Estimation de modèles autorégressifs vectoriels à noyaux à valeur opérateur: application à l'inférence de réseaux

Néhémy Lim

Université d'Évry Val d'Essonne
École Doctorale Sciences et Ingénierie

❧

Thèse

présentée par

Néhémy Lim

pour obtenir le grade de

Docteur de l'Université d'Évry Val d'Essonne
Spécialité : Informatique

Estimation de modèles autorégressifs vectoriels à noyaux à valeur opérateur

Application à l'inférence de réseaux

Soutenue le 2 avril 2015 devant le jury composé de

Alain Rakotomamonjy	Professeur, Université de Rouen	Rapporteur
Jean-Philippe Vert	Directeur de recherche, Mines ParisTech	Rapporteur
Jean-Marc Delosme	Professeur, Université d'Évry Val d'Essonne	Examinateur
Pierre Geurts	Associate Professor, Université de Liège	Examinateur
Massimiliano Pontil	Professor, University College London	Examinateur
Florence d'Alché-Buc	Professeur, Télécom ParisTech	Directrice
Cédric Auliac	Docteur, CEA LIST	Co-encadrant

Titre : Estimation de modèles autorégressifs vectoriels à noyaux à valeur opérateur : application à l'inférence de réseaux

Résumé

Dans l'analyse des séries temporelles multivariées, la plupart des modèles existants sont utilisés à des fins de prévision, c'est-à-dire pour estimer les valeurs futures du système étudié à partir d'un historique de valeurs observées dans le passé. Une autre tâche consiste à extraire des causalités entre les variables d'un système dynamique. C'est pour ce dernier problème à visée explicative que nous développons une série d'outils. À cette fin, nous définissons dans cette thèse une nouvelle famille de modèles autorégressifs vectoriels non paramétriques construits à partir de noyaux à valeur opérateur. En faisant l'hypothèse d'une structure sous-jacente creuse, la parcimonie du modèle est contrôlée en imposant dans la fonction de coût des contraintes de parcimonie aux paramètres du modèle (qui sont en l'occurrence des vecteurs qui pondèrent une combinaison linéaire de noyaux). Les noyaux étudiés possèdent parfois des hyperparamètres qui doivent être appris selon la nature du problème considéré. Lorsque des hypothèses de travail ou des connaissances expertes permettent de fixer les paramètres du noyau, le problème d'apprentissage se réduit à la seule estimation des paramètres du modèle. Pour optimiser la fonction de coût correspondante, nous développons un algorithme proximal. *A contrario*, lorsqu'aucune hypothèse relative aux variables n'est disponible, les paramètres de certains noyaux ne peuvent être fixés *a priori*. Il est alors nécessaire d'apprendre conjointement les paramètres du modèle et ceux du noyau. Pour cela, nous faisons appel à un schéma d'optimisation alterné qui met en jeu des méthodes proximales. Nous proposons ensuite d'extraire un estimateur de la matrice d'adjacence encodant le réseau causal sous-jacent en calculant une statistique des matrices jacobiennes instantanées. Dans le cas de la grande dimension, c'est-à-dire un nombre insuffisant de données par rapport au nombre de variables, nous mettons en œuvre une approche d'ensemble qui partage des caractéristiques du boosting et des forêts aléatoires. Afin de démontrer l'efficacité de nos modèles, nous les appliquons à deux jeux de données : des données simulées à partir de réseaux de régulation génique et des données réelles sur le climat.

Mots-clés : Modèles autorégressifs vectoriels, noyaux à valeur opérateur, régularisation, méthodes proximales, inférence de réseaux, jacobienne.

Title : Estimation of operator-valued kernel-based vector autoregressive models with an application to network inference

Abstract

In multivariate time series analysis, existing models are often used for forecasting, i.e. estimating future values of the observed system based on previously observed values. Another purpose is to find causal relationships among a set of state variables within a dynamical system. We focus on the latter and develop tools in order to address this problem. In this thesis, we define a new family of nonparametric vector autoregressive models based on operator-valued kernels. Assuming a sparse underlying structure, we control the model's sparsity by defining a loss function that includes sparsity-inducing penalties on the model parameters (which are basis vectors within a linear combination of kernels). The selected kernels sometimes involve hyperparameters that may need to be learned depending on the nature of the problem. On the one hand, when expert knowledge or working assumptions allow presetting the parameters of the kernel, the learning problem boils down to estimating only the model parameters. To optimize the corresponding loss function, we develop a proximal algorithm. On the other hand, when no prior knowledge is available, some other kernels may exhibit unknown parameters. Consequently, this leads to the joint learning of the kernel parameters in addition to the model parameters. We thus resort to an alternate optimization scheme which involves proximal methods. Subsequently, we propose to build an estimate of the adjacency matrix coding for the underlying causal network by computing a function of the instantaneous Jacobian matrices. In a high-dimensional setting, i.e. insufficient amount of data compared to the number of variables, we design an ensemble methodology that shares features of boosting and random forests. In order to emphasize the performance of the developed models, we apply them on two tracks : simulated data from gene regulatory networks and real climate data.

Keywords : Vector autoregressive models, operator-valued kernels, regularization, proximal methods, network inference, Jacobian

Remerciements

« Attention ! Ce flim n'est pas un flim sur le cyclimse. »

Je tiens d'abord à remercier mes encadrants Florence et Cédric sans lesquels cette thèse n'aurait pu être possible. Avant de connaître Florence, il y a bientôt six ans de cela, le monde de la recherche était pour moi aussi informe qu'une planète inconnue dans une galaxie lointaine. En m'offrant l'opportunité de travailler sur des thématiques à fort potentiel novateur, c'est tout un univers des possibles qui s'est révélé et je ne peux que montrer ma reconnaissance pour la confiance qu'elle m'a accordée. Et même si les premières pistes explorées ne furent pas concluantes, c'est cette confiance mutuelle qui a permis de construire et d'aboutir à une collaboration fructueuse.

Cédric, malgré ses contraintes professionnelles, a toujours su dégager du temps pour faire le point sur les avancées de mes travaux. J'ai ainsi pu bénéficier de son regard aussi perçant que celui de *Superman*, critique, mais toujours bienveillant.

Je suis également reconnaissant envers Alain Rakotomamonjy et Jean-Philippe Vert qui ont accepté d'être rapporteurs de ma thèse, ainsi que Jean-Marc Delosme, Pierre Geurts et Massimiliano Pontil qui m'ont honoré de leur présence en faisant partie de mon jury. J'ai apprécié leurs remarques qui m'ont permis d'envisager mes travaux sous un autre angle.

Au cours de ses pérégrinations, je suis convaincu qu'un (apprenti-)chercheur « *ne vit pas de pain seulement mais de toute parole qui sort de la bouche* » de toutes les personnes (ou devrais-je dire personnages) qu'il a eu la chance de côtoyer. Et ces trois années ont été l'occasion de nombreuses rencontres étant donné la multiplicité de mes lieux de recherche : CEA et IBISC, puis Télécom ParisTech dans les derniers mois de ma thèse.

D'IBISC, je retiens la bonne ambiance et la convivialité qui y règnent. Je ne pense pas me tromper en affirmant que Vincent y a fortement contribué en plus des autres doctorants : Adrien, Amélie, Fred, Lukasz, Stan, Seb et les stagiaires parmi lesquels William, Lise et JB qui ne le sont plus à l'heure où j'écris ces lignes. Désolé si j'oublie quelques noms. J'ai aussi eu des discussions fort intéressantes avec Jean-Christophe et Marie.

Du côté du CEA, je tiens à remercier principalement la machine à café *gratuite*, qui s'est révélée une compagne fidèle lors de longues journées. Plus sérieusement, je me remémore avec grand plaisir des conversations tantôt banales, tantôt passionnées, mais toujours courtoises auxquelles j'ai pu participer lors des diverses pauses-café. L'ambiance au LADIS était aussi au beau fixe et les occasions ne manquaient pas pour se réunir autour d'un pot (anniversaire, arrivée ou départ, diverses bonnes nouvelles). Merci à celles et ceux qui ont

partagé avec moi un thé, un café ou toute autre boisson : Aurore, Mathieu (photographe accrédité), Jérémy, Maxime S., Paul, Anne-Catherine, Flore, Fred, Yoann, Emira, Sylvain, Alberto, Guillaume, Cyrille, Antoine (M. Wikipédia), JP, les apprentis Tic et Tac, puis ceux qui leur ont succédé Bastien et Robin et tant d'autres.

Que serais-je devenu sans mes co-bureaux? Merci à mes compagnons de galère qui m'ont aidé à traverser ces trois années : Céline (ton talent en pâtisserie est à ma connaissance inégalé), Arnaud, Adel et Romain (« *Ben, qu'est ce que t'as à dire yep comme ça?* ») côté IBISC ; Laura, Crédo, Maxime M., Nicolas, Crédo et Antoine côté CEA.

Enfin, dans les moment heureux ou plus délicats, j'ai toujours pu compter sur le soutien inconditionnel de ma famille. Papa, Maman, mes sœurs et frères, merci d'avoir compris les motivations qui m'ont poussé à continuer en thèse et d'avoir cru que je pouvais arriver au bout. Pour finir cette liste de remerciements qui est loin d'être exhaustive, un grand merci à mes amis les plus proches qui m'ont permis de temps en temps de me vider l'esprit : Matthieu L. B., Quentin, « Croco », Gordon, Matthieu B., Anne, Clément, Flo, Sophie et mes colocs Gautier et Mahay.

Table des matières

Notations xiii

Abréviations xv

Introduction 1

I État de l'art 5

1 Précis introductif à l'apprentissage statistique 7

1.1 Introduction . . . 7
1.2 Apprentissage supervisé . . . 8
1.2.1 Formalisme . . . 9
1.2.2 Minimisation du risque empirique . . . 9
1.2.3 Biais, Variance, Complexité, Régularisation . . . 10
1.2.4 Mise en place d'une procédure d'apprentissage . . . 12
1.3 Apprentissage non supervisé . . . 12
1.4 Modèles classiques . . . 13
1.4.1 Régression linéaire . . . 13
1.4.2 Méthodes à noyaux . . . 15
1.5 Méthodes d'ensemble . . . 20
1.5.1 Bagging . . . 21
1.5.2 Boosting . . . 21
1.5.3 Forêts aléatoires . . . 22
1.6 Synthèse . . . 23

2 Éléments d'optimisation convexe pour la parcimonie 25

2.1 Introduction . . . 25
2.2 Rappels d'optimisation convexe et différentiable . . . 26
2.2.1 Premières définitions et propriétés . . . 26
2.2.2 Généralités sur les problèmes d'optimisation . . . 27
2.2.3 Problèmes d'optimisation convexe . . . 28
2.2.4 Algorithmes de référence . . . 30

2.3 Parcimonie . . . 34
2.3.1 Qu'est-ce que la parcimonie ? . . . 34
2.3.2 LASSO . . . 36
2.3.3 Normes et parcimonie structurée . . . 38
2.4 Méthodes proximales . . . 39
2.4.1 Définition et premières interprétations . . . 39
2.4.2 Propriétés fondamentales de l'opérateur proximal . . . 41
2.4.3 Algorithmes proximaux . . . 43
2.4.4 Calcul pratique des opérateurs proximaux . . . 47
2.5 Synthèse . . . 49

3 Modélisation de systèmes dynamiques 51
3.1 Introduction . . . 51
3.2 Généralités sur les systèmes dynamiques . . . 52
3.3 Modèles pour la prévision de séries temporelles . . . 54
3.3.1 Modèles scalaires . . . 54
3.3.2 Modèles vectoriels . . . 62
3.4 Modèles pour l'inférence de réseaux . . . 66
3.4.1 Problématique de l'inférence de réseaux . . . 66
3.4.2 Données statiques . . . 67
3.4.3 Données de séries temporelles . . . 69
3.5 Synthèse . . . 70

II Une nouvelle famille de modèles autorégressifs vectoriels à base de noyaux à valeur opérateur 73

4 Modèles autorégressifs vectoriels non paramétriques 75
4.1 Introduction . . . 75
4.2 Des modèles autorégressifs à noyaux scalaires vers des modèles autorégressifs à noyaux à valeur opérateur . . . 76
4.3 Noyaux à valeur opérateur et espaces fonctionnels associés . . . 80
4.3.1 Régularisation dans les RKHS à valeurs vectorielles . . . 84
4.3.2 RKBS et Régularisation parcimonieuse . . . 87
4.4 Une nouvelle famille de modèles autorégressifs non paramétriques : OKVAR 91
4.4.1 La famille de modèles OKVAR . . . 92
4.5 Apprentissage d'un modèle OKVAR . . . 98
4.5.1 Choix de la régularisation . . . 98

4.5.2 Algorithme proximal . . . 100
4.5.3 Résultats . . . 102
4.6 OKVAR-Boost . . . 107
4.6.1 Principe de l'algorithme . . . 107
4.6.2 Résultats . . . 109
4.7 Synthèse . . . 112

5 Extraction d'influences causales à partir de modèles autorégressifs vectoriels 115
5.1 Introduction . . . 115
5.2 Apprentissage de structure d'un système dynamique à partir de modèles autorégressifs vectoriels . . . 116
5.3 Choix des modèles à noyaux pour la tâche d'apprentissage de structure . . . 118
5.4 Apprentissage des paramètres du modèle OKVAR Hadamard . . . 121
5.5 OKVAR-Boost . . . 124
5.6 Applications . . . 127
5.6.1 Inférence de réseaux de régulation génique . . . 127
5.6.2 Données climat . . . 139
5.7 Synthèse . . . 142

Conclusion et perspectives 145

A Détails des calculs pour l'Algorithme 5.2 151

Bibliographie 153

Table des figures

2.1 Parcimonie : LASSO vs régression ridge 36

3.1 Illustration de la factorisation de la loi jointe d'un modèle de Markov caché. 61

4.1 Patterns de structure générés pour l'évaluation des modèles OKVAR en prévision 103
4.2 MSE moyenne de OKVAR-Boost en fonction du nombre de modèles agrégés pour diverses tailles de sous-ensembles aléatoires sur la base de données synthétiques de taille 10 « Scale-free » (bruit de covariance diagonale) . . . 110
4.3 Patterns de structure générés pour l'évaluation de OKVAR-Boost en prévision 111

5.1 Performances de OKVAR-Prox sur les réseaux de taille 10 de DREAM3 en fonction de la longueur de la série temporelle et du bruit 133
5.2 Performances de OKVAR-Prox sur les réseaux de taille 100 de DREAM3 en fonction de la longueur de la série temporelle 134
5.3 Performances de OKVAR-Prox sur les réseaux de DREAM3 en fonction du nombre de séries temporelles 135
5.4 Impact du type de régularisation choisi sur les performances de OKVAR-Prox sur les réseaux de DREAM3 137
5.5 Graphe causal obtenu par OKVAR-Prox pour une zone du nord Texas et résultat du clustering spectral sur les 125 zones des États-Unis 141

Liste des tableaux

3.1 Récapitulatif des modèles de prévision de séries temporelles existants et de leurs avantages respectifs . . . 65
3.2 Récapitulatif des types de modèles existants pour l'inférence de réseaux à partir de séries temporelles et de leurs avantages respectifs . . . 70

4.1 Récapitulatif des modèles OKVAR étudiés pour la tâche de prévision. . . . 105
4.2 Comparaison des performances en prédiction des modèles OKVAR, VAR(1) sur les bases de données synthétiques de dimension 10 . . . 106
4.3 Comparaison des performances en prédiction de OKVAR-Boost sur les bases de données synthétiques de dimension 10 . . . 111
4.4 Comparaison des performances en prédiction de OKVAR-Boost et VAR(1) sur les bases de données synthétiques de dimension 50 . . . 112

5.1 Récapitulatif des modèles OKVAR étudiés pour la tâche d'inférence de réseaux. . . . 130
5.2 Performances de la famille de modèles OKVAR sur les réseaux de taille 10 de DREAM3 en utilisant l'estimateur $\hat{J}_1$ . . . 131
5.3 Performances de la famille de modèles OKVAR sur les réseaux de taille 10 de DREAM3 en utilisant l'estimateur $\hat{J}_2$ . . . 131
5.4 Performances de la famille de modèles OKVAR sur les réseaux de taille 10 de DREAM3 en utilisant l'estimateur $\hat{J}_3$ . . . 132
5.5 Performances de OKVAR-Prox sur les données de DREAM3 en fonction de la taille de réseau et du bruit . . . 135
5.6 Impact du choix des hyperparamètres sur les performances de OKVAR-Prox sur le réseau E1 de taille 10 de DREAM3 . . . 136
5.7 Comparaison des performances de OKVAR-Prox et de méthodes existantes sur les réseaux de taille 10 de DREAM3 . . . 138
5.8 Comparaison des performances de OKVAR-Prox et de méthodes existantes sur les réseaux de taille 100 de DREAM3 . . . 139
5.9 Comparaison des performances de OKVAR-Prox et de OKVAR-Boost sur les réseaux de tailles 10 et 100 de DREAM3 . . . 140

5.10 Critère BIC pour la sélection des hyperparamètres de OKVAR-Prox pour les données climat 140

Notations

d	nombre de variables d'état
N	taille de l'ensemble d'apprentissage
$\mathcal{S}_N$	ensemble d'apprentissage de taille N
$\mathbf{x}_{1:N+1}$	série temporelle multivariée composée des vecteurs d'état observés $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1} \in \mathbb{R}^d$
x^i	i-ème coordonnée du vecteur $\mathbf{x}$
$\bar{z}$	conjugué du nombre complexe z
$\mathbb{N}_m$	pour $m \in \mathbb{N}^*$, ensemble des entiers de 1 à m
$\lvert \cdot \rvert$	selon le contexte, module d'un nombre complexe, cardinal d'un ensemble ou déterminant d'une matrice
$\mathcal{X}$	ensemble d'entrée
$\mathcal{Y}$	espace de Hilbert de sortie
$\mathcal{L}(\mathcal{Y})$	ensemble des opérateurs linéaires bornés de $\mathcal{Y}$ dans lui-même
A^*	opérateur adjoint de A
$\langle \cdot, \cdot \rangle_{\mathcal{Y}}$	produit scalaire associé à l'espace de Hilbert $\mathcal{Y}$
$\lVert \cdot \rVert_{\mathcal{Y}}$	norme induite par le produit scalaire $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$
$\mathcal{Y}^{\mathcal{X}}$	ensemble des fonctions de $\mathcal{X}$ dans $\mathcal{Y}$
$\mathcal{H}$	selon le contexte, espace d'hypothèses ou RKHS
$\mathcal{H}_K$	RKHS qui admet K comme noyau reproduisant
k	noyau scalaire $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
K	noyau à valeur opérateur $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$
$\mathbb{S}_+^d$	ensemble des matrices symétriques semi-définies positives de taille $d \times d$
ℓ	fonction de perte locale $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
Ω	terme de régularisation
$\mathcal{J}$	fonction de coût global
$\mathcal{T}_\mu$	opérateur de seuillage doux de paramètre $\mu > 0$
$A \circ B$	produit de Hadamard (terme à terme) des matrices A et B
$\lVert A \rVert_F$	norme de Frobenius de la matrice A
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$	loi normale multidimensionnelle d'espérance $\boldsymbol{\mu}$ et de matrice de variance-covariance Σ
$J(h)(\mathbf{x})$	matrice Jacobienne d'un modèle vectoriel h au point $\mathbf{x}$

