Habilitation à diriger des recherches

présentée devant

l'Université Paris Dauphine

par

Fabrice Rossi

Contribution à l'analyse des données complexes

Document de synthèse

Présenté le 23 Novembre 2006 devant le jury composé de :

Bennani	Younès	Université Paris XIII	Rapporteur
Besse	Philippe	Université Toulouse III	Rapporteur
Cazes	Pierre	Université Paris Dauphine	Coordinateur
Cottrell	Marie	Université Paris I	Rapporteur
Gallinari	Patrick	Université Paris VI	Rapporteur
Lechevallier	Yves	INRIA Rocquencourt	Examinateur
Pinson	Suzanne	Université Paris Dauphine	Examinatrice

Remerciements

Je remercie tout d'abord Pierre Cazes de m'avoir fait l'honneur de coordonner ce travail, Younès Bennani, Philippe Besse, Marie Cottrell et Patrick Gallinari d'avoir accepté d'être rapporteurs, ainsi que Yves Lechevallier et Suzanne Pinson en leur qualité de membres du Jury.

Ce travail n'aurait pu exister sans le soutien de Marie Cottrell et d'Yves Lechevallier. Quand j'ai souhaité quitter provisoirement l'université pour me consacrer pleinement à la recherche et préparer cette habilitation, Marie a recommandé ma candidature pour une délégation CNRS, Yves pour un poste similaire à l'INRIA (le choix, on s'en doute, fût difficile). Ce n'est qu'une manifestation parmi d'autres de l'aide qu'ils m'ont apportée de façon indéfectible depuis que je les connais. Je ne saurais leur exprimer suffisamment ma gratitude.

Je remercie de la même façon Michel Verleysen dont la confiance et le soutien ont été permanents. J'aurais pu inclure Michel dans le paragraphe précédent, tant notre collaboration m'a apporté, des articles écrits en commun aux participations à des comités d'encadrement, en passant par l'édition de numéraux spéciaux de *Neurocomputing*.

Je profite d'être momentanément en Belgique pour remercier les doctorants et collègues de Michel avec qui j'ai eu grand plaisir à travailler. Merci donc à Nicolas Delannay, Catherine Krier, Amaury Lendasse (qui affronte le froid de Finlande), Marc Meurens, et Vincent Wertz. Je classe Damien François à part, car outre une collaboration scientifique fructueuse, je lui dois de m'être mis à utiliser certains moyens de communication moderne que je dévoilerai pas ici pour éviter de me rendre ridicule. Merci aussi à Cédric Archambeau, John Lee, Geoffroy Simon et Frédéric Vrins, pour avoir contribué à rendre mes séjours à Louvain-la-Neuve et à Bruges encore plus agréables (et festifs).

J'ai eu le plaisir à l'INRIA de travailler dans un projet dynamique dont je remercie tous les membres. Je remercie plus particulièrement Brigitte Trousse pour avoir soutenu mes candidatures et m'avoir accueilli dans les meilleures conditions, ainsi que Stéphanie Aubin pour son efficacité et son professionnalisme rares. Merci aussi à notre collaborateur de longue date, Francisco Carvalho, et pas seulement pour les séjours (scientifiques !) au Brésil.

Je remercie aussi celles et ceux que je suis un peu gêné d'appeler mes "élèves", tant je les considère avant tout comme des collègues et amis. Le premier a été Frédérick Vautrain dont la réussite professionnelle est une magnifique récompense, pour lui et aussi un peu pour moi. Aïcha El Golli a rejoint la Tunisie avec les honneurs d'un poste de Maître assistant bien mérité, mais elle a laissé un vide bien difficile à combler dans le projet. Brieuc Conan-Guez n'est pas aussi loin, mais il manque tout autant. J'inclus dans cette liste Nathalie Villa, qui n'est pas une ex-doctorante mais qui bénéficie du privilège de l'âge. Je n'arriverais jamais à la convaincre de rejoindre la civilisation parisienne et c'est bien dommage.

Je termine ces remerciements par ceux destinés à ma famille, dans laquelle j'ose inclure mon ami Marc Csernel, que je ne suis pas loin de considérer comme un second père. Stéphanie et Joséphine n'ont pas besoin d'un discours, elles savent que je ne serais rien sans elles.

Table des matières

Introduction

Dans de nombreux domaines, comme par exemple l'analyse de documents multimédia, les observations associées aux problèmes rencontrés ne sont pas des données numériques classiques et on ne peut pas les représenter fidèlement par des vecteurs. Il est pourtant fréquent d'accepter une perte d'information en traduisant des observations complexes sous une forme vectorielle, car ceci permet l'utilisation de méthodes classiques. L'exemple le plus connu est peut être celui du texte : le modèle vectoriel simple [105] consiste à représenter un document par le vecteur des occurrences de chacun des mots du corpus considéré, éventuellement pondérées par diverses techniques. On néglige ainsi toute la structure du texte, ses enrichissements typographiques, etc.

Pour éviter la perte d'information qu'engendre la représentation vectorielle, il est de plus en plus fréquent d'utiliser des alternatives plus complexes, comme les représentations par ensemble (pour les données de taille variable), par liste (pour prendre en compte l'aspect temporel), par arbre ou graphe (pour décrire des dépendances structurelles, ou d'autre nature, entre composantes) ou par fonction (pour représenter par exemple un spectre, c'est-à-dire une fonction qui à une longueur d'onde associe une grandeur mesurée).

Mes travaux de recherche, résumés dans le présent document, portent sur l'analyse des données complexes pour lesquelles la représentation vectorielle naïve rencontre des limites. Ce domaine est animé par deux grands courants.

Une première stratégie consiste à modifier les méthodes d'analyse classiques afin de les adapter à certains types de données complexes : Somervuo propose par exemple dans [112] une version des cartes auto-organisatrices de Kohonen (le SOM, pour *Self Organizing Map* [78]) spécialement modifiée pour traiter des chaînes de caractères (et plus généralement des suites de symboles de longueurs quelconques).

Une deuxième stratégie consiste à définir des méthodes génériques applicables à une large classe de données complexes par l'intermédiaire d'une "mesure de comparaison" entre données. Cette mesure peut être une (dis)similarité ce qui conduit par exemple à la méthode de classification *Partitioning Around Medoids* [76] qui généralise les *k-means* [89]. On peut aussi s'appuyer sur un noyau, au sens des machines à noyau (cf, e.g., [109]), comme par exemple les machines à vecteurs de support. Ce deuxième courant inclut la construction de (dis)similarités et de noyaux adaptés à certains types de données complexes dans l'objectif de leur appliquer des méthodes génériques.

Mon travail apporte des contributions aux deux courants. Je me suis intéressé à l'analyse de données fonctionnelles (ADF), dont le but est la définition et l'étude de méthodes de traitement pour des observations décrites par des fonctions. Dans ce cadre, j'ai proposé des modifications et adaptations des méthodes neuronales classiques (perceptrons multi-couches, réseaux utilisant des fonctions à base radiale et cartes de Kohonen) qui leur permettent de traiter des données fonctionnelles. J'ai étudié les propriétés théoriques de ces modèles. J'ai

montré notamment que les perceptrons multi-couches fonctionnels sont des approximateurs universels. J'ai aussi étudié le problème de l'apprentissage dans ces modèles en montrant qu'on pouvait estimer de façon consistante les paramètres d'un modèle d'architecture fixe et qu'on pouvait aussi estimer la régression de T en G en adaptant l'architecture au nombre d'observations disponibles (T est à valeurs dans $\mathbb{R}$ et G dans un espace fonctionnel). Enfin, j'ai mis en œuvre ces modèles sur des données réelles, afin notamment de comparer leurs performances avec celles des solutions alternatives envisageables.

Je me suis aussi intéressé aux méthodes génériques. J'ai travaillé sur la définition de noyaux adaptés aux données fonctionnelles pour discriminer entre deux classes de fonctions grâce à une machine à vecteurs de support. J'ai montré qu'on pouvait ainsi construire un classifieur asymptotiquement optimal et j'ai étudié sa mise en œuvre sur des données réelles.

Dans le cadre de l'analyse exploratoire de données complexes, j'ai étudié des variantes des cartes auto-organisatrices de Kohonen applicables aux données décrites uniquement par un tableau de dissimilarités. J'ai notamment proposé un algorithme rapide qui permet d'analyser plusieurs milliers d'observations en quelques minutes sur un ordinateur personnel contre plusieurs heures avec la version d'origine de la variante du SOM étudiée. Cette accélération ne se fait pas au détriment de la qualité des résultats puisque je montre qu'ils sont strictement identiques à ceux obtenus par l'algorithme d'origine. J'ai appliqué cette variante du SOM à des données issues de l'usage des sites web de l'INRIA, ce qui m'a amené à étudier les dissimilarités adaptées à ce type de données.

Le reste de ce mémoire est organisé thématiquement. Le chapitre 1 regroupe les résultats concernant l'analyse de données fonctionnelles et comprend des contributions aux deux courants de l'analyse des données complexes que je viens de présenter. Le chapitre 2 concerne les méthodes basées sur des dissimilarités et regroupe les apports génériques (variante du SOM s'appliquant à toute dissimilarité) et spécifiques (analyse de l'usage des sites web).

Pour éviter d'alourdir ce texte, j'ai préféré résumer les résultats théoriques obtenus, en omettant systématiquement les preuves et une partie des hypothèses les plus techniques. Les développements complets sont contenus dans les articles annexés au mémoire.

Chapitre 1

Analyse de données fonctionnelles

Ce chapitre présente mes travaux sur l'analyse des données fonctionnelles par des techniques neuronales (au sens large). Ces travaux ont débuté fin 1998 quand j'ai commencé à encadrer la thèse de Brieuc Conan-Guez (Université Paris Dauphine et INRIA, projet AxIS). Nous avons d'abord proposé et étudié une extension des perceptrons multi-couches adaptée au traitement des données fonctionnelles et basée sur une approximation de certains calculs d'intégrales (cf section 1.2). Nous nous sommes ensuite intéressés à une méthodologie plus classique en analyse de données fonctionnelles, s'appuyant sur la représentation sur une base des fonctions étudiées (cf section 1.3.2).

J'ai commencé fin 2003 à participer à l'encadrement de Nicolas Delannay, dont la thèse est dirigée par Michel Verleysen (Université catholique de Louvain). J'ai aussi co-encadré la thèse d'Aïcha El Golli (Université Paris Dauphine et INRIA, projet AxIS) à partir de cette époque. A l'occasion de ces collaborations, j'ai poursuivi mes travaux sur l'approche par projection, notamment en l'appliquant à d'autres modèles neuronaux (réseaux utilisant des *Radial Basis Functions* et cartes de Kohonen, cf section 1.4), mais aussi en m'intéressant au problème du choix de la base de représentation des fonctions (cf section 1.4.2).

Enfin, je travaille depuis fin 2004 avec Nathalie Villa (Université Toulouse Le Mirail) sur l'utilisation des machines à vecteurs de support pour l'analyse de données fonctionnelles. Nous nous sommes intéressés , en particulier, à la construction de noyaux adaptés aux fonctions, par l'intermédiaire de projections et de transformations (cf section 1.5).

1.1 Les données fonctionnelles

1.1.1 Motivations

L'Analyse de Données Fonctionnelles (ADF, [100]) s'intéresse aux données pour lesquelles les individus étudiés sont décrits par des fonctions plutôt que par des vecteurs de $\mathbb{R}^n$. En d'autres termes, certaines variables observées sont à valeurs fonctionnelles plutôt que réelles. Ce principe est naturel dans de nombreux cas, comme l'illustrent les exemples suivants.

Le cas le plus fréquent est celui dans lequel les individus étudiés présentent une forme de variabilité temporelle. Un exemple simple de ce type est étudié dans [100] : chaque individu est une station météo décrite par la fonction qui, à une date, associe la température moyenne observée par la station (dans cette étude, la résolution est le mois). Dans le cas d'une unique série temporelle, une approche fonctionnelle reste possible. Considérons par exemple l'étude de l'évolution de diverses mesures géophysiques en un emplacement donné pendant

plusieurs années. Une année d'observations est la fonction qui au jour de l'année associe les grandeurs enregistrées. La périodicité approximative des phénomènes météorologiques amène alors à étudier la série temporelle des fonctions, comme dans la modélisation auto-régressive fonctionnelle du phénomène *El niño* réalisée dans [10]. On utilise en fait une approche avec deux échelles : les observations à haute résolution sont découpées en blocs et chacun d'eux est représenté par une fonction. On travaille ensuite sur la série temporelle des fonctions à basse résolution.

De nombreux autres domaines sont caractérisés par une forme de variabilité temporelle. On peut citer par exemple l'étude de la croissance d'enfants réalisée dans [100] (chaque enfant est décrit par une fonction qui, à son âge, associe sa taille). Une autre application est proposée dans [1] : il s'agit d'étudier le processus d'acidification de fromages, chaque échantillon étant décrit par l'évolution temporelle de son pH. Dans le domaine médical, on peut citer [103] (entre autres), qui étudie l'évolution du nombre de lymphocytes T4 dans le sang de malades atteints du SIDA.

Dans certaines situations, l'information temporelle est plus facile à analyser dans une représentation fréquentielle. Chaque individu est alors décrit par la transformée de Fourier de la fonction qui représentait son évolution temporelle. Cette approche est mise en œuvre dans [60], par exemple, pour une application de reconnaissance de phonèmes.

Les représentations fonctionnelles sont aussi naturelles dans d'autres cas. On peut citer, par exemple, la spectroscopie dans laquelle des échantillons sont caractérisés par leur spectre, une fonction qui, à une longueur d'onde, associe une mesure d'intérêt, comme l'absorbance (cf [61, 45]). Un autre exemple original est proposé dans [101] : à partir de radiographies, on détermine la forme de certains os qui est décrite par une courbe paramétrique. On étudie les fonctions ainsi construites pour faire apparaître les caractéristiques de forme utilisées par les experts pour déterminer la présence d'arthrose sur les os.

Le but de l'ADF est de profiter du caractère fonctionnel des données afin d'améliorer les performances des méthodes d'analyse, voire de rendre l'analyse possible dans certains cas (discrétisation irrégulière des fonctions observées par exemple). Hastie, Buja et Tibshirani montrent par exemple dans [60] comment mieux classer des observations fonctionnelles par analyse discriminante en remplaçant la pénalité *ridge* classique (basée sur les carrés des coefficients de l'hyperplan discriminant [66, 65]) par une pénalité qui tient compte de la nature fonctionnelle des données : on cherche ainsi une fonction discriminante "régulière", c'est-à-dire dont la norme de la dérivée seconde est minimale.

L'ADF est confrontée à des problèmes théoriques et pratiques. Au niveau théorique, il s'agit tout d'abord d'étudier le cas parfait dans lequel on suppose qu'on observe des fonctions au sens mathématique du terme. Dans ce contexte, on se pose d'abord la question de définition d'outils pertinents pour le traitement de telles données. La dimension infinie des observations est en effet un obstacle car de nombreux résultats théoriques en apprentissage sont basés sur les propriétés très particulières des espaces vectoriels de dimension finie (par exemple le fait que la dimension de Vapnik-Chervonenkis d'un hyperplan est finie en dimension finie, cf [39]). Une fois les outils construits, on étudie leur propriétés théoriques, en particulier la possibilité de les utiliser pour modéliser des données à partir d'un échantillon fini d'exemples d'apprentissage.

Une première étape vers l'aspect pratique consiste à analyser les effets de la discrétisation : en pratique, bien que les phénomènes physiques étudiés soient réellement fonctionnels (évolution temporelle, spectres, etc.), nous n'avons accès qu'à une version discrétisée des fonctions correspondantes, sous la forme de listes de couples entrée/sortie. Du point de vue théorique,

on doit alors étudier l'effet de la discrétisation sur les propriétés établies pour le cas parfait (capacité d'approximation, consistance des estimateurs, etc.).

Enfin, l'aspect pratique est ce qui motive au final l'ADF. Il s'agit alors de construire une méthodologie générale permettant de prendre en compte l'aspect fonctionnel des données. Celle-ci inclut (cf [101]), par exemple, des techniques de représentation des fonctions qui permettent de s'affranchir de problèmes de discrétisation (quand les fonctions ne sont pas évaluées aux mêmes points), des techniques de calage temporel, diverses solutions pour intégrer des pénalisations fonctionnelles dans la construction des modèles, etc.

1.1.2 Outils d'analyse

L'ADF s'est d'abord développée autour des outils linéaires d'analyse de données, en particulier l'analyse en composantes principales avec les travaux des pionniers Deville [38], et Dauxois et Pousse [35]. On trouvera dans [100] une présentation synthétique et complète des méthodes linéaires pour données fonctionnelles. Une vision générale plus récente de l'ADF est proposée dans [9].

Les méthodes non linéaires sont de développement plus récent. On peut citer par exemple les méthodes à noyau pour données fonctionnelles [44, 42, 45, 46], les approches de type centres mobiles [1], la régression inverse par tranche [34, 48, 49], la *projection pursuit* généralisée [73], les approches de type k plus proches voisins [11] ou encore des méthodes génératives [74].

1.1.3 Contribution personnelle

Ma contribution à l'analyse de données fonctionnelles a consisté à montrer qu'on pouvait adapter les modèles neuronaux classiques et des machines à vecteurs de support pour leur permettre de traiter des données fonctionnelles. Je me suis intéressé essentiellement au cas de la régression sur données fonctionnelles. Il s'agit, à partir d'un ensemble d'exemples d'apprentissage, les couples $(g^i, t^i)_{1 \leq i \leq N}$ où les g^i sont des fonctions et les t^i des vecteurs de $\mathbb{R}^o$, de construire un estimateur de $\mathbb{E}(T|G)$ en considérant les (g^i, t^i) comme des réalisations i.i.d. du couple (G, T). En pratique, ceci permet de prédire la valeur de t pour un g donné, par exemple d'estimer le taux de matières grasses dans un échantillon de viande à partir de son spectre infrarouge, comme dans [A–9]. Si T prend ses valeurs dans $\{-1, 1\}$, ou plus généralement dans un ensemble discret, on retrouve le cadre de la discrimination en plusieurs classes comme dans [A–5] par exemple. Enfin, si T est inconnu, on cherche à classer les fonctions en des groupes homogènes, comme dans [CI–13], par exemple.

1.2 Perceptrons multi-couches et données fonctionnelles

1.2.1 Modèle

Le modèle classique des perceptrons multi-couches (PMC) est destiné aux données réelles (cf e.g. [14]). Il est construit à partir d'un neurone numérique simple. Quand on lui donne p entrées numériques (un vecteur $x \in \mathbb{R}^p$), le neurone produit la sortie suivante :

$$T\left(b + \sum_{i=1}^{p} \beta_i x_i\right), \tag{1.1}$$

où T est la fonction d'activation du neurone (de $\mathbb{R}$ dans $\mathbb{R}$) et où b et les $\beta_1, \ldots, \beta_p$ sont les paramètres numériques du neurone (les poids).

Pour passer à des entrées fonctionnelles, on exploite simplement le fait que $\sum_{i=1}^{p} \beta_i x_i = \langle \beta, x \rangle$ (avec $\beta = (\beta_1, \ldots, \beta_p)$) : la sortie du neurone est obtenue par une transformation du produit scalaire de l'entrée vectorielle x avec un vecteur de paramètres. Il suffit alors de remplacer cette opération par l'équivalent fonctionnel, c'est-à-dire soit un produit scalaire (pour l'espace L^2), soit une forme linéaire (pour un espace L^p quelconque), comme décrit dans [A–10]. Un neurone avec une entrée fonctionnelle dans L^p associe à la fonction g la sortie suivante :

$$T\left(b + \int fg \, \mathrm{d}\lambda\right), \tag{1.2}$$

où b est un paramètre numérique (comme dans l'équation 1.1) et où f est une fonction paramètre (une **fonction de poids**) choisie telle que $fg \in L^1$ pour tout $g \in L^p$ (on peut prendre $f \in L^q$, où q est l'exposant conjugué de p). On étend facilement l'exemple de l'équation 1.2 au cas de plusieurs entrées fonctionnelles. Ce type de neurone est en fait un cas particulier des neurones généralisés proposés dans des travaux théoriques antérieurs [106, 107, 108, 116] (cf définition 2).

Comme la sortie d'un neurone fonctionnel est numérique, on ne les utilise que dans la première couche du PMC, les couches suivantes étant constituées de neurones classiques.

1.2.2 Approximation universelle

Une des propriétés intéressantes des PMC est leur capacité d'approximation universelle : toute fonction régulière peut être approchée arbitrairement bien par un PMC bien choisi (cf [116, 97] pour des présentations synthétiques des résultats d'approximation universelle pour les PMC). Une première question théorique naturelle est donc de savoir si cette propriété est conservée pour des entrées fonctionnelles.

On s'intéresse plus précisément à la propriété d'approximation universelle suivante :

Définition 1 *Soit X un espace topologique et $\mathcal{B}$ un ensemble de fonctions continues de X dans $\mathbb{R}$. $\mathcal{B}$ possède la propriété d'approximation universelle pour X si pour tout compact K de X, $\mathcal{B}$ est dense dans $C(K, \mathbb{R})$ (l'ensemble des fonctions continues de K dans $\mathbb{R}$) pour la norme uniforme.*

Il est connu qu'une seule couche cachée est suffisante pour obtenir l'approximation universelle dans le cas où $X = \mathbb{R}^p$. On s'intéresse donc aux fonctions exactement représentées par un PMC à une couche cachée constituée des neurones généralisés de [116], c'est-à-dire à l'ensemble défini comme suit :

Définition 2 *Soit X un ensemble quelconque, A un ensemble de fonction de X dans $\mathbb{R}$ et T une fonction de $\mathbb{R}$ dans $\mathbb{R}$. Alors $S_T^X(A)$ désigne l'ensemble des fonctions de la forme*

$$h(x) = \sum_{i=1}^{p} \alpha_i T(l_i(x) + b_i), \tag{1.3}$$

où p est un entier positif, les α_i et les b_i sont des réels et les l_i sont des éléments de A.

Pour obtenir un PMC fonctionnel à une couche cachée, il suffit de prendre pour A des fonctions de la forme $g \mapsto \int fg \, \mathrm{d}\lambda$.

