{n+1})} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} o_t^{(l)}\ \tilde{\mathbb{E}}^{(\mathbf{n})}[\mathbb{1}_{q_t^{(l)}=i}]}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \tilde{\mathbb{E}}^{(\mathbf{n})}[\mathbb{1}_{q_t^{(l)}=i}]} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} o_t^{(l)}\ P(q_t^{(l)} = i\ /\ o^{(l)}, \lambda^{(\mathbf{n})})}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} P(q_t^{(l)} = i\ /\ o^{(l)}, \lambda^{(\mathbf{n})})} \tag{21}$$

$$(\sigma_i)^{2(\mathbf{n+1})} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \left(o_t^{(l)} - \mu_i{}^{(\mathbf{n})}\right)^2\ \tilde{\mathbb{E}}^{(\mathbf{n})}[\mathbb{1}_{q_t^{(l)}=i}]}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \tilde{\mathbb{E}}^{(\mathbf{n})}[\mathbb{1}_{q_t^{(l)}=i}]} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \left(o_t^{(l)} - \mu_i{}^{(\mathbf{n})}\right)^2\ P(q_t^{(l)} = i\ /\ o^{(l)}, \lambda^{(\mathbf{n})})}{\sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} P(q_t^{(l)} = i\ /\ o^{(l)}, \lambda^{(\mathbf{n})})} \tag{22}$$

La méthode EM résout donc bien le système (14 - 17) sous une forme itérative de type "point-fixe" : $x^{(\mathbf{n+1})} = f(x^{(\mathbf{n})})$ en assurant que la vraisemblance des paramètres augmente à chaque étape. Le calcul des membres de dorite des équations associées nécessite bien de connaître

$$P(q_t^{(l)} = i\ /\ o^{(l)}, \lambda^{(\mathbf{n})}) \quad \text{et} \quad P(q_t^{(l)} = i,\ q_{t+1}^{(l)} = j\ /\ o^{(l)}, \lambda^{(\mathbf{n})}) \quad \text{(inférence)}$$

à chaque étape, ce qui se fait grâce au calcul des variables backward-forward associées à chaque séquence d'observations $o^{(l)}$ pour les valeurs courantes des paramètres du modèle. Ce calcul étant lui-même mené grâce aux formules de récursion (12) et (13) . On perçoit bien ici la complexité (au moins computationnelle) de l'ensemble des calculs mis en jeu, bien que les formules associées soient analytiquement exactes [6] dans le cadre de l'algorithme EM et pour le cas des HMMs. Il faut également noter l'importance de l'initialisation des paramètres [7] dans ce type d'algorithme puisque l'optimum atteint est local.

[6] Ceci est complètement différent dans le cadre des Champs de Markov en Traitement d'Images, où la phase d'Estimation de l'EM ne peut en général être menée de façon exacte, et doit se faire par simulation pour les valeurs courantes des paramètres.

[7] Voir remarques dans la conclusion de ce chapitre Section 9.

8 Recherche de la segmentation optimale

Rappelons ce qui est connu :

- une séquence d'observations o, de durée T.
- un modèle λ.

On cherche à trouver la segmentation optimale, c'est-à-dire une séquence d'états optimale, dans un sens à préciser, associée à la suite d'observations. Nous verrons dans la section suivante l'application pratique de la résolution de ce problème. Deux possibilités s'offrent alors :

a) on peut choisir choisir l'état q_t le plus probable de façon **locale** :
il s'agit de l'estimateur du Maximum de la Marginale a Posteriori (MPM). En effet d'après tout ce que l'on a vu précédemment on peut calculer exactement :

$$P(q_t = i \ / \ o, \ \lambda) = \frac{\alpha_t(i) \ \beta_t(i)}{P(o \ / \ \lambda)} \Rightarrow \boxed{\widehat{q_t} = \arg\max_{i \in E} \alpha_t(i) \ \beta_t(i)}$$

b) on peut choisir un chemin optimal **global** $\widehat{q}$ le plus probable :
il s'agit de l'estimateur du Maximum a Posteriori (MAP). En effet le théorème de Bayes nous permet d'affirmer que :

$$P(q \ / \ o, \ \lambda) = \frac{P(o \ / \ q, \ \lambda) \ P(q \ / \ \lambda)}{P(o \ / \ \lambda)}$$

La segmentation optimale au sens du MAP est donc telle que

$$\boxed{\widehat{q} = \arg\max_q P(q \ / \ o, \ \lambda) = \arg\max_q P(o \ / \ q, \ \lambda) \ P(q \ / \ \lambda) = \arg\max_q P(o, \ q \ / \ \lambda)}$$

$$= \arg\max_q \pi_{q_1} \ b_{q_1}(o_1) \ a_{q_1 q_2} \ b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} \ b_{q_T}(o_T)$$

Le logarithme de la quantité à maximiser

$$\log P(o \ / \ q, \ \lambda) \ P(q \ / \ \lambda)$$

s'écrit donc comme somme de termes :

- à une seule variable $\Phi(q_t) = \log b_{q_t}(o_t)$ associés à l'attache aux données (loi d'observation).
- à deux variables consécutives $\Psi(q_t, q_{t+1}) = \log a_{q_t q_{t+1}}$ associés à la chaîne sous-jacente (probabilités de transition).