Abréviations

ARMA	AutoRegressive Moving Average
AUPR	Aire sous la courbe précision-rappel
AUROC	Aire sous la courbe ROC (Receiver Operating Characteristic)
DAG	Directed Acyclic Graph
(D)BN	(Dynamic) Bayesian Network
GGM	Gaussian Graphical Model
GP	Gaussian process
GRN	Gene Regulatory Network
HMM	Hidden Markov Model
i.i.d.	indépendant et identiquement distribué
kNN	k Nearest Neighbors
LGSSM	Linear-Gaussian State Space Model
MLP	Multilayer Perceptron
NW	Nadaraya-Watson
ODE	Ordinary Differential Equation
OKVAR	Operator-valued Kernel Vector AutoRegressive
RKHS	Reproducing Kernel Hilbert Space
RNN	Recurrent Neural Network
SSM	State Space Model
SVR	Support Vector Regression
SVM	Support Vector Machine
VAR	Vector AutoRegressive
VC	Vapnik-Chervonenkis

Introduction

Motivations

Le cours des actions de la bourse, les prévisions économiques, le changement climatique, l'évolution de systèmes biologiques : ces quelques exemples tirés de champs applicatifs très divers attestent de l'omniprésence des systèmes dynamiques. Afin de rendre compte d'un phénomène d'intérêt, les données issues de ces systèmes résultent en général de mesures de plusieurs caractéristiques, appelées *variables d'état*, effectuées à plusieurs points de temps sur un laps de temps fini. Si l'on note d le nombre de variables d'état et N le nombre de points de temps de mesure, ces données prennent la forme d'une série temporelle multivariée de dimension d et de longueur N, notée $\mathbf{x}_{1:N} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subseteq \mathbb{R}^d$ où $\mathbf{x}_\ell$ désigne l'état du système à l'instant $t_\ell, \ell \in \mathbb{N}_N$. Traditionnellement, la recherche en modélisation des systèmes dynamiques s'est concentrée principalement sur la construction de modèles prédictifs qui, à partir d'une série temporelle donnée, permettent d'estimer l'état futur du système. Les travaux de cette thèse ont été initialement motivés par une problématique biologique autour des réseaux de régulation génique. Dans ce contexte, les variables d'état sont des mesures de l'expression de d gènes dont l'activité concertée permet par exemple de réaliser une fonction complexe spécifique au sein d'une cellule vivante. Afin de mieux appréhender le fonctionnement des organismes vivants, l'un des objectifs de la biologie moléculaire moderne consiste à identifier les influences régulatrices entre les gènes (via des protéines) (Segal et al., 2003). Cette tâche canonique en biologie des systèmes est connue sous le nom d'*inférence de réseaux*. Le vocable de *réseau* fait référence au graphe qui encode la structure des influences croisées entre les variables d'état. Dans cette thèse, nous souhaitons proposer un cadre méthodologique général pour traiter la tâche d'extraction d'*influences causales* à partir de données de série temporelle. Par ailleurs, nous tenons à insister sur le fait que bien que les motivations premières de nos travaux aient pour origine un problème biologique, les outils que nous développons demeurent néanmoins génériques, ce que nous montrons par une application sur des données de climat.

Contributions

Nous nous intéressons à des problèmes qui supposent l'existence d'un couplage entres les variables d'état. Classiquement, on admet que l'évolution de l'état d'un système dy-

namique est régie par une fonction h, telle que $\mathbf{x}_{t+1} = h(\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t+1-p}) + \mathbf{u}_{t+1}$ où t est une mesure discrète du temps et $\mathbf{u}_{t+1}$ est un terme de bruit centré. On dit alors que h est un modèle autorégressif d'ordre p. Dans cette thèse, nous faisons l'hypothèse que le processus sous-jacent est markovien d'ordre 1, c'est-à-dire $p = 1$. D'autre part, nous nous plaçons dans le cadre de l'apprentissage statistique. En supposant que h fait partie d'une famille de modèles $\mathcal{H}$, le paradigme de l'apprentissage supervisé consiste à construire un algorithme capable d'inférer un modèle $\hat{h} \in \mathcal{H}$ à partir des vecteurs d'état observés $\mathbf{x}_1, \ldots, \mathbf{x}_N$. L'apprentissage de la dynamique de la plupart des systèmes réels est confronté à plusieurs difficultés qui incluent notamment le caractère non linéaire de la dynamique et la présence de bruit qui entache les données. Dans certaines applications, s'ajoute le problème de la grande dimension lorsque le nombre de variables d'états est supérieur à la longueur de la série temporelle ($N \ll d$). Lorsque des connaissances expertes ne permettent pas d'imposer une forme *a priori* aux modèles, il est d'usage de recourir à des méthodes non paramétriques. Partant de ces hypothèses et observations, les travaux de cette thèse ont consisté à définir une nouvelle famille de modèles autorégressifs vectoriels non paramétriques (*Operator-valued Kernel-based Vector AutoRegressive* OKVAR) (Néhémy Lim et al., 2014) fondés sur la théorie des noyaux à valeur opérateur (Senkene et Tempel'man, 1973) qui est appropriée pour l'apprentissage de fonctions à valeurs vectorielles (C. Micchelli et M. Pontil, 2005). Ces modèles constituent une généralisation des modèles autorégressifs linéaires VAR(1) au cas non linéaire. Nous avons développé un ensemble d'algorithmes d'optimisation fondés sur des méthodes proximales pour l'estimation des paramètres de ces modèles sous contraintes de parcimonie adaptée et l'apprentissage du noyau lui-même.

Pour traiter le problème de la grande dimension, nous avons également conçu une méthode d'ensemble OKVAR-Boost (N. Lim et al., 2013) qui partage des caractéristiques du L_2-boosting (J. Friedman, 2001 ; Peter Bühlmann et Yu, 2003) et des forêts aléatoires (Leo Breiman, 2001). Une autre contribution substantielle de nos travaux concerne la mise en place d'un cadre méthodologique général pour la tâche d'extraction de la structure causale sous-jacente d'un système via le calcul de la matrice jacobienne d'un modèle autorégressif vectoriel comme estimateur de la matrice d'adjacence (N. Lim et al., 2013 ; Néhémy Lim et al., 2014).

Organisation du document

Le présent document est organisé de la façon suivante :

— Le chapitre 1 vise à donner les concepts en apprentissage statistique auxquels nous faisons référence tout le long du mémoire.

— Le chapitre 2 aborde la question de l'optimisation qui découle généralement d'un problème d'apprentissage. Après quelques rappels en optimisation différentiable et convexe, nous discutons du bien-fondé de la parcimonie dans les modèles. Ensuite, nous nous focalisons sur une classe de techniques d'optimisation, les méthodes proximales qui sont particulièrement adaptées pour minimiser les fonctions de coût que nous définissons par la suite.

— Dans le chapitre 3, nous dressons un panorama de la modélisation des systèmes dynamiques. Nous présentons notamment un rapide état de l'art relatif à la tâche d'inférence de réseaux en précisant les mérites et les manques des principales méthodes.

— Dans l'amorce du chapitre 4, nous motivons l'intérêt de considérer les modèles couplés. Après avoir donné des éléments de la théorie des RKHS pour des fonctions à valeurs vectorielles, nous posons les fondements de la famille de modèles OKVAR que nous avons introduite au cours de la thèse. L'apprentissage de ces modèles nous a conduit à nous intéresser à la minimisation de fonctions de coût qui font intervenir plusieurs schémas de régularisation parcimonieuse. L'estimation des paramètres des modèles est alors réalisée via l'élaboration d'algorithmes proximaux dédiés. Dans l'optique de la grande dimension, s'ensuit la présentation de la méthode d'ensemble OKVAR-Boost. Nous terminons ce chapitre en donnant quelques résultats en prédiction de plusieurs modèles de la famille OKVAR et d'OKVAR-Boost.

— Le chapitre 5 est consacré au problème de l'inférence de réseaux. Pour cela, nous introduisons plusieurs estimateurs de la matrice jacobienne et montrons comment des *a priori* sur la structure sous-jacente guident le choix du modèle OKVAR considéré. Ce noyau à valeur matricielle possède un hyperparamètre qui encode les indépendances entre les variables d'état. Cet hyperparamètre doit être appris en sus des paramètres du modèle. Nous avons alors proposé deux stratégies. La première solution est la mise en œuvre d'un schéma d'optimisation alterné qui permet d'apprendre conjointement les deux types de paramètres. Dans le cas d'OKVAR-Boost, nous considérons un apprentissage découplé des deux paramètres : l'hyperparamètre est estimé via un test statistique d'indépendance puis on estime les paramètres du modèle.

  Enfin, nous évaluons les performances de la famille de modèles OKVAR sur deux jeux de données : des bases de données *in silico* de réseaux de régulation issues de la compétition DREAM3 et des données réelles du climat.

Le document se clôt par une conclusion suivie de perspectives.

Première partie

État de l'art

Chapitre 1

Précis introductif à l'apprentissage statistique

Sommaire

1.1 Introduction 7
1.2 Apprentissage supervisé 8
1.2.1 Formalisme 9
1.2.2 Minimisation du risque empirique 9
1.2.3 Biais, Variance, Complexité, Régularisation 10
1.2.4 Mise en place d'une procédure d'apprentissage 12
1.3 Apprentissage non supervisé 12
1.4 Modèles classiques 13
1.4.1 Régression linéaire 13
1.4.2 Méthodes à noyaux 15
1.5 Méthodes d'ensemble 20
1.5.1 Bagging 21
1.5.2 Boosting 21
1.5.3 Forêts aléatoires 22
1.6 Synthèse 23

1.1 Introduction

Notre travail s'inscrit dans le cadre de l'apprentissage statistique. Ce chapitre a pour but d'introduire les notions et concepts clés de la théorie de l'apprentissage statistique que nous sommes amenés à utiliser le long du mémoire. Après avoir présenté les grandes familles de problèmes rencontrés en apprentissage statistique (supervisé, non supervisé), nous abordons les stratégies d'évaluation d'une méthode d'apprentissage ainsi que la problématique de sélection de modèle. Nous présentons ensuite plusieurs modèles de référence puis nous terminons sur les méthodes d'ensemble.

L'apprentissage statistique peut être vu comme une formalisation du domaine du *Machine Learning*, terme anglais consacré, traduit en français tantôt par apprentissage automatique tantôt par apprentissage artificiel selon la sensibilité des auteurs. Mitchell (1997) définit le concept d'apprentissage de la manière suivante : « On dit qu'un programme apprend d'une expérience `E` pour une classe de tâches donnée `T` et pour une mesure de performance donnée `P` si sa performance à accomplir les tâches dans `T`, mesurée par `P`, s'améliore avec l'expérience `E` ». Cette définition suggère l'idée qu'un algorithme d'apprentissage est capable d'inférer un comportement, des règles à partir d'exemples qui lui sont fournis et de les généraliser à des situations nouvelles.

Par le vocable d'apprentissage statistique, on désigne à la fois la conception de méthodes pour décrire, analyser et prédire des phénomènes (partie *algorithmique*) mais aussi l'étude théorique des garanties (e.g. consistance, bornes sur l'erreur de généralisation) qu'offrent les algorithmes ainsi produits (partie *statistique*). Les applications de l'apprentissage automatique sont nombreuses et touchent des domaines aussi variés que le traitement du signal et des images, la reconnaissance de formes, la recherche d'information, la bio-informatique ou la finance, entre autres. On distingue généralement deux types de problèmes : apprentissage supervisé ou non supervisé.

1.2 Apprentissage supervisé

En apprentissage supervisé, l'on cherche à attribuer une étiquette à une observation décrite par un ensemble de caractéristiques à partir d'une base finie d'observations étiquetées. Prenons deux exemples :

1. Les boîtes de mails proposent un service qui permet de filtrer les spams. Un mail peut être représenté par le nombre de mots qu'il contient, des valeurs booléennes qui indiquent la présence de certains mots-clés,.... L'étiquette associée pour un mail est alors `spam` ou `non spam` ; au lieu d'une chaîne de caractères, l'on préfère utiliser une étiquette numérique, respectivement +1 ou -1.

2. Une autre illustration de tâche supervisée est la prédiction du prix de vente (*étiquette*) d'un bien immobilier, connaissant sa surface, son âge, son nombre de pièces (*caractéristiques*).

Il est alors courant de distinguer les deux types de tâches selon la plage de valeurs que peuvent prendre les étiquettes, si ces dernières sont continues (réelles) comme dans la deuxième tâche, on parle de *régression*, tandis que si elles sont discrètes comme dans la première tâche, on parle de *classification*.

1.2.1 Formalisme

On note $\mathcal{X}$ le domaine des observations ou entrées $\mathbf{x}$. Les étiquettes, encore appelées réponses ou sorties y associées aux entrées prennent leurs valeurs dans un ensemble noté $\mathcal{Y}$. On rappelle la distinction de cas : tâche de régression si $\mathcal{Y} \subseteq \mathbb{R}$, tâche de classification lorsque $\mathcal{Y}$ est discret, c'est-à-dire isomorphe à une partie finie de $\mathbb{N}$. Les couples $(\mathbf{x}, y)$ sont des tirages indépendants et identiquement distribués (i.i.d.) d'une loi de probabilité conjointe $P(X, Y)$ caractérisant le phénomène étudié où X désigne un vecteur aléatoire associé aux observations et Y est la variable aléatoire correspondant aux réponses. La distribution $P(X, Y)$ est fixe mais *inconnue*.

Définition 1.1. *Un ensemble d'apprentissage ou base d'entraînement $\mathcal{S}_N$ est un ensemble fini d'exemples étiquetés (couples entrées/sorties) $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$ i.i.d. échantillonnés selon la distribution jointe $P(X, Y)$.*

Le processus d'apprentissage, tel que le décrit V. N. Vapnik (1995), repose sur les éléments suivants :

1. un générateur aléatoire d'observations $\mathbf{x} \in \mathcal{X}$ tirées i.i.d. d'une loi $P(X)$ fixe mais inconnue ;
2. un superviseur ou oracle qui retourne une sortie $y \in \mathcal{Y}$ pour chaque entrée $\mathbf{x}$ selon une distribution $P(Y|X)$ qui est également fixe mais inconnue ;
3. un programme ou une machine capable d'implémenter un ensemble de fonctions h, c'est-à-dire que nous définissons un espace d'hypothèses $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$.

Le problème de l'apprentissage consiste à sélectionner l'hypothèse $h \in \mathcal{H}$ qui approxime au mieux les réponses de l'oracle. Ce choix doit être guidé par les informations contenues dans l'ensemble d'apprentissage $\mathcal{S}_N$. La formulation proposée par Vapnik amène à attaquer le problème de l'apprentissage sous l'angle de l'approximation de fonction.

1.2.2 Minimisation du risque empirique

Il s'agit de traduire en termes mathématiques ce que signifie « hypothèse $h \in \mathcal{H}$ qui approxime au mieux les réponses de l'oracle ». On mesure l'inadéquation d'une hypothèse h aux données observées via une *fonction de coût* ou *fonction de perte* ℓ : $\mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$. Ainsi, $\ell(y, h(\mathbf{x}))$ quantifie l'écart entre la sortie y associée à une observation $\mathbf{x}$ et la prédiction $h(\mathbf{x})$ fournie par l'hypothèse h.

Définition 1.2. *Le risque réel ou risque théorique d'une hypothèse h est l'espérance de la fonction de perte ℓ :*

$$R(h) = \mathbb{E}\left[\ell(Y, h(X))\right] = \int_{\mathcal{X} \times \mathcal{Y}} \ell(y, h(\mathbf{x})) dP(\mathbf{x}, y)$$

L'objectif est d'identifier une hypothèse $h^\star$ qui minimise le risque réel dans l'espace d'hypothèses $\mathcal{H}$:

$$h^\star \in \underset{h\in\mathcal{H}}{\operatorname{argmin}} R(h)$$

Cependant, avant même d'aborder une quelconque stratégie d'optimisation, on remarque que le risque réel ne peut être calculé puisque la loi jointe $P(X,Y) = P(Y|X)P(X)$ est inconnue. Étant donné que la seule information disponible sur cette distribution est contenue dans l'ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, on substitue, dans l'expression du risque réel, à la mesure $dP(\mathbf{x}, y)$ inconnue la mesure de probabilité empirique $dP_{emp}(\mathbf{x}, y) = \frac{1}{N}\sum_{i=1}^N \delta_{\mathbf{x}_i}(\mathbf{x})\delta_{y_i}(y)$ où δ est le delta de Kronecker. On obtient alors le *risque empirique* défini par :

$$R_N(h) = \frac{1}{N}\sum_{i=1}^{N} \ell(y_i, h(\mathbf{x}_i))$$

Un processus d'apprentissage fondé sur le principe de minimisation du risque empirique cherche à trouver une hypothèse $\hat{h}_N \in \mathcal{H}$ qui minimise $R_N(h)$:

$$\hat{h}_N \in \underset{h\in\mathcal{H}}{\operatorname{argmin}} R_N(h)$$

Vapnik démontre la pertinence de ce principe en définissant la notion de consistance.

Définition 1.3. *Une méthode d'apprentissage est consistante si le risque réel de l'hypothèse $\hat{h}_N$, solution du problème de minimisation du risque empirique, converge vers la valeur minimale du risque réel dans l'espace d'hypothèses $\mathcal{H}$:*

$$R\left(\hat{h}_N\right) \xrightarrow[N\to+\infty]{} \inf_{h\in\mathcal{H}} R(h) = R(h^\star)$$

1.2.3 Biais, Variance, Complexité, Régularisation

Même si l'on peut garantir la consistance d'un processus d'apprentissage pour un échantillon de taille infinie, en pratique l'on dispose d'un unique ensemble d'apprentissage $\mathcal{S}_N$ fini de N données. Après avoir obtenu un estimateur $\hat{h}_N$ à partir de $\mathcal{S}_N$, l'approche fréquentiste pose la question de l'incertitude de l'estimation. Illustrons cette problématique dans le cas de la régression : on suppose qu'il existe une relation entre les variables d'entrée $\mathbf{x}$ et la variable de sortie y et que les observations disponibles sont entachées d'un bruit ϵ, variable aléatoire gaussienne centrée de variance σ^2. Le modèle de régression s'écrit alors :

$$y = h^\star(\mathbf{x}) + \epsilon$$

En considérant la fonction de perte quadratique, l'on sait que la solution optimale au sens du risque théorique pour une observation $\mathbf{x}$ est donnée par la fonction de régression $h^\star(\mathbf{x}) = \mathbb{E}[Y|\mathbf{x}]$. Si l'on suppose qu'on dispose de plusieurs ensembles d'apprentissage $\mathcal{S}_N^{(i)}$ de taille N, tous tirés i.i.d. de la loi $P(X,Y)$, un algorithme d'apprentissage produit pour chaque ensemble $\mathcal{S}_N^{(i)}$ une hypothèse $\hat{h}(\cdot;\mathcal{S}_N^{(i)})$. L'on peut alors démontrer que le risque réel d'une hypothèse apprise $\hat{h}$ peut se décomposer de la manière suivante :

$$\begin{aligned} R\left(\hat{h}\right) &= \mathbb{E}_{\mathcal{S}_N}\left[\left(Y - \hat{h}(X;\mathcal{S}_N)\right)^2\right] \\ &= \int_{\mathcal{X}} \left\{\mathbb{E}_{\mathcal{S}_N}\left[\hat{h}(\mathbf{x};\mathcal{S}_N)\right] - h^\star(\mathbf{x})\right\}^2 dP(\mathbf{x}) \\ &\quad + \int_{\mathcal{X}} \mathbb{E}_{\mathcal{S}_N}\left[\left(\hat{h}(\mathbf{x};\mathcal{S}_N) - \mathbb{E}_{\mathcal{S}_N}\left[\hat{h}(\mathbf{x};\mathcal{S}_N)\right]\right)^2\right] dP(\mathbf{x}) \\ &\quad + \int_{\mathcal{X}\times\mathcal{Y}} \left(h^\star(\mathbf{x}) - y\right)^2 dP(\mathbf{x},y) \end{aligned}$$

Le premier terme correspond au *biais* de l'estimateur et quantifie l'écart entre la prédiction moyenne des hypothèses $\hat{h}(\cdot;\mathcal{S}_N^{(i)})$ et la valeur donnée par la fonction de régression $h^\star$. Le second terme fait référence à la *variance* de l'estimateur et mesure à quel point chaque modèle appris $\hat{h}(\cdot;\mathcal{S}_N^{(i)})$ diffère de la moyenne de ces modèles. La variance constitue donc un indicateur de la sensibilité de l'algorithme d'apprentissage au choix particulier des données d'apprentissage. Le dernier terme est égal à σ^2 la variance du bruit intrinsèque à l'observation des données. Cette quantité indépendante du modèle appris représente une borne inférieure pour le risque réel.