Divers auteurs se sont intéressés aux possibilités d'approximation universelle pour les $S_T^X(A)$ en fonction des propriétés de T, X et A. Les premiers travaux remontent à [24] et à [106] (pour $S_T^{L^p}(L^q)$), dont les résultats sont améliorés et étendus dans [23, 26, 108, 116]. Les résultats les plus généraux sont fournis par [116] (théorème 5.1 et corollaires 5.1.2 et 5.1.3) : Stinchcombe propose des conditions suffisantes sur A et sur T pour que $S_T^X(A)$ possède la propriété d'approximation universelle pour X (avec de très faibles hypothèses sur X).

Dans [A–10], j'ai montré comment appliquer les résultats de [116] aux cas particulier des données fonctionnelles. La difficulté vient des propriétés que doivent vérifier les éléments de A, c'est-à-dire les fonctions de la forme $g \mapsto \int fg \, \mathrm{d}\lambda$. J'ai montré qu'on pouvait se contenter d'approcher (par exemple par des PMC classiques) les fonctions f utilisées pour définir les intégrales des neurones fonctionnels. En pratique, cela signifie qu'une fonction continue de L^2 dans $\mathbb{R}$ peut être approchée sur un compact de L^2 à une précision donnée par un PMC fonctionnel dont les fonctions de poids sont elles-mêmes obtenue par des PMC numériques (par d'autres modèles classiques, neuronaux ou non). Techniquement, on a les deux corollaires suivants :

Corollaire 1 (Rossi & Conan-Guez 2005 [A–10]) *Soit μ une mesure borélienne positive finie sur $\mathbb{R}^n$. Soit $1 < p \leq \infty$ et son exposant conjugué q, et soit V un sous-ensemble dense de $L^q(\mu)$. On note A_V les formes linéaires sur $L^p(\mu)$ de la forme $l(g) = \int fg \, \mathrm{d}\mu$ pour $g \in V$. Soit enfin T une fonction de $\mathbb{R}$ dans $\mathbb{R}$ telle que l'ensemble des PMC à une couche cachée utilisant T comme fonction d'activation possède la propriété d'approximation universelle pour $\mathbb{R}$. Alors $S_T^{L^p(\mu)}(A_V)$ possède la propriété d'approximation universelle pour $L^p(\mu)$.*

Corollaire 2 (Rossi & Conan-Guez 2005 [A–10]) *Soit μ une mesure borélienne positive finie et à support compact sur $\mathbb{R}^n$. Soit V un sous-ensemble de $L^\infty(\mu)$ dense dans $C(\mathbb{R}^n, \mathbb{R})$. On note A_V les formes linéaires sur $L^1(\mu)$ de la forme $l(g) = \int fg \, \mathrm{d}\mu$ pour $g \in V$. Soit enfin T une fonction de $\mathbb{R}$ dans $\mathbb{R}$ telle que l'ensemble des PMC à une couche cachée utilisant T comme fonction d'activation possède la propriété d'approximation universelle pour $\mathbb{R}$. Alors $S_T^{L^1(\mu)}(A_V)$ possède la propriété d'approximation universelle pour $L^1(\mu)$.*

Pour obtenir l'approximation universelle fonctionnelle, il suffit donc d'exhiber des ensembles denses dans $L^q(\mu)$ et dans $C(\mathbb{R}^n, \mathbb{R})$. On peut par exemple utiliser des PMC numérique, comme démontré dans [69, 70, 85] (cf la discussion détaillée dans [A–10], section 3.3).

1.2.3 Estimation des paramètres

En pratique, on doit construire un PMC à partir d'exemples de couples entrées/sorties. Dans le cadre fonctionnel, l'approximation d'une fonction F de L^p dans $\mathbb{R}$ se fera donc à partir d'exemples de la forme (g^i, t^i) (cf section 1.1.3). Cependant, pour réaliser une analyse théorique réaliste, on doit tenir compte du fait que les fonctions ne sont jamais connues de façon parfaite. Chaque fonction g^i est ainsi associée à une suite de discrétisation, les $(x_j^i)_j$: on observe les couples $(x_j^i, y_j^i)_j$, où $y_j^i = g^i(x_j^i) + \varepsilon_j^i$ (ce qui permet de prendre en compte un bruit d'observation sur les fonctions discrétisées). Ce cadre est très général et couvre la plupart des cas réels. Bien qu'une discrétisation identique pour chaque fonction soit le cas le plus fréquent, certaines applications correspondent à des situations plus complexes pour lesquelles chaque fonction possède sa propre discrétisation. C'est le cas, par exemple, pour certaines applications médicales (cf, e.g., [72, 71]) ou en reconnaissance d'écriture cursive (cf, e.g., [6]).

Étant donnés N exemples de la forme proposée au dessus et un PMC fonctionnel d'architecture fixée, l'apprentissage consiste à trouver les paramètres numériques de ce PMC qui minimisent un critère d'erreur (par exemple l'erreur quadratique moyenne commise par le PMC sur l'ensemble d'apprentissage). Une première difficulté vient de la définition même du neurone fonctionnel (équation 1.2) qui s'appuie sur un calcul d'intégrale impossible à réaliser exactement en présence d'une fonction discrétisée. En pratique, on remplace donc l'équation 1.2 par

$$T\left(b+\frac{1}{m^i}\sum_{j=1}^{m^i} f(x_j^i)y_j^i\right), \tag{1.4}$$

pour une fonction g^i discrétisée en m^i points. Pour que la moyenne utilisée dans cette expression soit une bonne approximation de l'intégrale de l'expression d'origine, il faudrait utiliser des pondérations correspondant à un schéma de quadrature. Plutôt que de faire l'hypothèse d'une discrétisation déterministe qui est assez restrictive, nous préférons supposer que les x_i^j sont des réalisations i.i.d. d'une variable d'observation X (excepté les travaux de [1], ce modèle très général n'a été étudié que dans mes travaux, à ma connaissance). On note μ la mesure engendrée par X sur $\mathbb{R}^u$, l'espace de départ des données fonctionnelles. Sous certaines hypothèses additionnelles, la loi des grands nombres implique alors (pour f et g continues)

$$\lim_{m^i\to\infty}\frac{1}{m^i}\sum_{j=1}^{m^i} f(x_j^i)y_j^i = \mathbb{E}\left(f(X)g^i(X)\right) = \int fg^i\,\mathrm{d}\mu, \tag{1.5}$$

et l'approximation proposée est donc bien adaptée au problème, sans nécessiter l'introduction de pondérations.

Une deuxième difficulté vient de l'utilisation de fonctions de poids quelconques (ou restreintes à un ensemble dense dans L^q comme dans le corollaire 1 par exemple). En pratique, il n'est pas possible d'optimiser un critère d'erreur par rapport à des variables fonctionnelles sans introduire une représentation compatible avec une implémentation informatique. Nous choisissons ici une représentation paramétrique, ce qui couvre l'essentiel des cas pratiques. Techniquement, on suppose donnée, pour chaque neurone fonctionnel, une fonction F de $W\times\mathbb{R}^u$ dans $\mathbb{R}$, où W est un compact de $\mathbb{R}^v$, l'espace des paramètres de la fonction. Pour une valeur fixée de $w\in W$, $F(w,.)$ est une fonction de L^q, la fonction de poids du neurone. La sortie d'un tel neurone est donc

$$T\left(b+\frac{1}{m^i}\sum_{j=1}^{m^i} F(w,x_j^i)y_j^i\right). \tag{1.6}$$

Le réglage du neurone consiste alors à bien choisir les paramètres numériques b et w.

Grâce à cette version empirique et paramétrique du neurone fonctionnel, on peut construire un PMC à une entrée fonctionnelle, par simple combinaison avec des neurones numériques classiques. Par exemple, un PMC à une couche cachée de k neurones et muni d'un neurone linéaire dans sa couche de sortie, associe à la fonction g^i donnée par les m^i couples $(x_j^i,y_j^i)_{1\le j\le m^i}$ la valeur

$$\widetilde{H}(w,g^i)=\sum_{l=1}^{k}\alpha_l T\left(b_l+\frac{1}{m^i}\sum_{j=1}^{m^i} F_l(w_l,x_j^i)y_j^i\right), \tag{1.7}$$

où w désigne l'ensemble de tous les paramètres numériques du PMC, c'est-à-dire les α_l, les b_l et les w_l. De façon plus générale, on note $\widetilde{H}(w, g^i)$ la sortie associée à la fonction g^i par un PMC construit à partir des neurones fonctionnels empiriques et paramétriques qui viennent d'être proposés. On note $H(w, g^i)$ la sortie associée à la fonction g^i par le PMC de même architecture mais dans lequel les intégrales sont calculées de façon exacte (i.e., comme si on connaissait exactement g^i). Par exemple, l'équation 1.7 correspond à une approximation de la version exacte suivante :

$$H(w, g^i) = \sum_{l=1}^{k} \alpha_l T \left(b_l + \int F_l(w_l, x) g^i(x) \, \mathrm{d}\mu(x) \right). \tag{1.8}$$

Considérons maintenant un critère d'erreur c qui mesure la qualité de la prédiction de t^i par h, par exemple l'erreur quadratique

$$c(h, t^i) = \left\| h - t^i \right\|^2. \tag{1.9}$$

L'erreur théorique commise par le PMC H peut être définie par

$$\lambda(w) = \mathbb{E}\left(c\left(H(w, G), T \right) \right). \tag{1.10}$$

L'apprentissage consiste alors à choisir une valeur de l'ensemble des paramètres w qui minimise λ. En pratique, nous disposons seulement de N observations discrétisées et nous ne pouvons que minimiser l'erreur empirique suivante

$$\widetilde{\lambda}_N^m(w) = \frac{1}{N} \sum_{i=1}^{N} c\left(\widetilde{H}(w, g^i), t^i \right), \tag{1.11}$$

où $m = \inf_{1 \leq i \leq N} m^i$. Cette expression fait apparaître une double approximation : l'espérance de l'équation 1.10 est remplacée par une moyenne empirique (la somme sur i) et la sortie exacte du réseau est remplacée par une sortie approchée (approximation des intégrales discutée précédemment).

Dans [A–10], je montre que les paramètres obtenus en minimisant le critère empirique convergent vers les paramètres optimaux au sens du critère théorique.

Théorème 1 (Rossi & Conan-Guez 2005 [A–10]) *On note $\widetilde{w}_N^m$ une valeur de w qui minimise $\widetilde{\lambda}_N^m(w)$ sur le compact W dans lequel on cherche les paramètres du PMC. On note W^* l'ensemble des w qui minimisent $\lambda(w)$ sur W. Sous les hypothèses de [A–10] (section 4), on a*

$$\lim_{N \to \infty} \lim_{m \to \infty} d(\widetilde{w}_N^m, W^*) = 0, \ [p.s.]. \tag{1.12}$$

La convergence obtenue est séquentielle et fait intervenir les deux sources d'erreur : les paramètres optimaux théoriques sont la limite quand le nombre d'exemples tend vers l'infini ($N \to \infty$) de la limite des paramètres empiriques quand le nombre de points de discrétisation tend vers l'infini ($m = \inf_{1 \leq i \leq N} m^i \to \infty$). Cela montre qu'en pratique, à condition de disposer d'une bonne discrétisation associée à un nombre suffisant d'exemples, les paramètres obtenus seront proches des paramètres optimaux. La situation est donc sensiblement la même que pour les PMC classiques (cf [126]), à condition d'avoir une discrétisation suffisamment fine pour que l'approximation induite par l'utilisation de $\widetilde{H}$ à la place de H ne change pas significativement la valeur du critère d'erreur.

Les hypothèses du résultat, assez techniques, se trouvent dans [A–10] (section 4) et dans [A–11]. Elles sont assez classiques : indépendance des variables aléatoires, régularité des fonctions utilisées (les F^l, c et T), existence d'un moment au sens de c pour la variable à expliquer T, etc. Le seul aspect véritablement restrictif des hypothèses correspond aux conditions imposées aux fonctions observées (les g^i) : elles doivent être continues et définies sur un compact Z de $\mathbb{R}^u$. Ces conditions sont cependant moins restrictives que celles imposées dans le seul autre travail qui considère une discrétisation aléatoire des observations [1].

Le théorème s'appuie sur les travaux de White, en particulier [126] dont il suit la méthode de preuve. Je montre en particulier une loi des grands nombres uniforme en m'appuyant sur [4]. Cette loi permet à la fois de traiter le cas de données à valeurs fonctionnelles, mais aussi de prendre en compte la discrétisation.

1.2.4 Mise en œuvre pratique

Bien que proche d'un PMC classique, le PMC fonctionnel construit à partir des neurones empiriques (équation 1.6) présente des spécificités qui demandent une implémentation informatique dédiée. En effet, l'optimisation de l'erreur empirique $\widetilde{\lambda}(w)$ est, en général, réalisée à partir du gradient de celle-ci. Dans l'algorithme de rétro-propagation (cf e.g. [14]) le gradient est obtenu à partir de celui de la combinaison linéaire effectuée par chaque neurone. Plus précisément, on utilise le gradient suivant

$$\nabla_\beta \left(b + \sum_{i=1}^{p} \beta_i x_i \right) = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_p \end{pmatrix}. \tag{1.13}$$

Dans le cas du neurone empirique de l'équation 1.6, on a donc

$$\nabla_w \left(b + \frac{1}{m^i} \sum_{j=1}^{m^i} F(w, x_j^i) y_j^i \right) = \begin{pmatrix} 1 \\ \frac{1}{m^i} \sum_{j=1}^{m^i} \left(\frac{\partial F}{\partial w}(w, x_j^i) y_j^i \right) \end{pmatrix}. \tag{1.14}$$

On doit ainsi calculer la dérivée partielle de F par rapport à ses paramètres : il est par conséquent impossible d'utiliser directement un logiciel classique proposant des PMC. Une adaptation est nécessaire pour le calcul du gradient. En outre, si le modèle F est complexe, les temps de calcul peuvent être élevés. On remarque de plus que le calcul de la dérivée par rapport à un paramètre numérique, qui est instantané dans le cas des données classiques, est ici en $O(m)$ (où m désigne le nombre de points de discrétisation de la fonction considérée).

Bien qu'un code spécifique ait été développé par Brieuc Conan-Guez pendant sa thèse (à partir de la bibliothèque GSL [B–1], dont j'ai programmé les algorithmes d'optimisation), nous avons cherché à palier ce problème ainsi que celui des temps de calcul. Dans le cas d'une fonction F générale, en particulier non linéaire par rapport à ses paramètres, aucune simplification n'est possible. Par contre, le cas linéaire est particulièrement intéressant. On écrit alors F sous la forme suivante

$$F(w, x) = \sum_{k=1}^{l} w_k \psi_k(x). \tag{1.15}$$

Cette formulation est très générale et parfaitement compatible avec les hypothèses des corollaires d'approximation universelle : il suffit par exemple de considérer un réseau de neurones de type RBF (*Radial Basis Function*) pour voir qu'on peut obtenir un sous ensemble dense de $C(\mathbb{R},\mathbb{R})$ (par exemple) avec des fonctions de la forme donnée par l'équation 1.15 (cf [95]).

Le gros avantage de cette simplification est qu'elle permet de factoriser certains calculs. On a en effet

$$\frac{1}{m^i}\sum_{j=1}^{m^i} F(w,x_j^i)y_j^i = \sum_{k=1}^{l} w_k \left(\frac{1}{m^i}\sum_{j=1}^{m^i}\psi_k(x_j^i)y_j^i\right), \tag{1.16}$$

ce qui conduit à associer à la fonction g^i, donnée par les m^i couples $(x_j^i,y_j^i)_{1\leq j\leq m^i}$, le vecteur de $\mathbb{R}^p$ suivant

$$\widetilde{v}(g^i) = \begin{pmatrix} \frac{1}{m^i}\sum_{j=1}^{m^i}\psi_1(x_j^i)y_j^i \\ \vdots \\ \frac{1}{m^i}\sum_{j=1}^{m^i}\psi_l(x_j^i)y_j^i \end{pmatrix}. \tag{1.17}$$

On constate alors que la sortie d'un neurone fonctionnel construit à partir de F est obtenu par

$$T\left(b+\sum_{k=1}^{l} w_k\widetilde{v}(g^i)_k\right). \tag{1.18}$$

On obtient ainsi un neurone numérique classique. En pratique, on peut donc implémenter un PMC fonctionnel sous la forme d'un simple pré-traitement : chaque fonction est représentée sous une forme vectorielle unifiée. Les vecteurs obtenus sont alors traités par un PMC classique. Notons que cette approche n'est possible que si tous les neurones fonctionnels utilisent la même fonction paramétrique F.

Cette nouvelle vision du PMC fonctionnel est très intéressante pour diverses raisons. Tout d'abord, elle simplifie la mise en œuvre puisqu'elle ramène le problème à un simple pré-traitement. De plus, ce pré-traitement est pertinent du point de vue fonctionnel. En effet, le vecteur $\widetilde{v}(g^i)$ est en fait une approximation du vecteur suivant

$$v(g^i) = \begin{pmatrix} \int \psi_1 g^i \,\mathrm{d}\mu \\ \vdots \\ \int \psi_l g^i \,\mathrm{d}\mu \end{pmatrix}. \tag{1.19}$$

Le pré-traitement consiste donc simplement à calculer les intégrales des produits de la fonction étudiée avec des fonctions bien choisies. Si on considère pour les ψ_k des fonctions de type RBF, on réalise par exemple une sorte de moyenne locale de la fonction g^i. Si les ψ_k sont des fonctions orthogonales et de normes unitaires de L^2, on calcule en fait les coordonnées de g^i sur le sous-espace vectoriel engendré par les ψ_k (cf la section 1.3 pour un développement sur cet aspect). Plus généralement, cette approche permet à la fois de s'affranchir des différences dans la discrétisation des fonctions, mais aussi de découpler la taille de la représentation d'origine de celles-ci de la quantité d'information transmise au réseau. En effet, la dimension du vecteur $\widetilde{v}(g^i)$ est fixée librement lors du pré-traitement et ne dépend pas directement du nombre de points de discrétisation m^i. On peut donc adapter la finesse de la représentation (le nombre de fonctions ψ_k qui forment F) en tenant compte de la régularité des fonctions g^i, même si celles-ci sont discrétisées de façon très fine. On évite ainsi de transmettre au PMC des informations redondantes et de très grande dimension, ce qui réduit le risque de problèmes à

l'apprentissage. En outre, le pré-traitement nécessite un temps négligeable comparativement à l'apprentissage : contrairement au cas de F quelconque, le coût algorithmique de la forme particulière utilisée ici dépend de la complexité de F (nombre de fonctions ψ_k), sous le contrôle de l'utilisateur, plutôt que la finesse de la discrétisation.

La principale difficulté reste alors le choix des ψ_k, c'est-à-dire à la fois la nature des fonctions utilisées et leur nombre. En pratique, le nombre de fonctions devient un méta-paramètre de plus (qui s'ajoute au nombre de neurones par exemple) à déterminer par une méthode de sélection de modèles appropriée (validation croisée, par exemple). La nature des fonctions est un élément plus délicat, bien que certaines familles génériques, comme les B-splines [36], donnent généralement de bons résultats.

1.2.5 Résultats expérimentaux

Le PMC fonctionnel présenté ci dessus a été appliqué à divers jeux de données. Sur des données simulées, nous avons montré dans [CI–17] que l'utilisation d'une fonction F non linéaire (par rapport à w) donnait des résultats très intéressants. Cependant, les temps de calcul prohibitifs observés avec des données discrétisées finement nous ont conduits à abandonner cette voie pour nous concentrer sur le cas des fonctions F linéaires par rapport à leurs paramètres.

Nous avons étudié les performances des PMC fonctionnels sur une version spécifique des vagues de Breiman [16] : au lieu d'utiliser une discrétisation en 21 points, nous utilisons 101 points, comme [45, 46]. Nous avons montré dans [A–10, A–8] que les PMC obtenaient d'excellent résultats, les meilleurs des méthodes comparées, tout en étant très parcimonieux (3 neurones fonctionnels avec une dizaine de fonctions ψ_k, des B-splines, pour chaque neurone).

Nous avons appliqué le modèle à un problème de spectrométrie utilisé dans [45, 46] (et dérivé de [120, 122]). Il s'agit de données réelles (des spectres discrétisés en 100 points) pour lesquelles on doit réaliser une discrimination en deux classes. Là encore, les performances obtenues sont très satisfaisantes, en particulier grâce à l'utilisation de transformations fonctionnelles : on montre en effet sur ce problème qu'il est particulièrement fructueux de travailler sur une estimation des dérivées secondes des spectres étudiés. Les PMC fonctionnels obtiennent de nouveau les meilleurs performances, en utilisant peu de ressources (3 ou 4 neurones fonctionnels avec une vingtaine de fonctions ψ_k, des B-splines).

Les expériences réalisées dans [A–10, A–8] montrent la compétitivité du modèle de PMC fonctionnel. L'avantage principal semble résider dans le contrôle de complexité opéré par l'intermédiaire des fonctions F. Grâce à lui, on adapte la puissance de calcul du modèle à la régularité des fonctions, ce qui évite certains problèmes liés à la grande dimensionnalité des données. Il faut noter en effet que les jeux de données considérés sont en fait des données classiques de dimension relativement élevée ($\mathbb{R}^{100}$). Nous avons donc comparé les PMC fonctionnels à des PMC classiques et montré la supériorité des premiers sur les seconds. Or, techniquement, nous réalisons simplement un pré-traitement fonctionnel (qui peut d'ailleurs intégrer des transformations fonctionnelles, comme une dérivation, par exemple). La pertinence de celui-ci est justifiée par un cadre théorique complet, mais d'un point de vue pragmatique, il s'agit finalement d'exploiter une connaissance experte sur les données (il s'agit de fonctions) pour améliorer les performances. Si on compare mon approche aux travaux antérieurs sur les PMC pour données fonctionnelles, en particulier [26], on constate que c'est l'idée de ne pas représenter les fonctions par un vecteur de discrétisation qui est nouvelle. Or, il s'avère en pratique que c'est justement cette idée qui conduit à de meilleurs

résultats.

1.3 Représentation des fonctions

1.3.1 Principe

Dans les travaux présentés dans la section 1.2, j'ai développé une approche originale basée sur une l'approximation des intégrales utilisées dans un modèle neuronal défini pour des entrées fonctionnelles quelconques. Je me suis ensuite intéressé à une approche plus classique en ADF qui consiste à remplacer les fonctions par des représentants plus faciles à manipuler. Il s'agit en fait de projeter les fonctions étudiées sur un sous-espace de dimension finie de l'espace fonctionnel de départ. En munissant ce sous-espace d'une base bien choisie, on se contente de travailler sur les coordonnées des projetées, ce qui ramène donc le problème fonctionnel à un problème classique.