$\Rightarrow$ c'est donc ici que la programmation dynamique s'impose comme méthode particulièrement adaptée à la tache de segmentation proposée.

9 Récapitulation pour les HMM

Procédons pour conclure à une récapitulation de l'approche statistique dans la modélisation et l'utilisation des HMMs, en rapport avec les trois problèmes envisagés plus haut Section 5.

9.1 apprentissage d'un modèle (λ_u)

Rappelons que nous avons décrit la méthode EM qui consiste dans les étapes suivantes :

1) initialisation des paramètres du modèle $[\ \pi_i, a_{ij}, \mu_i, \sigma_i\ ]^{(\mathbf{0})}$:
- en ce qui concerne la chaîne cachée on introduit alors souvent un instant initial "virtuel" $t = 0$ et un état initial "déterministe" $i = 0$ ($\pi_0 = 1$), en imposant :

- soit un modèle ergodique : $a_{01} = \dfrac{1}{M}$ uniforme [8] , π_i et a_{ij} également uniformes.
- ou un modèle gauche-droite : $a_{01} = 1$, de sorte que l'état à $t = 1$ est $i = 1$. Pour les valeurs des états courants on ne retient que les coefficients : a_{ii} et $a_{i\ i+1} = 1 - a_{ii}$.

- en ce qui concerne la loi d'observation supposée gaussienne, on choisit souvent des moyennes μ_i équi-réparties dans un domaine admissible d'observations, et des variances $(\sigma_i)^2$ en conséquence.

2) à l'itération (n) : on utilise les variables backward-forward $\alpha_t(i)$ et $\beta_t(i)$ qui doivent être calculées pour chaque séquence d'apprentissage $o^{(l)}$.

$\Rightarrow$ On a donc résolu ici le problème (3) .

9.2 reconnaissance : observation

On dispose d'une séquence d'obervations $o^{(r)}$ dite "de test". On veut calculer le "score" du modèle λ_u pour cette observation. On procéde pour cela en calculant la vraisemblance du modèle pour la séquence d'obervations de test :
a) soit de façon exacte d'après (7) :

$$P(o^{(r)}\ /\ \lambda_u) = \sum_{i \in M_u} \alpha_t(i)\ \beta_t(i) \qquad \text{(avec } t = 1 \text{ ou } T^{(r)} \text{ très souvent)}$$

grâce au calcul encore une fois des variables backward-forward $\alpha_t(i)$ et $\beta_t(i)$ associées à l'observation $o^{(r)}$ et au modèle λ_u.

$\Rightarrow$ On a donc utilisé ici la résolution du problème (1) .
b) soit par l'approximation de Viterbi : on ne retient dans l'expression de la vraisemblance cherchée que la séquence d'états cachés optimale *a posteriori*

$$\begin{aligned} P(o^{(r)}\ /\ \lambda_u) &= \sum_q P(o^{(r)},\ q\ /\ \lambda_u) \approx P(o^{(r)},\ \widehat{q}\ /\ \lambda_u) \\ \text{avec } \widehat{q} &= \arg\max_q P(o^{(r)},\ q\ /\ \lambda) \quad \text{(segmentation optimale MAP)} \end{aligned}$$

On emploie donc pour cela la programmation dynamique vue dans la section précédente, puis l'on calcule ensuite facilement $P(o^{(r)},\ \widehat{q}\ /\ \lambda_u) = P(o^{(r)}\ /\ \widehat{q},\ \lambda_u)\ P(\widehat{q},\ \lambda_u)$
(ou les logarithmes de chacun de ces termes) pour la séquence optimale $\widehat{q}$ ainsi trouvée.

$\Rightarrow$ On a donc utilisé ici la résolution du problème (2) .

[8] ou a_{0i} si on part de l'état i au temps $t = 1$ de façon certaine.

Références

[Cornuéjols and Miclet(2002)] A. Cornuéjols and L. Miclet. *Apprentissage artificiel. Concepts et Algorithmes*. Eyrolles, 2002.

[Meyn and Tweedie(1993)] S.P. Meyn and R.L. Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag, London, 1993.

[Rabiner(1989)] L. Rabiner. A tutorial on Hidden Markov Models and selected applications in Speech Recognition http://www.ai.mit.edu/courses/6.867-f02/papers/rabiner.pdf. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[Redner and Walker(1984)] R.A. Redner and H.F. Walker. Mixture Densities, Maximum Likelihood and the EM algorithm. *SIAM Review*, 26:195–239, 1984.