Cette décomposition du risque apporte un nouvel éclairage sur la question de la minimisation du risque. L'on peut obtenir (1) soit des modèles trop flexibles pouvant atteindre un risque empirique faible voire nul (biais faible) mais assujettis à une forte variance (2) soit des modèles plus simples présentant une faible variance mais donnant des prédictions relativement éloignées des valeurs des observations (biais élevé). La notion de *complexité* d'un modèle découle du compromis entre biais et variance. Vapnik quantifie la complexité d'un modèle en donnant une mesure de la « taille » de l'espace d'hypothèses $\mathcal{H}$ auquel il appartient, mesure appelée dimension de Vapnik-Chervonenkis (dimension VC). Le principe de minimisation du risque *structurel* qu'il propose consiste alors à définir une structure d'espaces emboîtés pour laquelle la complexité est une fonction croissante de la dimension VC et le risque empirique est une fonction décroissante de la dimension VC.

Dans le présent mémoire, nous prenons le parti d'attaquer le problème de la complexité via le paradigme de la régularisation (Tikhonov et Arsenin, 1977). La minimisation du risque empirique peut constituer un problème mal posé (non existence, non unicité d'une solution ou bien cas d'un système d'équations linéaires sous-déterminé ou sur-déterminé).

Il s'agit alors de transformer un tel problème en imposant des propriétés *a priori* sur le modèle qui prennent la forme d'un terme de régularisation qui s'ajoute au risque empirique. Le problème de l'apprentissage devient alors :

$$\underset{h \in \mathcal{H}}{\operatorname{argmin}} R_N(h) + \lambda \Omega(h) \tag{1.1}$$

où Ω est une fonction qui pénalise la complexité du modèle et l'hyperparamètre $\lambda \geq 0$ contrôle la force de la pénalité. La résolution du problème (1.1) pour une valeur de λ élevée aboutit à un modèle de complexité faible (biais élevé et variance faible) et vice-versa.

1.2.4 Mise en place d'une procédure d'apprentissage

Un praticien de l'apprentissage se pose généralement les deux questions suivantes :

— Étant donné un échantillon fini d'exemples $\mathcal{S}_N$, comment peut-on évaluer la capacité d'un modèle $\hat{h}_N$ appris à partir de $\mathcal{S}_N$ à généraliser à de nouveaux exemples ?

— Comment choisir le ou les hyperparamètres d'un algorithme d'apprentissage ?

La première question est celle de l'estimation du risque réel $R\left(\hat{h}_N\right)$ et la deuxième fait référence au problème de la sélection de modèle. Une mise en place rigoureuse d'un processus d'apprentissage nécessite une partition de l'ensemble $\mathcal{S}_N$ en trois sous-ensembles :

— un ensemble d'apprentissage utilisé pour entraîner des modèles $\hat{h}$

— un ensemble de validation dédié à la sélection de modèle, c'est-à-dire au choix des meilleurs hyperparamètres. On obtient alors une hypothèse $\hat{h}_{\Lambda^*}$

— un ensemble de test à partir duquel on calcule le risque du modèle sélectionné $R_{test}\left(\hat{h}_{\Lambda^*}\right)$

Lorsque les données disponibles ne sont pas suffisamment nombreuses, l'on recommande plutôt une procédure de validation croisée.

1.3 Apprentissage non supervisé

Contrairement à l'apprentissage supervisé, les exemples en apprentissage non supervisé, sont dépourvus d'étiquettes. En l'absence de « vérité », la notion de risque utilisée en apprentissage supervisé n'a plus cours. Il s'agit dans ce cas de déterminer quelques traits saillants ou encore de dégager une structure sous-jacente aux exemples. Parmi les tâches en apprentissage non supervisé, on trouve notamment :

— la classification non supervisée ou clustering qui consiste à regrouper dans des groupes ou clusters des objets de telle sorte que les objets au sein d'un même cluster sont plus semblables entre eux qu'à ceux appartenant à d'autres clusters

— la réduction de dimension. Quand les données sont de grande dimension, l'on peut

faire l'hypothèse que certaines des variables ou caractéristiques des données sont soit redondantes, soit non pertinentes, soit non informatives. Des techniques de sélection de variables ou d'extraction de nouvelles variables synthétiques présentent plusieurs avantages : elles améliorent l'interprétabilité des données, diminuent le temps de calcul des algorithmes d'apprentissage et augmentent la capacité de généralisation en réduisant le risque de surapprentissage.

Dans le présent mémoire, l'on ne s'étendra pas sur cette classe de modèles et nous nous focalisons sur des problèmes *supervisés*.

1.4 Modèles classiques

Dans cette section, nous présentons les modèles les plus communément utilisés en apprentissage supervisé. Plus précisément, nous introduisons les grandes classes de modèles par rapport auxquelles nous nous sommes positionnés et ce faisant, nous rappelons un certain nombre de concepts clés qui seront abondamment utilisés dans la présentation de nos contributions.

1.4.1 Régression linéaire

En régression, on cherche à modéliser la relation entre d variables explicatives $X_1, \ldots, X_d$ aussi appelées covariables ou prédicteurs et une variable réponse Y,

$$Y \quad = \quad h(X_1, \ldots, X_d) + \epsilon$$

où ϵ est une variable aléatoire associée à du bruit (erreurs de mesure,...). Dans le cas de la régression linéaire, l'espace d'hypothèses $\mathcal{H}$ est l'ensemble des combinaisons linéaires des d prédicteurs, i.e. h est de la forme :

$$h(X_1, \ldots, X_d) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_d X_d$$

où $\beta_0, \ldots, \beta_d \in \mathbb{R}$ sont les coefficients de la régression. Le processus d'apprentissage consiste à identifier le modèle $h \in \mathcal{H}$ qui minimise le risque empirique, ce qui revient à estimer les paramètres $\beta_0, \ldots, \beta_d$. Étant donné un ensemble d'apprentissage de N observations $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ avec $\mathbf{x}_i = (x_i^1 \ldots x_i^d)^T \in \mathbb{R}^d$, le modèle de régression linéaire peut également s'écrire sous forme matricielle :

$$\mathbf{y} = X\mathbf{b} + \mathbf{e}$$

où $\mathbf{y} = (y_1 \dots y_N)^T \in \mathbb{R}^N$, $\mathbf{b} = (\beta_0 \dots \beta_d)^T \in \mathbb{R}^{d+1}$, $\mathbf{e} = (\epsilon_1 \dots \epsilon_N)^T \in \mathbb{R}^N$ et $X = \begin{pmatrix} 1 & x_1^1 & \cdots & x_1^d \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_N^1 & \cdots & x_N^d \end{pmatrix} \in \mathbb{R}^{N \times (d+1)}$. Le risque empirique, pour une fonction de perte quadratique, devient :

$$\begin{aligned} R_N(h) &= \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} 1 \\ \mathbf{x}_i \end{pmatrix}^T \mathbf{b} \right)^2 \\ &= \frac{1}{N} \|\mathbf{y} - X\mathbf{b}\|_2^2 \end{aligned}$$

$R_N(h)$ est, en l'occurrence, une fonction convexe et différentiable en $\mathbf{b}$. Les conditions du premier ordre permettent alors de trouver un minimiseur (cf. Section 2.2) :

$$\begin{aligned} \nabla_{\mathbf{b}} \left\{ \frac{1}{N} (\mathbf{y} - X\mathbf{b})^T (\mathbf{y} - X\mathbf{b}) \right\} = 0 \quad &\Leftrightarrow \quad \frac{1}{N} 2 (-X)^T (\mathbf{y} - X\mathbf{b}) = 0 \\ &\Leftrightarrow \quad \left(X^T X \right) \mathbf{b} = X^T \mathbf{y} \end{aligned}$$

On obtient ainsi un système d'équations linéaires, dites équations normales. Si la matrice $X^T X \in \mathbb{R}^{(d+1)\times(d+1)}$ est inversible, alors le problème de minimisation du risque empirique admet une unique solution :

$$\hat{\mathbf{b}} = \left(X^T X \right)^{-1} X^T \mathbf{y}$$

$\hat{\mathbf{b}}$ est appelé estimateur des Moindres Carrés Ordinaires (MCO). Il n'y a plus unicité de la solution lorsque la matrice $X^T X$ est singulière : soit les variables explicatives ne sont pas linéairement indépendantes (certaines variables sont redondantes) soit le nombre d'observations N est inférieur au nombre de variables explicatives d, ce qui est le cas des problèmes en grande dimension. On peut alors envisager de régulariser le risque empirique en contrôlant la complexité du modèle, c'est-à-dire ici en pénalisant le carré de la norme ℓ_2 du paramètre $\mathbf{b}$. C'est le problème de la régression *ridge*. Le risque empirique régularisé à minimiser s'écrit alors :

$$R_N^\lambda(h) = \frac{1}{N} \|\mathbf{y} - X\mathbf{b}\|_2^2 + \lambda \|\mathbf{b}\|_2^2 \tag{1.2}$$

avec $\lambda > 0$. De la même façon, l'étude des conditions du premier ordre donne la solution suivante :

$$\hat{\mathbf{b}}^\lambda = \left(X^T X + N\lambda Id \right)^{-1} X^T \mathbf{y}$$

1.4.2 Méthodes à noyaux

Les modèles de régression linéaire sont très populaires car conceptuellement simples et faciles à mettre en œuvre. Toutefois, les modèles linéaires sont limités à cause de cette hypothèse de linéarité qui est peu vraisemblable dans la plupart des problèmes réels. Les méthodes à noyaux constituent une classe de modèles qui étendent *astucieusement* les méthodes linéaires au cas non linéaire. Elles ont prouvé leur efficacité dans une série de tâches et se sont révélées incontournables notamment grâce à leurs performances dans des domaines d'application tels que la biologie computationnelle (Bernhard Schölkopf et al., 2004) où elles font référence dans l'état de l'art. Les méthodes à noyaux sont basées sur la manipulation et l'évaluation de fonctions appelées *noyaux* et que nous définissons ici d'un point de vue fonctionnel.

Éléments de théorie des RKHS

Étant donné un ensemble $\mathcal{X}$, $\mathbb{R}^{\mathcal{X}}$ est l'ensemble des fonctions de $\mathcal{X}$ dans $\mathbb{R}$.

Définition 1.4. *$\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ est un* espace de Hilbert à noyau reproduisant *(en anglais Reproducing Kernel Hilbert Space, RKHS) si, et seulement si :*

1. *$\mathcal{H}$ est un sous-espace vectoriel de $\mathbb{R}^{\mathcal{X}}$*
2. *$\mathcal{H}$ est un espace de Hilbert, c'est-à-dire muni d'un produit scalaire $\langle \cdot, \cdot \rangle$ (préhilbertien) et toute suite de Cauchy sur $\mathcal{H}$ est convergente et sa limite est dans $\mathcal{H}$ (complet)*
3. *Pour tout $\mathbf{z} \in \mathcal{X}$, la fonction d'évaluation linéaire $\delta_{\mathbf{z}} : \mathcal{H} \to \mathbb{R}$, définie pour tout $h \in \mathcal{H}$ par $\delta_{\mathbf{z}}(h) = h(\mathbf{z})$, est bornée*

Les RKHS surviennent dans plusieurs domaines d'étude tels que la théorie de l'approximation, les statistiques ou encore la théorie de représentation de groupe. On trouve trace de la théorie des RKHS dans les travaux précurseurs de Aronszajn (1950).

Si $\mathcal{H}$ est un RKHS, alors pour tout $\mathbf{z} \in \mathcal{X}$, $\delta_{\mathbf{z}}$ est une forme linéaire bornée sur $\mathcal{H}$ donc continue. D'après le théorème de représentation de Riesz, il existe un unique élément $k_{\mathbf{z}} \in \mathcal{H}$ tel que :

$$\forall h \in \mathcal{H}, h(\mathbf{z}) = \langle h, k_{\mathbf{z}} \rangle$$

Définition 1.5. *La fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ définie par :*

$$\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, k(\mathbf{x}, \mathbf{z}) = k_{\mathbf{z}}(\mathbf{x})$$

est appelée noyau reproduisant associé à $\mathcal{H}$.

En particulier, notons qu'en prenant $h = k_{\mathbf{x}} = k(\cdot, \mathbf{x})$, on a la propriété de reproduction :

$$k(\mathbf{x}, \mathbf{z}) = \langle k(\cdot, \mathbf{x}), k(\cdot, \mathbf{z}) \rangle$$

On peut alors définir la fonction $\phi : \mathcal{X} \to \mathcal{H}$ définie par $\phi(\mathbf{x}) = k(\cdot, \mathbf{x})$. Cette fonction appelée en anglais *feature map* joue un rôle central, notamment dans la méthode des Séparateurs à Vaste Marge (SVM) détaillée dans la sous-section suivante.

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$$

Ce résultat fournit une interprétation sur les noyaux qui peuvent être vus comme des produits scalaires dans un espace des caractéristiques (*feature space*, en anglais). Ceci explique que l'on présente souvent les noyaux comme une mesure de similarité. Il s'agit là d'un changement de paradigme majeur. En effet, chaque objet $\mathbf{x} \in \mathcal{X}$ est désormais représenté par un vecteur $\phi(\mathbf{x}) \in \mathcal{H}$, l'astuce dite des noyaux (Aizerman et al., 1964) consiste à ne pas calculer explicitement $\phi(\mathbf{x})$, $\mathcal{H}$ pouvant être de dimension infinie, mais plutôt à calculer uniquement des produits scalaires dans l'espace des caractéristiques, ce qui revient simplement à évaluer des noyaux. Par la facilité de sa mise en œuvre, l'astuce des noyaux a permis de transformer une série de méthodes linéaires (régression linéaire, analyse en composantes principales,...) en des méthodes non linéaires en remplaçant le produit scalaire dans l'espace de départ par des noyaux non linéaires et ce pour un coût algorithmique nul.

Définition 1.6. *Soit une fonction $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. s est une fonction semi-définie positive si pour tout $n \in \mathbb{N}$ et pour tout $\{(\mathbf{x}_i, \alpha_i)\}_{i=1}^n \subseteq \mathcal{X} \times \mathbb{R}$, on a :*

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j s(\mathbf{x}_i, \mathbf{x}_j) \geq 0$$

ou encore que la matrice $(s(\mathbf{x}_i, \mathbf{x}_j))_{1 \leq i,j \leq n}$ est semi-définie positive.

Proposition 1.1. *Soit $\mathcal{H}$ un RKHS sur $\mathbb{R}^{\mathcal{X}}$ qui admet k pour noyau reproduisant. Alors k est une fonction semi-définie positive.*

Théorème 1.1 (Théorème de Moore-Aronszajn)**.** *Soit une fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Si k est une fonction semi-définie positive, alors il existe un unique RKHS $\mathcal{H}$ sur $\mathbb{R}^{\mathcal{X}}$ tel que k est le noyau reproduisant associé à $\mathcal{H}$.*

Voici quelques exemples de noyaux couramment utilisés dans la littérature :

— le noyau linéaire défini par $k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}$, il s'agit simplement du produit scalaire usuel dans l'espace d'entrée

— le noyau polynomial : $k(\mathbf{x}, \mathbf{z}) = \left(\mathbf{x}^T \mathbf{z} + c\right)^p$ où $p \geq 1$ est un entier et $c \in \mathbb{R}$ une constante

— le noyau gaussien : $k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|_2^2\right)$ avec $\gamma > 0$

Le noyau gaussien fait partie d'une classe particulière de noyaux aux propriétés intéressantes, les noyaux *universels*.

Définition 1.7. *(*Steinwart*, 2002) Soit un espace métrique compact $(\mathcal{X}, \Delta)$, $C(\mathcal{X})$ désigne l'ensemble des fonctions continues sur $\mathcal{X}$. Un noyau k est dit* universel *si le RKHS $\mathcal{H}_k$ associé à k est dense dans $C(\mathcal{X})$, i.e. pour tout $f \in C(\mathcal{X})$ et pour tout $\epsilon > 0$, il existe une fonction $g \in \mathcal{H}_k$ telle que $\|f - g\|_\infty \leq \epsilon$*

Steinwart prouve notamment que les noyaux RBF (Radial Basis Function) ci-après sont universels.

Théorème 1.2. *Les noyaux* $\exp\left(-\gamma \|\cdot - \cdot\|_2^2\right)$ *et* $\exp\left(-\gamma \|\cdot - \cdot\|_2\right)$ *avec* $\gamma > 0$ *sont universels sur toute partie compacte de* $\mathbb{R}^d$.

Propriétés fonctionnelles. $\mathcal{H}$ étant un sous-espace vectoriel de $\mathbb{R}^{\mathcal{X}}$, la stabilité par combinaison linéaire permet de définir $\mathcal{H}$ comme l'ensemble des fonctions $h : \mathcal{X} \to \mathbb{R}$ de la forme :

$$h = \sum_{i=1}^{m} \alpha_i k(\cdot, \mathbf{x}_i)$$

où $m > 0, \mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathcal{X}$ et $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$ auxquelles on ajoute les limites ponctuelles de suites de Cauchy. Par la propriété de reproduction, le produit scalaire de deux éléments de $\mathcal{H}$, $g = \sum_{j=1}^{n} \beta_j k(\cdot, \mathbf{z}_j)$ et $h = \sum_{i=1}^{m} \alpha_i k(\cdot, \mathbf{x}_i)$ s'exprime comme suit :

$$\langle g, h \rangle = \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \beta_j k(\mathbf{x}_i, \mathbf{z}_j)$$

En particulier, la norme RKHS de h associée à $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ est donnée par :

$$\|h\|_{\mathcal{H}}^2 = \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)$$

La norme RKHS d'un modèle h est un indicateur de la complexité, dans le sens où une fonction est d'autant plus lisse (*smooth* en anglais) que sa norme RKHS est faible. Il est important de noter que la notion de régularité ou de « lissage » dépend du noyau utilisé. C'est un *a priori* sur les propriétés fonctionnelles désirées qui guide l'utilisateur dans le choix ou la conception d'un noyau.

Les méthodes à noyaux que nous étudions aboutissent généralement à un problème de minimisation d'un risque empirique régularisé de la forme :

$$\operatorname*{argmin}_{h \in \mathcal{H}} R_N(h) + \lambda \|h\|_{\mathcal{H}}^2$$

avec $\lambda > 0$. La résolution de ce problème est grandement facilitée par un théorème appelé *théorème du représentant*.