Le recours à une projection est très fréquent en ADF (comme l'illustre [100]), à la fois pour des raisons pratiques d'implémentation, mais aussi pour des raisons théoriques. On montre en effet que certains problèmes simples en dimension finie, par exemple l'estimation de la régression linéaire de Y en X, n'admettent pas de solution directe en dimension infinie (par exemple, l'estimateur naïf de la régression linéaire ne converge pas vers la valeur théorique). Une solution simple pour contourner ces problèmes consiste à se ramener à la dimension finie par projection, comme dans [20] pour le modèle linéaire. La différence fondamentale entre les projections utilisées pour des raisons pratiques et celles motivées par des considérations théoriques est que ces dernières sont souvent effectuées sur des espaces qui dépendent des données, alors que pour les premières, des considérations expertes interviennent.

Techniquement, la représentation des fonctions par projection peut être utilisée quand celles-ci viennent d'un espace de Hilbert, généralement L^2. On se donne alors un sous-espace de dimension finie V, et on remplace une observation g^i par $\Pi_V(g^i)$, la projection orthogonale de g^i sur V. Dans le cas de fonctions discrétisées, représentées par des listes de couples $(x^i_j, y^i_j)_j$, on doit calculer une projection approximative. Pour ce faire, on se donne une base (pas nécessairement orthogonale) de V, les $(\psi_k)_{1\leq k\leq p}$, et on cherche la projection optimale de g^i au sens des moindres carrés, c'est-à-dire le vecteur $\widetilde{v}(g^i)$ de $\mathbb{R}^p$ qui minimise

$$\frac{1}{m^i}\sum_{j=1}^{m^i}\left(y^i_j - \sum_{k=1}^{p}\widetilde{v}(g^i)_k\psi_k(x^i_j)\right)^2. \tag{1.20}$$

Cette erreur est en fait une approximation de l'erreur théorique suivante

$$\mathbb{E}\left(\left(g(X) - \sum_{k=1}^{p}\widetilde{v}(g^i)_k\psi_k(X)\right)^2\right) = \int\left(g - \sum_{k=1}^{p}\widetilde{v}(g^i)_k\psi_k\right)^2 \mathrm{d}\mu, \tag{1.21}$$

en utilisant le modèle de discrétisation aléatoire présentée à la section 1.2.3. On a donc bien une correspondance entre la solution pratique et le modèle théorique. On montre d'ailleurs que $\widetilde{v}(g^i)$ converge vers $v(g^i)$, les coordonnées de la projection de g^i sur V dans la base des ψ_k (cf [28]).

1.3.2 Perceptrons multi-couches sur fonctions projetées

Quand on travaille sur des données projetées, le neurone fonctionnel associe à la fonction g^i la valeur suivante

$$T\left(b + \int f \Pi_V(g^i)\,\mathrm{d}\lambda\right) = T\left(b + \int \Pi_V(f)\Pi_V(g^i)\,\mathrm{d}\lambda\right). \tag{1.22}$$

La deuxième formulation insiste sur le fait que la fonction paramètre du neurone n'intervient dans le calcul que par l'intermédiaire de sa projection orthogonale sur V. Comme dans la version du PMC fonctionnel étudié dans la section 1.2.4, on se ramène donc, grâce à la projection, au cas d'un PMC numérique classique. En effet, en utilisant une base de V, $(\psi_k)_{1\leq k\leq p}$ orthogonale, le neurone fonctionnel associe à g^i la valeur

$$T\left(b + \sum_{k=1}^{p} v(f)_k v(g^i)_k\right), \tag{1.23}$$

où $v(h)$ est le vecteur de coordonnées de la projection de la fonction h sur V pour la base des ψ_k (cf la section 1.4 pour le cas d'une base non orthogonale). On se retrouve donc, comme à la section 1.2.4, avec un PMC classique travaillant sur une représentation vectorielle des fonctions d'entrées. Si les ψ_k utilisés dans la section 1.2.4 forment une base orthogonale, la différence entre les deux approches est très faible (il s'agit en fait de deux solutions approximatives différentes d'un même problème, cf la section 5.2 de [A–10]). En fait, l'approche par projection est une sorte de cas particulier du modèle développé dans la section 1.2 : l'introduction de la projection rend totalement inutile l'utilisation d'une fonction paramétrique complexe F pour représenter les fonctions de poids, car celles-ci seront toujours prises en compte par l'intermédiaire de leur projection sur V.

Comme le modèle par projection est plus contraint que le PMC fonctionnel général, les théorèmes démontrés pour ce dernier ne s'appliquent plus. Les résultats évoqués dans la section 1.2 sont redémontrés pour le modèle avec projection dans [CI–15] et [28].

1.3.3 Approximation universelle

Concernant l'approximation universelle, il s'agit de montrer qu'en composant un opérateur de projection avec un PMC fonctionnel, on conserve la propriété d'approximation universelle, à condition de bien choisir la projection. J'ai montré dans [A–7] un résultat assez général reproduit ici. On commence par définir une suite de projections de plus en plus précises :

Définition 3 *Soit μ une mesure de Borel positive σ-finie définie sur $\mathbb{R}^n$. On considère une suite doublement indicée de fonctions de $L^2(\mu)$, les $(\psi_{p,k})_{p\in\mathbb{N}^*,1\leq k\leq p}$, telle que pour tout p, $(\psi_{p,k})_{1\leq k\leq p}$ soit un système orthogonal*[1]*. On note V_p le sous-espace de $L^2(\mu)$ engendré par $(\psi_{p,k})_{1\leq k\leq p}$ (et Π_p le projecteur orthogonal sur V_p).*

Soit $\mathcal{G}$ un sous-ensemble de $L^2(\mu)$. On dit que la suite $(\Pi_p)_{p\in\mathbb{N}^*}$ possède la propriété d'approximation universelle simple pour $\mathcal{G}$, si pour toute fonction $g \in \mathcal{G}$, $\lim_{p\to\infty} \|\Pi_p(g) - g\|_2 = 0$.

On a alors le théorème suivant :

[1] Il suffit en fait que le système soit libre, mais le cas orthogonal simplifie la présentation et la rédaction des preuves.

Théorème 2 (Rossi & Conan-Guez 2006 [A–7]) *Soit T une fonction continue non polynomiale de $\mathbb{R}$ dans $\mathbb{R}$ et soit $(\psi_{p,k})_{p\in\mathbb{N}^*,1\leq k\leq p}$, une suite de fonctions de $L^2(\mu)$ possédant la propriété d'approximation universelle simple pour $L^2(\mu)$. On note $S(T,(\psi_{p,k})_{p\in\mathbb{N}^*,1\leq k\leq p})$ l'ensemble des fonctions de $L^2(\mu)$ dans $\mathbb{R}$ de la forme*

$$h(g)=\sum_{i=1}^{l}\alpha_i T\left(b_i+\sum_{k=1}^{p}\beta_{ik}v_p(g)_k\right), \tag{1.24}$$

où l et p sont des entiers positifs arbitraires, les α_i, β_{ik} et b_i des réels quelconques et où $v_p(g)$ désigne les coordonnées de $\Pi_p(g)$ sur la base $(\psi_{p,k})_{1\leq k\leq p}$.

Alors $S(T,(\psi_{p,k})_{p\in\mathbb{N}^*,1\leq k\leq p})$ a la propriété d'approximation universelle pour $L^2(\mu)$.

Comme les corollaires 1 et 2, ce théorème s'appuie sur les résultats de [116]. Il peut être vu comme une extension des résultats de [25, 107] : je considère en effet un schéma de projection complexe (la suite de la définition 3) qui est plus général que les simples bases tronquées. Dans [25, 107], les auteurs se limitent à des projections construites à partir d'une base Hilbertienne de $L^2(\mu)$: on considère les p premières coordonnées des fonctions dans une telle base, et on fait croître p. Dans le théorème 2, on peut changer de base à chaque augmentation de p : en pratique, on rencontre ce cas quand on travaille avec des B-splines, comme dans [A–9].

1.3.4 Estimation des paramètres

Dans [CI–15] et [28], j'étudie le problème de l'estimation des paramètres d'un PMC fonctionnel d'architecture fixée à partir d'un nombre fini de fonctions discrétisées, dans le même cadre théorique que celui de la section 1.2.3. Je me contente de résumer les résultats obtenus car ils sont similaires à ceux indiqués à la section 1.2.3.

Grâce à la projection, on peut définir une erreur empirique dont la minimisation donne des paramètres optimaux empiriques. Techniquement, on remplace l'approximation définie par l'équation 1.7 (dans le cas d'un PMC à une couche cachée) par la suivante

$$\widetilde{H}(w,g^i)=\sum_{l=1}^{q}\alpha_l T\left(b_l+\sum_{k=1}^{p}\beta_{lk}\widetilde{v}_p(g^i)_k\right), \tag{1.25}$$

où $\widetilde{v}_p(g^i)$ est le vecteur de coordonnées de la projection approximative de g^i sur l'espace V_p (cf équation 1.20). $\widetilde{H}(w,g^i)$ est donc l'approximation de

$$H(w,g^i)=\sum_{l=1}^{q}\alpha_l T\left(b_l+\sum_{k=1}^{p}\beta_{lk}v_p(g^i)_k\right). \tag{1.26}$$

En utilisant $\widetilde{H}(w,g^i)$, on définit une erreur empirique $\widetilde{\lambda}_N^m(w)$ comme dans l'équation 1.11. L'erreur théorique correspondante est donnée par

$$\lambda(w)=\mathbb{E}\left(c\left(H(w,\Pi_p(G)),T\right)\right). \tag{1.27}$$

On montre alors le théorème suivant :

Théorème 3 (Conan-Guez & Rossi 2002 [CI–15] et [28]) *On note $\widetilde{w}_N^m$ une valeur de w qui minimise $\widetilde{\lambda}_N^m(w)$ sur le compact W dans lequel on cherche les paramètres du PMC. On*

note W^* l'ensemble des w qui minimisent $\lambda(w)$ sur W. Sous les hypothèses de [CI–15] et [28], on a

$$\lim_{N\to\infty}\lim_{m\to\infty} d(\widetilde{w}_N^m, W^*) = 0, \ [p.s.]. \tag{1.28}$$

On montre ainsi que les paramètres empiriques convergent vers les paramètres théoriques, toujours au sens d'une limite séquentielle. La principale différence avec le modèle sans projection est que l'erreur de calcul sur les intégrales est ici remplacée par une erreur sur les coordonnées des projections des fonctions.

1.3.5 Consistance

Les résultats sur l'estimation des paramètres sont intéressants car ils montrent qu'on ne risque pas d'erreur systématique en choisissant les paramètres d'un PMC fonctionnel (avec ou sans projection) en minimisant l'erreur empirique. Cependant, ils ne sont pas suffisants pour assurer la pertinence du modèle obtenu : ils garantissent en effet l'obtention d'un modèle d'erreur minimale, mais ne renseignent pas sur la valeur de cette erreur. Si l'architecture du PMC est mal choisie (pas assez de neurones, par exemple), on peut obtenir en pratique un très mauvais modèle.

Il est donc intéressant d'étudier le comportement des PMC quand on adapte la puissance du réseau (le nombre de neurones cachés) à la taille des données. Nous cherchons à montrer qu'on peut obtenir ainsi asymptotiquement une bonne approximation de la relation entre la variable à prédire T et la variable explicative fonctionnelle, G. Il s'agit en fait d'estimer $\mathbb{E}(T|G)$. On se place ici dans le cadre d'une connaissance parfaite des fonctions, même si des résultats récents commencent à lever ces limitations (cf [CN–2, A–2] et [7]).

Les premiers résultats de consistance pour les données fonctionnelles concernent le cas des méthodes à noyaux [44]. Ils sont cependant limités par des hypothèses très fortes sur la distribution de la variable fonctionnelle G. Des résultats sans hypothèse sur G sont donnés dans [11] dans le cas de la discrimination en deux classes (T à valeurs dans $\{-1, 1\}$). Bien que l'article lui-même se contente de montrer la consistance universelle des k-plus proches voisins fonctionnels, la méthode proposée est très générale et s'applique à toute combinaison d'une méthode universellement consistante en dimension finie avec une projection sur une base Hilbertienne tronquée, à condition que les méta-paramètres de la méthode (le nombre de voisins par exemple) soient choisis dans un ensemble dénombrable. On peut donc combiner les résultats classiques sur les PMC (cf [39] chapitre 30, par exemple) avec ceux [11] pour conclure à la consistance universelle des PMC fonctionnels avec projection, dans le cas de la discrimination à deux classes (cf la section 1.5 pour le cas des machines à vecteurs de support).

Dans [A–7] je propose un résultat différent : en m'appuyant sur les résultats de [87], j'établis la consistance pour l'estimation de $\mathbb{E}(T|G)$ sans restriction sur T et sur G, mais avec une limite séquentielle sur deux paramètres (la dimension de l'espace de projection et le nombre d'observations). Contrairement à l'approche de [11] nous ne sommes pas limités au cas de la discrimination, mais nous n'intégrons pas le choix automatique de la dimension de l'espace de projection.

Nous considérons n couples de variables aléatoires

$$D_n = ((G^1, T^1), \ldots, (G^n, T^n)), \tag{1.29}$$

indépendants et identiquement distribués comme le couple (G, T). Un modèle h_n (construit

à partir de D_n) est universellement consistant si

$$\mathbb{E}\left((h_n(G) - \mathbb{E}\left(T|G\right))^2|D_n\right)^{\frac{1}{2}} \xrightarrow[n\to\infty]{} 0 \ p.s. \tag{1.30}$$

Pour construire un PMC fonctionnel universellement consistant, nous partons d'une base Hilbertienne de $L^2(\mu)$, $(\psi_p)_{p\in\mathbb{N}^*}$ et nous notons $v_p(g)$ le vecteur des p premières coordonnées d'une fonction g de $L^2(\mu)$ sur cette base. Étant données une suite d'entiers $(L_n)_{n\in\mathbb{N}^*}$ et une suite de réels positifs $(a_n)_{n\in\mathbb{N}^*}$, nous définissons une suite d'ensembles de PMC fonctionnels à une couche cachée, les $\mathcal{H}_{np}$, donnés par

$$\mathcal{H}_{np} = \left\{ h \in C(L^2(\mu),\mathbb{R}) \middle| h(g) = \sum_{l=1}^{L_n} \alpha_l T\left(b_l + \sum_{k=1}^{p} \beta_{lk} v_p(g)_k\right), \sum_{k=1}^{L_n} |\alpha_l| \leq a_n \right\}. \tag{1.31}$$

Dans la classe $\mathcal{H}_{np}$, on choisit à partir des données le PMC h_{np} qui minimise l'erreur quadratique empirique, c'est-à-dire tel que

$$\frac{1}{n}\sum_{i=1}^{n}(h_{np}(G^i) - T^i)^2 \leq \frac{1}{n}\sum_{i=1}^{n}(h(G^i) - T^i)^2, \ \forall h \in \mathcal{H}_{np}, \tag{1.32}$$

Moyennant quelques hypothèses techniques sur T (croissante, limite 0 en $-\infty$ et 1 en $+\infty$) et sur les suites $(L_n)_{n\in\mathbb{N}^*}$ et $(a_n)_{n\in\mathbb{N}^*}$ (cf [A–7], section 4, en particulier le théorème 2), je montre le théorème suivant :

Théorème 4 (Rossi & Conan-Guez 2006 [A–7])

$$\lim_{p\to\infty}\lim_{n\to\infty} \mathbb{E}\left((h_{np}(G) - \mathbb{E}\left(T|G\right))^2|D_n\right)^{\frac{1}{2}} = 0 \ p.s. \tag{1.33}$$

Nous obtenons donc la consistance universelle des PMC fonctionnels : asymptotiquement, à condition de contrôler la puissance du PMC (c'est le rôle des hypothèses sur les suites $(L_n)_{n\in\mathbb{N}^*}$ et $(a_n)_{n\in\mathbb{N}^*}$), le meilleur PMC sur l'ensemble d'apprentissage approche de façon de plus en plus précise l'espérance conditionnelle de T sachant G.

Comme mentionné plus haut, la limite de ce résultat est qu'il n'offre pas de choix automatique de la dimension de l'espace de projection. Il montre cependant qu'on ne commet pas d'erreur systématique en utilisant un PMC fonctionnel pour estimer la régression de T en G.

1.3.6 Résultats expérimentaux

Dans [A–10, A–8], j'ai comparé l'approche proposée dans la section 1.2.4 et celle par projection proposée ci-dessus, sur les données simulées et réelles évoquées dans la section 1.2.5. La projection optimale est déterminée par validation croisée, comme les autres méta-paramètres. En pratique, les deux modèles donnent des résultats assez proches, sans qu'il soit possible de vraiment les départager : cela est parfaitement normal puisque le modèle fonctionnel simplifié (basé sur une fonction paramétrique F linéaire par rapport à ses paramètres) et le modèle par projection sont deux réalisations pratiques alternatives d'un même modèle théorique (dans le cas de l'espace fonctionnel L^2).

1.4 Analyse de données dans un espace de Hilbert

1.4.1 Principe

L'avantage de travailler dans L^2 (ou plus généralement dans un espace de Hilbert) ne se limite pas à la possibilité de réaliser des projections orthogonales. Les espaces de Hilbert sont en effet la généralisation en dimension quelconque des espaces euclidiens et partagent avec ces derniers de nombreuses propriétés. Or, beaucoup de méthodes d'analyse de données sont construites sans référence particulière à la dimension finie de $\mathbb{R}^n$. Elles utilisent au contraire les opérations "abstraites" associées à la structure vectorielle et euclidienne : combinaison linéaire de vecteurs, calcul de produits scalaires et de normes. Toute méthode écrite exclusivement à partir de ces opérations élémentaires peut s'appliquer, au moins formellement, dans un espace de Hilbert quelconque, donc en particulier dans L^2, par exemple. Cette idée a été exploitée dès les premiers travaux sur l'analyse de données fonctionnelles, comme dans [35] par exemple.

Le neurone fonctionnel défini par l'équation 1.2 illustre ce principe. Quand on le restreint à une entrée fonctionnelle g dans L^2, la sortie peut s'écrire

$$T\left(b + \langle f, g\rangle\right), \tag{1.34}$$

si on désigne par $\langle ., .\rangle$ le produit scalaire dans L^2 (où plus généralement dans un Hilbert).

Dans [A–9, CI–13], j'exploite ce principe pour adapter les réseaux de neurones RBF et les cartes auto-organisatrices de Kohonen (SOM pour *Self Organizing Map*) aux données fonctionnelles. Je montre que ces deux méthodes peuvent s'exprimer directement dans un espace de Hilbert et qu'elles peuvent donc traiter des données fonctionnelles. J'applique ensuite la méthode générale de projection décrite dans la section précédente pour mettre en œuvre les méthodes ainsi définies.

L'utilisation d'une projection demande un peu plus d'attention dans le cas de méthodes construites à partir d'un calcul de norme (réseaux RBF et SOM) que dans le cas des PMC construits à partir d'un produit scalaire. En effet, la représentation des fonctions sur une base se fait souvent à partir d'une famille de fonctions non orthogonales : il est fréquent d'utiliser par exemple des B-splines (pour leur très bonnes propriétés pratiques de stabilité et de localité). Or, si on travaille directement sur les coordonnées des fonctions dans ce type de base, la structure euclidienne utilisée n'est pas la même que celle de L^2. Considérons en effet deux fonctions f et g, projetées en $v(f)$ et $v(g)$ sur l'espace V. On muni V d'une base quelconque $(\psi_k)_{1\leq k\leq p}$ et on note $\gamma(f)$ (resp. $\gamma(g)$) les coordonnées de $v(f)$ (resp. $v(g)$) sur cette base (il s'agit d'un vecteur de $\mathbb{R}^p$). On a alors

$$\langle v(f), v(g)\rangle_{L^2} = \sum_{k=1}^{p}\sum_{l=1}^{p} \gamma(f)_k \gamma(g)_l \langle \psi_k, \psi_l\rangle_{L^2}. \tag{1.35}$$

Le point important est que l'application coordonnées, qui à $v(f) \in V \subset L^2$ associe le vecteur de $\mathbb{R}^p$ $\gamma(f)$, n'est pas en général une isométrie, si on muni $\mathbb{R}^p$ de son produit scalaire canonique. Si la base $(\psi_k)_{1\leq k\leq p}$ est orthonormée, on a bien sûr

$$\langle v(f), v(g)\rangle_{L^2} = \langle \gamma(f), \gamma(g)\rangle_{\mathbb{R}^p}, \tag{1.36}$$

mais c'est faux pour une base quelconque (comme une base de B-splines, par exemple). En pratique, si on soumet les coordonnées sur une base de B-splines des fonctions manipulées

à une méthode quelconque, les résultats obtenus ne seront pas les mêmes que ceux obtenus avec une base orthonormée. En outre, ils ne correspondront pas à ceux qu'on obtiendrait si on pouvait travailler directement dans l'espace fonctionnel V.

Pour maintenir une cohérence entre l'espace fonctionnel et la représentation dans $\mathbb{R}^p$, il faut donc contourner le problème. Une solution très simple consiste à transformer les coordonnées. Si on note Ψ la matrice $(\langle \psi_k, \psi_l \rangle)_{kl}$, l'équation 1.35 devient :

$$\langle v(f), v(g) \rangle_{L^2} = \gamma(f)^T \Psi \gamma(g). \tag{1.37}$$

On calcule alors la décomposition de Cholesky de $\Psi = U^T U$, et on remplace $\gamma(f)$ par $\rho(f) = U\gamma(f)$. On a alors :

$$\langle v(f), v(g) \rangle_{L^2} = \langle \rho(f), \rho(g) \rangle_{\mathbb{R}^p}. \tag{1.38}$$

Grâce à cette transformation élémentaire, on peut utiliser directement les coordonnées des fonctions manipulées (les $\rho(g^i)$) pour les soumettre à une méthode classique prévue pour traiter des données de $\mathbb{R}^p$ (c'est la technique utilisée dans [A–9, CI–13]).

On peut noter qu'un tel travail n'est pas nécessaire pour un PMC. On a en effet

$$\langle v(f), v(g) \rangle_{L^2} = \langle \Psi\gamma(f), \gamma(g) \rangle_{\mathbb{R}^p}. \tag{1.39}$$

La sortie associée à une fonction $v(g)$ par un neurone fonctionnel travaillant avec la fonction de poids $v(f)$ est donc égale à la sortie associée au vecteur $\gamma(g)$ par un neurone numérique travaillant avec le vecteur de poids synaptique $\Psi\gamma(f)$. Pour un réseau RBF ou un SOM, il n'est pas possible de réaliser une transformation des paramètres qui rende les calculs équivalents.

1.4.2 Choix de la base

La principale difficulté pratique des méthodes de représentation réside dans le choix de la base (nature des fonctions et dimension de l'espace de projection). Nous avons déjà rencontré ce problème à la section 1.2.4, dans un contexte légèrement différent : il s'agissait alors de choisir les fonctions utilisées pour représenter les fonctions poids (dans le cas des projections, la base est la même pour les fonctions observées et pour les fonctions de poids).

La solution la plus pertinente *a priori*, mise en œuvre dans [A–10, A–8] et déjà décrite dans les sections 1.2.4, consiste à considérer la base comme un méta-paramètre comme les autres et à l'intégrer dans la méthode de sélection de modèles choisie (validation croisée, *bootstrap*, etc.). Le défaut de cette approche est son coût très élevé.

Il est donc naturel de chercher une solution moins coûteuse. Dans [A–9, CI–13], j'ai retenu l'idée simple qui consiste à préserver au mieux les fonctions. Pour ce faire, on utilise un critère de type *leave-one-out* pour déterminer la meilleure base pour l'ensemble des fonctions : il s'agit simplement d'ajouter les critères leave-one-out de chaque fonction observée pour obtenir une mesure globale de la qualité d'une base (voir la section 3.3 de [A–9] pour des détails).

L'avantage de cette méthode est sa rapidité pour certaines bases. Pour les B-splines, par exemple, le calcul du critère leave-one-out n'est pas plus coûteux que celui des coordonnées des fonctions sur la base : il faut $O(mpN)$ opérations quand on travaille avec N fonctions, discrétisées en m points et représentées sur une base à p B-splines (cf, e.g., [100]). L'inconvénient principal de la méthode est que le critère optimisé correspond à la qualité de la représentation des fonctions et non pas celle du modèle de régression obtenu au final, ce qui conduit à une représentation sous-optimale (parfois trop grossière, mais le plus souvent trop précise).

Les expériences menées dans [A–9, CI–13] montrent cependant que cette méthode simple conduit à des résultats très satisfaisants, même en présence d'une discrétisation irrégulière des fonctions (données manquantes, par exemple). Elle peut être d'ailleurs combinée avec une Analyse en Composantes Principales (ACP), afin de réduire la dimensionnalité des données, une fois celles-ci projetées dans un espace de dimension finie. Il s'agit en fait d'une des méthodes d'ACP fonctionnelle (cf [100]). J'ai utilisé cette approche dans [CI–10, A–9]. Le recours à l'ACP implique cependant de choisir le nombre de composantes à conserver, et donc un recours soit à un critère de qualité de représentation (comme dans [CI–10]), soit à une méthode de sélection de modèle (comme dans [A–9]).

En pratique, le principal défaut de cette approche est qu'elle conduit souvent à sélectionner des bases assez précises, c'est-à-dire avec un nombre élevé de fonctions par rapport au nombre de points d'évaluation. Pour limiter la dimension des vecteurs de coordonnées soumis aux modèles neuronaux, on peut appliquer une méthode de sélection de variables [A–1, CI–1] : on conserve ainsi seulement certaines coordonnées, ce qui revient à réduire la taille de la base de représentation.

1.4.3 Résultats expérimentaux

J'ai tout d'abord comparé les PCM fonctionnels et les réseaux RBF fonctionnels sur des données réelles. Pour les deux modèles, la projection optimale est choisie par un critère de leave-one-out, comme décrit dans la section 1.4.2. Les résultats sont rapportés dans la section 4 de [A–9]. Les méthodes sont comparées sur un jeu de données spectrométrique, le même que celui utilisé pour [A–10, A–8] mais pour de la régression. Les expériences confirment la supériorité connue des PMC sur les réseaux RBF, mais ces performances accrues sont obtenues au prix d'un temps de calcul très élevé (l'apprentissage d'un PMC dure 200 fois plus longtemps que celui d'un réseau RBF sur le problème étudié). En outre, l'approche fonctionnelle, en particulier les transformations réalisées sur les données (dérivation, centrage fonctionnel, etc.), est beaucoup plus utile dans le cas des réseaux RBF que dans celui des PMC : l'erreur de prédiction d'un réseau RBF est divisée par 6 quand on passe du modèle classique à un traitement fonctionnel contre une amélioration de 10% dans le cas d'un PMC.

Les simulations de [A–9] explorent aussi la robustesse de l'approche fonctionnelle vis à vis de la discrétisation. On supprime aléatoirement 10% points de discrétisation dans les spectres utilisés, puis on applique la chaîne de traitement complète : projection et détermination par leave-one-out de la base optimale, puis construction d'un modèle neuronal sur les coordonnées. Les expériences montrent la grande robustesse des méthodes fonctionnelles, car les performances ne sont pas significativement dégradées. En outre, les méthodes d'imputation classiques sont totalement incapables de retrouver des valeurs pertinentes pour les données manquantes si on ne tient pas compte du fait que les observations sont des fonctions. Des expériences additionnelles (non publiées) ont, de plus, montré qu'une approche par inter/extrapolation appliquée à chaque spectre donne de moins bons résultats (erreur 20% plus grande) que la détermination globale de la projection optimale.

Des expériences de portée plus limitée sont incluses dans [CI–10, CI–13]. Dans [CI–10], je montre que la régularisation induite par la projection sur une base fonctionnelle est suffisante pour améliorer (d'environ 7%) les performances d'un PMC sur un problème de reconnaissance de phonèmes. Dans [CI–13] j'illustre l'intérêt d'utiliser des transformations fonctionnelles (dérivation par exemple) avant de soumettre des données fonctionnelles à un SOM : cela permet, par exemple, de s'intéresser aux formes des fonctions plutôt qu'à leur valeur moyenne.

Dans le cadre de la régression sur données fonctionnelles, j'ai obtenu des résultats intéressants (rapportés dans [A–1, CI–1]) en réduisant la base choisie grace au critère leave-one-out par une procédure de sélection de variables. Les performances numériques finales ne sont pas très supérieures à celles obtenues avec d'autres modèles, mais les fonctions (ici des spectres) sont représentées de façon très parcimonieuse : on retient un petit nombre de fonctions de base, chacune avec un support réduit. En pratique, cela permet de déterminer des plages d'intérêt dans les données d'origine et donc d'interpréter les résultats obtenus.

1.5 Machines à vecteurs de support

On s'intéresse dans cette section au problème de la discrimination en deux classes. On suppose donc donné n observations $(x_i, y_i)_{1\leq i\leq n}$ avec $x_i \in \mathcal{X}$ et $y_i \in \{-1, 1\}$. L'ensemble $\mathcal{X}$ pourra désigner un espace de Hilbert mais aussi un ensemble quelconque.

1.5.1 Principe

Classifieur linéaire à marge maximale et données fonctionnelles

Quand deux classes sont linéairement séparables, il existe en général une infinité d'hyperplans séparateurs. L'une des idées fondatrices des machines à vecteurs de support (MVS) est de rechercher parmi ces hyperplans celui de marge maximale, c'est-à-dire celui pour lequel la distance entre les observations et l'hyperplan séparateur est la plus grande possible (cf [33], par exemple).

Ce principe peut être mis en œuvre dans tout espace de Hilbert $\mathcal{X}$, car il revient en fait à résoudre un problème d'optimisation quadratique sous contrainte, défini uniquement à partir du produit scalaire dans $\mathcal{X}$:

$$(P_0) \quad \min_{w\in\mathcal{X},\ b\in\mathbb{R}} \langle w, w\rangle_{\mathcal{X}},\ \text{avec } y_i(\langle w, x_i\rangle_{\mathcal{X}} + b) \geq 1,\ 1 \leq i \leq n. \tag{1.40}$$

Comme pour les méthodes étudiées à la section 1.4, les MVS peuvent donc s'appliquer directement à des observations fonctionnelles.

En dimension finie et quand n est plus grand que 2 fois la dimension, il est peu probable que le problème (P_0) possède une solution car les données ne seront généralement plus linéairement séparables [32]. On relâche alors les contraintes pour obtenir le problème suivant :

$$\begin{aligned}(P_C) \quad & \min_{w\in\mathcal{X},\ b\in\mathbb{R},\ \xi\in\mathbb{R}^n} \langle w, w\rangle_{\mathcal{X}} + C\sum_{i=1}^{n}\xi_i, \\ & \text{avec} \quad y_i(\langle w, x_i\rangle_{\mathcal{X}} + b) \geq 1 - \xi_i,\ 1 \leq i \leq n, \\ & \qquad \xi_i \geq 0,\ 1 \leq i \leq n,\end{aligned} \tag{1.41}$$

où C désigne une constante numérique positive mesurant l'importance des erreurs dans le problème : plus C est petit, plus on favorise une petite valeur de $\langle w, w\rangle_{\mathcal{X}}$ (qui s'apparente à l'inverse de la marge) au détriment de la qualité du classifieur.

On pourrait croire qu'en dimension infinie le recours à (P_C) est inutile, car il est toujours possible, sauf dans des cas particuliers pathologiques, de séparer linéairement n fonctions dans L^2, par exemple. Cependant, comme le souligne par exemple [62] en dimension finie, le classifieur obtenu risque d'avoir de mauvaises performances en généralisation tant il colle aux données. En fait, il s'agit d'un problème connu en ADF : le modèle linéaire ne peut

pas être implémenté de façon naïve sur des données fonctionnelles car il est alors victime de sur-apprentissage (cf [100], chapitres 9 et 10).

Or, le problème (P_C) est en fait une forme régularisée du problème (P_0). On montre en effet (cf [63], par exemple) que (P_C) est équivalent au problème suivant :

$$(R_\lambda) \min_{w\in\mathcal{X},\ b\in\mathbb{R}} \frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - y_i(\langle w, x_i\rangle_{\mathcal{X}} + b)\right) + \lambda\langle w, w\rangle_{\mathcal{X}}, \tag{1.42}$$

avec $\lambda = \frac{1}{Cn}$. Il s'agit donc de trouver un classifieur linéaire dont le paramètre est de norme minimale dans l'espace $\mathcal{X}$, ce qui permet de limiter le sur-apprentissage (si λ est bien choisi).

Cette formulation montre aussi les limites de (P_C) pour les données fonctionnelles. Bien que le coût associé à une erreur ne soit pas le coût quadratique de la régression linéaire (mais le *hinge loss*, i.e., $h(u, v) = \max(0, 1 - uv)$, spécifique aux MVS), la formulation (R_λ) montre que (P_C) est proche d'une régression *ridge*, dont on connaît les limites pour les données fonctionnelles [60]. Les expériences réalisées dans [A–5] confirment cette intuition (cf section 1.5.3) : les MVS construits par résolution de (P_C) sur des données fonctionnelles discrétisées finement (et donc avec n petit devant la dimension de l'espace) ont des performances assez mauvaises.

Noyau

On peut contourner ces problèmes par l'utilisation d'un noyau. En dimension finie, ceux-ci sont utilisés pour construire des MVS non linéaires. En effet, le cas où n est plus grand que 2 fois la dimension est très fréquent et le problème peut donc être intrinsèquement non linéaire. Le recours à (P_C) ne résout alors rien. On introduit donc une fonction ϕ de l'espace de départ $\mathcal{X}$ (qui est maintenant un espace quelconque) vers un espace de Hilbert H, et on construit une MVS linéaire dans H sur les données transformées $(\phi(x_1), y_1), \ldots, (\phi(x_n), y_n)$. Le problème (P_C) s'écrit maintenant :

$$\begin{aligned}(P_C)\quad & \min_{w\in H,\ b\in\mathbb{R},\ \xi\in\mathbb{R}^n} \langle w, w\rangle_H + C\sum_{i=1}^{n}\xi_i, \\ & \text{avec}\quad y_i(\langle w, \phi(x_i)\rangle_H + b) \geq 1 - \xi_i,\ 1 \leq i \leq n, \\ & \qquad \xi_i \geq 0,\ 1 \leq i \leq n.\end{aligned} \tag{1.43}$$

Si ϕ n'est pas linéaire, le classifieur correspondant $x \mapsto \text{signe}(\langle w, \phi(x)\rangle_H + b)$ n'est pas non plus linaire en tant que fonction de $\mathcal{X}$ dans $\{-1, 1\}$.

On peut éviter le calcul explicite (et même la définition) de ϕ en considérant le problème dual de (P_C). On montre en effet (pour des espaces de départ très généraux, cf [86]) que (P_C) est équivalent au problème suivant :

$$\begin{aligned}(D_C)\quad & \max_{\alpha\in\mathbb{R}^n} \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \langle\phi(x_i), \phi(x_j)\rangle_H, \\ & \text{avec}\quad \textstyle\sum_{i=1}^{n}\alpha_i y_i = 0, \\ & \qquad 0 \leq \alpha_i \leq C,\ 1 \leq i \leq n.\end{aligned} \tag{1.44}$$

Cette formulation a le mérite d'éviter l'utilisation directe de ϕ car elle ne repose que sur la fonction noyau K, de $\mathcal{X}\times\mathcal{X}$ définie par :

$$K(x_i, x_j) = \langle\phi(x_i), \phi(x_j)\rangle_H. \tag{1.45}$$

De même, le classifieur correspondant est la fonction suivante :

$$x \mapsto \text{signe}\left(\sum_{i=1}^{n} \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle_H + b\right), \tag{1.46}$$

qui s'exprime uniquement en fonction de K.

Dès lors, il suffit de se donner un noyau K positif, c'est-à-dire tel que

$$\forall\, n \geq 1,\ \forall\, (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n,\ \forall\, (x_1, \ldots, x_n) \in (\mathcal{X})^n,\ \sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) \geq 0, \tag{1.47}$$

pour définir le problème (D_C). Le théorème de Moore-Aronszajn [5] assure alors la validité théorique de la construction : quand K est un noyau positif, il existe un espace de Hilbert H et une fonction ϕ de $\mathcal{X}$ dans H tel que pour tout x et y dans $\mathcal{X}$, $K(x, y) = \langle \phi(x), \phi(y) \rangle_H$ (K est en fait le noyau reproduisant de H).

Données fonctionnelles

Comme je l'ai indiqué précédemment, la difficulté avec les données fonctionnelles réside avant tout dans la nécessité d'une régularisation plutôt que d'un recours à une méthode non linéaire. Or, l'équivalence entre (P_C) et (R_λ) (cf équation (1.42) est valable dans le cas de l'utilisation d'un noyau. En fait, on montre (cf [40] par exemple) que (P_C) avec un noyau K est équivalent au problème suivant

$$(R_\lambda) \min_{f \in H} \frac{1}{n} \sum_{i=1}^{n} \max\left(0, 1 - y_i f(x_i)\right)) + \lambda \langle f, f \rangle_H, \tag{1.48}$$

où H désigne l'espace de Hilbert associé à K. L'introduction d'un noyau a donc pour effet à la fois de construire une MVS non linéaire, mais aussi d'introduire une régularisation différente de la pénalité *ridge* classique (cf aussi [111] pour les liens entre les opérateurs de régularisation et les noyaux).

Je propose donc dans [A–5] d'utiliser des noyaux adaptés aux données fonctionnelles, notamment pour contourner les limitations de la régularisation *ridge* pour ces données. L'approche la plus simple consiste à exploiter la structure hilbertienne de l'espace fonctionnel considéré (comme dans la section 1.4). En effet, certains noyaux pour données classiques sont construits exclusivement à partir des opérations euclidiennes. C'est le cas par exemple :

– des noyaux gaussiens donnés par $K(u, v) = e^{-\sigma \|u - v\|^2}$;
– des noyaux polynomiaux donnés par $K(u, v) = (1 + \langle u, v \rangle)^D$.

Plus généralement, je propose la combinaison d'opérateurs de transformations fonctionnelles avec des noyaux. Si P est une fonction d'un espace fonctionnel $\mathcal{X}$ vers un autre espace $\mathcal{D}$ sur lequel un noyau K est défini, alors la fonction Q définie sur $\mathcal{X}^2$ par $Q(u, v) = K(P(u), P(v))$ est un noyau. Je mentionne quelques choix possibles pour P dans la section 4 de [A–5], par exemple :

– des transformations fonctionnelles (centrage, dérivation, etc.) ;
– des projections (sur un sous-espace de dimension finie, comme dans la section 1.3).

En pratique, ces noyaux sont implémentés par des approximations semblables à celles utilisées dans les sections 1.2 et 1.3 qui permettent d'appliquer la méthodologie définie dans L^2 à des fonctions discrétisées.

1.5.2 Consistance

On sait que les MVS conduisent à des estimateurs universellement consistants quand on travaille en dimension finie (cf [115]). La procédure développée dans [11] peut donc être adaptée de façon simple pour montrer qu'on peut obtenir un estimateur universellement consistant de $\mathbb{E}(T|G)$ (avec T à valeurs dans $\{-1,1\}$) en combinant une projection des fonctions observées sur une base hilbertienne tronquée avec une MVS.

Dans [A–5], je propose une extension plus riche en intégrant le choix automatique de la constante de régularisation (le paramètre C du problème (D_C), cf équation 1.44) dans la procédure. La preuve proposée dans [11] est en effet limitée aux cas où les méta-paramètres de l'algorithme utilisé en dimension finie sont choisis dans un ensemble dénombrable. Or, l'algorithme de [62] permet de choisir C dans un intervalle complet (sans discrétisation) en un temps de calcul raisonnable. Il est donc intéressant d'intégrer cette possibilité dans l'analyse de la consistance des MVS fonctionnelles.

Comme dans la section 1.3.2, nous supposons donné un couple (G,T), avec G à valeurs dans un borné d'un espace de Hilbert $\mathcal{X}$ séparable, et T à valeurs dans $\{-1,1\}$ (discrimination à deux classes). Les observations sont n réplications indépendants du couple (cf équation 1.29). La procédure inspirée de [11] est basée sur un découpage de D_n en un ensemble d'apprentissage $((G^i,T^i))_{1\leq i\leq l_n}$ et un ensemble de validation $((G^i,T^i))_{1+l_n\leq i\leq n}$. L'ensemble de validation est utilisé pour choisir les méta-paramètres de la MVS ainsi que la projection utilisée.

Comme pour le théorème 4, nous nous donnons en effet une base hilbertienne $(\psi_k)_{k\geq 1}$ de $\mathcal{X}$, et nous appelons $v_d(g)$ les d premières coordonnées d'un élément g de $\mathcal{X}$ sur la base. Nous choisissons alors un ensemble de méta-paramètres pour les MVS considérées : pour tout $d\geq 1$, l'utilisation des d coordonnées des fonctions permet en effet de se ramener dans $\mathbb{R}^d$ et on doit donc construire une MVS consistante dans cet espace, ce qui demande le choix du paramètre de régularisation C et celui du noyau.

Selon [115], il faut alors utiliser un noyau universel [114], c'est-à-dire tel que les fonctions de la forme

$$x\mapsto \sum_{i=1}^{n}\alpha_i y_i\langle\phi(x_i),\phi(x)\rangle_H + b, \tag{1.49}$$

possèdent la propriété d'approximation universelle pour $C(\mathbb{R}^d,\mathbb{R})$[2].

Il faut en outre que le noyau ne soit pas trop "fin", au sens des nombres de couverture qu'il induit. Plus précisément, on rappelle que le nombre de couverture d'un espace métrique (Z,d), $\mathcal{N}((Z,d),\epsilon)$, est le plus petit nombre de boules de rayon ϵ qui couvrent Z (cf [39] par exemple). A un noyau K défini sur un ensemble X sont associés des nombres de couverture par l'intermédiaire de la fonction ϕ et de l'espace de Hilbert Y induits par K. On définit en effet la métrique d_K sur X de la façon suivante :

$$d_K(u,v) = \|\phi(u)-\phi(v)\|_Y. \tag{1.50}$$

La condition de consistance donnée par Steinwart dans [115] est que pour tout compact X de $\mathbb{R}^d$, il existe $\nu>0$ tel que $\mathcal{N}((X,d_K),\epsilon)=\mathcal{O}(\epsilon^{-\nu})$. Ces deux propriétés (universalité et borne sur les nombres de couverture) sont satisfaites par tout noyau gaussien, par exemple.

Pour tout d, on se donne donc un ensemble fini de noyaux définis sur $\mathbb{R}^d$, $\mathcal{K}_d$, contenant au moins un noyau K_d universel et associé à un réel positif ν_d tel que $\mathcal{N}((X,d_{K_d}),\epsilon)=\mathcal{O}(\epsilon^{-\nu_d})$ pour tout compact X de $\mathbb{R}^d$. Notons que nous parlons ici de noyaux, pas de classe de noyaux.

[2] La définition de [114] est légèrement différente, mais cela ne change rien dans notre contexte.

Si on considère la classe des noyaux gaussiens, par exemple, il faut choisir la valeur de σ pour définir un noyau. Si l'ensemble $\mathcal{K}_d$ est ne contient que des noyaux gaussiens, on ne considérera donc qu'un nombre fini de valeurs possibles pour σ.

On se donne enfin pour tout d un réel $\mathcal{C}_d > 1$. Je montre alors le théorème suivant :

Théorème 5 (Rossi & Villa 2006 [A–5]) *Soit une suite de réels $(\lambda_d)_{d\geq 1}$ telle que*

$$\sum_{d\geq 1} |\mathcal{K}_d| e^{-2\lambda_d^2} < +\infty,$$

et une suite d'entiers positifs $(l_n)_{n\geq 1}$ telle que

$$\lim_{n\to+\infty} l_n = +\infty \qquad \lim_{n\to+\infty} n - l_n = +\infty$$
$$\lim_{n\to+\infty} \frac{l_n \log(n-l_n)}{n-l_n} = 0.$$

Pour tout $d \geq 1$, tout $K \in K_d$ et tout $C \in [0, \mathcal{C}_d]$ on construit la MVS $f_{n,d,K,C}$ de noyau K_d, solution du problème (D_C) pour les observations $((G^i, T^i))_{1\leq i\leq l_n}$. On définit enfin f_n comme la MVS qui minimise

$$\frac{1}{n-l_n} \sum_{i=l_n+1}^{n} \mathbb{I}_{\{f_{n,d,K,C}(G^i)\neq T^i\}} + \frac{\lambda_d}{\sqrt{n-l_n}},$$

sur les trois méta-paramètres d, K et C.