Théorème 1.3. *(Kimeldorf et Wahba, 1971 ; Schölkopf et al., 2001) Soient un RKHS $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ sur $\mathbb{R}^{\mathcal{X}}$ qui admet k pour noyau reproduisant, $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subseteq \mathcal{X}$ et $\Psi : \mathbb{R}^{m+1} \to \mathbb{R}$ une fonction strictement croissante en son dernier argument. Alors toute solution du problème*

$$\underset{h\in\mathcal{H}}{\operatorname{argmin}}\, \Psi\left(h(\mathbf{x}_1), \ldots, h(\mathbf{x}_m), \|h\|_{\mathcal{H}}\right) \tag{1.3}$$

admet une représentation de la forme :

$$\hat{h} = \sum_{i=1}^{m} \alpha_i k(\cdot, \mathbf{x}_i)$$

avec $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$

Le théorème du représentant induit un avantage calculatoire substantiel. En effet, toute solution de (1.3) appartient à un sous-espace de $\mathcal{H}$ de dimension au plus m, le nombre d'observations alors même que le problème d'optimisation porte sur l'espace $\mathcal{H}$ qui est possiblement de dimension infinie. En pratique, résoudre (1.3) se réduit à un problème dans $\mathbb{R}^m$ en estimant les m paramètres $\alpha_1, \ldots, \alpha_m$.

Séparateurs à Vaste Marge

Les Séparateurs à Vaste Marge (Support Vector Machines en anglais) ou SVM (Boser et al., 1992 ; Cortes et V. Vapnik, 1995) sont une méthode à noyaux dédiée à la classification supervisée. Nous nous restreignons au cas binaire : $\mathcal{Y} = \{+1, -1\}$ et supposons que l'on dispose d'un ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$ avec $\mathcal{X} \subseteq \mathbb{R}^d$. Si les observations sont linéairement séparables, la frontière de décision recherchée peut prendre la forme d'un hyperplan H dont une équation est $h(\mathbf{x}) = \omega^T\mathbf{x} + b = 0$ où $\omega \in \mathbb{R}^d$ et $b \in \mathbb{R}$. La règle de décision est donnée par la fonction : $f(\mathbf{x}) = \text{signe}\left(\omega^T\mathbf{x} + b\right)$. Ainsi un exemple est bien classé si $y_i\left(\omega^T\mathbf{x}_i + b\right) \geq 0$. Le principe de minimisation du risque empirique, fondé sur la minimisation du nombre d'exemples mal classés ou erreur de classification, ne permet pas de déterminer un unique couple de paramètres (ω, b) à partir de $\mathcal{S}_N$. L'algorithme des SVM consiste alors à choisir les paramètres qui maximisent un critère géométrique : l'hyperplan séparateur optimal est celui qui crée la plus grande *marge* entre les observations de la classe positive ($y_i = +1$) et celles de la classe négative ($y_i = -1$). Cette marge vaut en l'occurrence $2/\|\omega\|_2$.

Dans le cas où les observations ne sont pas séparables, on peut conserver l'objectif de maximiser la marge mais en tolérant que certains exemples soient mal classés, l'on introduit

alors N variables d'écart positives $\xi_1, \ldots, \xi_N$. $\xi_i > 0$ signifie que la i-ème observation est mal classée de sorte que $\sum_{i=1}^N \xi_i$ est une estimation du nombre total d'erreurs. S'ensuit le problème d'optimisation :

$$\begin{array}{ll} \underset{\omega,b,\xi_1,\ldots,\xi_N}{\operatorname{argmin}} & \frac{1}{2}\|\omega\|_2^2 + C\sum_{i=1}^N \xi_i \\ \text{s.c.} & \left\{ \begin{array}{ll} y_i\left(\omega^T \mathbf{x}_i + b\right) \geq 1 - \xi_i, & i \in \mathbb{N}_N \\ \xi_i \geq 0 & i \in \mathbb{N}_N \end{array} \right. \end{array} \tag{1.4}$$

où $C > 0$ est un hyperparamètre qui réalise le compromis entre taille de la marge et nombre d'exemples mal classés.

On peut généraliser le problème des SVM au cas non linéaire en plongeant les données dans un espace de caractéristiques $\mathcal{H}$ de plus grande dimension via une fonction $\phi : \mathcal{X} \to \mathcal{H}$. L'hyperplan recherché dans $\mathcal{H}$ admet alors une équation de la forme $h(\mathbf{x}) = \omega^T \phi(\mathbf{x}) + b = 0$. Nous proposons ici de transformer la formulation du problème sous contraintes (1.4) en un problème équivalent sans contraintes :

$$\underset{h\in\mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^N \max(0, 1 - y_i h(\mathbf{x}_i)) + \lambda \|h\|_{\mathcal{H}}^2 \tag{1.5}$$

où $\lambda > 0$ est un hyperparamètre qui joue un rôle inverse à C. Cette formulation fait apparaître la fonction de coût charnière et justifie l'application du théorème du représentant. Le coût charnière n'étant pas différentiable, l'approche classique consiste à calculer le Lagrangien associé au problème (1.4), à dériver le problème dual et à utiliser les conditions de Karush-Kühn-Tucker. Néanmoins, l'on peut noter le travail de Chapelle (2007) qui propose de résoudre directement dans le primal le problème (1.5).

Kernel Ridge Regression

Nous avons présenté dans la section précédente la régression ridge dans le cas linéaire (cf. équation (1.2)). La transposition du problème dans le cas non linéaire se fait via l'utilisation d'un noyau k sur $\mathcal{X}$. Soit un ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathbb{R}$, en notant $\mathcal{H}_k$ le RKHS sur $\mathbb{R}^{\mathcal{X}}$ associé à k, le problème de la Kernel Ridge Regression consiste à apprendre une hypothèse $h \in \mathcal{H}_k$ qui minimise le coût régularisé suivant :

$$\sum_{i=1}^N (y_i - h(\mathbf{x}_i))^2 + \lambda \|h\|_{\mathcal{H}_k}^2 \tag{1.6}$$

avec $\lambda > 0$. Le théorème du représentant s'applique et un minimiseur de (1.6) admet une expression de la forme :

$$\hat{h} = \sum_{i=1}^{N} \alpha_i k(\cdot, \mathbf{x}_i)$$

En injectant l'expression de $\hat{h}$ dans (1.6), le problème devient :

$$\underset{\boldsymbol{\alpha} \in \mathbb{R}^N}{\operatorname{argmin}} \mathcal{J}(\boldsymbol{\alpha}) = \|\mathbf{y} - \mathbf{K}\boldsymbol{\alpha}\|_2^2 + \lambda \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \tag{1.7}$$

où $\mathbf{y} = (y_1 \dots y_N)^T \in \mathbb{R}^N, \boldsymbol{\alpha} = (\alpha_1 \dots \alpha_N)^T \in \mathbb{R}^N$ et $\mathbf{K} = (k(\mathbf{x}_i, \mathbf{x}_j))_{1 \leq i,j \leq N}$ est la matrice de Gram associée aux entrées $\mathbf{x}_1, \dots, \mathbf{x}_N$. Étudions les conditions du premier ordre :

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} \mathcal{J}(\boldsymbol{\alpha}) = 0 \quad &\Leftrightarrow \quad -2\mathbf{K}^T(\mathbf{y} - \mathbf{K}\boldsymbol{\alpha}) + 2\lambda \mathbf{K}\boldsymbol{\alpha} = 0 \\ &\Leftrightarrow \quad \mathbf{K}\left[(\mathbf{K} + \lambda Id)\boldsymbol{\alpha} - \mathbf{y}\right] = 0 \end{aligned}$$

$\mathbf{K}$ étant une matrice symétrique, elle est diagonalisable dans une base orthonormale avec $\ker(\mathbf{K}) \perp \operatorname{Im}(\mathbf{K})$. Dans cette base, on remarque que $(\mathbf{K} + \lambda Id)^{-1}$ laisse $\ker(\mathbf{K})$ et $\operatorname{Im}(\mathbf{K})$ invariants. Le problème (1.7) devient alors équivalent à :

$$\begin{aligned} & (\mathbf{K} + \lambda Id)\boldsymbol{\alpha} - \mathbf{y} \in \ker(K) \\ \Leftrightarrow \quad & \boldsymbol{\alpha} - (\mathbf{K} + \lambda Id)^{-1}\mathbf{y} \in \ker(\mathbf{K}) \\ \Leftrightarrow \quad & \boldsymbol{\alpha} = (\mathbf{K} + \lambda Id)^{-1}\mathbf{y} + \epsilon \text{ avec } \mathbf{K}\epsilon = 0. \end{aligned}$$

Soit $\hat{h}' = \sum_{i=1}^{N} \alpha_i' k(\cdot, \mathbf{x}_i)$ avec $\boldsymbol{\alpha}' = (\alpha_1' \dots \alpha_N')^T \in \mathbb{R}^N$ tel que $\boldsymbol{\alpha}' = \boldsymbol{\alpha} + \epsilon$ et $\mathbf{K}\epsilon = 0$, alors :

$$\|\hat{h} - \hat{h}'\|_{\mathcal{H}_k}^2 = (\boldsymbol{\alpha} - \boldsymbol{\alpha}')^T \mathbf{K} (\boldsymbol{\alpha} - \boldsymbol{\alpha}') = 0 \,.$$

Par conséquent, $\hat{h} = \hat{h}'$. On obtient alors la solution suivante :

$$\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda Id)^{-1}\mathbf{y} \,.$$

1.5 Méthodes d'ensemble

Les méthodes d'ensemble reposent sur un paradigme en apprentissage qui consiste à construire un ensemble d'hypothèses à partir de variations d'un ensemble d'apprentissage donné puis à combiner leurs prédictions. L'idée sous-jacente est qu'un ensemble peut exhiber des performances supérieures à n'importe quelle hypothèse individuelle qui compose l'ensemble. Hansen et Salamon (1990) montrent dans le cas de la classification qu'une condition nécessaire et suffisante pour qu'un ensemble de classifieurs soit plus précis que

n'importe lequel d'entre eux est que chacun des classifieurs soit précis (erreur de classification $R < 0.5$) et que ces classifieurs soient suffisamment « différents » les uns des autres. Nous présentons ici trois techniques d'ensemble qui mettent en œuvre cette condition : le bagging, le boosting et la forêt aléatoire.

1.5.1 Bagging

Le *Bagging* ou *Bootstrap Aggregating* (Leo Breiman, 1996) consiste à entraîner un algorithme d'apprentissage sur différents sous-ensembles de l'ensemble d'apprentissage initial $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. En pratique, on génère B sous-ensembles $\mathcal{S}_N^{(b)}$ ($b \in \mathbb{N}_B$) en tirant pour chacun d'eux aléatoirement N fois avec remise dans $\mathcal{S}_N$. Les sous-ensembles $\mathcal{S}_N^{(b)}$ sont appelés échantillons bootstraps. Le tirage avec remise implique que certaines observations peuvent se trouver en plusieurs exemplaires au sein d'un même échantillon bootstrap. Pour $b \in \mathbb{N}_B$, un modèle $\hat{h}_N^{(b)}$ est alors entraîné sur l'échantillon $\mathcal{S}_N^{(b)}$ et on obtient ainsi un ensemble de B modèles. Pour une nouvelle entrée $\mathbf{x}$, les prédictions de chaque hypothèse $\hat{h}_N^{(b)}(\mathbf{x})$ sont combinées en moyennant leurs sorties (en régression) ou en procédant à un vote majoritaire (en classification). Le bagging s'avère particulièrement efficace pour des algorithmes d'apprentissage « instables » ; sont déclarés de cette nature les algorithmes produisant des hypothèses très sensibles à toute modification même mineure de l'ensemble d'apprentissage. Arbres de décision et réseaux de neurones, entre autres, répondent à cette acception.

1.5.2 Boosting

Le boosting regroupe un ensemble d'algorithmes. Nous présentons les grandes lignes du plus connu d'entre eux AdaBoost (Freund et R. Schapire, 1995), dédié initialement à la classification binaire ($\mathcal{Y} = \{+1, -1\}$). Ses auteurs partent du constat que construire un classifieur ayant une haute précision, appelé *apprenant fort* s'avère une tâche ardue tandis qu'il peut être bien plus aisé de concevoir des hypothèses, appelées *apprenants faibles*, qui affichent une précision modérée, c'est-à-dire une erreur de classification légèrement inférieure à 0.5. À titre d'exemple, si l'on considère le problème de la détection de spams, soit la règle : « Si le mot `achetez` apparaît dans un mail, alors prédire `spam` », clairement, cette règle à elle seule ne permet pas de traiter tous les types de messages, mais donne des prédictions significativement meilleures que le pur hasard.

AdaBoost applique un algorithme d'apprentissage M fois sur des données qui sont modifiées à chaque itération, produisant en conséquence un ensemble de M classifieurs faibles $\hat{h}_1, \ldots, \hat{h}_M$. Les prédictions de chacune de ces hypothèses sont combinées via un

vote majoritaire pondéré :

$$\hat{H}(\mathbf{x}) = \text{signe}\left(\sum_{m=1}^{M} \alpha_m \hat{h}_m(\mathbf{x})\right)$$

où $\alpha_1, \ldots, \alpha_M$ représentent le poids respectif attribué à chaque classifieur, un poids plus important étant accordé aux hypothèses les plus précises. À chaque étape de boosting, les données d'apprentissage se voient appliquer un poids $w_1, \ldots, w_N$. Les poids sont initialement tous égaux à $1/N$. À l'étape $m > 1$, les observations mal classées par l'hypothèse $\hat{h}_{m-1}$ voient leur poids augmenter alors que celui des observations bien classées diminue. Par conséquent, au fil des itérations, les observations qui n'ont pas été correctement classées par les classifieurs précédents dans l'ensemble reçoivent une attention croissante et AdaBoost encourage l'algorithme d'apprentissage à produire à l'étape ultérieure une hypothèse qui se concentre davantage sur ces observations particulièrement difficiles. Le succès d'AdaBoost a ouvert la voie à l'application du boosting à divers types de problèmes qui dépassent le cadre de la classification, notamment en théorie des jeux (Freund et R.E. Schapire, 1996), en approximation de fonction (J. Friedman, 2001) ou encore en programmation linéaire (Demiriz et al., 2002).

1.5.3 Forêts aléatoires

La *forêt aléatoire* (Leo Breiman, 2001) est une technique d'ensemble pour laquelle les apprenants faibles sont des *arbres de décision*. Ces derniers, aussi appelés arbres de classification et de régression (*Classification And Regression Trees*, CART) (L. Breiman et al., 1984) sont des modèles qui se fondent sur une partition hiérarchique de l'espace d'entrée, que l'on peut représenter par un arbre. À chaque nœud terminal ou *feuille* de cet arbre correspond un modèle local. La prédiction pour un point $\mathbf{x}$ est obtenue de la manière suivante : on commence à partir de la racine de l'arbre, les nœuds internes sont étiquetés par des « questions » (tests sur les caractéristiques) et les branches par les réponses possibles. Les nœuds parcourus par $\mathbf{x}$ jusqu'à atteindre une feuille dépendent des réponses données aux nœuds précédents. La prédiction pour $\mathbf{x}$ ne dépend que des données contenues dans la feuille f à laquelle appartient $\mathbf{x}$. La prédiction pour $\mathbf{x}$ est alors donnée par la moyenne des sorties appartenant à cette feuille :

$$h_f(\mathbf{x}) = \frac{1}{|f|} \sum_{(\mathbf{x}_i, y_i) \in f} y_i \tag{1.8}$$

Comme le bagging, la forêt aléatoire consiste en une collection d'arbres de décisions, chacun appris sur un échantillon bootstrap de l'ensemble d'apprentissage. Ce qui différencie la forêt aléatoire du bagging réside dans la manière dont chaque arbre est construit. Spéci-

fiquement, lors de la phase d'apprentissage d'un arbre de base, la caractéristique à utiliser pour segmenter à chaque nœud n'est plus choisie parmi toutes les variables mais seulement parmi un sous-ensemble *aléatoire* de $q \leq d$ variables. L'ajout de cette composante aléatoire se justifie par la similitude des arbres issus d'un échantillon bootstrap ordinaire. En effet, si une ou plusieurs caractéristiques pèsent substantiellement dans la variable à expliquer, les mêmes caractéristiques auront tendance à être sélectionnées par la plupart des arbres de base, empêchant *de facto* une variété au sein de l'ensemble. Selon la taille du sous-ensemble aléatoire q, l'on observe différents comportements :

- $q = d$: on revient au cas usuel d'un bagging d'arbres de décision
- $q = 1$: une variable est choisie aléatoirement à chaque nœud
- $1 < q < d$: en général, on préconise une taille de sous-ensemble très petite ($q \ll d$) afin d'encourager une diversité dans l'ensemble. Breiman suggère notamment trois valeurs possibles pour q : $\frac{1}{2}\sqrt{d}$, $\sqrt{d}$ et $2\sqrt{d}$

Les forêts aléatoires bénéficient de temps de calcul relativement faible et sont capables d'atteindre d'excellentes performances même en présence de données manquantes ou d'ensembles d'apprentissage déséquilibrés. Elles ont néanmoins une propension à sur-apprendre lorsque les données sont particulièrement bruitées.

1.6 Synthèse

Ce chapitre nous a permis de définir quelques concepts clés en apprentissage statistique auxquels nous faisons référence de manière récurrente dans le mémoire. Nous avons notamment présenté deux modèles de référence en apprentissage supervisé : les modèles de régression linéaire et les méthodes à noyaux. Pour améliorer la performance des modèles, nous avons vu que des modèles de base pouvaient être combinés via des méthodes d'ensemble. Dans le chapitre suivant, nous abordons les aspects qui ont trait à l'optimisation en présentant des méthodes qui permettent d'estimer les paramètres d'un modèle.

Chapitre 2

Éléments d'optimisation convexe pour la parcimonie

Sommaire

2.1 Introduction 25
2.2 Rappels d'optimisation convexe et différentiable 26
2.2.1 Premières définitions et propriétés 26
2.2.2 Généralités sur les problèmes d'optimisation 27
2.2.3 Problèmes d'optimisation convexe 28
2.2.4 Algorithmes de référence 30
2.3 Parcimonie 34
2.3.1 Qu'est-ce que la parcimonie ? 34
2.3.2 LASSO 36
2.3.3 Normes et parcimonie structurée 38
2.4 Méthodes proximales 39
2.4.1 Définition et premières interprétations 39
2.4.2 Propriétés fondamentales de l'opérateur proximal 41
2.4.3 Algorithmes proximaux 43
2.4.4 Calcul pratique des opérateurs proximaux 47
2.5 Synthèse 49

2.1 Introduction

Dans le chapitre précédent, nous avons vu qu'un problème d'apprentissage bien posé aboutissait à un problème d'optimisation via le principe de minimisation du risque empirique, risque pouvant être régularisé. Le problème se présente généralement sous la forme :

$$\underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, \mathcal{J}(\mathbf{c}) = f(\mathbf{c}) + \lambda\Omega(\mathbf{c}) \tag{2.1}$$

où $f : \mathcal{C} \to \mathbb{R}$ est une fonction qui traduit un coût d'attache aux données, $\Omega : \mathcal{C} \to \mathbb{R}$ est une fonction de pénalité qui incorpore un *a priori* sur la nature des données et induit

de la *parcimonie* sur le paramètre $\mathbf{c}$ du modèle, concept défini ci-après. $\lambda > 0$ est un hyperparamètre qui règle le compromis entre les deux objectifs.