On a alors

$$Lf_n \xrightarrow{n\to+\infty} L^*,$$

où $Lf = P(f(G) \neq T)$ et $L^* = \inf_{f:H\to\{-1,1\}} Lf$.

Le théorème est basé sur une inégalité "oracle" similaire à celle de [11], mais construite de façon plus complexe car l'ensemble de recherche pour C, $[0, \mathcal{C}_d]$, n'est pas dénombrable.

1.5.3 Résultats expérimentaux

Les MVS fonctionnelles ont été appliquées sur trois jeux de données dans [A–5]. La première expérience utilise le problème de reconnaissance de phonèmes étudié dans [11]. Comme dans cet article, j'utilise une base de Fourier pour l'espace L^2, ce qui revient en pratique à s'appuyer sur une transformée de Fourier rapide pour obtenir les coordonnées des fonctions. J'applique la méthodologie consistante du théorème 5. Les résultats obtenus pour les MVS fonctionnelles sont très satisfaisants. Ils confirment en particulier que l'utilisation d'une MVS linéaire dans l'espace fonctionnel donne de très mauvaises performances, la régularisation *ridge* n'étant pas adaptée aux données fonctionnelles. On peut noter que les données comportent 100 observations discrétisées en 8192 points : le modèle fonctionnel est donc particulièrement justifié dans ce cas.

La deuxième expérience utilise une partie du problème de reconnaissance de phonème exploité dans [60]. On dispose ici de 639 fonctions discrétisées en 256 points. J'utilise de nouveau la méthodologie consistante du théorème 5, mais en m'appuyant cette fois-ci sur une base d'ondelettes. Dans cette situation, l'approche fonctionnelle n'améliore que marginalement la qualité des résultats par rapport à une MVS construite directement dans $\mathbb{R}^{256}$. Ceci montre

que les MVS sont suffisamment robustes pour être utilisées en grande dimension, quand le problème reste raisonnable, c'est-à-dire quand le ratio entre le nombre d'observations et la dimension est supérieur à 2 et donc quand les données ont peu de chance d'être linéairement séparables.

Dans une troisième expérience, j'étudie des donnés spectrométriques [120, 122]. J'abandonne ici la méthodologie consistante afin d'explorer les possibilités offertes par les noyaux utilisant une transformation fonctionnelle. Je montre en particulier, comme je l'avais fait dans le cas des PMC sur les mêmes données (cf sections 1.2.5 et 1.4.3), l'intérêt d'utiliser un opérateur de dérivation.

1.6 Conclusion et perspectives

Les résultats théoriques et expérimentaux de ce chapitre ont démontré qu'il était possible d'utiliser des outils neuronaux pour résoudre efficacement des problèmes de modélisation (régression, discrimination et classification) dans lesquels les observations sont décrites par des fonctions.

Les questions théoriques abstraites sont, dans une certaine mesure, bien résolues : si on suppose que les fonctions observées sont parfaitement connues et qu'on peut les manipuler de façon exacte (par exemple calculer un produit scalaire), on dispose de résultats intéressants, notamment sous la forme d'estimateurs consistants de $\mathbb{E}(T|G)$ quand T est à valeurs dans $\{-1, 1\}$ (pour les perceptrons multi-couches en combinant [11] et [39], pour les machines à vecteurs de support grâce à [A–5]). Le cas général de l'estimation de la régression de T en G (pour T à valeurs réelles) est résolu dans [A–7], mais d'une façon moins complète (pas de choix automatique de la dimension de projection). L'approche de [11] semble cependant extensible au cas de la régression général et devrait pouvoir conduire à un estimateur consistant concurrent de ceux proposés dans [44].

Le cas de l'estimation des paramètres est aussi résolu dans [A–10] (le théorème 1 s'applique en cas de connaissance parfaite en identifiant m à ∞, c'est-à-dire en ne conservant que l'approximation induite par l'échantillon fini). On ne dispose pas cependant d'une distribution asymptotique des paramètres, contrairement au cas de la dimension finie (cf [126]). Ce type de résultats, ou mieux encore, un résultat comme celui de [104], serait particulièrement utile en pratique pour justifier une procédure de sélection d'architecture (basée par exemple sur la log-vraisemblance pénalisée comme dans [104]).

Pour se rapprocher de la pratique, il faut tenir compte de la discrétisation des fonctions, un thème relativement peu abordé en ADF. Les résultats de [A–10] montrent qu'on peut estimer correctement les paramètres d'un perceptron multi-couches fonctionnel, même en présence de discrétisation, mais au prix d'une limite séquentielle. Des résultats similaires pour l'estimation de $\mathbb{E}(T|G)$ quand T est à valeurs dans $\{-1, 1\}$ sont proposés dans [7] (toujours avec une limite séquentielle). J'étudie actuellement avec Nathalie Villa le cas des machines à vecteur de support, en intégrant dans l'étude l'utilisation d'opérateurs fonctionnels de dérivation [CN–2, A–2] : nous obtenons un estimateur consistant de $\mathbb{E}(T|G)$, dans le cas de la discrimination à deux classes (et toujours avec une limite séquentielle).

En pratique, enfin, les expériences réalisées sont très satisfaisantes, mais elles ne sont finalement que confirmer ce qu'on retient de la lecture de [100] : quand les données sont fonctionnelles, les méthodes qui utilisent cette information donnent de meilleurs résultats que celles qui l'ignorent. En pratique, le caractère fonctionnel s'exprime soit sous la forme

d'une dimension très élevée pour les observations (associée à une forte corrélation entre les variables), soit sous la forme d'une dépendance entre des caractéristiques fonctionnelles des observations (courbure, localisation des extrema, etc.) et la variable d'intérêt. Dans certains cas particuliers, les fonctions sont en outre discrétisées de façon indépendante pour chacune d'elles, ce qui impose le recours à une forme de modélisation fonctionnelle.

Le fossé entre pratique et théorie reste de ce fait assez large. On s'autorise en pratique à choisir la base de projection des fonctions de façon *ad hoc* (cf la section 1.4.2, par exemple) ou à calculer des dérivées ou d'autres transformées. Les travaux récents menés avec Nathalie Villa [CN–2, A–2] apportent des éléments de justification à l'utilisation d'opérateurs de dérivation, mais les autres techniques restent sans justification. Or, la procédure consistante proposée dans [11], et étendue dans [A–5] et [7], est très coûteuse en pratique (en terme de temps de calcul), sauf quand elle est combinée avec des classifieurs simples comme les k-plus proches voisins, dont on connaît les limitations. C'est cette lourdeur qui justifie le recours à des constructions plus rapides, comme un choix *a priori de la base de projection*, dont les propriétés théoriques restent à explorer. Leur analyse risque cependant de demander des hypothèses fortes sur la distribution de G ou sur $\mathbb{E}(T|G)$, hypothèses dont la validité en pratique n'est généralement pas vérifiable.

Un autre problème important en pratique est celui du calage dans les données fonctionnelles, par exemple quand on observe $g \circ c_g$ plutôt que g, c_g étant une fonction de $\mathbb{R}$ dans $\mathbb{R}$ qui décale la fonction g (cf [100] chapitre 5). De nombreuses méthodes de calage, qui cherchent à estimer c_g ou à s'affranchir de son effet sur g, ont été proposées (cf e.g. [125, 77, 13, 52]). La prise en compte du calage dans les méthodes de modélisation non linéaire est un problème ouvert important, en particulier pour certaines applications, notamment l'analyse de trajectoires [6], pour lesquels la principale source de bruit dans les observations réside justement dans les décalages des points de discrétisation.

Plus généralement, il me semble important de dépasser le modèle fonctionnel "simple". L'analyse de séries temporelles, par exemple, peut se faire par méthode fonctionnelle en choisissant deux échelles temporelles : on représente par exemple l'évolution de la série sur une journée par une fonction, et on s'intéresse à la série temporelle des fonctions (cf, e.g., [10]). Du point de vue théorique, ce genre de modèle empêche l'utilisation de l'hypothèse d'indépendance entre les observations que j'ai souvent utilisée. Comme dans [43, 42], il faut donc étudier le cas des données avec dépendance entre les observations. En pratique, on peut même choisir automatiquement l'échelle temporelle correspondant aux fonctions, comme nous le faisons dans [CI–5]. L'étude des propriétés théoriques de ce genre d'approche reste entièrement à faire.

Chapitre 2

Analyse de tableaux de dissimilarités

Ce chapitre présente mes travaux sur l'analyse de données décrites par un tableau de dissimilarités. Ces travaux ont débuté quand j'ai rejoint le projet AxIS de l'INRIA en octobre 2003. J'ai participé à l'encadrement de la thèse d'Aïcha El Golli à partir de cette époque. Nous avons d'abord étudié les variantes de l'algorithme de cartes auto-organisatrices de Kohonen qui permettent le traitement des tableaux de dissimilarités. Nous avons proposé une nouvelle variante (cf section 2.2.2), puis nous nous sommes intéressés à une implémentation efficace de ce type d'algorithmes (cf section 2.2.3).

La motivation applicative de nos travaux, au sein du projet AxIS, est l'analyse de l'usage des systèmes d'information. Nous avons donc appliqué les algorithmes proposés aux *logs* d'un serveur web, ce qui nous a conduit à la définition et à l'étude de dissimilarités adaptées à ce problème (cf section 2.3).

2.1 Introduction

2.1.1 Analyse de données non vectorielles

Dans certaines applications, les données ne peuvent pas être décrites de façon satisfaisante par un nombre fixé de variables à valeurs numériques, c'est-à-dire sous une forme vectorielle dans laquelle chaque observation est un élément de $\mathbb{R}^p$. Dans le chapitre 1 nous nous sommes intéressés à un cas particulier de cette situation, celui des données fonctionnelles. Bien que les données fonctionnelles soient très proches des données vectorielles, le passage en dimension infinie suffit à justifier, en théorie comme en pratique, le développement de méthodes adaptées.

Cependant, les données n'ont parfois rien de vectoriel ou même de numérique. Parmi les nombreuses applications pratiques qui conduisent à ce type de situation, on peut évoquer quelques exemples concrets. Les données textuelles constituent peut être le cas le plus connu : bien qu'il soit possible de représenter un texte sous une forme vectorielle [105], on perd ainsi sa structure, ce qui limite la qualité et la richesse des traitements envisageables. En représentant le texte sous une forme semi-structurée (par exemple en XML), on conserve la richesse des données initiales et on obtient des résultats plus pertinents (cf [37], par exemple, qui traite plus généralement de documents structurés, sans se limiter au texte pur).

Un autre exemple intéressant est celui des données issues des études métriques comme

la bibliométrie ou la webométrie [127]. En bibliométrie, par exemple, on travaille sur un graphe volumineux dont les sommets sont les auteurs, les articles, les actes de conférences, les revues, etc. et dont les arêtes représentent les relations entre ces entités (par exemple "auteur de", "publié dans" ou "cité par"). Les études les plus fines portent par exemple sur des auteurs individuels, dont on veut comprendre la position dans le champs du savoir concerné, le réseau social, etc. Les auteurs sont donc décrits par une partie du graphe initial. Les études plus générales portent sur des informations agrégées, par exemple au niveau des revues. Dans la plupart des cas, les entités statistiques considérées (auteurs, revues, pays, etc.) sont décrites d'une façon riche et complexe, dans laquelle le graphe des relations deux à deux a une importance cruciale. Il n'est bien sûr pas possible de le résumer par une description vectorielle simple.

Outre ces deux exemples précis, on peut évoquer brièvement certains cas généraux (voir aussi [56] pour un état de l'art récent sur les méthodes neuronales pour ces données non vectorielles). Les données temporelles sont, par exemple, très fréquentes en pratique : on ne considère pas ici l'analyse d'une unique série temporelle, mais le cas où chaque observation est une série temporelle. Quand la série est à valeurs numériques, on peut l'étudier avec les méthodes de l'analyse de données fonctionnelles (cf chapitre 1), mais ces techniques ne sont pas vraiment adaptées au cas où les séries sont à valeurs symboliques, comme c'est le cas dans de très nombreuses applications. On pense en particulier à l'analyse du génome, à celle des navigations d'utilisateurs sur un site web (cf section 2.3), etc. Diverses méthodes d'analyses sont applicables à ce type de données temporelles [102], comme par exemple [57] (qui s'applique aussi à des données structurées).

En autre cas général est celui des données structurées (ou semi-structurées) sous forme d'arbres ou de graphes. Outre l'exemple du texte sous forme XML, déjà évoqué plus haut, qui se décrit naturellement sous forme d'arbres, on peut citer le cas de la structure atomique de molécules, représentée sous forme de graphe ou encore l'approche de [55] qui décrit un signal grâce à un graphe extrait de sa représentation temps-fréquence (voir aussi [17] pour des exemples récents d'utilisation de graphes pour représenter des données).

2.1.2 Dissimilarités et noyaux

Pour créer des méthodes d'analyse et de traitement de données non vectorielles, il est possible de s'appuyer sur une "mesure de comparaison" entre données. Le paradigme des machines à noyau [109] est fondé sur cette idée : on se donne une fonction K positive (cf la section 1.5.1), puis on définit à partir d'elle divers outils, comme les machines à vecteurs de support, par exemple. J'ai décrit dans la section 1.5 mes travaux sur l'application des machines à noyau aux données fonctionnelles.

Une autre solution générique consiste à s'appuyer sur une mesure de similarité ou de dissimilarité. Cette pratique est courante en analyse de données, la plupart des méthodes de classification hiérarchique ascendante étant formulées de sorte à être applicable directement à une matrice de dissimilarités contenant le résultat de la comparaison deux-à-deux des individus étudiés. On peut aussi citer l'exemple de la méthode PAM (*Partitioning Around Medoids*) [76] qui généralise les *k-means* [89] au cas d'un tableau de dissimilarités (cf aussi la généralisation proposée dans [21]).

L'avantage de ces approches réside dans leur aspect générique : elles découplent la machinerie algorithmique des données en limitant de façon drastique les hypothèses faites sur ces dernières. Alors que la grande majorité des méthodes d'analyse de données travaillent

exclusivement sur des données vectorielles, les méthodes basées sur des (dis)similarités ou des noyaux s'appliquent à tout type de données, à condition de pouvoir définir une mesure de comparaison adaptée. Or, une telle mesure se construit relativement facilement car les hypothèses à vérifier sont très faibles, contrairement aux contraintes imposées par le modèle vectoriel. En pratique, le bénéfice est notable car le découplage a une conséquence importante en terme d'implémentation : la mise en œuvre d'une méthode générique est généralement elle-même générique. Pour exploiter un programme implémentant des machines à vecteurs de support, il suffit de programmer le calcul du noyau utilisé. Ceci permet d'exploiter des bibliothèques hautement optimisées, généralement disponibles sous une licence libre, comme par exemple libSVM pour les machines à vecteurs de support [22]. La vitalité scientifique du domaine des approches génériques est d'ailleurs attesté à la fois par la disponibilité de tels outils très performants, mais aussi par les très nombreuses publications traitant de ces sujets (un numéro spécial de *Pattern Recognition* est par exemple consacré aux méthodes basées sur les dissimilarités est sous presse en juin 2006 [12]).

2.1.3 Contribution personnelle

Mes contributions à l'analyse de données non vectorielles portent à la fois sur l'étude de méthodes génériques et sur celle de dissimilarités pour certains types de données.

Je me suis d'abord intéressé aux modifications des cartes auto-organisatrices leur permettant de traiter des tableaux de dissimilarités. J'ai étudié une variante des adaptations existantes, basée sur la combinaison du modèle de [81, 82] avec un critère d'affectation inspiré des travaux de [64]. Plus généralement, j'ai travaillé à l'optimisation d'algorithmes de la forme de celui de [81, 82], en proposant notamment une version extrêmement rapide et produisant des résultats strictement équivalents à ceux des algorithmes classiques.

J'ai appliqué ce modèle ainsi que d'autres méthodes de classification sur tableau de dissimilarités à l'analyse de l'usage d'un site web. Je me suis intéressé en particulier à la définition de dissimilarités adaptées aux données obtenues dans ce contexte.

2.2 Cartes auto-organisatrices sur tableaux de dissimilarités

2.2.1 Introduction

Les cartes auto-organisatrices de Kohonen (le SOM pour *Self Organzing Map* [78]) sont un algorithme qui combine classification et projection non linéaire. L'algorithme classique (stochastique ou *batch*) s'appuie sur la structure euclidienne de $\mathbb{R}^n$ et est donc limité aux donnés vectorielles. L'adaptation aux données fonctionnelles ne pose pas de problème particulier (cf [CI–13] et la section 1.4), mais il en va autrement du cas des données non vectorielles, ce qui a motivé la définition de variantes spécifiquement adaptées à certains types de données : données structurées (séries temporelles, arbres et graphes) dans [57], chaînes de caractères (et de symboles) dans [112] ou encore données qualitatives dans [30, 31].

Bien que ces solutions soient performantes, elles sont limitées par leur spécificité. Il donc intéressant de chercher une solution générique s'appuyant seulement sur une mesure de dissimilarité. Plus précisément, on suppose donné un ensemble d'observations $\Omega = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, associé à une fonction d, de $\Omega \times \Omega$ dans $\mathbb{R}^+$ qui vérifie les propriétés suivantes :

- d est symétrique, i.e., $d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i)$;
- d est positive, i.e., $d(\mathbf{x}_i, \mathbf{x}_j) \geq 0$;

– $d(\mathbf{x}_i, \mathbf{x}_i) = 0$.

On cherche à construire un SOM pour Ω, sans faire aucune hypothèse additionnelle.

Ce problème est étudié depuis une dizaine d'années, le premier travail publié sur ce sujet, à ma connaissance, étant [2], dans lequel les auteurs adaptent au cas des dissimilarités leur interprétation probabiliste du SOM (proposée dans [3]). L'idée fondatrice, qui sera reprise dans de nombreux autres travaux, est d'utiliser des éléments de Ω comme prototypes dans les neurones du SOM. Ce principe est aussi central dans l'adaptation décrite dans [79, 81, 82]. Cette solution peut être vue comme une formulation SOM de l'algorithme des *k-means* généralisé [21] mentionné plus haut (cf la section 2.2.2 pour une présentation détaillée).

Une approche assez différente est issue des travaux de [18, 67, 68]. Dans ceux-ci, Hofmann et Buhmann définissent un critère de qualité d'une partition en M classes de Ω, construit uniquement à partir de la dissimilarité d [18] :

$$\sum_{c=1}^{M} \frac{1}{2|\mathcal{C}_c|} \sum_{i \in \mathcal{C}_c} \sum_{j \in \mathcal{C}_c} d(\mathbf{x}_i, \mathbf{x}_j), \tag{2.1}$$

où les $(\mathcal{C}_c)_{1 \leq c \leq M}$ sont les classes de la partition de Ω et où $|A|$ désigne le cardinal de l'ensemble A. L'optimisation de ce critère par rapport à la partition est un problème NP complet, ce qui conduit Hofmann et Buhmann à proposer l'utilisation d'un algorithme de recuit par champ moyen (*Mean Field Annealing*). Cette approche est étendue au SOM dans [53, 54, 110].

2.2.2 Modèle étudié

J'ai étudié une variante de l'extension proposée dans [79, 81, 82]. Elle est construite à partir d'une énergie, minimisée grâce à une heuristique (cf [CI–11, A–12, A–4]). Je présente brièvement ici le modèle, les détails étant disponibles dans les articles sus-cités.

La carte auto-organisatrice est construite à partir d'une structure *a priori*, décrite par un graphe (C, Γ). C désigne les M *neurones* de la carte. Chaque neurone est associé à un prototype et à une classe (on aura donc M classes). L'organisation *a priori* provient de l'ensemble d'arêtes Γ : deux neurones c et r sont connectés directement et donc voisins dans la carte si $(c, r) \in \Gamma$. Cette structure de graphe induit une distance discrète δ sur la carte : pour tout couple de neurones (c, r) de la carte, la distance $\delta(c, r)$ est définie comme étant la longueur du plus court chemin, dans (C, Γ), entre c et r.

Le but de l'algorithme SOM est, partant d'un ensemble de N observations, les $\Omega = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, d'associer à chaque neurone $c \in C$ un prototype $\mathbf{p}_c$ et un sous-ensemble $\mathcal{C}_c$ de Ω. On demande que les $(\mathcal{C}_c)_{c \in C}$ forment une partition de Ω et que pour tout c, $\mathbf{p}_c$ représente de façon satisfaisante les éléments de $\mathcal{C}_c$ (il s'agit d'une mesure de qualité de la partition) : ceci correspond à l'aspect classificatoire de l'algorithme SOM. De plus il faut que la structure *a priori* soit respectée, c'est-à-dire que si c et r sont des neurones proches (au sens de la distance δ induite par le graphe Γ), alors $\mathbf{p}_c$ doit représenter correctement les éléments de $\mathcal{C}_r$ (de même pour $\mathbf{p}_r$ par rapport à $\mathcal{C}_c$).

Dans [CI–11, A–12, A–4], nous proposons de représenter le prototype par un sous-ensemble de Ω contenant q éléments distincts. Nous généralisons ainsi le principe proposé initialement dans [2, 79]. Le but de cette généralisation est de limiter l'impact de la discrétisation de l'espace induite par l'impossibilité de recourir à d'autres objets que ceux de Ω.

En s'inspirant de [64], on définit alors l'énergie associée à la carte par

$$E^T\left((\mathcal{C}_c)_{c\in C},(A_c)_{c\in C}\right)=\sum_{\mathbf{x}_i\in\Omega}\sum_{c\in C}K^T\left(\delta(f(\mathbf{x}_i),c)\right)\sum_{\mathbf{x}_j\in A_c}d(\mathbf{x}_i,\mathbf{x}_j), \tag{2.2}$$

dans laquelle A_c désigne le prototype associé à la classe c (qui est donc un sous-ensemble de Ω) et f la fonction d'affectation qui associe à une observation son neurone. K^T désigne un noyau permettant de transformer la distance discrète dans le graphe (C,Γ) en une fonction de voisinage qui impose le respect de la structure *a priori*. Comme dans [121], par exemple, nous construisons K^T à partir d'une fonction noyau K de $\mathbb{R}^+$ dans $\mathbb{R}^+$, décroissante et telle que $K(0)=1$ et $\lim_{x\to\infty}K(x)=0$ (par exemple $K(x)=e^{-x^2}$). On pose ensuite $K^T(x)=K\left(\frac{x}{T}\right)$. Le paramètre T joue le rôle d'une température : quand T est élevé, $K^T(x)$ reste proche de 1 même pour de grandes valeurs de x ; au contraire, une valeur faible engendre une fonction K^T qui décroît très vite vers 0.

L'optimisation de E^T se fait en alternant deux phases, comme dans la version *batch* du SOM classique. Pour ce faire on définit

$$\gamma^T(\mathbf{x},r)=\sum_{c\in C}K^T\left(\delta(r,c)\right)\sum_{\mathbf{x}_j\in A_c}d(\mathbf{x},\mathbf{x}_j), \tag{2.3}$$

ce qui donne

$$E^T\left(\mathcal{P},\mathcal{R}\right)=\sum_{\mathbf{x}_i\in\Omega}\gamma^T(\mathbf{x}_i,f(\mathbf{x}_i)), \tag{2.4}$$

dans laquelle on utilise les notations $\mathcal{P}=(\mathcal{C}_c)_{c\in C}$ et $\mathcal{R}=(A_c)_{c\in C}$.

La première phase de l'algorithme (dite d'*affectation*) consiste à minimiser E^T par rapport à $\mathcal{P}$, en gardant $\mathcal{R}$ fixé. Ceci revient à trouver, pour tout i, $r=f(\mathbf{x}_i)$ qui minimise $\gamma^T(\mathbf{x}_i,r)$.

La deuxième phase de l'algorithme (dite de *représentation*) consiste à minimiser E^T par rapport à $\mathcal{R}$, en gardant $\mathcal{P}$ fixé. Or, E^T se décompose en la somme sur c des énergies suivantes

$$E_c^T(A)=\sum_{\mathbf{x}_i\in\Omega}K^T\left(\delta(f(\mathbf{x}_i),c)\right)\sum_{\mathbf{x}_j\in A}d(\mathbf{x}_i,\mathbf{x}_j), \tag{2.5}$$

qu'on peut optimiser indépendamment les unes des autres. De plus, pour trouver le minimum de E_c^T sur l'ensemble des parties à q éléments distincts, il suffit de trouver les q éléments de Ω, qui donnent les q plus petites valeurs pour $E_c^T(\{\mathbf{x}\})$. Ceci peut se faire par force brute, c'est-à-dire en calculant $E_c^T(\{\mathbf{x}\})$ pour tout $\mathbf{x}\in\Omega$.

En combinant ces deux phases dans une boucle qui les alterne, on obtient l'algorithme 1 (qu'on appelle le DSOM pour *Dissimilarity* SOM). On montre dans [A–4] que la preuve de convergence proposée dans [27] pour le SOM *batch* s'adapte parfaitement à l'algorithme ci-dessus. Des détails sur l'implémentation et une discussion des liens avec les algorithmes concurrents sont aussi inclus dans [A–4] (sections 3.3 et 3.4). L'algorithme est mis en œuvre sur des donnés synthétiques et réelles dans [A–12, CI–11, CI–8, A–4, A–3] (cf aussi la section 2.3.5).

2.2.3 Implémentation efficace

Coût algorithmique

Le principal défaut des adaptations du SOM aux tableaux de dissimilarités est leur coût algorithmique élevé, qui conduit à des temps de calcul déraisonnables car incompatibles avec

Algorithme 1 Les cartes auto-organisatrices pour tableau de dissimilarités (DSOM)

1: Choisir une valeur initiale pour les prototypes $(A_c)_{c\in C}$ {Étape d'initialisation}
2: **Pour** $l = 1$ à L **faire**
3: **Pour tout** élément $\mathbf{x}$ de Ω **faire** {Étape d'affectation}
4: calculer

$$f(\mathbf{x}) = \arg\min_{r\in C} \gamma^T(\mathbf{x}, r)$$

5: **Fin pour**
6: **Pour tout** neurone $c \in C$ **faire** {Étape de représentation}
7: calculer

$$A_c = \arg \min_{A\subset\Omega,\ |A|=q} \sum_{\mathbf{x}_i\in\Omega} K^T\left(\delta(f(\mathbf{x}_i), c)\right) \sum_{\mathbf{x}_j\in A} d(\mathbf{x}_i, \mathbf{x}_j),$$

où $|A|$ désigne le cardinal de l'ensemble A.
8: **Fin pour**
9: **Fin pour**

une véritable utilisation en analyse de données exploratoires. Une implémentation optimisée de l'algorithme 1 conduit, par exemple, à une durée d'apprentissage d'une heure et vingt minutes pour un ensemble de 3200 observations (sur un PC standard, cf [A–3]). Je me suis donc intéressé à l'implémentation efficace de l'algorithme 1, afin d'atteindre des temps de calcul acceptables (de l'ordre de la minute).

Intrinsèquement, le problème des dissimilarités est au moins en $\mathcal{O}(N^2)$ pour N observations : il ne semble pas possible, en effet, d'optimiser l'erreur de l'équation 2.1 ou celle de l'équation 2.2 sans la calculer. Il est bien sûr envisageable d'utiliser des techniques d'approximation par échantillonnage (comme dans [93] pour le *multidimensional scaling*), mais pour s'attaquer au problème théorique, il faut impérativement connaître la matrice de dissimilarités complète. Cette dépendance quadratique avec le nombre de données rend le cas des tableaux de dissimilarités intrinsèquement plus coûteux que le SOM classique dont le temps de calcul est en $\mathcal{O}(NMp)$ pour une carte à M neurones appliquées à N observations en dimension p (cf [80] pour une implémentation utilisant des approximations pour réduire encore ce coût). Dans le cas où p est grand devant N, on peut espérer une meilleure complexité pour le DSOM que pour le SOM, mais le temps de calcul de la matrice de dissimilarités elle-même risque de devenir prohibitif.

En outre, le coût d'une implémentation naïve de l'algorithme 1 est supérieur à $\mathcal{O}(N^2)$, en raison de l'étape de représentation. On constate que dans le cas le plus simple ($q = 1$, i.e., quand chaque prototype est représenté par une seule observation), le coût de cette phase est en $\mathcal{O}(N^2M)$. Pour chaque neurone, il faut en effet tester toutes les observations comme prototype, ce qui impose d'évaluer NM fois l'énergie de l'équation 2.5. A ce coût s'ajoute aussi celui de la phase d'affectation, en $\mathcal{O}(NM^2)$. Cette phase peut même dominer les calculs quand M est grand, mais cette situation est rare : contrairement au SOM, il est indispensable, pour les variantes adaptées aux tableaux de dissimilarités, que chaque neurone reçoive au moins un individu dans la phase d'affectation. On doit donc en pratique limiter le ratio M/N et M est généralement de l'ordre de $\sqrt{N}$ au maximum.

Le coût en $\mathcal{O}(N^2M)$ n'est pas spécifique à l'algorithme 1 présenté ici, mais correspond en fait au cas général des adaptations du SOM aux tableaux de dissimilarités.

Un algorithme rapide

Je propose dans [A–3] une modification de l'algorithme de force brute utilisé pour réaliser la phase de représentation de l'algorithme 1. Pour $q = 1$, E_c^T se simplifie en

$$E_c^T(\mathbf{x}_k) = \sum_{i=1}^{N} K^T\left(\delta(f(\mathbf{x}_i), c)\right) d(\mathbf{x}_i, \mathbf{x}_k), \tag{2.6}$$

ce qui peut s'écrire

$$E_c^T(\mathbf{x}_k) = \sum_{u \in C} K^T\left(\delta(u, c)\right) D(u, \mathbf{x}_k), \tag{2.7}$$

avec

$$D(u, \mathbf{x}_k) = \sum_{i \in \mathcal{C}_u} d(\mathbf{x}_i, \mathbf{x}_k). \tag{2.8}$$

Il y a NM valeurs $D(u, \mathbf{x}_k)$ et leur calcul peut se faire en $\mathcal{O}(N^2)$ opérations. Le calcul de E_c^T se fait alors en $\mathcal{O}(M)$, ce qui conduit à un coût total de $\mathcal{O}(N^2 + NM^2)$, généralement dominé par le terme en N^2 : on gagne ainsi un facteur M par rapport à l'algorithme naïf. En outre, cette modification algorithmique ne change absolument rien aux calculs effectués et donc aux résultats de l'algorithme[1].

Dans [A–3], je combine cette amélioration avec deux astuces d'implémentation qui ne changent pas la complexité théorique, mais qui permettent de gagner jusqu'à un facteur trois en temps de calcul effectif.

La première astuce consiste à interrompre prématurément le calcul de $E_c^T(\mathbf{x}_k)$ pour un candidat prototype $\mathbf{x}_k$ quand on s'aperçoit que la valeur déjà atteinte pendant la sommation dépasse la meilleure valeur obtenue jusqu'à présent. Si on ordonne correctement le passage des candidats prototypes, par exemple en s'appuyant sur l'ordre créé progressivement par le DSOM lui-même, on réduit considérablement les calculs inutiles et donc le coût effectif (cf [A–3], section 3.3, pour des détails).

La deuxième astuce consiste à réutiliser les valeurs des $D(u, \mathbf{x}_k)$ d'une itération de l'algorithme sur l'autre. On constate en effet que ces valeurs dépendent seulement de la partition engendrée par le DSOM. Or, celle-ci tend à se stabiliser au fil des itérations. On peut donc superviser les changements d'affectation pour les itérations afin de savoir quelles valeurs doivent être recalculées. En pratique, on réduit considérablement le temps de calcul avec cette technique, surtout en fin d'apprentissage (cf [A–3], section 3.4, pour des détails).

Il est important de noter que les heuristiques d'accélération proposées ne changent strictement rien aux résultats de l'algorithme. En outre, elles ont été choisies afin d'engendrer un sur-coût minimal, même quand elles ne sont jamais activées (par exemple quand on ne peut jamais interrompre prématurément le calcul de $E_c^T(\mathbf{x}_k)$). Dans le cas de la réutilisation des valeurs de $D(u, \mathbf{x}_k)$, le sur-coût est effectivement minimal. Par contre, la première astuce induit un sur-coût qui ralentit parfois légèrement l'algorithme. Il est donc judicieux de n'utiliser que quand M est relativement grand et donc que la probabilité d'une interruption augmente.

[1] Comme l'ordre des calculs n'est pas le même dans l'algorithme d'origine que dans la version proposée, les erreurs d'arrondi pourraient entraîner l'apparition de différences non significatives entre les résultats. En pratique, je n'ai pas observé d'occurrence de ce phénomène.

Évaluation expérimentale

L'algorithme proposé et les astuces d'implémentation ont été implémentés en Java[2], puis testés sur des données simulées et sur des données réelles dans [A–3]. Les expériences montrent que les modèles de coût théorique représentent la réalité de façon satisfaisante. L'accélération des calculs est très importante, en particulier sur des données volumineuses. Pour un jeu de données artificielles comportant $N = 3000$ observations, l'apprentissage d'un DSOM comportant $M = 49$ neurones prend par exemple 865 secondes (plus de 14 minutes) pour l'algorithme classique, contre 22 secondes pour l'algorithme amélioré (sans les astuces). La version intégrant les astuces tourne en un peu plus de 9 secondes dans les mêmes conditions. Sur un jeu de $N = 3200$ données réelles, on passe de 4700 secondes à 104 puis à 70, selon le niveau d'optimisation de l'algorithme, pour un DSOM comportant $M = 225$ neurones.

L'analyse détaillée des résultats (cf section 4 de [A–3]) montre que l'interruption des calculs est efficace quand on utilise beaucoup de neurones (M grand) alors que la réutilisation des $D(u, \mathbf{x}_k)$ d'une itération à l'autre est plus efficace pour les petites valeurs de M : les deux heuristiques se combinent en outre assez efficacement et divisent le temps de calcul par un facteur allant jusqu'à 3,2 sur les expériences réalisées.

2.3 Analyse de l'usage des sites web

Je présente dans cette section mes travaux sur l'analyse de l'usage des sites web par des méthodes fondées sur la comparaison deux à deux des objets étudiés.

2.3.1 Motivation

La construction puis la maintenance continue d'un site Web de taille importante demandent un travail considérable sans lequel le site perd peu à peu tout intérêt aux yeux du public. Le contenu lui-même doit bien entendu correspondre au public visé, mais cela ne suffit pas. L'organisation hyper-textuelle d'un site Web induit en effet un mode de parcours totalement différent de celui des médias traditionnels : il n'y a plus de début et de fin, l'utilisateur étant libre d'interrompre à tout moment un parcours linéaire pour suivre un hyperlien, puis revenir au document précédent grâce à l'opération "page précédente" de son navigateur.

A cette complexité du média, s'ajoute celle induite par les ressources externes. L'indexation d'un site par les moteurs de recherche de référence peut par exemple créer une structure de navigation totalement différente de celle envisagée par les concepteurs du site. L'inclusion du site dans des listes de favoris ou dans des annuaires thématiques peut créer des rapprochements incongrus ou de nouveaux modes de navigation.

Les responsables d'un site Web ne peuvent donc pas se contenter de simples statistiques d'accès pour comprendre l'utilisation de leur site par les internautes. Pour les raisons évoquées au dessus, il est nécessaire en effet de confronter la conception du site à sa perception par les utilisateurs. Pour ce faire, il est possible d'utiliser les traces laissées par les visiteurs d'un site sous forme des fichiers logs des serveurs concernés. Il s'agit alors de réaliser une forme particulière de *Web Usage Mining* (WUM) dans laquelle on cherche à se focaliser sur la perception de l'organisation et du contenu d'un site par ses utilisateurs. Le WUM est utilisé depuis une dizaine d'années dans le but de comprendre et d'améliorer les sites Web (cf [113] par exemple pour une présentation synthétique des objectifs principaux du WUM).

[2]Code disponible sur GForge : `http://somlib.gforge.inria.fr/`

Une méthode d'analyse dirigée par l'usage consiste à réaliser une classification du contenu du site à partir des navigations enregistrées dans les logs du serveur. Les classes ainsi obtenues sont constituées de pages qui ont tendance à être visitées ensembles. Elles traduisent donc les préférences des utilisateurs. Une autre méthode d'analyse consiste à classer les navigations elles-mêmes pour établir une typologie des utilisateurs du site.

La principale difficulté du WUM réside dans la nature des données elles-mêmes. Les navigations sont en effet des séries temporelles à valeurs dans un graphe de ressources (le site). En outre, elles sont observées par l'intermédiaire des *logs* et doivent être reconstruites à partir de ceux-ci, ce qui induit généralement des données de qualité moyenne.

Je présente dans cette section mes travaux en analyse de l'usage. Ma contribution a consisté en la définition et l'étude de dissimilarités adaptées à la comparaison de données d'usage, ainsi qu'en l'application de la méthode générique décrite dans la section précédente à ces données.

2.3.2 Méthodologie

Les données d'usage d'un site Web proviennent essentiellement des fichiers log des serveurs concernés. Ceux-ci sont généralement écrits dans le format CLF (*Common Logfile Format*, [88]) ou dans sa version étendue qui comporte plus d'informations, en particulier le *User Agent* un élément très important pour la reconstruction des navigations (il s'agit, en général, du nom du logiciel de navigation utilisé ainsi que d'une information sur le système d'exploitation, par exemple `"Mozilla/5.0 (X11 ; U ; Linux i686 ; rv :1.7.3) Gecko/20041001 Firefox/0.10.1"` correspond au logiciel Firefox utilisé sous Linux). Une des premières difficultés du WUM est de reconstruire le comportement de chaque utilisateur à partir des logs. Les logs sont en effet constitués de lignes indépendantes, ordonnées selon les dates des requêtes, et contenant, entre autres, l'adresse IP du client associé à une requête et son *User Agent*. Il faut donc combiner les lignes associées pour reconstruire l'historique d'un utilisateur.

Pour réaliser mes analyses, j'ai utilisé les algorithmes d'extraction de navigations développés dans mon équipe [118, 119] (cf aussi [A–4], section 4, pour des détails). Ceux-ci permettent de travailler sur des données multi-sites, en supprimant les requêtes provenant de robots et en reconstruisant efficacement les navigations des utilisateurs (un utilisateur est défini par un couple adresse IP et *User Agent*). Chaque observation est donc une navigation constituée par la liste horodatée des requêtes envoyées par l'utilisateur correspondant (cf la table 2.1 pour un exemple simplifié).

"date"	URL
1	`http://www-sop.inria.fr/`
2	`http://www-sop.inria.fr/act_recherche/les_projets_fr.shtml`
3	`http://www.inria.fr/recherche/equipes/axis`
4	`http://www-sop.inria.fr/axis/`
5	`http://www-sop.inria.fr/axis/ra.html`
6	`http://www.inria.fr/rapportsactivite/RA2003/axis2003/axis_tf.html`

TAB. 2.1 – Un exemple de navigation sur le site web de l'INRIA

Chaque requête correcte contient un URL (*Uniform Ressource Locators*, un cas particulier des *Uniform Ressource Identifiers* [8]). Un URL est de la forme simplifiée suivante : `http:`

`//<host>/<path>` (on ne tient pas compte de la partie recherche qui peut terminer un URL). La partie `<host>` correspond au nom DNS du serveur considéré alors que la partie `<path>` correspond au chemin d'accès au document demandé sur le serveur. L'URL `http://www-sop.inria.fr/axis/` correspond ainsi au serveur `www-sop.inria.fr` et au document `axis/` sur ce serveur.

La plupart des documents d'un site Web sont des pages au format (X)HTML [124, 99] qui contiennent des hyperliens, c'est-à-dire des références vers d'autres documents accessibles sur le Web (sous forme d'URLs). En raison des références internes, un site Web est donc un graphe dont les noeuds sont les documents et les arêtes les liens inclus dans les documents. Comme indiqué précédemment, une navigation est donc une série temporelle à valeurs dans l'ensemble des sommets d'un graphe. Cette série est assez particulière car elle peut contenir des requêtes incorrectes et des requêtes manquantes (quand on revient en arrière avec un navigateur, celui n'envoie généralement pas une nouvelle requête). Une navigation est aussi décrite par d'autres variables, en particulier le navigateur et système d'exploitation utilisés, ainsi que l'adresse IP d'origine, ce qui donne des informations géographiques sur l'internaute.

2.3.3 Simplifications et recodage

Généralisation

L'analyse des navigations, ou du contenu du site à partir de ces dernières, doit donc se baser sur des données non vectorielles. En analyse de l'usage, il est très courant de rechercher des sous-séquences fréquentes de pages ou plus simplement des groupes de pages fréquents parmi les navigations étudiées (cf, e.g., [91, 113]). Cependant, quand le site étudié est très volumineux, ces approches rencontrent des difficultés liées aux faibles supports des sous-séquences à découvrir [90]. En outre, l'analyse des sous-séquences se focalise sur des comportements locaux (généralement induits par la structure du site) au détriment d'une vision globale des navigations. D'autres approches, comme par exemple les modèles génératifs utilisés dans [19], sont confrontés à des problèmes similaire : il est très difficile d'extraire directement des informations pertinentes de l'analyse de l'usage d'un site web très volumineux.

Pour palier ce problème, particulièrement important sur le site de l'INRIA dont les pages se comptent en dizaine de milliers, je m'appuie dans [CI–8, CN–5, A–4] sur un mécanisme de *généralisation* : on classe le contenu du site analysé de sorte à obtenir des groupes de pages homogènes (au sens de la structure de graphe, ou encore du contenu des pages, etc.). Au lieu de travailler au niveau des pages, on travaille sur les classes.

Une solution très simple, qui donne des résultats satisfaisants sur le site de l'INRIA, consiste à tronquer les URLs (comme dans [51]). On exploite ainsi l'organisation hiérarchique du site en supprimant les informations les plus détaillées. Dans l'URL `http://www-sop.inria.fr/axis/Publications/`, par exemple, on retrouve le serveur de l'unité de recherche de l'INRIA située à Sophia-Antipolis (`www-sop.inria.fr`), le projet de recherche AxIS (`axis`) et la liste de publications de ses membres (`Publications`). Suivant le niveau de détails souhaité, on travaillera sur l'URL complet ou sur une version tronquée comme `http://www-sop.inria.fr/axis/`. La table 2.2 donne un exemple de représentation avec deux niveaux conservés (ainsi que le serveur) à partir de la navigation de la table 2.1.

Serveur	Niveau 1	Niveau 2
www-sop.inria.fr		
www-sop.inria.fr	act_recherche	les_projets_fr.shtml
www.inria.fr	recherche	equipes
www-sop.inria.fr	axis	
www-sop.inria.fr	axis	ra.html
www.inria.fr	rapportsactivite	RA2003

TAB. 2.2 – Une représentation simplifiée de la navigation de la table 2.1

Représentation vectorielle

Quand le nombre de "pages" est ramené à une valeur raisonnable, on peut adopter un modèle vectoriel simple inspiré de celui utilisé pour le texte [105] : on compte le nombre de passage de la navigation dans chaque page, ce qui revient à perdre l'information temporelle (comme dans [92], par exemple). Dans le cas du texte, certains succès obtenus par le modèle vectoriel peuvent s'expliquer par les contraintes induites par la grammaire de la langue sur la structure des textes. Celles-ci sont suffisamment fortes pour que l'extraction du vocabulaire seul conserve une part importante de la sémantique du texte. Dans le cas d'une navigation, la structure hypertextuelle joue le rôle d'une grammaire extrêmement stricte : l'ordre de parcours du site est imposé de façon plus ou moins stricte par sa structure. De ce fait, les simples statistiques de passage dans les pages contiennent beaucoup d'information.

Il est bien sûr possible de conserver la structure temporelle des navigations, comme le font de nombreux auteurs (cf [19], par exemple). Je n'ai pas encore exploré cette possibilité car je me suis surtout focalisé (excepté dans [A–4]) sur l'analyse du contenu du site par l'intermédiaire des données d'usage. Dans cette situation, on étudie les pages, décrites par les navigations. La prise en compte de l'aspect temporel (ou plus simplement de l'ordre de parcours) est alors plus délicat, car il est difficile de donner un sens au fait qu'une page soit avant une autre dans une navigation, d'autant plus que cet ordre est avant tout une conséquence de la structure du site.

2.3.4 Dissimilarités

Navigations

Le recours au modèle vectoriel pour le codage des informations n'impose pas l'exploitation de la structure euclidienne de l'espace de description. Dans [A–4], j'analyse les navigations réalisées sur le site web de l'INRIA à partir du modèle vectoriel induit par la troncature des URLs de sorte à garder un niveau et le serveur, mais en utilisant une métrique non euclidienne, le coefficient d'affinité (cf section 4.4.2 de l'article pour des détails). Je m'appuie alors sur le DSOM (cf section 2.3.5).