Les fonctions f et Ω que nous considérons dans le manuscrit sont *convexes*. Ce chapitre vise à définir des notions-clés d'optimisation convexe et à passer en revue des algorithmes de l'état de l'art pertinents dans la résolution du problème (2.1).

2.2 Rappels d'optimisation convexe et différentiable

Les éléments présentés dans cette section sont traités dans la plupart des ouvrages d'optimisation convexe, nous recommandons notamment celui de Boyd et Lieven Vandenberghe (2009).

2.2.1 Premières définitions et propriétés

Nous commençons par rappeler quelques définitions et propriétés sur les fonctions convexes.

Définition 2.1. *(Ensemble convexe) Un ensemble $\mathcal{C}$ est convexe si, et seulement si, pour tout $(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{C}^2$ et pour tout $\theta \in [0,1]$, $\theta\mathbf{x}_1 + (1-\theta)\mathbf{x}_2 \in \mathcal{C}$.*

Par exemple, soit $\|\cdot\|$ une norme sur $\mathbb{R}^m$. En utilisant les propriétés sur les normes, on peut montrer que la boule de rayon $r > 0$ de centre $\mathbf{x}_C \in \mathbb{R}^m$ définie par $\mathcal{B}(\mathbf{x}_C, r) = \{\mathbf{x} \in \mathbb{R}^m \mid \|\mathbf{x} - \mathbf{x}_C\| \leq r\}$ est convexe.

Définition 2.2. *Soit une fonction $f : \mathbb{R}^m \to \mathbb{R}$,*

- *le* domaine de définition *de f est : **dom** $f = \{\mathbf{x} \in \mathbb{R}^m \mid \exists y \in \mathbb{R},\ y = f(\mathbf{x})\}$*
- *le* graphe *de f est : **graph** $f = \{(\mathbf{x}, f(\mathbf{x})) \mid \mathbf{x} \in$ **dom** $f\}$*
- *l'*épigraphe *de f est : **epi** $f = \{(\mathbf{x}, y) \mid \mathbf{x} \in$ **dom** $f,\ f(\mathbf{x}) \leq y\}$*

Définition 2.3. *(Fonction convexe) Une fonction $f : \mathbb{R}^m \to \mathbb{R}$ est convexe si, et seulement si, **dom** f est un ensemble convexe et*

$$\forall (\mathbf{x}, \mathbf{z}) \in (\textbf{dom } f)^2, \forall \theta \in [0,1],\ f\left(\theta\mathbf{x} + (1-\theta)\mathbf{z}\right) \leq \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{z}) \tag{2.2}$$

f est dite *strictement convexe* si l'inégalité (2.2) est stricte pour tout $\mathbf{x} \neq \mathbf{z}$ et pour tout $0 < \theta < 1$. f est *concave* si $-f$ est convexe et *strictement concave* si $-f$ est strictement convexe.

Proposition 2.1. *Une fonction $f : \mathbb{R}^m \to \mathbb{R}$ est convexe si, et seulement si, son épigraphe **epi** f est un ensemble convexe.*

Conditions du premier ordre.

Proposition 2.2. *Soit $f : \mathbb{R}^m \to \mathbb{R}$ une fonction différentiable, c'est-à-dire son gradient ∇f est défini en tout point de l'ouvert **dom** f. Alors f est convexe si, et seulement si, **dom** f est convexe et*

$$\forall (\mathbf{x}, \mathbf{z}) \in (\textbf{dom } f)^2, \ f(\mathbf{z}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T (\mathbf{z} - \mathbf{x}) \tag{2.3}$$

L'inégalité (2.3) affirme que pour une fonction convexe, l'approximation donnée par un développement de Taylor du premier ordre est toujours inférieure à la valeur de la fonction. Réciproquement, si l'approximation donnée par un développement de Taylor du premier ordre d'une fonction est toujours inférieure à sa valeur, alors cette fonction est convexe. Ainsi, à partir d'informations *locales* (valeur de la fonction et de son gradient en un point donné) sur une fonction convexe, on peut tirer des informations *globales* (approximation inférieure à la fonction). Cette remarque capitale est le fondement des propriétés remarquables des fonctions convexes et justifie le développement de toute une littérature autour des problèmes d'optimisation convexe. Voici une première conséquence qui découle directement de l'inégalité (2.3).

Proposition 2.3. *Soit $f : \mathbb{R}^m \to \mathbb{R}$ une fonction convexe et différentiable. Alors $\nabla f(\mathbf{x}^\star) = 0$ si, et seulement si, $\mathbf{x}^\star$ est un minimiseur global de f : $\mathbf{x}^\star \in \operatorname{argmin} f$.*

Conditions du second ordre. Nous supposons maintenant que f est deux fois différentiable, c'est-à-dire que sa Hessienne $\nabla^2 f$ est définie en chaque point de l'ouvert **dom** f.

Proposition 2.4. *Soit $f : \mathbb{R}^m \to \mathbb{R}$ une fonction deux fois différentiable. Alors f est convexe si, et seulement si, **dom** f est convexe et sa Hessienne est semi-définie positive, c'est-à-dire :*

$$\forall \mathbf{x} \in \textbf{dom } f, \ \nabla^2 f(\mathbf{x}) \in \mathbb{S}_+^m \tag{2.4}$$

où $\mathbb{S}_+^m = \{X \in \mathbb{R}^{m \times m} \mid X \succeq 0\}$ est l'ensemble des matrices carrées semi-définies positives d'ordre m.

2.2.2 Généralités sur les problèmes d'optimisation

Un problème d'optimisation peut se présenter sous la forme standard suivante :

$$\begin{array}{lll} \text{minimiser} & f(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^m \\ \text{sous contraintes} & g_i(\mathbf{x}) \leq 0, & i \in \mathbb{N}_p \\ & h_j(\mathbf{x}) = 0, & j \in \mathbb{N}_q \end{array} \tag{2.5}$$

où f est la fonction objectif ou fonction de coût à minimiser et pour laquelle une solution $\mathbf{x}_0 \in \mathbb{R}^m$ respecte à la fois les conditions $g_i(\mathbf{x}) \leq 0,\ i \in \mathbb{N}_p$, appelées contraintes d'inégalités et $h_j(\mathbf{x}) = 0,\ j \in \mathbb{N}_q$ sont les contraintes d'égalité. Si de telles contraintes sont inexistantes dans le problème (2.5), on parle de problème sans contraintes.

L'ensemble $\mathcal{D}$ des points pour lesquels les fonctions f, g_i, h_j sont définies

$$\mathcal{D} = \mathbf{dom}\ f \cap \bigcap_{i=1}^{p} \mathbf{dom}\ g_i \cap \bigcap_{j=1}^{q} \mathbf{dom}\ h_j$$

est appelé domaine du problème d'optimisation de (2.5). Un point $\mathbf{x} \in \mathcal{D}$ est dit *admissible* s'il satisfait toutes les contraintes d'inégalité et d'égalité. L'ensemble de tous les points admissibles est appelé *ensemble admissible* et est noté $\mathcal{A}$. Enfin, la valeur optimale $f^\star$ du problème (2.5) est définie par :

$$f^\star = \inf \{f(\mathbf{x}) \mid g_i(\mathbf{x}) \leq 0,\ i \in \mathbb{N}_p,\ h_j(\mathbf{x}) = 0,\ j \in \mathbb{N}_q\}$$

Solution globale, solution locale. On dit que $\mathbf{x}^\star$ est une *solution* ou un *point optimal* de (2.5) si $\mathbf{x}^\star$ est admissible et $f(\mathbf{x}^\star) = f^\star$. Si un tel point $\mathbf{x}^\star$ existe, on dit que la valeur optimale est atteinte en $\mathbf{x}^\star$.

On dit qu'un point admissible $\mathbf{x}_0$ est une *solution locale* ou un *optimum local* de (2.5) s'il existe un voisinage V de $\mathbf{x}_0$ tel que :

$$f(\mathbf{x}_0) = \inf \{f(\mathbf{z}) \mid g_i(\mathbf{z}) \leq 0,\ i \in \mathbb{N}_p,\ h_j(\mathbf{z}) = 0,\ j \in \mathbb{N}_q,\ \mathbf{z} \in V\}$$

Dans la littérature, certains auteurs utilisent le vocable d'optimum « global » en lieu et place d'optimum pour mettre en exergue la différence entre les deux types de solutions.

2.2.3 Problèmes d'optimisation convexe

Les problèmes d'optimisation convexe se présentent sous la forme générale :

$$\begin{array}{lll} \text{minimiser} & f(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^m \\ \text{sous contraintes} & g_i(\mathbf{x}) \leq 0, & i \in \mathbb{N}_p \\ & \mathbf{a}_j^T \mathbf{x} = b_j, & j \in \mathbb{N}_q \end{array} \tag{2.6}$$

Il s'agit d'une instanciation particulière du problème d'optimisation général (2.5) avec les conditions requises suivantes :

— la fonction objectif f est convexe

— les fonctions des contraintes d'inégalité $g_i,\ i \in \mathbb{N}_p$ sont convexes

— les fonctions des contraintes d'égalité, de la forme $h_j(\mathbf{x}) = \mathbf{a}_j^T\mathbf{x} - b_j,\ j \in \mathbb{N}_q$ avec $\mathbf{a}_j \in \mathbb{R}^m, b_j \in \mathbb{R}$, sont affines

On peut remarquer d'emblée que l'ensemble admissible du problème (2.6) est un ensemble convexe. En effet, le domaine du problème est bien convexe. De plus, les p lignes de sous-niveau $\{\mathbf{x} \in \mathbb{R}^m \mid g_i(\mathbf{x}) \leq 0\}$ et les q hyperplans $\{\mathbf{x} \in \mathbb{R}^m \mid \mathbf{a}_j^T\mathbf{x} = b_j\}$ sont convexes. Le résultat vient alors par intersection de ces ensembles. Ainsi, dans un problème d'optimisation convexe, on minimise une fonction objectif convexe sur un ensemble convexe. Si f est strictement convexe, il existe au plus un optimum global.

Optimum local, optimum global. Nous énonçons ensuite une propriété fondamentale des problèmes d'optimisation convexe.

Proposition 2.5. *Tout optimum local du problème d'optimisation convexe (2.6) est un optimum global.*

Démonstration. Soit $\mathbf{x}_0$ un optimum local de (2.6), on note $\mathcal{A}$ l'ensemble admissible du problème. On a :

$$\exists r > 0, \forall \mathbf{x} \in \mathcal{A}, \|\mathbf{x} - \mathbf{x}_0\| \leq r \Rightarrow f(\mathbf{x}_0) \leq f(\mathbf{x})$$

Soit $\tilde{\mathbf{x}} \in \mathcal{A}$ un point admissible avec $\tilde{\mathbf{x}} \neq \mathbf{x}_0$. Deux cas se présentent :

1. Soit $\|\tilde{\mathbf{x}} - \mathbf{x}_0\| \leq r$, alors $f(\mathbf{x}_0) \leq f(\tilde{\mathbf{x}})$
2. Soit $\|\tilde{\mathbf{x}} - \mathbf{x}_0\| > r$, on pose alors :

$$\mathbf{z} = \left(1 - \frac{r}{2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|}\right)\mathbf{x}_0 + \frac{r}{2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|}\tilde{\mathbf{x}}$$

Comme $0 < r/\left(2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|\right) < 1/2$, par convexité de $\mathcal{A}$, $\mathbf{z}$ est un point admissible. Ensuite, on vérifie aisément que $\|\mathbf{z} - \mathbf{x}_0\| = r/2$ et donc que $f(\mathbf{x}_0) \leq f(\mathbf{z})$. Par convexité de f, on obtient ainsi

$$f(\mathbf{x}_0) \leq f(\mathbf{z}) \leq \left(1 - \frac{r}{2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|}\right) f(\mathbf{x}_0) + \frac{r}{2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|} f(\tilde{\mathbf{x}})$$

soit $0 \leq r\left(f(\tilde{\mathbf{x}}) - f(\mathbf{x}_0)\right) / \left(2\|\tilde{\mathbf{x}} - \mathbf{x}_0\|\right)$ et finalement $f(\mathbf{x}_0) \leq f(\tilde{\mathbf{x}})$.

□

Critère d'optimalité pour les fonctions différentiables

On rappelle que pour toute fonction f convexe et différentiable, on a :

$$\forall (\mathbf{x}, \mathbf{z}) \in (\mathbf{dom}\ f)^2,\ f(\mathbf{z}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{z} - \mathbf{x})$$

Proposition 2.6. *Si la fonction objectif f du problème d'optimisation convexe (2.6) est différentiable, $\mathbf{x}^\star$ est un point optimal si, et seulement si, $\mathbf{x}^\star$ est un point admissible et pour tout point admissible $\mathbf{z}$,*

$$\nabla f(\mathbf{x}^\star)^T(\mathbf{z}-\mathbf{x}^\star) \geq 0 \tag{2.7}$$

Dans le cas des problèmes sans contraintes, le critère d'optimalité (2.7) se réduit à la condition nécessaire et suffisante :

$$\nabla f(\mathbf{x}^\star) = 0$$

Deux types de problèmes d'optimisation convexe ont fait l'objet d'études poussées :

1. les programmes linéaires (Todd (2002) et références incluses) de la forme :

$$\begin{array}{lll} \text{minimiser} & \mathbf{c}^T\mathbf{x}, & \mathbf{x} \in \mathbb{R}^m \\ \text{sous contraintes} & G\mathbf{x} \preceq \mathbf{h} & \\ & A\mathbf{x} = \mathbf{b} & \end{array} \tag{2.8}$$

   où $\preceq$ désigne la comparaison vectorielle élément par élément, $\mathbf{c} \in \mathbb{R}^m, G \in \mathbb{R}^{p\times m}, \mathbf{h} \in \mathbb{R}^p, A \in \mathbb{R}^{q\times m}$ et $\mathbf{b} \in \mathbb{R}^q$

2. les programmes quadratiques de la forme :

$$\begin{array}{lll} \text{minimiser} & \frac{1}{2}\mathbf{x}^T U\mathbf{x} + \mathbf{c}^T\mathbf{x} + r, & \mathbf{x} \in \mathbb{R}^m \\ \text{sous contraintes} & G\mathbf{x} \preceq \mathbf{h} & \\ & A\mathbf{x} = \mathbf{b} & \end{array} \tag{2.9}$$

   avec $U \in \mathbb{S}^m_+, \mathbf{c} \in \mathbb{R}^m, r \in \mathbb{R}, G \in \mathbb{R}^{p\times m}, \mathbf{h} \in \mathbb{R}^p, A \in \mathbb{R}^{q\times m}$ et $\mathbf{b} \in \mathbb{R}^q$. Le problème de régression linéaire des moindres carrés est un exemple de programme quadratique sans contraintes.

2.2.4 Algorithmes de référence

Nous présentons dans cette sous-section quelques algorithmes classiques dédiés à la résolution de problèmes d'optimisation convexe sans contraintes dans des cas plutôt favorables.

$$\text{minimiser} \quad f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^m \tag{2.10}$$

Ainsi, nous considérons que la fonction objectif $f : \mathbb{R}^m \to \mathbb{R}$ est convexe et de classe $\mathcal{C}^2$ sur l'ouvert $\mathbf{dom}\ f$. De plus, nous supposons que le problème (2.10) admet au moins un point optimal $\mathbf{x}^\star$ dont la valeur optimale est notée $f^\star = f(\mathbf{x}^\star) = \inf_{\mathbf{x}} f(\mathbf{x})$. Nous avons vu précédemment dans ce cas-ci qu'une condition nécessaire et suffisante pour que $\mathbf{x}^\star$ soit

optimal est qu'il annule le gradient de f :

$$\nabla f(\mathbf{x}^\star) = 0 \tag{2.11}$$

Par conséquent, résoudre le problème sans contraintes (2.10) revient à trouver l'ensemble solution du problème (2.11) qui correspond à un système de m équations à m inconnues. La résolution analytique (i.e. l'existence d'une solution en forme close) de (2.11) n'est envisageable que dans de rares cas, il est davantage fréquent de recourir à un algorithme *itératif*, dans le sens où ce dernier construit une suite $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots \in \mathbf{dom}\ f$ avec la propriété de convergence $f(\mathbf{x}^{(\ell)}) \to f^\star$ quand $\ell \to +\infty$. L'algorithme s'arrête lorsqu'un critère d'arrêt est satisfait ou bien après un nombre d'itérations prédéfini. Un critère d'arrêt classique consiste à vérifier si $f\left(\mathbf{x}^{(\ell)}\right) - f^\star \leq \epsilon$, où $\epsilon > 0$ est un paramètre de seuil ou de tolérance sur l'erreur d'approximation.

Méthodes de descente

Les algorithmes que nous décrivons produisent une suite d'itérés $\mathbf{x}^{(\ell)},\ \ell = 1, 2, \ldots$ de la forme :

$$\mathbf{x}^{(\ell+1)} = \mathbf{x}^{(\ell)} + \alpha^{(\ell)} \Delta\mathbf{x}^{(\ell)}$$

où $\alpha^{(\ell)} \geq 0$ est le *pas* et $\Delta\mathbf{x}^{(\ell)} \in \mathbb{R}^m$ correspond à une direction de recherche. Les méthodes que nous étudions sont des *méthodes de descente* dans le sens où $f\left(\mathbf{x}^{(\ell+1)}\right) < f\left(\mathbf{x}^{(\ell)}\right)$ sauf lorsque $\mathbf{x}^{(\ell)}$ est optimal. Par convexité de f, la caractérisation du premier ordre donne : $f\left(\mathbf{x}^{(\ell+1)}\right) - f\left(\mathbf{x}^{(\ell+1)}\right) \geq \nabla f\left(\mathbf{x}^{(\ell)}\right)^T (\mathbf{x}^{(\ell+1)} - \mathbf{x}^{(\ell)})$. Pour obtenir $f\left(\mathbf{x}^{(\ell+1)}\right) < f\left(\mathbf{x}^{(\ell)}\right)$, il faut nécessairement :

$$\nabla f\left(\mathbf{x}^{(\ell)}\right)^T \Delta\mathbf{x}^{(\ell)} < 0$$

Ainsi, les méthodes de descente imposent une direction de recherche $\Delta\mathbf{x}^{(\ell)}$ qui doit faire un angle aigu avec l'opposé du gradient $-\nabla f\left(\mathbf{x}^{(\ell)}\right)$. Dans ce contexte, $\Delta\mathbf{x}^{(\ell)}$ est alors appelé direction de descente pour f au point $\mathbf{x}^{(\ell)}$. Les étapes d'une méthode de descente sont déclinées dans l'algorithme générique 2.1.
Les méthodes de descente s'articulent autour de deux étapes charnières :

1. le calcul d'une direction de descente $\Delta\mathbf{x}^{(\ell)}$
2. la sélection d'un pas $\alpha^{(\ell)} \geq 0$ qui permet de localiser le prochain itéré sur la demi-droite $\{\mathbf{x}^{(\ell)} + \alpha\Delta\mathbf{x}^{(\ell)} \mid \alpha \in \mathbb{R}_+\}$. Cette deuxième étape est appelée recherche linéaire (*line search* en anglais).

Recherche linéaire. L'on distingue deux stratégies de recherche linéaire. La première option qui est la recherche linéaire *exacte* consiste à choisir $\alpha^{(\ell)}$ de telle sorte à minimiser

Algorithme 2.1 Méthode de descente

Entrées : $\mathbf{x}^{(0)} \in \mathbf{dom}\ f$
Initialisation : $\ell := 0$; `stop` :=`faux`
tant que `stop`=`faux` **faire**
1. Déterminer une direction de descente $\Delta\mathbf{x}^{(\ell)}$
2. Choisir un pas $\alpha^{(\ell)} \geq 0$
3. $\mathbf{x}^{(\ell+1)} = \mathbf{x}^{(\ell)} + \alpha^{(\ell)}\Delta\mathbf{x}^{(\ell)}$
4. `stop` := « critère d'arrêt satisfait ? »
5. $\ell := \ell + 1$
fin tant que
Sorties : $\mathbf{x}^{(\ell)}$

f sur la demi-droite $\{\mathbf{x}^{(\ell)} + \alpha\Delta\mathbf{x}^{(\ell)} \mid \alpha \in \mathbb{R}_+\}$:

$$\alpha^{(\ell)} = \underset{\alpha\in\mathbb{R}_+}{\operatorname{argmin}}\ f\left(\mathbf{x}^{(\ell)} + \alpha\Delta\mathbf{x}^{(\ell)}\right) \tag{2.12}$$

On peut effectuer une recherche linéaire exacte lorsque le coût algorithmique pour résoudre le problème (2.12) est faible par rapport au coût de calcul de la direction de descente. Si tel n'est pas le cas, la recherche linéaire se fait généralement de façon *inexacte*, on se propose alors de résoudre de manière approchée le problème (2.12). Parmi les méthodes de recherche linéaire inexacte, nous présentons la technique du *backtracking* (Algorithme 2.2) qui est très simple à mettre en œuvre et qui s'avère efficace en pratique.

Algorithme 2.2 Backtracking pour la recherche linéaire

Entrées : $\mathbf{x} \in \mathbf{dom}\ f$; une direction de descente $\Delta\mathbf{x}$ pour f en $\mathbf{x}$; $\beta \in (0, 0.5)$; $\gamma \in (0, 1)$

Initialisation : $\alpha := 1$
tant que $f(\mathbf{x} + \alpha\Delta\mathbf{x}) > f(\mathbf{x}) + \beta\alpha\nabla f(\mathbf{x})^T\Delta\mathbf{x}$ **faire**
$\alpha := \gamma\alpha$
fin tant que
Sorties : α

Méthode de descente de gradient. Le choix le plus évident pour la direction de descente pour f à un point $\mathbf{x}$ consiste à prendre $\Delta\mathbf{x} = -\nabla f(\mathbf{x})$. L'algorithme 2.1 instancié avec ce choix particulier à l'étape 1 pour la direction de descente est appelé *algorithme de descente de gradient*. Une analyse des résultats théoriques et empiriques des méthodes de descente de gradient (Boyd et Lieven Vandenberghe, 2009) permet de tirer les enseignements suivants :

— l'algorithme de descente de gradient affiche approximativement une convergence linéaire, c'est-à-dire que l'erreur $f\left(\mathbf{x}^{(\ell)}\right) - f^\star$ converge vers 0 approximativement

comme une série géométrique ;
— le choix des paramètres β et γ dans l'algorithme de backtracking a un impact avéré mais relativement limité sur la convergence. Même si une recherche linéaire exacte améliore la convergence de l'algorithme, le gain par rapport à une recherche inexacte demeure toutefois généralement modeste pour qu'il vaille vraiment la peine d'effectuer une recherche exacte ;
— le taux de convergence dépend essentiellement du conditionnement de la Hessienne $\nabla^2 f$. Dans le cas de problèmes mal conditionnés, la convergence peut être dramatiquement ralentie, voire si lente que la méthode est inutile en pratique.

Méthode de la plus forte pente. Une approximation du premier ordre de Taylor de $f(\mathbf{x}+\mathbf{v})$ autour de $\mathbf{x}$ est donnée par :

$$f(\mathbf{x}+\mathbf{v}) \approx f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{v}$$

Le terme $\nabla f(\mathbf{x})^T \mathbf{v}$ est la *dérivée directionnelle* de f au point $\mathbf{x}$ dans la direction $\mathbf{v}$. $\mathbf{v}$ est une direction de descente pour f en $\mathbf{x}$ si la dérivée directionnelle $\nabla f(\mathbf{x})^T \mathbf{v}$ est négative. Soit $\|\cdot\|$ une norme sur $\mathbb{R}^m$, on peut alors chercher une direction $\mathbf{v}$ normée (i.e. $\|\mathbf{v}\| = 1$) pour laquelle la dérivée directionnelle est la plus faible possible :

$$\Delta \mathbf{x}_{\text{pfp}} \in \underset{\mathbf{v}\in\mathbb{R}^m}{\operatorname{argmin}} \left\{ \nabla f(\mathbf{x})^T \mathbf{v} \mid \|\mathbf{v}\| = 1 \right\}$$

$\Delta \mathbf{x}_{\text{pfp}}$ est une direction de recherche normée qui entraîne la plus grande décroissance dans l'approximation linéaire de f. C'est la raison pour laquelle $\Delta \mathbf{x}_{\text{pfp}}$ est appelée direction de descente *de plus forte pente*. La méthode dite de la plus forte pente est une instanciation de l'algorithme 2.1 qui utilise une direction de descente de plus forte pente dans l'étape de calcul de la direction de recherche. Un élément déterminant pour cette méthode réside dans le choix de la norme. Dans le cas de la norme euclidienne ℓ_2 $\|\cdot\|_2$, la direction de descente de plus forte pente est : $\Delta \mathbf{x}_{\text{pfp}} = -\nabla f(\mathbf{x})/\|\nabla f(\mathbf{x})\|_2$ et on revient à une méthode de descente de gradient classique. D'autres choix classiques sont la norme ℓ_1 ou encore la P-norme définie par $\|\mathbf{z}\|_P = \left(\mathbf{z}^T P \mathbf{z}\right)^{1/2}$ où P est une matrice définie positive. L'utilisation d'une norme appropriée pour un problème donné permet d'obtenir de meilleurs taux de convergence qu'une méthode de descente de gradient.

Méthode de Newton. Un développement de Taylor du second ordre de $f(\mathbf{x}+\mathbf{v})$ autour de $\mathbf{x}$ donne :

$$f(\mathbf{x}+\mathbf{v}) \approx \hat{f}(\mathbf{v}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{v} + \frac{1}{2} \mathbf{v}^T \nabla^2 f(\mathbf{x}) \mathbf{v}$$

$\hat{f}$ est une fonction convexe quadratique en $\mathbf{v}$. Appliquons la condition d'optimalité du premier ordre :

$$\nabla_{\mathbf{v}} \hat{f}(\mathbf{v}) = 0 \quad \Leftrightarrow \quad \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x})\mathbf{v} = 0$$

Ainsi $\Delta\mathbf{x}_{\text{n}} = -\nabla^2 f(\mathbf{x})^{-1}\nabla f(\mathbf{x})$ est l'optimum de $\hat{f}$. Il s'ensuit que $\mathbf{x} + \Delta\mathbf{x}_{\text{n}}$ est le minimiseur de l'approximation du second ordre de f en $\mathbf{x}$. $\Delta\mathbf{x}_{\text{n}}$ est appelé *pas de Newton* pour f au point $\mathbf{x}$. Vérifions que $\Delta\mathbf{x}_{\text{n}}$ est une direction de descente :

$$\nabla f(\mathbf{x})^T \Delta\mathbf{x}_{\text{n}} = -\nabla f(\mathbf{x})^T \nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x}) \leq 0$$

Le résultat vient du caractère défini positif de la Hessienne $\nabla^2 f(\mathbf{x})$. La méthode de Newton est une méthode de descente qui utilise le pas de Newton comme direction de descente. On peut considérer la méthode de Newton comme un cas particulier de la méthode de la plus forte pente pour la norme définie par la Hessienne, à savoir $\|\mathbf{z}\|_{\nabla^2 f(\mathbf{x})} = \left(\mathbf{z}^T \nabla^2 f(\mathbf{x}) \mathbf{z}\right)^{1/2}$. La méthode de Newton présente plusieurs avantages compétitifs :

- sa convergence est en général rapide et quadratique près de l'optimum $\mathbf{x}^\star$ (Kantorovich, 1948)
- elle se comporte bien lors d'un passage à l'échelle : on obtient des performances similaires sur les problèmes dans $\mathbb{R}^{10}$ et sur ceux dans $\mathbb{R}^{10000}$ pour un nombre d'itérations équivalent

L'inconvénient majeur de la méthode de Newton est le coût de calcul et de stockage de la matrice Hessienne ainsi que le calcul du pas de Newton qui nécessite la résolution d'un système d'équations linéaires.

Nous invitons le lecteur intéressé par une analyse plus approfondie de ces méthodes de descente à se référer aux ouvrages de Dennis Jr et Schnabel (1996) et de Ortega et Rheinboldt (2000).

2.3 Parcimonie

2.3.1 Qu'est-ce que la parcimonie ?

Revenons au problème général (2.1) :

$$\underset{\mathbf{c} \in \mathcal{C}}{\operatorname{argmin}} \, \mathcal{J}(\mathbf{c}) = f(\mathbf{c}) + \lambda \Omega(\mathbf{c})$$

Pour illustrer ce dernier, considérons un problème de régression. Soient d variables explicatives $X_1, \ldots, X_d$ et Y une variable aléatoire à expliquer. L'on dispose d'un ensemble

d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$, l'on note $\mathbf{y} = (y_1 \dots y_N)^T \in \mathbb{R}^N$ le vecteur des sorties observées, $X \in \mathbb{R}^{N \times d}$ la matrice des données associée où le coefficient x_i^j désigne la valeur de la variable X_j pour la i-ème observation. L'on rappelle qu'en régression linéaire, le modèle s'écrit :

$$\hat{y} = h_{\boldsymbol{\beta}}(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta}$$

où $\boldsymbol{\beta} = (\beta_1 \dots \beta_d)^T \in \mathbb{R}^d$ est le paramètre de régression. Comme nous l'avons vu dans le chapitre précédent, l'estimation de $\boldsymbol{\beta}$ peut être réalisée en minimisant une fonction de perte régularisée afin d'éviter l'écueil du surapprentissage.

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^d}{\operatorname{argmin}} \mathcal{J}_{\text{ridge}}(\boldsymbol{\beta}) = f(\boldsymbol{\beta}) + \lambda \Omega(\boldsymbol{\beta})$$

où :

— $f(\boldsymbol{\beta}) = \|\mathbf{y} - X\boldsymbol{\beta}\|_2^2$
— $\Omega(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2$

La régularisation ridge pénalise le carré de la norme ℓ_2 du paramètre $\boldsymbol{\beta}$ et a pour effet de réduire les valeurs de $\boldsymbol{\beta}$. Au-delà d'une simple diminution, l'on peut désirer que plusieurs coefficients de $\boldsymbol{\beta}$ soient exactement égaux à zéro de telle sorte que les variables explicatives X_j associées à un coefficient β_j nul soient écartées du modèle : on obtient en conséquence un modèle *parcimonieux* et on réalise dans le même temps une sélection de variables ou sélection de caractéristiques (*feature selection* en anglais). Le principe de parcimonie est un concept fondamental dans plusieurs sciences, il stipule que parmi toutes les explications qui rendent compte d'un phénomène donné, l'on doit préférer la plus simple ou la moins complexe d'entre elles. En apprentissage statistique, la problématique de la parcimonie peut être vue sous l'angle de la sélection de caractéristiques.

Définition 2.4. *(Support) Le support d'un vecteur $\mathbf{z} \in \mathbb{R}^d$ est défini par :*

$$supp(\mathbf{z}) = \{j \in \mathbb{N}_d \mid z_j \neq 0\}$$

Définition 2.5. *(Norme ℓ_q, pseudo-norme ℓ_0) Pour $q \geq 1$, la norme ℓ_q d'un vecteur $\mathbf{z} \in \mathbb{R}^d$ est $\|\mathbf{z}\|_q = \left(\sum_{j=1}^d |z_j|^q\right)^{1/q}$, $\|\mathbf{z}\|_\infty = \max_{1 \leq j \leq d} |z_j| = \lim_{q \to \infty} \|\mathbf{z}\|_q$. Enfin, l'on définit la pénalité ℓ_0 ou pseudo-norme ℓ_0 de $\mathbf{z}$ par le nombre de coefficients non nuls de $\mathbf{z}$: $\|\mathbf{z}\|_0 = |supp(\mathbf{z})| = \lim_{q \to 0^+} \|\mathbf{z}\|_q$.*

En régression linéaire, un modèle est parcimonieux si une faible fraction des coefficients β_j sont non nuls ($\|\boldsymbol{\beta}\|_0$ est « petit »). Une première idée pour garantir la parcimonie du modèle est de considérer la minimisation directe d'un coût régularisé par la pseudo-norme

ℓ_0 du paramètre $\boldsymbol{\beta}$:

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}} \|\mathbf{y} - X\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0 \tag{2.13}$$

Cependant le problème (2.13) est un problème combinatoire difficile (Natarajan, 1995 ; Tropp, 2004). Il est alors d'usage de remplacer la pseudo-norme ℓ_0 par une pénalité convexe dont l'expression fait intervenir des normes. Les estimateurs $\hat{\boldsymbol{\beta}}$ qui en résultent sont alors solutions de problèmes d'optimisation convexe. Faire apparaître la problématique de l'estimation parcimonieuse comme un problème d'optimisation convexe induit deux avantages majeurs : d'une part, cela justifie l'utilisation d'algorithmes d'estimation dédiés et efficaces comme nous le montrons dans la suite du chapitre ; deuxièmement, cela permet de travailler dans un cadre théorique où l'on est en mesure d'apporter des réponses aux questions concernant la qualité des prédictions fournies et la consistance de l'estimation.

2.3.2 LASSO

L'exemple canonique de régularisation parcimonieuse est celui de la régression linéaire pénalisée par la nome ℓ_1 :

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}}\, \mathcal{J}_{\text{lasso}}(\boldsymbol{\beta}) = \|\mathbf{y} - X\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \tag{2.14}$$

Le problème (2.14) conduit au LASSO (Tibshirani, 1996) en statistiques ou de manière équivalente à sa formulation duale, appelée *basis pursuit* (Chen et al., 1998) dans la communauté du traitement du signal. Pour comprendre pourquoi la norme ℓ_1 induit de la parcimonie contrairement à une régularisation en norme ℓ_2, nous proposons d'analyser la figure 2.1. Sur cette figure, le centre de l'ellipse (le point noté $\hat{\boldsymbol{\beta}}^{OLS}$) correspond à

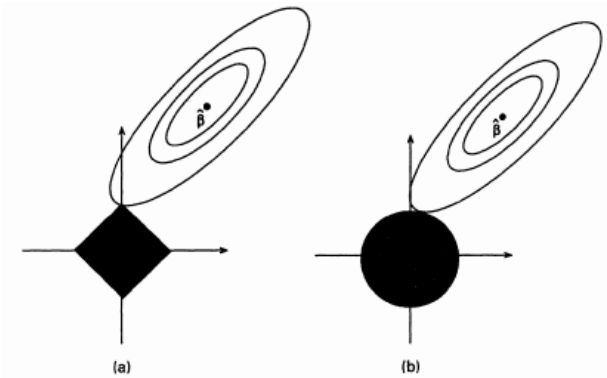


Figure 2.1 – Parcimonie : LASSO vs régression ridge. *Source :* Tibshirani (1996).

l'estimateur des moindres carrés ordinaires non régularisé. Chaque contour elliptique centré en $\hat{\boldsymbol{\beta}}^{OLS}$ est une ligne de niveau de l'erreur quadratique $f(\hat{\boldsymbol{\beta}}) = \|\mathbf{y} - X\hat{\boldsymbol{\beta}}\|_2^2$, i.e. $N_c(f) =$

$\left\{\hat{\boldsymbol{\beta}} \in \mathbb{R}^d \mid \|\mathbf{y} - X\hat{\boldsymbol{\beta}}\|_2^2 = c\right\}$ représente l'ensemble des estimateurs $\hat{\boldsymbol{\beta}}$ qui conduisent à la même erreur quadratique c. Pénaliser par une norme ℓ_q ($q \geq 1$) est équivalent à minimiser sous la contrainte $\|\boldsymbol{\beta}\|_q \leq \mu$, pour un certain $\mu > 0$. Les boules pour les normes ℓ_1 et ℓ_2, notées $\mathcal{B}_{\ell_1}$ et $\mathcal{B}_{\ell_2}$, prennent respectivement la forme d'un losange et d'un disque tous deux centrés en 0. La solution du problème pour un λ donné (ou le μ approprié) est donnée par la première ellipse qui intersecte la boule. Tandis que $\mathcal{B}_{\ell_2}$ est isotropique et ne favorise aucune direction de l'espace, $\mathcal{B}_{\ell_1}$ est anisotropique et possède des points singuliers (les « coins » du losange) dus à la non-dérivabilité de la norme ℓ_1. Cette propriété a pour conséquence qu'une ligne de niveau de f a une plus grande probabilité d'être tangente aux coins de $\mathcal{B}_{\ell_1}$. De plus, les coins de $\mathcal{B}_{\ell_1}$ sont situés sur les axes de $\mathbb{R}^d$ et ce sont précisément à ces endroits-là que les coordonnées s'annulent et qu'on obtient ainsi des solutions parcimonieuses. Par ailleurs, plusieurs méthodes efficaces ont été proposées pour résoudre (2.14) (Markowitz, 1956 ; Fu, 1998 ; Efron et al., 2004 ; Wu et Lange, 2008).

Deux résultats théoriques majeurs concernant le LASSO sont à noter. Supposons que les données sont générées par un paramètre $\boldsymbol{\beta}^\star$ parcimonieux. Premièrement, nous venons de voir que la régularisation par la norme ℓ_1 permet d'obtenir une solution parcimonieuse $\hat{\boldsymbol{\beta}}$, mais rien n'assure que les coefficients $\hat{\beta}_j$ mis à 0 correspondent bien à des variables non pertinentes. Plusieurs travaux ont analysé la nature des zéros trouvés par le LASSO et ont démontré qu'il était possible dans certaines conditions de retrouver le support exact (propriété de consistance de la sélection de modèle), i.e. $\mathrm{supp}(\hat{\boldsymbol{\beta}}) \xrightarrow[N\infty]{} \mathrm{supp}(\boldsymbol{\beta}^\star)$ (Zhao et Yu, 2006 ; Hui Zou, 2006 ; Yuan et Lin, 2006 ; Wainwright, 2009). La deuxième contribution d'importance affirme que sous certaines hypothèses, l'on peut garantir des prédictions de qualité en grande dimension, dès lors que le nombre de caractéristiques est au plus exponentiel en le nombre d'observations : $\log d = O(N)$, i.e. $\frac{1}{N}\|X\hat{\boldsymbol{\beta}} - X\boldsymbol{\beta}^\star\|_2^2 \xrightarrow[N\infty]{} 0$ (propriété de consistance des prédictions) (Zhao et Yu, 2006 ; Lounici et al., 2008 ; Bickel et al., 2009 ; Meinshausen et Yu, 2009 ; Wainwright, 2009).