Pages

L'essentiel de mon travail a porté sur l'analyse des pages d'un site. En utilisant la représentation vectorielle présentée dans la section 2.3.3, on obtient un tableau dont les lignes sont les navigations et les colonnes les pages (ou les groupes de pages). Sa transposition donne

immédiatement une représentation vectorielle pour les pages, illustrée par la table 2.3 dans le cas du site de l'INRIA (avec une troncature des URLs).

Navigations / Rubriques	N_1	N_2	...	N_{3969}
R_1 = inria	0	2	...	0
R_2 = Recherche	1	0	...	0
⋮	⋮	⋮	⋮	⋮
R_{196} = SOP-freesoft	0	0	...	0

TAB. 2.3 – Tableau décrivant les 196 groupes de pages du site de l'INRIA en fonction des navigations

En général, cette représentation vectorielle est problématique car elle est de très grande dimension : dans l'exemple étudié dans [A–4], on considère 196 groupes de pages décrits par 3969 navigations, alors que dans [CI–8, CN–5] on étudie 107 groupes de pages décrits par 16717 navigations. Dans de telles situations, la métrique euclidienne n'est pas du tout adaptée (cf [CN–5] par exemple) et on utilise donc d'autres mesures de dissimilarités, ce qui justifie le recours à des outils génériques comme le DSOM présenté à la section 2.2.2. On se place en fait dans le cadre des métriques non standard (cf [59]).

Dans [CI–8, CN–5, A–4], j'ai utilisé l'indice de Jaccard pour comparer les pages, comme suggéré dans [50], par exemple. Pour cet indice, la similarité entre deux pages A et B est donnée par la probabilité qu'une navigation qui passe par au moins une des deux pages passe par les deux. Les résultats obtenus avec le DSOM (cf la section 2.3.5) sont très satisfaisants, malgré la pauvreté de la représentation des données initiales.

Pour analyser l'impact de la dissimilarité choisie sur les résultats, j'étudie un site de petite taille (91 pages) en collaboration avec Francisco De Carvalho et Alzennyr Da Silva du Centro de Informatica (CIn) de l'UPFE (Recife, Brésil). Nous nous intéressons au site du CIn qui, outre sa petite taille, est organisé de façon très stricte, mais souffre aussi d'une surabondance de liens. Dans [CN–3, CI–2], nous construisons une partition experte du site en 13 classes, puis nous comparons trois dissimilarités (Jaccard, cosinus et une métrique de corrélation basée une pondération de type $tf \times idf$) grâce aux résultats de classification qu'elles produisent (nous utilisons une classification hiérarchique et une méthode de type PAM [21]).

Les résultats obtenus confirment ceux de [CI–8, CN–5, A–4] et montrent que la dissimilarité de Jaccard semble particulièrement adaptée à la classification de pages en fonction de l'usage d'un site. Aucune des trois dissimilarités étudiées ne donne cependant des résultats parfaits, puisque certaines classes expertes ne sont pas retrouvées. L'analyse détaillée des résultats (cf [CN–3, CI–2]) suggère une possible amélioration de ceux-ci par l'inclusion d'informations sur la structure du site dans le calcul des dissimilarités : l'idée serait d'insister sur les rapprochements inattendus entre pages, c'est-à-dire sur les pages qui apparaissent souvent dans une même navigation alors que la structure du site n'incite pas particulièrement à passer de l'une à l'autre.

2.3.5 Application du DSOM

Dans [CI–8, CN–5, A–4], j'ai appliqué le DSOM à l'analyse de l'usage d'un site web (celui de l'INRIA) en utilisant les dissimilarités décrites à la section 2.3.4. Les analyses détaillées

sont données dans les articles cités. Elles ont permis de bien comprendre la façon dont le site de l'INRIA est utilisé.

On a constaté, par exemple, qu'une part importante des navigations sur le site pouvait être qualifiée de navigations "internes" [A–4] portant sur des pages en accès restreint et ainsi que sur des pages contenant des informations pratiques (description des services informatiques, de l'aide à la recherche, etc.). On remarque, dans l'analyse sur les pages, que ces pages internes sont proches de la racine du site de l'unité de recherche de Sophia, ce qui montre que celle-ci joue bien son rôle de pointeur vers des informations utiles [CI–8].

En outre, toutes les analyses font ressortir le relatif isolement des différentes parties du site de l'INRIA : le serveur principal `www.inria.fr` semble bien jouer son rôle d'instrument de communication (en particulier pour les recrutements, cf [CN–5, CI–8]), mais pas celui de point d'entrée pour les sites des projets de recherche. Les navigations qui les concernent correspondent plutôt à des entrées directes sur les sites (par l'intermédiaire d'une recherche sur un moteur, par exemple).

Un exemple intéressant de communication réussie est fourni par l'ex-projet Robotvis, dont le site est l'un des plus visités à l'INRIA. L'analyse fait ressortir que son succès est avant tout du à la disponibilité de démonstrations en ligne des algorithmes développés par le projet [CN–5].

Globalement, l'analyse visuelle permise grâce au DSOM (et aux autres techniques proposées dans [CN–5]), associée à la qualité de l'indice de Jaccard, donne des résultats très intéressants qui permettent une véritable analyse exploratoire des logs du site de l'INRIA.

2.4 Conclusion et perspectives

L'adaptation des cartes auto-organisatrices de Kohonen aux tableaux de dissimilarités proposée dans ce chapitre est doublement efficace : l'algorithme rapide développé dans [A–3] permet d'explorer des données volumineuses en un temps très raisonnable ; les expériences en analyse de l'usage (cf section 2.3.5) donnent des résultats très satisfaisants sur des données réelles.

Il existe cependant de nombreuses autres variantes de cartes auto-organisatrices adaptées aux tableaux de dissimilarités (cf section 2.2.1). Certaines ont des propriétés théoriques intéressantes : celle de [54], par exemple, optimise un critère pertinent même quand la dissimilarité étudiée n'est pas métrique. Il semble donc important de réaliser une étude comparative, tant en termes théoriques (énergie optimisée, convergence, etc.) que pratiques (qualité du résultat testé sur des données réelles). Ceci pose le problème de l'adaptation aux cas des dissimilarités des nombreuses méthodes d'évaluation de la qualité d'un SOM (cf, e.g., [98]).

Les expériences réalisées en analyse de l'usage sont intéressantes, mais elles relèvent plus, pour l'instant, du domaine des métriques non standard [59] que de celui des données non vectorielle [56]. J'ai en effet utilisé une transformation des données d'origine vers une forme vectorielle à laquelle j'applique ensuite une métrique non euclidienne. Dans un tel contexte, il est envisageable de développer un SOM directement applicable à ces modèles, sans passer par un tableau de dissimilarités. L'avantage serait de ne pas restreindre les prototypes à être des données, limitant ainsi les effets néfastes de la quantification qu'induit cette contrainte.

Une autre piste est bien sûr la définition de dissimilarités exploitant de façon plus complète toute la richesse des données d'usage, en s'appuyant par exemple sur une analyse de la structure du site étudié. Il semble aussi intéressant de guider le processus de généralisation par

troncature au moyen des données d'usage. Ceci nécessite cependant la définition de sites de référence, pour lesquels la sémantique et les cas d'utilisation sont bien connus et maîtrisés, travail que j'ai amorcé dans [CN–3, CI–2].

Plus généralement, on peut se demander si les méthodes génériques sur tableaux de dissimilarités sont compétitives par rapport aux méthodes spécifiquement construites pour certains types de données complexes. Le DSOM (algorithme 1), par exemple, est sensible au problème de quantification de l'espace de départ : si les données d'origine sont mal réparties, la carte obtenue peut être de qualité médiocre, alors que dans le SOM classique, une forme d'interpolation est réalisée par certains neurones. D'un autre côté, on ne sait pas traiter de façon directe certains types de données. En bibliométrie, par exemple, l'analyse des liens entre auteurs est réalisée quasi-exclusivement par le biais des (co)-citations (cf, e.g., [127, 15]), informations qui sont intrinsèquement sous la forme d'un tableau de similarités. Il semble donc important de poursuivre le développement de méthodes génériques, tout en les comparant, sur des données réelles, aux méthodes spécifiques qui sont parfois disponibles.

Chapitre 3

Bibliographie personnelle

Ce chapitre recense mes publications et donne une analyse d'impact succincte de celles-ci. Il s'agit de publications acceptées par un comité de lecture, à l'exception de ma thèse [T–1] et de l'ouvrage collectif consacré à la *Gnu Scientific Library* [B–1].

3.1 Analyse d'impact

3.1.1 Décompte

Le tableau suivant décompte mes publications :

Support	Nombre d'articles
Revue internationale	8
Revue nationale	4
Ouvrage collectif	1
Chapitre d'ouvrage collectif	1
Conférence internationale	25
Conférence nationale	14

3.1.2 Invitation

Certains communications dans des congrès, listées dans les sections suivantes, ont été effectuées suite à des invitations ou en raison de l'organisation d'une session spéciale :

- [CN–9] : invitation des organisateurs du congrès (session plénière) ;
- [CI–7] : invitation des organisateurs d'une session spéciale sur les données fonctionnelles ;
- [CI–4] : session spéciale organisée par mes soins ;
- [CI–3] : invitation des organisateurs d'une session spéciale sur la visualisation.

3.1.3 Facteur d'impact

Le tableau suivant donne les facteurs d'impact des journaux dans lesquels j'ai publié des articles (quand ces facteurs sont disponibles dans la base de l'ISI).

Journal	Impact 2004	Impact 2005
Chemometrics and Intelligent Laboratory Systems	1.899	1.770
Neural Networks	1.736	1.665
Neurocomputing	0.641	0.790
Neural Processing Letters	0.605	0.701
Comptes rendus de l'Académie des Sciences – Série I	0.284	0.469

3.1.4 Citations

J'indique ici les citations dont mes articles ont fait l'objet, en excluant celles venant de publications comptant au moins un auteur en commun avec l'article cité. Cette liste ne prétend pas à l'exhaustivité.

- [A–9] est cité par [29, 94, 117, 123]
- [A–10] est cité par [47, 94]
- [CI–9] est cité par [117]
- [CI–13] est cité par [47, 59, 83, 117, 123]
- [CI–16] est cité par [96]
- [CI–18] est cité par [56, 58]
- [CI–22] est cité par [75, 84]
- [CI–25] est cité par [41]

3.2 Articles (revues avec comité de lecture)

[A–1] F. Rossi, D. François, V. Wertz, and M. Verleysen. Fast selection of spectral variables with b-spline compression. *Chemometrics and Intelligent Laboratory Systems*, 2006. In Press. `http://dx.doi.org/10.1016/j.chemolab.2006.06.007`.

[A–2] N. Villa and F. Rossi. Un résultat de consistance pour des svm fonctionnels par interpolation spline. *Comptes Rendus Mathématiques*, 343(8) :555–560, Octobre 2006.

[A–3] B. Conan-Guez, F. Rossi, and A. El Golli. Fast algorithm and implementation of dissimilarity self-organizing maps. *Neural Networks*, 19(6–7) :855–863, July–August 2006.

[A–4] A. El Golli, F. Rossi, B. Conan-Guez, and Y. Lechevallier. Une adaptation des cartes auto-organisatrices pour des données décrites par un tableau de dissimilarités. *Revue de Statistique Appliquée*, LIV(3) :33–64, 2006.

[A–5] F. Rossi and N. Villa. Support vector machine for functional data classification. *Neurocomputing*, 69(7–9) :730–742, March 2006.

[A–6] F. Rossi, A. Lendasse, D. François, V. Wertz, and M. Verleysen. Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemometrics and Intelligent Laboratory Systems*, 80(2) :215–226, February 2006.

[A–7] F. Rossi and B. Conan-Guez. Theoretical properties of projection based multilayer perceptrons with functional inputs. *Neural Processing Letters*, 23(1) :55–70, February 2006.

[A–8] F. Rossi and B. Conan-Guez. Un modèle neuronal pour la régression et la discrimination sur données fonctionnelles. *Revue de Statistique Appliquée*, LIII(4) :5–30, 2005.

[A–9] F. Rossi, N. Delannay, B. Conan-Guez, and M. Verleysen. Representation of functional data in neural networks. *Neurocomputing*, 64 :183–210, March 2005.

[A–10] F. Rossi and B. Conan-Guez. Functional multi-layer perceptron : a nonlinear tool for functional data analysis. *Neural Networks*, 18(1) :45–60, January 2005.

[A–11] F. Rossi and B. Conan-Guez. Estimation consistante des paramètres d'un modèle non linéaire pour des données fonctionnelles discrétisées aléatoirement. *Comptes rendus de l'Académie des Sciences - Série I*, 340(2) :167–170, January 2005.

[A–12] A. El Golli, B. Conan-Guez, and F. Rossi. Self organizing map and symbolic data. *Journal of Symbolic Data Analysis*, 2(1), November 2004.

3.3 Article soumi

[PP–1] D. François, F. Rossi, V. Wertz, and M. Verleysen. Resampling methods for parameter-free and robust feature selection with mutual information. Technical report, INRIA, July 2006. Submitted to Neurocomputing.

3.4 Editorial

[E–1] J. J. Steil, G. C. Cawley, and F. Rossi. New issues in neurocomputing. *Neurocomputing*, 69(7–9), 2006.

3.5 Thèse

[T–1] F. Rossi. *Calcul de différentielles dans les réseaux de neurones généralisés : algorithmes, complexité, implantation logicielle et applications.* Thèse de doctorat, Paris IX Dauphine, Décembre 1996.

3.6 Ouvrage collectif

[B–1] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi. *GNU Scientific Library Reference Manual.* Network Theory Ltd., second edition, February 2003.

3.7 Chapitre d'ouvrage collectif

[BC–1] F. Rossi and B. Conan-Guez. Multi-layer perceptron and symbolic data. In E. Diday and M. Noirhomme-Fraiture, editors, *Symbolic Data Analysis and the SODAS Software.* Wiley, 2006. To be published.

3.8 Conférences internationales avec comité de lecture et publication des actes

[CI–1] F. Rossi, D. François, V. Wertz, and M. Verleysen. A functional approach to variable selection in spectrometric problems. In *Proceedings of ICANN 2006*, Athens, Greece, September 2006.

[CI–2] F. Rossi, F. De Carvalho, Y. Lechevallier, and A. Da Silva. Dissimilarities for web usage mining. In V. Batagelj, H.-H. Bock, A. Ferligoj, and A. Žiberna, editors, *Data Science and Classification (Proceedings of IFCS 2006)*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 39–46, Ljubljana, Slovenia, July 2006. Springer.

[CI–3] F. Rossi. Visualization methods for metric studies. In *Proceedings of the International Workshop on Webometrics, Informetrics and Scientometrics*, pages 356–366, Nancy, France, May 2006.

[CI–4] F. Rossi. Visual data mining and machine learning. In *Proceedings of XIVth European Symposium on Artificial Neural Networks (ESANN 2006)*, pages 251–264, Bruges (Belgium), April 2006.

[CI–5] T. Kärnä, F. Rossi, and A. Lendasse. Ls-svm functional network for time series prediction. In *Proceedings of XIVth European Symposium on Artificial Neural Networks (ESANN 2006)*, pages 473–478, Bruges (Belgium), April 2006.

[CI–6] B. Conan-Guez, F. Rossi, and A. El Golli. A fast algorithm for the self-organizing map on dissimilarity data. In *Proceedings of the 5th Workshop on Self-Organizing Maps (WSOM 05)*, pages 561–568, Paris (France), September 2005.

[CI–7] F. Rossi and N. Villa. Classification in Hilbert spaces with support vector machines. In *Proceedings of the XIth International Symposium on Applied Stochastic Models and Data Analysis (ASMDA 2005)*, pages 635–642, Brest (France), May 2005.

[CI–8] F. Rossi, A. El Golli, and Y. Lechevallier. Usage guided clustering of web pages with the median self organizing map. In *Proceedings of XIIIth European Symposium on Artificial Neural Networks (ESANN 2005)*, pages 351–356, Bruges (Belgium), April 2005.

[CI–9] N. Villa and F. Rossi. Support vector machine for functional data classification. In *Proceedings of XIIIth European Symposium on Artificial Neural Networks (ESANN 2005)*, pages 467–472, Bruges (Belgium), April 2005.

[CI–10] B. Conan-Guez and F. Rossi. Phoneme discrimination with functional multilayer perceptron. In D. Banks, L. House, F. R. McMorris, P. Arabie, and W. Gaul, editors, *Classification, Clustering, and Data Mining Applications (Proceedings of IFCS 2004)*, pages 157–165, Chicago, Illinois (USA), July 2004. IFCS, Springer.

[CI–11] A. El Golli, B. Conan-Guez, and F. Rossi. A self organizing map for dissimilarity data. In D. Banks, L. House, F. R. McMorris, P. Arabie, and W. Gaul, editors, *Classification, Clustering, and Data Mining Applications (Proceedings of IFCS 2004)*, pages 61–68, Chicago, Illinois (USA), July 2004. IFCS, Springer.

[CI–12] F. Rossi and B. Conan-Guez. Functional preprocessing for multilayer perceptrons. In *Proceedings of XIIth European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 319–324, Bruges (Belgium), April 2004.

[CI–13] F. Rossi, B. Conan-Guez, and A. El Golli. Clustering functional data with the SOM algorithm. In *Proceedings of XIIth European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 305–312, Bruges (Belgium), April 2004.

[CI–14] N. Delannay, F. Rossi, B. Conan-Guez, and M. Verleysen. Functional radial basis function network. In *Proceedings of XIIth European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 313–318, Bruges (Belgium), April 2004.

[CI–15] B. Conan-Guez and F. Rossi. Multilayer perceptrons for functional data analysis : a projection based approach. In J. R. Dorronsoro, editor, *Artificial Neural Networks – ICANN 2002*, pages 667–672, Madrid (Spain), August 2002. Springer.

[CI–16] F. Rossi and B. Conan-Guez. Multilayer perceptron on interval data. In A. S. K. Jajuga and H.-H. Bock, editors, *Classification, Clustering, and Data Analysis (IFCS 2002)*, pages 427–434, Cracow (Poland), July 2002. Springer.

[CI–17] F. Rossi, B. Conan-Guez, and F. Fleuret. Functional data analysis with multilayer perceptrons. In *Proceedings of IJCNN 2002 (WCCI 2002)*, volume 3, pages 2843–2848, Honolulu, Hawai (USA), May 2002. IEEE/NNS/INNS.

[CI–18] F. Rossi, B. Conan-Guez, and F. Fleuret. Theoretical properties of functional multilayer perceptrons. In *Proceedings of Xth European Symposium on Artificial Neural Networks (ESANN 2002)*, pages 7–12, Bruges (Belgium), April 2002.

[CI–19] F. Rossi and F. Vautrain. Expert constrained clustering : A symbolic approach. In J. K. Djamel A. Zighed and J. Zytkow, editors, *Principles of Data Mining and Knowledge Discovery*, pages 605–612, Lyon, September 2000.

[CI–20] F. Rossi. Geometrical selection of important inputs with feedforward neural network. In D. W. Pearson, N. C. Steele, and R. F. Albrecht, editors, *Int. Conf. on Artificial Neural Nets and Genetic Algorithms*, April 1997.

[CI–21] F. Rossi. Second Differentials in Arbitrary Feed-Forward Neural Networks. In *Int. Conf. on Neural Networks*, volume I, pages 418–423, Washington (USA), June 1996. IEEE.

[CI–22] F. Rossi. Attribute suppression with multilayer perceptron. In *CESA Multiconference*, volume Symposium on Robotics and Cybernetics, pages 542–547, Lille (France), July 1996. IMACS.

[CI–23] C. Gégout, B. Girau, and F. Rossi. Generic Back-Propagation in Arbitrary Feedforward Neural Networks. In D. W. Pearson, N. C. Steele, and R. F. Albrecht, editors, *Int. Conf. on Artificial Neural Nets and Genetic Algorithms*, pages 168–171, Alès (France), April 1995. Springer Verlag.

[CI–24] C. Gégout, B. Girau, and F. Rossi. NSK, an Object-Oriented Simulator Kernel for Arbitrary Feedforward Neural Networks. In *Int. Conf. on Tools with Artificial Intelligence*, pages 93–104, New Orleans, Louisiana (USA), November 1994. IEEE.

[CI–25] F. Rossi and C. Gégout. Geometrical Initialization, Parametrization and Control of Multilayer Perceptrons : Application to Function Approximation. In *Proceedings of WCCI ICNN*, volume I, pages 546–550, Orlando, Florida (USA), June 1994. IEEE.

3.9 Conférences nationales avec comité de lecture et publication des actes

[CN–1] B. Conan-Guez, F. Rossi, and A. El Golli. Un algorithme efficace pour les cartes auto-organisatrices de kohonen appliquées aux tableaux de dissimilarités. In M. Nadif and F.-X. Jollois, editors, *Actes des treizièmes rencontres de la Société Francophone de Classification*, pages 73–76, Metz, France, September 2006.

[CN–2] N. Villa and F. Rossi. Svm fonctionnels par interpolation spline. In *Actes des 38ièmes Journées de Statistique de la SFDS*, Clamart, France, May/June 2006.

[CN–3] F. Rossi, F. De Carvalho, Y. Lechevallier, and A. Da Silva. Comparaison de dissimilarités pour l'analyse de l'usage d'un site web. In G. Ritschard and C. Djeraba, editors, *Actes des 6ème journées Extraction et Gestion des Connaissances (EGC 2006), Revue des Nouvelles Technologies de l'Information (RNTI-E-6)*, volume II, pages 409–414, Villeneuve d'Ascq, France, January 2006.

[CN–4] F. Rossi, D. François, V. Wertz, and M. Verleysen. Sélection de groupes de variables spectrales par information mutuelle grâce à une représentation spline. In *Actes de la conférence Chimiométrie 2005*, pages 52–55, Villeneuve d'Ascq, France, November–December 2005.

[CN–5] F. Rossi, Y. Lechevallier, and A. El Golli. Visualisation de la perception d'un site web par ses utilisateurs. In S. Pinzon and N. Vincent, editors, *Actes des 5ème journées Extraction et Gestion des Connaissances (EGC 2005), Revue des Nouvelles Technologies de l'Information (RNTI-E-3)*, volume II, pages 563–574, Paris, France, January 2005. Cépaduès-Éditions.