Dans la suite de l'exposé, nous allons considérer d'autres types de modèles parcimonieux que le LASSO. En effet, d'une part, l'hypothèse de linéarité s'avère peu réaliste puisque la plupart des problèmes rencontrés sont non linéaires. Ainsi, le modèle h peut notamment être un modèle à noyaux. Pour les méthodes d'optimisation que nous présentons, nous nous limitons à des fonctions de coût locales ℓ convexes et lisses de sorte que la fonction de coût globale f reste convexe et différentiable, ce qui exclut le coût charnière même si l'étude de ce dernier eût été d'un intérêt certain. Nous considérons toutefois que cette classe de fonctions demeure suffisamment large pour les problèmes que nous traitons. D'autre part, nous allons voir que l'utilisation d'autres normes que la norme ℓ_1 permet de tirer avantage de la structure *a priori* des données et réduire ainsi substantiellement la complexité algorithmique des méthodes d'optimisation.

2.3.3 Normes et parcimonie structurée

Nous considérons à présent le problème général (2.1) :

$$\underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, \mathcal{J}(\mathbf{c}) = f(\mathbf{c}) + \lambda\Omega(\mathbf{c})$$

où $\mathcal{C}$ est un ensemble convexe fermé de $\mathbb{R}^d$, $f : \mathcal{C} \to \mathbb{R}$ est une fonction convexe et différentiable et $\Omega : \mathcal{C} \to \mathbb{R}$ est une fonction convexe, éventuellement non lisse. Comme nous l'avons vu précédemment, l'on peut introduire de la parcimonie en choisissant $\Omega(\mathbf{c}) = \|\mathbf{c}\|_1$. Dans certaines situations, les coefficients de $\mathbf{c}^\star$, une solution optimale de (2.1), sont partitionnés en *groupes* de variables. Prenons l'exemple d'une application biomédicale, des variables ordinales peuvent être utilisées pour coder la niveau d'une caractéristique telle que le degré de sévérité d'un symptôme. Il est d'usage d'introduire autant de variables binaires que de valeurs possibles pour la caractéristique donnée. Dans ce cas, il est naturel de vouloir soit conserver, soit écarter *simultanément* toutes les variables qui font partie d'un même groupe. Toute norme qui exploite explicitement cette structure de groupe est appelée norme de groupe ℓ_1 ou norme ℓ_1/ℓ_q ($q \geq 1$) :

$$\Omega(\mathbf{c}) = \sum_{g\in\mathcal{G}} w_g \|\mathbf{c}_g\|_q = \sum_{g\in\mathcal{G}} w_g \left\{ \sum_{j\in g} |c_j|^q \right\}^{1/q} \tag{2.15}$$

où $\mathcal{G}$ est une partition de $\mathbb{N}_d$, $(w_g)_{g\in\mathcal{G}}$ sont des poids strictement positifs et $\mathbf{c}_g \in \mathbb{R}^{|g|}$ est le vecteur qui contient les coefficients de $\mathbf{c}$ correspondant aux indices dans g. Plusieurs travaux montrent qu'elle permet d'améliorer les performances en prédiction et éventuellement de gagner en interprétabilité sur les modèles appris (Turlach et al., 2005 ; Yuan et Lin, 2006 ; Huang, T. Zhang et al., 2010 ; Lounici et al., 2009 ; Obozinski et al., 2010 ; Roth et Fischer, 2008). La norme de groupe la plus simple est définie pour $q = 2$ par :

$$\Omega(\mathbf{c}) = \sum_{g\in\mathcal{G}} w_g \|\mathbf{c}_g\|_2 \tag{2.16}$$

La norme ainsi définie par (2.16) est connue sous le nom de *norme mixte* ℓ_1/ℓ_2. Elle possède comme caractéristique d'agir comme une norme ℓ_1 sur le vecteur $(\|\mathbf{c}_g\|_2)_{g\in\mathcal{G}}$. L'on obtient ainsi un effet parcimonieux au niveau de groupe puisque la norme ℓ_1 incite à annuler chaque vecteur $\mathbf{c}_g$, alors que la norme ℓ_2 ne promeut aucune parcimonie au sein de chaque groupe $g \in \mathcal{G}$. Ce type de parcimonie structurée appliqué au cas de la régression linéaire avec une régularisation des coefficients de la régression par une norme mixte ℓ_1/ℓ_2 conduit à la formulation du *goupe LASSO* (Turlach et al., 2005 ; Yuan et Lin, 2006). Bien sûr d'autres normes mixtes ℓ_1/ℓ_q plus générales sont aussi utilisées dans la littérature (Zhao,

Rocha et al., 2009). Toutefois, les normes ℓ_1/ℓ_2 et ℓ_1/ℓ_∞ demeurent les choix les plus populaires.

Au-delà des normes ℓ_1/ℓ_q, il est utile, pour certaines applications, de pouvoir relâcher la contrainte de partition sur $\mathcal{G}$. En d'autres termes, en autorisant $\mathcal{G}$ à contenir des groupes de variables qui *se chevauchent*, les normes qui en résultent offrent des schémas de parcimonie structurée plus sophistiqués et permettent notamment d'encoder des relations de hiérarchie entre les variables. Nous invitons le lecteur intéressé par ce sujet à se référer aux travaux cités ci-après : Bach (2009), Jacob et al. (2009), Jenatton et al. (2011), Seyoung Kim et Xing (2010), Schmidt et Kevin P Murphy (2010) et Zhao, Rocha et al. (2009).

2.4 Méthodes proximales

Cette section passe en revue une classe de techniques connues sous le nom de *méthodes proximales*, qui permettent de minimiser efficacement des fonctions objectifs de la forme (2.1). De la même manière que la méthode de Newton est un outil standard pour la résolution de problèmes convexes différentiables et sans contraintes de taille modeste, les algorithmes proximaux peuvent être considérés comme son pendant pour des problèmes convexes non lisses sous contraintes à grande échelle (Amir Beck et Marc Teboulle, 2009 ; Combettes et Pesquet, 2011 ; Nesterov, 2007 ; Wright et al., 2009). Ces algorithmes ont suscité ces dernières années un intérêt croissant de la part de la communauté en apprentissage automatique, grâce à leur aspect général et de par leur facilité de mise en œuvre dans une série de problèmes récents impliquant des bases de données de grande dimension. Pour continuer le parallèle avec la méthode de Newton, si les opérations de base pour cette dernière consistent essentiellement à calculer des gradients et des Hessiennes, les algorithmes proximaux, quant à eux, se situent à un niveau d'abstraction plus élevé car ils nécessitent l'évaluation d'*opérateurs proximaux*, ce qui conduit à résoudre un problème d'optimisation convexe intermédiaire. En général, ces sous-problèmes peuvent être résolus « facilement » et admettent souvent des solutions en forme close.

2.4.1 Définition et premières interprétations

Définition 2.6 (Opérateur proximal (Moreau, 1962))**.** *Soit $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ une fonction convexe fermée et propre, i.e. l'épigraphe de g est un ensemble convexe fermé non vide. L'opérateur proximal $\boldsymbol{prox}_{\mu g} : \mathbb{R}^n \to \mathbb{R}^n$ de la fonction μg, où $\mu > 0$ est défini, pour tout $\mathbf{v} \in \mathbb{R}^n$, par :*

$$\boldsymbol{prox}_{\mu g}(\mathbf{v}) = \underset{\mathbf{x} \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ g(\mathbf{x}) + \frac{1}{2\mu} \|\mathbf{x} - \mathbf{v}\|_2^2 \right\} \tag{2.17}$$

Notons que la fonction à minimiser dans la définition de l'opérateur proximal est strictement convexe et possède donc un unique minimiseur pour tout $\mathbf{v} \in \mathbb{R}^n$. Pour comprendre l'intérêt des méthodes proximales, considérons le problème qui nous intéresse :

$$\underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, \mathcal{J}(\mathbf{c}) = f(\mathbf{c}) + \lambda\Omega(\mathbf{c})$$

Un algorithme itératif pour résoudre (2.1) consiste à chaque itération à linéariser la fonction f au point courant et à résoudre un problème de la forme :

$$\underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, f(\mathbf{c}^{(m)}) + \nabla f(\mathbf{c}^{(m)})^T \left(\mathbf{c} - \mathbf{c}^{(m)}\right) + \lambda\Omega(\mathbf{c}) + \frac{L}{2}\left\|\mathbf{c} - \mathbf{c}^{(m)}\right\|_2^2 \tag{2.18}$$

Le terme quadratique assure que le prochain itéré demeure dans un voisinage du point courant $\mathbf{c}^{(m)}$, $L > 0$ est une constante de Lipschitz pour ∇f, le gradient de f. Le problème (2.18) peut aussi se réécrire de la manière suivante :

$$\underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, \frac{1}{2}\left\|\mathbf{c} - \left(\mathbf{c}^{(m)} - \frac{1}{L}\nabla f(\mathbf{c}^{(m)})\right)\right\|_2^2 + \frac{\lambda}{L}\Omega(\mathbf{c}) \tag{2.19}$$

Or (2.19) est exactement le problème proximal :

$$\mathbf{prox}_{\frac{\lambda}{L}\Omega}\left(\mathbf{c}^{(m)} - \frac{1}{L}\nabla f(\mathbf{c}^{(m)})\right) \tag{2.20}$$

Remarquons que lorsque le terme non lisse Ω est absent, la solution de (2.19) consiste en une étape de gradient classique : $\mathbf{c}^{(m+1)} := \mathbf{c}^{(m)} - \frac{1}{L}\nabla f(\mathbf{c}^{(m)})$, ce qui suggère une relation étroite entre les opérateurs proximaux et les méthodes de gradient.

Revenons au cas général et à la définition de l'opérateur proximal (2.17). Lorsque g est la fonction indicatrice d'un ensemble convexe fermé non vide E, i.e.

$$\iota_E(\mathbf{x}) = \begin{cases} 0 & \mathbf{x} \in E \\ +\infty & \mathbf{x} \notin E \end{cases}$$

l'opérateur proximal correspondant est la projection euclidienne sur E :

$$\mathbf{prox}_{\iota_E}(\mathbf{v}) = \Pi_E(\mathbf{v}) = \underset{\mathbf{x}\in E}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{v}\|_2 \tag{2.21}$$

C'est donc à juste titre que les opérateurs proximaux peuvent être considérés comme des projections généralisées.

2.4.2 Propriétés fondamentales de l'opérateur proximal

Nous étudions ensuite trois propriétés fondamentales de l'opérateur proximal. La première d'entre elles fait le lien entre les opérateurs proximaux et la théorie sur les points fixes.

Proposition 2.7 (Point fixe)**.** $\mathbf{x}^\star$ *réalise l'infimum de* g *si, et seulement si,* $\mathbf{x}^\star$ *est un point fixe de* $\boldsymbol{prox}_g$ *:*

$$\mathbf{x}^\star \in \operatorname{argmin} g \Leftrightarrow \mathbf{x}^\star = \boldsymbol{prox}_g(\mathbf{x}^\star) \tag{2.22}$$

Démonstration. Premièrement, supposons que $\mathbf{x}^\star$ minimise g, i.e. pour tout $\mathbf{x}$, $g(\mathbf{x}) \geq g(\mathbf{x}^\star)$, on a alors :

$$g(\mathbf{x}) + \frac{1}{2}\left\|\mathbf{x}-\mathbf{x}^\star\right\|_2^2 \geq g(\mathbf{x}^\star) = g(\mathbf{x}^\star) + \frac{1}{2}\left\|\mathbf{x}^\star-\mathbf{x}^\star\right\|_2^2, \quad \text{pour tout } \mathbf{x}$$

Donc $\mathbf{x}^\star$ minimise la fonction $\mathbf{x} \mapsto g(\mathbf{x}) + \frac{1}{2}\left\|\mathbf{x}-\mathbf{x}^\star\right\|_2^2$ et on a bien $\mathbf{x}^\star = \mathbf{prox}_g(\mathbf{x}^\star)$.

Réciproquement, soit $\mathbf{x}^\star$ tel que $\mathbf{x}^\star = \mathbf{prox}_g(\mathbf{x}^\star)$, i.e. $\mathbf{x}^\star$ minimise la fonction $G : \mathbf{x} \mapsto g(\mathbf{x}) + \frac{1}{2}\left\|\mathbf{x}-\mathbf{x}^\star\right\|_2^2$. Nous allons utiliser la caractérisation du minimum d'une fonction convexe par le sous-différentiel : un point $\tilde{\mathbf{x}}$ minimise G si, et seulement si,

$$0 \in \partial G(\tilde{\mathbf{x}}) = \partial g(\tilde{\mathbf{x}}) + (\tilde{\mathbf{x}} - \mathbf{x}^\star)$$

où pour $\mathbf{x} \in \mathbb{R}^n$, $\partial g(\mathbf{x}) \subseteq \mathbb{R}^n$ est le sous-différentiel de g en $\mathbf{x}$, défini par :

$$\partial g(\mathbf{x}) = \left\{\mathbf{y} \in \mathbb{R}^n \mid g(\mathbf{z}) \geq g(\mathbf{x}) + \mathbf{y}^T(\mathbf{z}-\mathbf{x}), \quad \text{pour tout } \mathbf{z} \in \mathbb{R}^n\right\} \tag{2.23}$$

En prenant $\tilde{\mathbf{x}} = \mathbf{x}^\star$, il vient alors : $0 \in \partial g(\mathbf{x}^\star)$ et on en conclut que $\mathbf{x}^\star$ est un minimiseur de g. □

Comme les minimiseurs de g sont les points fixes de $\mathbf{prox}_g$, il suffit de trouver un point fixe de g pour minimiser g. Si $\mathbf{prox}_g$ était une contraction, c'est-à-dire une application lipschitzienne de rapport strictement inférieur à 1, alors appliquer itérativement $\mathbf{prox}_g$ permettrait de trouver un tel point fixe. Cependant, en général, $\mathbf{prox}_g$ n'est pas une contraction, mais possède une autre propriété.

Proposition 2.8. *L'opérateur* $\boldsymbol{prox}_g$ *est une application* fermement non expansive*, i.e.*

$$\forall (\mathbf{x},\mathbf{z}) \in \mathbb{R}^n \times \mathbb{R}^n, \ \left\|\boldsymbol{prox}_g(\mathbf{x}) - \boldsymbol{prox}_g(\mathbf{z})\right\|_2^2 \leq (\mathbf{x}-\mathbf{z})^T\left(\boldsymbol{prox}_g(\mathbf{x}) - \boldsymbol{prox}_g(\mathbf{z})\right) \tag{2.24}$$

Les opérateurs fermement non expansifs sont des cas particuliers des opérateurs non expansifs (applications lipschtziennes de rapport 1). L'itération d'opérateurs non expansifs

n'aboutit pas nécessairement à un point fixe, il suffit de considérer l'exemple des rotations ou de $-Id$. Toutefois, si N est un opérateur non expansif, alors l'opérateur T défini par :

$$T = (1-\alpha)Id + \alpha N, \quad \text{pour } \alpha \in (0,1) \tag{2.25}$$

possède les mêmes points fixes que N et une itération de l'opérateur T converge vers un point fixe de T et donc de N. En d'autres termes, le schéma itératif suivant :

$$\mathbf{x}^{(m+1)} = (1-\alpha)\mathbf{x}^{(m)} + \alpha N(\mathbf{x}^{(m)}) \tag{2.26}$$

converge vers un point fixe de N. Les opérateurs de la forme (2.25) sont appelés opérateurs *α-moyennés*. Les opérateurs fermement non expansifs sont moyennés, plus précisément, ce sont des opérateurs (1/2)-moyennés. La justification de la convergence de schémas itératifs fondés sur des opérateurs moyennés est donnée par le théorème suivant :

Théorème 2.1 (Théorème de Krasnoselskii-Mann (Mann, 1953))**.** *Soit T un opérateur moyenné défini par (2.25). Si T possède au moins un point fixe, alors la méthode itérative :*

$$\mathbf{x}^{(0)} \in \mathbb{R}^n$$
$$\mathbf{x}^{(m+1)} := T(\mathbf{x}^{(m)})$$

est convergente : $\left\|\mathbf{x}^{(m)} - T(\mathbf{x}^{(m)})\right\| \xrightarrow[m\to\infty]{} 0$ et la suite $(\mathbf{x}^{(m)})$ converge vers un point fixe de T.

Une seconde propriété importante de l'opérateur proximal concerne son lien avec l'opérateur sous-différentiel défini en (2.23). L'opérateur sous-différentiel ∂g d'une fonction g convexe fermée propre et sous-différentiable est une relation sur $\mathbb{R}^n$ qui à tout $\mathbf{x} \in \mathbf{dom}\ g$ associe l'ensemble sous-différentiel $\partial g(\mathbf{x})$. Tout point $\mathbf{y} \in \partial g(\mathbf{x})$ est appelé un *sous-gradient* de g en $\mathbf{x}$. Lorsque g est différentiable, l'on a : $\partial g(\mathbf{x}) = \{\nabla g(\mathbf{x})\}$ pour tout $\mathbf{x}$.

Proposition 2.9 (Résolvante du sous-différentiel)**.** *Soit $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ une fonction convexe fermée propre et sous-différentiable. L'opérateur proximal $\boldsymbol{prox}_{\mu g}$ et l'opérateur sous-différentiel ∂g sont liés par la relation suivante :*

$$\boldsymbol{prox}_{\mu g} = (Id + \mu\partial g)^{-1} \tag{2.27}$$

La relation $(Id + \mu\partial g)^{-1}$ est appelée *résolvante du sous-différentiel* ∂g de paramètre $\mu > 0$. En d'autres termes, l'opérateur proximal est la résolvante de l'opérateur sous-différentiel.

Démonstration. Soit $\mathbf{x} \in \mathbf{dom}\ g$, alors

$$\begin{aligned}
\mathbf{z} \in (Id + \mu\partial g)^{-1}(\mathbf{x}) &\Leftrightarrow \mathbf{x} \in (Id + \mu\partial g)(\mathbf{z}) = \mathbf{z} + \mu\partial g(\mathbf{z}) \\
&\Leftrightarrow 0 \in \partial g(\mathbf{z}) + \frac{1}{\mu}(\mathbf{z} - \mathbf{x}) \\
&\Leftrightarrow 0 \in \partial_{\mathbf{z}} \left(g(\mathbf{z}) + \frac{1}{2\mu} \|\mathbf{z} - \mathbf{x}\|_2^2 \right) \\
&\Leftrightarrow \mathbf{z} \in \underset{\mathbf{v}}{\operatorname{argmin}} \left\{ g(\mathbf{v}) + \frac{1}{2\mu} \|\mathbf{v} - \mathbf{x}\|_2^2 \right\}
\end{aligned}$$

On vient donc de montrer que $\mathbf{z} \in (Id + \mu\partial g)^{-1}(\mathbf{x})$ si et seulement si $\mathbf{z} = \mathbf{prox}_{\mu g}(\mathbf{x})$, ce qui amène le résultat. □

La dernière propriété que nous exposons ici met en exergue le lien entre opérateurs proximaux et la notion de dualité.

Proposition 2.10 (Décomposition de Moreau)**.** *Soit $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ une fonction convexe fermée propre. La* décomposition de Moreau *de $\mathbf{v} \in \mathbb{R}^n$ est donnée par :*

$$\mathbf{v} = \boldsymbol{prox}_g(\mathbf{v}) + \boldsymbol{prox}_{g^*}(\mathbf{v}) \tag{2.28}$$

où g^* est la transformée de Fenchel-Legendre ou conjuguée de g définie, pour tout $\mathbf{y} \in \mathbb{R}^n$, par : $g^*(\mathbf{y}) = \sup_{\mathbf{x} \in \mathbb{R}^n} \left\{ \mathbf{y}^T\mathbf{x} - g(\mathbf{x}) \right\}$.