[CN–6] A. Lendasse, D. François, F. Rossi, V. Wertz, and M. Verleysen. Sélection de variables spectrales par information mutuelle multivariée pour la construction de modèles non-linéaires. In *Actes de la conférence Chimiométrie 2004*, pages 44–47, Paris, France, Décembre 2004.

[CN–7] A. El Golli, B. Conan-Guez, F. Rossi, D. Tanasa, B. Trousse, and Y. Lechevallier. Une application des cartes topologiques auto-organisatrices à l'analyse des fichiers logs. In *Actes des onzièmes journées de la Société Francophone de Classification*, pages 181–184, Bordeaux, France, Septembre 2004.

[CN–8] F. Rossi and B. Conan-Guez. Estimation consistante d'un modèle paramétrique fonctionnel en présence de discrétisation aléatoire. In *Actes des XXXVèmes journées de Statistique de la SFdS*, pages 819–822, Lyon, France, Juin 2003.

[CN–9] F. Rossi and B. Conan-Guez. Modélisation supervisée de données fonctionnelles par perceptron multi-couches. In *Actes des neuvièmes journées de la Société Francophone de Classification (conférence invitée)*, pages 93–100, Toulouse, France, Septembre 2002.

[CN–10] B. Conan-Guez and F. Rossi. Approche régularisée du traitement de données fonctionnelles par un perceptron multi-couches. In *Actes des neuvièmes journées de la Société Francophone de Classification*, pages 169–172, Toulouse, France, Septembre 2002.

[CN–11] F. Rossi and F. Vautrain. Traitement symbolique de contraintes expertes en classification automatique. In A. N. F. Le Ber, J.F. Mari and A. Simon, editors, *Actes*

des Septièmes journées de la Société Francophone de Classification, pages 221–227, Nancy, France, Septembre 1999.

[CN–12] J.-P. Aboa Yapo, B. Tang Ahanda, R. Emilion, and F. Rossi. Deux méthodes de segmentation sur un tableau de données symboliques. In *Actes des Sixièmes journées de la Société Francophone de Classification*, Montpellier, France, Septembre 1998.

[CN–13] C. Gégout and F. Rossi. Initialisation des Réseaux de Neurones Non Récurrents à coefficients réels par Algorithmes Evolutionnistes. In *Journées internationales sur Les Réseaux Neuromimétiques et leurs Applications*, pages 416–424, Marseille, Décembre 1994. Neuro-Nîmes.

[CN–14] C. Gégout, B. Girau, and F. Rossi. NSK, un noyau pour la simulation orientée objets de réseaux de neurones. In *Journées internationales sur Les Réseaux Neuromimétiques et leurs Applications*, pages 123–131, Marseille, Décembre 1994. Neuro-Nîmes.

Bibliographie

[1] C. Abraham, P.-A. Cornillon, E. Matzner-Lober, and N. Molinari. Unsupervised curve clustering using b-splines. *Scandinavian Journal of Statistics*, 30(3) :581–595, September 2003.

[2] C. Ambroise and G. Govaert. Analyzing dissimilarity matrices via Kohonen maps. In *Proceedings of 5th Conference of the International Federation of Classification Societies (IFCS 1996)*, volume 2, pages 96–99, Kobe (Japan), March 1996.

[3] C. Ambroise and G. Govaert. Constrained clustering and kohonen self-organizing maps. *Journal of Classification*, 13(2) :299–313, 1996.

[4] D. W. K. Andrews. Consistency in nonlinear econometric models : A generic uniform law of large numbers. *Econometrica*, 55(6) :1465–1471, November 1987.

[5] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3) :337–404, May 1950.

[6] C. Bahlmann and H. Burkhardt. The writer independent online handwriting recognition system *frog on hand* and cluster generative statistical dynamic time warping. *IEEE Trans. Pattern Anal. and Mach. Intell.*, 26(3) :299–310, Mar. 2004.

[7] A. Berlinet, G. Biau, and L. Rouvière. Functional classification with wavelets. *submitted*, 2005.

[8] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI) : Generic Syntax. RFC 2396, The Internet Society, August 1998. `http://www.ietf.org/rfc/rfc2396.txt`.

[9] P. Besse and H. Cardot. Modélisation statistique de données fonctionnelles. In G. Govaert, editor, *Analyse des données*, chapter 6, pages 167–198. Hermès/Lavoisier, 2003.

[10] P. Besse, H. Cardot, and D. Stephenson. Autoregressive forecasting of some functional climatic variations. *Scandinavian Journal of Statistics*, 4 :673–688, 2000.

[11] G. Biau, F. Bunea, and M. Wegkamp. Functional classification in Hilbert spaces. *IEEE Transactions on Information Theory*, 51 :2163–2172, 2005.

[12] M. Bicego, V. Murino, M. Pelillo, and A. Torsello. Similarity-based pattern recognition. *Pattern Recognition*, 2006. In press.

[13] J. Bigot. Landmark-based registration of curves via the continuous wavelet transform. *Journal of Computational and Graphical Statistics*, 2006. To be published.

[14] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[15] K. Börner, C. Chen, and K. Boyack. Visualizing knowledge domains. In B. Cronin, editor, *Annual Review of Information Science & Technology*, volume 37, chapter 5,

pages 179–255. Information Today, Inc./American Society for Information Science and Technology, Medford, NJ, 2003.

[16] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[17] L. Brun and M. Vento. Graph-based representations. *Pattern Recognition*, 39(4) :499–586, April 2006. Special Issue.

[18] J. M. Buhmann and T. Hofmann. A maximum entropy approach to pairwise data clustering. In *Proceedings of the International Conference on Pattern Recognition*, volume II, pages 207–212, Hebrew University, Jerusalem (Israel), 1994. IEEE Computer Society Press.

[19] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Model-based clustering and visualization of navigation patterns on a web site. *Journal of Data Mining and Knowledge Discovery*, 7(4), 2003.

[20] H. Cardot, F. Ferraty, and P. Sarda. Functional linear model. *Statist. & Prob. Letters*, 45 :11–22, 1999.

[21] G. Celeux, E. Diday, G. Govaert, Y. Lechevallier, and H. Ralambondrainy. *Classification Automatique des Données*. Bordas, Paris, 1989.

[22] C.-C. Chang and C.-J. Lin. *LIBSVM : a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[23] T. Chen. A unified approach for neural network-like approximation of non-linear functional. *Neural Networks*, 11 :981–983, 1998.

[24] T. Chen and H. Chen. Approximation of continuous functionals by neural networks with application to dynamic systems. *IEEE Transactions on Neural Networks*, 4(6) :910–918, November 1993.

[25] T. Chen and H. Chen. Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. *IEEE Transactions on Neural Networks*, 6(4) :904–910, July 1995.

[26] T. Chen and H. Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4) :911–917, July 1995.

[27] Y. Cheng. Convergence and ordering of Kohonen's batch map. *Neural Computation*, 9(8) :1667–1676, November 1997.

[28] B. Conan-Guez. *Modélisation supervisée de données fonctionnelles par perceptron multicouches*. Thèse de doctorat, Paris IX Dauphine, Décembre 2002.

[29] F. Corona and A. Lendasse. Input selection and function approximation using the som : an application to spectrometric modeling. In *Proceedings of the 5th Workshop on Self-Organizing Maps (WSOM 05)*, pages 653–660, Paris (France), September 2005.

[30] M. Cottrell, S. Ibbou, and P. Letrémy. SOM-based algorithms for qualitative variables. *Neural Networks*, 17(8–9) :1149–1167, October-November 2004.

[31] M. Cottrell and P. Letrémy. How to use the kohonen algorithm to simultaneously analyze individuals and modalities in a survey. *Neurocomputing*, 63 :193–207, January 2005.

[32] T. M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. On Electronic Computers*, 14 :326–334, 1965.

[33] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

[34] J. Dauxois, L. Ferré, and A.-F. Yao. Un modèle semi-paramétrique pour variables aléatoires hilbertiennes. *C. R. Acad. Sci. Paris*, 333 :947–952, 2001.

[35] J. Dauxois and A. Pousse. *Les analyses factorielles en calcul des probabilités et en statistiques : essai d'étude synthétique*. Thèse d'état, Université Paul Sabatier, Toulouse, 1976.

[36] C. de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer, 1978.

[37] L. Denoyer and P. Gallinari. Bayesian network model for semi-structured document classification. *Information Processing & Management*, 40(5) :807–827, September 2004.

[38] J. Deville. Méthodes statistiques et numériques de l'analyse harmonique. *Annales de l'INSEE*, 15 :3–97, Janvier–Avril 1974.

[39] L. Devroye, L. Györfi, and G. Lugosi, editors. *A Probabilistic Theory of Pattern Recognition*, volume 21 of *Applications of Mathematics*. Springer, 1996.

[40] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1) :1–50, 2000.

[41] M. Fernandez-Redondo and C. Hernandez-Espinosa. A comparison among weight initialization methods for multilayer feedforward networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'00)*, volume 4, pages 543–548, Como, Italy, July 2000.

[42] F. Ferraty, A. Goia, and P. Vieu. Functional nonparametric model for time series : a fractal approach for dimension reduction. *TEST*, 11(2) :317–344, December 2002.

[43] F. Ferraty, A. Goia, and P. Vieu. Régression non-paramétrique pour des variables aléatoires fonctionnelles mélangeantes. *C. R. Acad. Sci. Paris*, 334 :217–220, 2002. Série I.

[44] F. Ferraty and P. Vieu. Dimension fractale et estimation de la régression dans des espaces vectoriels semi-normés. *C. R. Acad. Sci. Paris*, 330 :139–142, 2000. Série I.

[45] F. Ferraty and P. Vieu. The functional nonparametric model and application to spectrometric data. *Computational Statistics*, 17(4), 2002.

[46] F. Ferraty and P. Vieu. Curves discriminations : a nonparametric functional approach. *Computational Statistics and Data Analysis*, 44(1–2) :161–173, 2003.

[47] L. Ferré and N. Villa. Multilayer perceptron with functional inputs : an inverse regression approach. *Scandinavian Journal of Statistics*, 2006. In Press. DOI : `10.1111/j.1467-9469.2006.00496.x`.

[48] L. Ferré and A.-F. Yao. Functional sliced inverse regression analysis. *Statistics*, 37(6) :475–488, November/December 2003.

[49] L. Ferré and A.-F. Yao. Smoothed functional inverse regression. *Statistica Sinica*, 15(3) :665–683, 2005.

[50] A. Foss, W. Wang, and O. R. Zaïane. A non-parametric approach to web log analysis. In *Proc. of Workshop on Web Mining in First International SIAM Conference on Data Mining (SDM2001)*, pages 41–50, Chicago, IL, April 2001.

[51] Y. Fu, K. Sandhu, and M.-Y. Shih. A generalization-based approach to clustering of web usage sessions. In Masand and Spiliopoulou, editors, *Web Usage Analysis and User Profiling*, volume 1836 of *Lecture Notes in Artificial Intelligence*, pages 21–38. Springer, 2000.

[52] F. Gamboa, J.-M. Loubes, and E. Maza. Shifts estimation for high dimensional data. *Under revision by Annals of Stats*, 2005.

[53] T. Graepel, M. Burger, and K. Obermayer. Self-organizing maps : Generalizations and new optimization techniques. *Neurocomputing*, 21 :173–190, November 1998.

[54] T. Graepel and K. Obermayer. A stochastic self-organizing map for proximity data. *Neural Computation*, 11(1) :139–155, 1999.

[55] V. Guigue, A. Rakotomamonjy, and S. Canu. Translation-invariant classification of non-stationary signals. *Neurocomputing*, 69(7–9) :743–753, March 2006.

[56] B. Hammer and B. J. Jain. Neural methods for non-standard data. In *Proceedings of XIIth European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 281–292, Bruges (Belgium), April 2004.

[57] B. Hammer, A. Micheli, A. Sperduti, and M. Strickert. A general framework for unsupervised processing of structured data. *Neurocomputing*, 57 :3–35, March 2004.

[58] B. Hammer and T. Villmann. Mathematical aspects of neural networks. In *Proceedings of XIth European Symposium on Artificial Neural Networks (ESANN 2003)*, pages 59–72, Bruges (Belgium), April 2003.

[59] B. Hammer and T. Villmann. Classification using non standard metrics. In *Proceedings of XIIIth European Symposium on Artificial Neural Networks (ESANN 2005)*, pages 303–316, Bruges (Belgium), April 2005.

[60] T. Hastie, A. Buja, and R. Tibshirani. Penalized discriminant analysis. *Annals of Statistics*, 23 :73–102, 1995.

[61] T. Hastie and C. Mallows. A discussion of "A statistical view of some chemometrics regression tools" by I.E. Frank and J.H. Friedman. *Technometrics*, 35 :140–143, 1993.

[62] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5 :1391–1415, October 2004.

[63] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer-Verlag, 2001.

[64] T. Heskes and B. Kappen. Error potentials for self-organization. In *Proceedings of 1993 IEEE International Conference on Neural Networks (Joint FUZZ-IEEE'93 and ICNN'93 [IJCNN93])*, volume III, pages 1219–1223, San Francisco, California, 1993. IEEE/INNS.

[65] A. E. Hoerl and R. W. Kennard. Ridge regression : Application to non orthogonal problems. *Technometrics*, 12(2) :69–82, 1970.

[66] A. E. Hoerl and R. W. Kennard. Ridge regression : Biased estimation for non orthogonal problems. *Technometrics*, 12(1) :55–67, 1970.

[67] T. Hofmann and J. M. Buhmann. Hierarchical pairwise data clustering by mean-field annealing. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN'95)*, pages 197–202. Springer, 1995.

[68] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1) :1–14, January 1997.

[69] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2) :251–257, 1991.

[70] K. Hornik. Some new results on neural network approximation. *Neural Networks*, 6(8) :1069–1072, 1993.

[71] G. M. James and T. J. Hastie. Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society Series B*, 63 :533–550, 2001.

[72] G. M. James, T. J. Hastie, and C. A. Sugar. Principal component models for sparse functional data. *Biometrika*, 87(3) :587–602, September 2000.

[73] G. M. James and B. Silverman. Functional adaptive model estimation. *Journal of the American Statistical Association*, 100 :565–576, 2005.

[74] G. M. James and C. A. Sugar. Clustering for sparsely sampled functional data. *Journal of American Statistical Association*, 98 :397–408, 2003.

[75] B. Jeong and H. Cho. Feature selection techniques and comparative studies for large-scale manufacturing processes. *The International Journal of Advanced Manufacturing Technology*, 28(9) :1006–1011, April 2006.

[76] L. Kaufman and P. J. Rousseeuw. Clustering by means of medoids. In Y. Dodge, editor, *Statistical Data Analysis Based on the L1-Norm and Related Methods*, pages 405–416. North-Holland, 1987.

[77] A. Kneip, X. Li, K. MacGibbon, and J. Ramsay. Curve registration by local regression. *Canadian Journal of Statistics*, 28(1) :19–29, 2000.

[78] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, third edition, 1995. Last edition published in 2001.

[79] T. Kohonen. Self-organizing maps of symbol strings. Technical report a42, Laboratory of computer and information science, Helsinki University of technoligy, Finland, 1996.

[80] T. Kohonen, S. Kaski, K. Lagus, J. Salöjarvi, J. Honkela, V. Paatero, and A. Saarela. Self organization of a massive text document collection. *IEEE Transactions on Neural Networks*, 11(3) :574–585, May 2000.

[81] T. Kohonen and P. J. Somervuo. Self-organizing maps of symbol strings. *Neurocomputing*, 21 :19–30, 1998.

[82] T. Kohonen and P. J. Somervuo. How to make large self-organizing maps for nonvectorial data. *Neural Networks*, 15(8) :945–952, 2002.

[83] J. A. Lee and M. Verleysen. Generalization of the lp norm for time series and its application to self-organizing maps. In *Proceedings of the 5th Workshop on Self-Organizing Maps (WSOM 05)*, pages 733–740, Paris (France), September 2005.

[84] P. Leray and P. Gallinari. Feature selection with neural networks. *Behaviormetrika*, 26(1) :145–166, 1999. Special issue on Analysis of Knowledge Representation in Neural Network Models.

[85] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6) :861–867, 1993.

[86] C.-J. Lin. Formulations of support vector machines : a note from an optimization point of view. *Neural Computation*, 2(13) :307–317, 2001.

[87] G. Lugosi and K. Zeger. Nonparametric estimation via empirical risk minimization. *IEEE Transactions on Information Theory*, 41(3) :677–687, May 1995.

[88] A. Luotonen. The common logfile format. `http://www.w3.org/pub/WWW/Daemon/User/Config/Logging.html`, 1995.

[89] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley, USA, 1967. University of California Press.

[90] F. Masseglia, D. Tanasa, and B. Trousse. Web usage mining : Sequential pattern extraction with a very low support. In *Advanced Web Technologies and Applications : 6th Asia-Pacific Web Conference, APWeb 2004, Hangzhou, China. Proceedings*, volume 3007 of *LNCS*, pages 513–522. Springer-Verlag, 14-17 April 2004.

[91] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Communication of ACM*, 43(8) :142–151, August 2000.

[92] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery*, 6(1) :61–82, January 2002.

[93] A. Morrison and M. Chalmers. A pivot-based routine for improved parent-finding in hybrid mds. *Information Visualization*, 3(2) :109–122, 2004.

[94] B.-J. Park, W. Pedrycz, and S.-K. Oh. Fuzzy polynomial neurons as neurofuzzy processing units. *Neural Computing & Applications*, 15(3–4) :310–327, June 2006.

[95] J. Park and I. Sandberg. Universal approximation using radial-basis-function. *Neural Computation*, 3 :246–257, 1991.

[96] R. E. Patiño-Escarcina, B. R. Callejas Bedregaland, and A. Lyra. Interval computing in neural networks : One layer interval neural networks. In *Proceeding of 7th International Conference on Information Technology*, volume 3356 of *Lecture Notes in Computer Science*, Hyderabad (India), December 2004.

[97] A. Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, pages 143–195, 1999.

[98] D. Polani. Measures for the organization of self-organizing maps. In L. Jain and U. Seiffert, editors, *Self-Organizing Neural Networks. Recent Advances and Applications*. Springer, 2001.

[99] D. Raggett, A. Le Hors, and I. Jacobs. HTML 4.01 specification. W3C recommandation, W3C, December 1999. `http://www.w3.org/TR/html4/`.

[100] J. Ramsay and B. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer Verlag, June 1997.

[101] J. Ramsay and B. Silverman. *Applied Functional Data Analysis : Methods and Case Studies*. Springer Verlag, 2002.

[102] C. A. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, and M. Vlachos. *Data Mining and Knowledge Discovery Handbook : A Complete Guide for Practitioners and Researchers*, chapter Mining Time Series Data. Kluwer Academic Publishers, 2005.

[103] J. A. Rice and C. O. Wu. Nonparametric mixed effects models for unequally sampled noisy curves. *Biometrics*, 57(1) :253–259, March 2001.

[104] J. Rynkiewicz. Consistent estimation of the architecture of multilayer perceptrons. In *Proceedings of XIVth European Symposium on Artificial Neural Networks (ESANN 2006)*, pages 149–154, Bruges (Belgium), April 2006.

[105] G. Salton, C. Yang, and A. Wong. A vector space model for automatic indexing. *Communications of the ACM*, 18(11) :613–620, 1975.

[106] I. W. Sandberg. General structures for classification. *IEEE Transactions on Circuits and Systems–I : Fundamental Theory and Applications*, 41(5) :372–376, May 1994.

[107] I. W. Sandberg. Notes on weighted norms and network approximation of functionals. *IEEE Transactions on Circuits and Systems–I : Fundamental Theory and Applications*, 43(7) :600–601, July 1996.

[108] I. W. Sandberg and L. Xu. Network approximation of input-output maps and functionals. *Circuits Systems Signal Processing*, 15(6) :711–725, 1996.

[109] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[110] S. Seo and K. Obermayer. Self-organizing maps and clustering methods for matrix data. *Neural Networks*, 17(8–9) :1211–1229, October–November 2004.

[111] A. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11 :637–649, 1998.

[112] P. J. Somervuo. Online algorithm for the self-organizing map of symbol strings. *Neural Networks*, 17(1231–1239), 2004.

[113] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining : Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, 1(2) :12–23, 2000.

[114] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2 :67–93, November 2001.

[115] I. Steinwart. Support vector machines are universally consistent. *J. Complexity*, 18(3) :768–791, September 2002.

[116] M. B. Stinchcombe. Neural network approximation of continuous functionals and continuous functions on compactifications. *Neural Networks*, 12(3) :467–477, 1999.

[117] M. Strickert, U. Seiffert, N. Sreenivasulu, W. Weschke, T. Villmann, and B. Hammer. Generalized relevance LVQ (GRLVQ) with correlation measures for gene expression analysis. *Neurocomputing*, 69(7–9) :651–659, March 2006.

[118] D. Tanasa and B. Trousse. Advanced data preprocessing for intersites web usage mining. *IEEE Intelligent Systems*, 19(2) :59–65, March-April 2004.

[119] D. Tanasa and B. Trousse. Data preprocessing for wum. *IEEE Potentials*, 23(3) :22–25, August-September 2004.

[120] Tecator dataset. Available on statlib : `http://lib.stat.cmu.edu/datasets/tecator`.

[121] S. Thiria, Y. Lechevallier, O. Gascuel, and S. Canu. *Statistique et méthodes neuronales.* Dunod, Paris, 1997.

[122] H. H. Thodberg. A review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE Trans. on Neural Networks*, 7(1) :56–72, January 1996.

[123] A. Vellido, P. J. Lisboa, and D. Vicente. Robust analysis of MRS brain tumour data using t-GTM. *Neurocomputing*, 69(7–9) :754–768, March 2006.

[124] W3C HTML Working Group. XHTML 1.0 the Extensible HyperText Markup Language. W3C recommandation, W3C, August 2002. Second Edition. `http://www.w3.org/TR/xhtml1/`.

[125] K. Wang and T. Gasser. Synchronizing sample curves nonparametrically. *Annals of Statistics*, 27(2) :439–460, 1999.

[126] H. White. Learning in Artificial Neural Networks : A Statistical Perspective. *Neural Computation*, 1(4) :425–464, 1989.

[127] C. S. Wilson. Informetrics. In M. E. Williams, editor, *Annual Review of Information Science and Technology (ARIST)*, volume 34, pages 107–247. Information Today, Inc., 1999.