La décomposition de Moreau peut être vue comme une généralisation de la décomposition orthogonale induite par un sous-espace. En effet, soit F un sous-espace de $\mathbb{R}^n$, le complément orthogonal est le sous-espace $F^\perp$ défini par : $F^\perp = \left\{ \mathbf{y} \in \mathbb{R}^n \mid \mathbf{y}^T\mathbf{x} = 0, \text{ pour tout } \mathbf{x} \in F \right\}$. En remarquant que $(\iota_F)^* = \iota_{F^\perp}$, où l'on rappelle que ι_F est l'indicatrice de F, l'on a, d'après (2.28), pour tout $\mathbf{v} \in \mathbb{R}^n$,

$$\begin{aligned}
\mathbf{v} &= \mathbf{prox}_{\iota_F}(\mathbf{v}) + \mathbf{prox}_{(\iota_F)^*}(\mathbf{v}) \\
&= \Pi_F(\mathbf{v}) + \Pi_{F^\perp}(\mathbf{v})
\end{aligned}$$

Et l'on obtient bien le résultat classique de la décomposition orthogonale.

2.4.3 Algorithmes proximaux

Algorithme du point proximal

Les opérateurs proximaux étant moyennés, la méthode proximale la plus simple est une application directe du théorème de Krasnoselskii-Mann (2.1), appelée *algorithme du point proximal* et décrite ci-dessous.

Algorithme 2.3 Algorithme du point proximal

Entrées : $\mathbf{x}^{(0)} \in \mathbb{R}^n$; $\mu > 0$
Initialisation : $m := 0$; `stop` :=`faux`
tant que `stop`=`faux` **faire**
0. $m := m + 1$
1. $\mathbf{x}^{(m)} := \mathbf{prox}_{\mu g}(\mathbf{x}^{(m-1)})$
2. `stop` := « critère d'arrêt satisfait ? »
fin tant que
Sorties : $\mathbf{x}^{(m)}$

Une variante possible de l'algorithme du point proximal consiste à modifier la valeur du paramètre μ à chaque itération. La convergence de l'algorithme est garantie dès lors que la suite $(\mu^{(m)})$ est positive et que $\sum_{m=0}^{\infty} \mu^{(m)} = \infty$. La méthode du point proximal s'avère peu utilisée en pratique. En effet, à chaque itération, il faut minimiser la fonction g plus un terme quadratique, de sorte que cet algorithme n'a de véritable intérêt que lorsque minimiser g (objectif initial) est difficile et minimiser g plus un terme quadratique est plus facile.

Algorithme de gradient proximal

Le prochain algorithme justifie l'intérêt que nous portons aux méthodes proximales. Considérons le problème général :

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\operatorname{argmin}} F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \tag{2.29}$$

où $f : \mathbb{R}^n \to \mathbb{R}$ et $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ sont deux fonctions convexes fermées propres. f est *differentiable* et g est éventuellement *non lisse*. Avant d'aller plus loin, il est à noter que le problème de minimisation (2.1) qui nous intéresse est un cas particulier du problème (2.29), où la fonction f représente un coût d'attache aux données et $g = \lambda\Omega$ est un terme de régularisation non lisse qui induit de la parcimonie.

Lorsque l'on décompose une fonction F convexe en deux termes, de la façon décrite ci-dessus, on dit qu'on a procédé à un *éclatement* (*splitting*, en anglais) de la fonction objectif. La méthode de gradient proximal décrite par l'algorithme 2.4 repose essentiellement sur l'étape de calcul :

$$\mathbf{x}^{(m+1)} := \mathbf{prox}_{\mu^{(m)} g}\left(\mathbf{x}^{(m)} - \mu^{(m)} \nabla f(\mathbf{x}^{(m)})\right) \tag{2.30}$$

Lorsque ∇f est lipschitzienne de rapport L, on peut montrer que la méthode converge à un taux $O(1/m)$ où m est le nombre d'itérations de l'algorithme, si l'on utilise un

Algorithme 2.4 Algorithme de gradient proximal

Entrées : $\mathbf{x}^{(0)} \in \mathbb{R}^n$; $\mu^{(0)} > 0$
Initialisation : $m := 0$; `stop` :=`faux`
tant que `stop`=`faux` **faire**
0. $m := m + 1$
1. $\mathbf{x}^{(m)} := \mathbf{prox}_{\mu^{(m-1)}g}\left(\mathbf{x}^{(m-1)} - \mu^{(m-1)}\nabla f(\mathbf{x}^{(m-1)})\right)$
2. `stop` := « critère d'arrêt satisfait ? »
3. $\mu^{(m)} := \text{update}(\mu^{(m-1)})$
fin tant que
Sorties : $\mathbf{x}^{(m)}$

pas constant $\mu^{(m)} = \mu \in (0, 1/L]$ (Combettes et Pesquet, 2011). Si le rapport L est inconnu, ou coûteux à calculer, les pas $\mu^{(m)}$ peuvent être ajustés à chaque itération par une recherche linéaire (A. Beck et M. Teboulle, 2010).

Quelques cas particuliers sont à noter. Lorsque $g = \iota_{\mathcal{C}}$, l'indicatrice d'un ensemble convexe $\mathcal{C}$, $\mathbf{prox}_{\mu g}$ est la projection sur $\mathcal{C}$ et l'étape donnée par l'équation (2.30) revient à la méthode du gradient projeté. Lorsque $f = 0$, on tombe sur l'algorithme du point proximal. Enfin, quand $g = 0$, on retrouve la méthode de descente de gradient classique.

L'algorithme de gradient proximal peut être interprété comme une méthode de point fixe. En effet, $\mathbf{x}^\star$ est une solution de (2.29) si, et seulement, si :

$$0 \in \nabla f(\mathbf{x}^\star) + \partial g(\mathbf{x}^\star)$$

Pour tout $\mu > 0$, on a les équivalences suivantes (toutes équivalentes à la condition d'optimalité donnée ci-dessus) :

$$\begin{aligned} 0 &\in \mu f(\mathbf{x}^\star) + \mu \partial g(\mathbf{x}^\star) \\ 0 &\in \mu f(\mathbf{x}^\star) - \mathbf{x}^\star + \mathbf{x}^\star + \mu \partial g(\mathbf{x}^\star) \\ \mathbf{x}^\star &= (Id + \mu \partial g)^{-1} (Id - \mu f)(\mathbf{x}^\star) \\ \mathbf{x}^\star &= \mathbf{prox}_{\mu g}(\mathbf{x}^\star - \mu \nabla f(\mathbf{x}^\star)) \end{aligned}$$

On a remplacé le symbole d'appartenance $\in$ par une égalité dans les deux dernières expressions car l'opérateur proximal est mono-valué. Ainsi, $\mathbf{x}^\star$ minimise $f + g$ si, et seulement si, $\mathbf{x}^\star$ est un point fixe de l'opérateur $(Id + \mu\partial g)^{-1}(Id - \mu f)$, ce dernier est appelé *opérateur explicite-implicite* (*forward-backward operator*), la partie explicite correspondant au calcul du gradient de f et la partie implicite au calcul de l'opérateur proximal de g. C'est la raison pour laquelle l'algorithme de gradient proximal est aussi connu sous le nom de méthode d'éclatement explicite-implicite (*forward-backward splitting method*). L'algorithme applique l'opérateur explicite-implicite de manière itérative pour obtenir un point fixe

et donc une solution au problème (2.29). La condition $\mu \in (0, 1/L]$ garantit le caractère moyenné de l'opérateur $(Id + \mu\partial g)^{-1}(Id - \mu f)$ et donc de la convergence de la méthode vers un point fixe.

Versions accélérées. Plusieurs auteurs ont proposé des versions dites « accélérées »de l'algorithme de gradient proximal. Certaines incluent une étape d'*extrapolation* supplémentaire :

$$\begin{aligned} \mathbf{y}^{(m+1)} &:= \mathbf{x}^{(m)} + \omega^{(m)}\left(\mathbf{x}^{(m)} - \mathbf{x}^{(m-1)}\right) \\ \mathbf{x}^{(m+1)} &:= \mathbf{prox}_{\mu^{(m)}g}\left(\mathbf{y}^{(m+1)} - \mu^{(m)}\nabla f(\mathbf{y}^{(m+1)})\right) \end{aligned}$$

où $\omega^{(m)} \in [0, 1)$ est un paramètre d'extrapolation. Les paramètres $\omega^{(m)}$ et $\mu^{(m)}$ doivent être choisis de manière appropriée pour accélérer la convergence de la méthode. Plusieurs stratégies possibles ont été proposées à cet égard (A. Beck et M. Teboulle, 2010 ; L. Vandenberghe, 2010). L'algorithme de gradient proximal peut alors atteindre un taux de convergence $O(1/m^2)$. Nesterov affirme qu'il s'agit d'une méthode optimale du premier ordre car ce taux de convergence ne peut être amélioré (Nesterov, 1983). D'autres versions d'accélération ont aussi été proposées (Nesterov, 2007 ; Tseng, 2008).

Algorithme d'éclatement explicite-implicite généralisé

L'algorithme de gradient proximal est utile en pratique dès lors que l'opérateur proximal de g dans (2.29) est facile (peu coûteux) à calculer, une telle fonction est dite *proximable*. Dans certains cas, la fonction g n'est pas proximable mais peut se présenter sous la forme d'une somme de termes proximables. Ce constat a motivé le travail récent de Raguet et al. (2013) qui considèrent le problème :

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\operatorname{argmin}} F(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^{s} g_i(\mathbf{x}) \tag{2.31}$$

où $f : \mathbb{R}^n \to \mathbb{R}$ et $g_i : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, $(i \in \mathbb{N}_s)$ sont des fonctions convexes fermées propres. f est *differentiable*, ∇f est lipschitzienne de rapport L et chaque g_i, $(i \in \mathbb{N}_s)$ est *proximable*. L'algorithme 2.5 généralise la méthode d'éclatement explicite-implicite originelle.

Le théorème 2.1 dans Raguet et al. (2013) détaille les conditions, en particulier les valeurs des pas $\mu^{(m)}$ et $\eta^{(m)}$, pour lesquelles l'algorithme 2.5 converge vers un minimiseur de (2.31). La méthode a notamment été appliquée avec succès pour l'estimation de matrices avec des contraintes de parcimonie et de faible rang (Richard et al., 2012).

Algorithme 2.5 Algorithme d'éclatement explicite-implicite généralisé

Entrées : $(\mathbf{z}_i^{(0)})_{i=1}^s \in (\mathbb{R}^n)^s$; $\eta^{(0)} \in (0, 2/L)$; $\mu^{(0)} > 0$; $(\omega_i)_{i=1}^s \in (0,1)^s$ avec $\sum_{i=1}^s \omega_i = 1$

Initialisation : $m := 0$; $\mathbf{x}^{(0)} = \sum_{i=1}^s \omega_i \mathbf{z}_i^{(0)}$; `stop` :=`faux`
tant que `stop`=`faux` **faire**
0. $m := m + 1$
1.
pour $i \in \mathbb{N}_s$ **faire**
$\mathbf{z}_i^{(m)} := \mathbf{z}_i^{(m-1)} + \mu^{(m-1)} \left(\mathbf{prox}_{\frac{\eta^{(m-1)}}{\omega_i} g_i} \left(2\mathbf{x}^{(m-1)} - \mathbf{z}_i^{(m-1)} - \eta^{(m-1)} \nabla f(\mathbf{x}^{(m-1)}) \right) - \mathbf{x}^{(m-1)} \right)$
fin pour
2. $\mathbf{x}^{(m)} = \sum_{i=1}^s \omega_i \mathbf{z}_i^{(m)}$
3. `stop` := « critère d'arrêt satisfait ? »
4. $(\eta^{(m)}, \mu^{(m)}) := \text{update}(\eta^{(m-1)}, \mu^{(m-1)})$
fin tant que
Sorties : $\mathbf{x}^{(m)}$

2.4.4 Calcul pratique des opérateurs proximaux

L'efficience des algorithmes présentés réside essentiellement dans la capacité à évaluer efficacement l'opérateur proximal. Par définition, le calcul de l'opérateur proximal consiste à résoudre un problème d'optimisation convexe. Nous avons qualifié de proximable toute fonction pour laquelle l'opérateur proximal peut être calculé de manière efficace. Dans le cas qui nous intéresse, nous allons voir que $g = \Omega$ est proximable lorsque Ω est une norme qui induit de la parcimonie.

Soit $g = \|\cdot\|$ une norme sur $\mathbb{R}^n$, alors la conjuguée de g est $g^* = \iota_{\mathcal{B}}$ où $\mathcal{B}$ est la boule unité pour la norme duale de $\|\cdot\|$. En appliquant la décomposition de Moreau (2.28), on obtient :

$$\begin{aligned} \mathbf{prox}_{\mu g}(\mathbf{v}) &= \mathbf{v} - \mu \mathbf{prox}_{g^*/\mu}(\mathbf{v}/\mu) \\ \mathbf{prox}_{\mu\|\cdot\|}(\mathbf{v}) &= \mathbf{v} - \mu \Pi_{\mathcal{B}}(\mathbf{v}/\mu) \end{aligned} \tag{2.32}$$

Ce résultat montre que l'opérateur proximal d'une norme peut se calculer comme le résidu d'une projection euclidienne sur la boule d'une autre norme (duale).

Norme euclidienne. Par exemple, soit $g = \|\cdot\|_2$, la norme ℓ_2 de $\mathbb{R}^n$, la projection sur la boule unité pour la norme ℓ_2 est :

$$\Pi_{\mathcal{B}_{\ell_2}}(\mathbf{v}) = \begin{cases} \frac{\mathbf{v}}{\|\mathbf{v}\|_2} & \|\mathbf{v}\|_2 > 1 \\ \mathbf{v} & \|\mathbf{v}\|_2 \leq 1 \end{cases}$$

Il s'ensuit :

$$\mathbf{prox}_{\mu\|\cdot\|_2}(\mathbf{v}) = \left(1 - \frac{\mu}{\|\mathbf{v}\|_2}\right)_+ \mathbf{v} = \begin{cases} \left(1 - \frac{\mu}{\|\mathbf{v}\|_2}\right)\mathbf{v} & \|\mathbf{v}\|_2 \geq \mu \\ 0 & \|\mathbf{v}\|_2 < \mu \end{cases}$$

Norme ℓ_1. De la même manière, la norme duale de la norme ℓ_1 est la norme ℓ_∞ et l'on sait que la boule unité $\mathcal{B}_{\ell_\infty}$ de la norme ℓ_∞ est une boîte de sorte que sa projection est définie, coordonnée par coordonnée, pour $i \in \mathbb{N}_n$ par :

$$\left(\Pi_{\mathcal{B}_{\ell_\infty}}(\mathbf{v})\right)_i = \begin{cases} 1 & v_i > 1 \\ v_i & |v_i| \leq 1 \\ -1 & v_i < -1 \end{cases}$$

L'opérateur proximal de la norme ℓ_1 est alors donné, composante par composante, pour $i \in \mathbb{N}_n$ par :

$$\left(\mathbf{prox}_{\mu\|\cdot\|_1}(\mathbf{v})\right)_i = \begin{cases} v_i - \mu & v_i > \mu \\ 0 & |v_i| \leq \mu \\ v_i + \mu & v_i < -\mu \end{cases}$$

Ce dernier est aussi connu sous le nom d'opérateur de *seuillage doux* (*soft thresholding*) $\mathcal{T}_\mu : \mathbb{R}^n \to \mathbb{R}^n$ et défini plus synthétiquement élément par élément par :

$$(\mathcal{T}_\mu(\mathbf{v}))_i = \left(\mathbf{prox}_{\mu\|\cdot\|_1}(\mathbf{v})\right)_i = \left(1 - \frac{\mu}{|v_i|}\right)_+ v_i \tag{2.33}$$

Régularisation filet élastique. Ces exemples peuvent être combinés astucieusement. Pour illustrer cela, nous allons considérer la régularisation *filet élastique* (*elastic net*) (H. Zou et T. Hastie, 2005) qui est la combinaison linéaire d'une pénalité en norme ℓ_1 et d'une pénalité quadratique en norme ℓ_2 et définie par :

$$g(\mathbf{x}) = \|\mathbf{x}\|_1 + \frac{\gamma}{2}\|\mathbf{x}\|_2^2 \tag{2.34}$$

avec $\gamma > 0$. g n'est pas une norme mais son opérateur proximal peut être obtenu sous forme close :

$$\mathbf{prox}_{\mu\left(\|\cdot\|_1 + \frac{\gamma}{2}\|\cdot\|_2^2\right)}(\mathbf{v}) = \mathbf{prox}_{\frac{\mu\gamma}{2}\|\cdot\|_2^2} \circ \mathbf{prox}_{\mu\|\cdot\|_1}(\mathbf{v}) = \frac{1}{1+\mu\gamma}\mathbf{prox}_{\mu\|\cdot\|_1}(\mathbf{v})$$

Norme mixte ℓ_1/ℓ_2. Soit $\mathcal{G}$ une partition de $\mathbb{N}_n$, la norme duale de la norme mixte ℓ_1/ℓ_2 $\Omega : \mathbf{x} \mapsto \sum_{g\in\mathcal{G}} \|\mathbf{x}_g\|_2$ est la norme ℓ_∞/ℓ_2. On peut montrer que la projection orthogonale sur la boule unité de la norme ℓ_∞/ℓ_2 est réalisée en projetant séparément chaque sous-vecteur $\mathbf{x}_g$ ($g \in \mathcal{G}$) sur la boule unité de la norme ℓ_2 dans $\mathbb{R}^{|g|}$. L'on a alors, pour tout

$g \in \mathcal{G}$:

$$\Big(\mathbf{prox}_{\mu\Omega}(\mathbf{v})\Big)_g = \left(1 - \frac{\mu}{\|\mathbf{v}_g\|_2}\right)_+ \mathbf{v}_g$$

Cet opérateur est parfois appelé opérateur de seuillage doux par bloc.

2.5 Synthèse

Après avoir fait quelques rappels d'optimisation convexe, nous avons introduit le concept central de parcimonie. Dans le cadre d'une régularisation en norme parcimonieuse, nous avons mis en exergue les méthodes proximales qui sont des algorithmes pertinents lorsque le problème d'optimisation d'intérêt est composé d'un coût différentiable et d'un ou plusieurs termes non lisses. Pour les modèles que nous définissons, l'estimation des paramètres est réalisée par le biais d'algorithmes d'inspiration « proximale ».

Chapitre 3

Modélisation de systèmes dynamiques

Sommaire

3.1 Introduction 51
3.2 Généralités sur les systèmes dynamiques 52
3.3 Modèles pour la prévision de séries temporelles 54
3.3.1 Modèles scalaires 54
3.3.2 Modèles vectoriels 62
3.4 Modèles pour l'inférence de réseaux 66
3.4.1 Problématique de l'inférence de réseaux 66
3.4.2 Données statiques 67
3.4.3 Données de séries temporelles 69
3.5 Synthèse 70

3.1 Introduction

En toute généralité, les *systèmes dynamiques* sont des objets mathématiques utilisés pour modéliser des phénomènes physiques dont l'*état* ou *description instantanée* change au cours du temps. Ces systèmes sont présents dans toute une série d'applications : les modèles de systèmes dynamiques servent notamment aux prévisions économiques et financières, aux diagnostics médicaux, à l'analyse du vieillissement d'équipements industriels. Ces exemples, loin d'être exhaustifs, sont donnés à titre indicatif pour montrer l'omniprésence des systèmes dynamiques. La plupart de ces applications peuvent être classées en trois catégories selon leur finalité :

1. la *prédiction* ou la *prévision* (on parle aussi de finalité générative), dont l'objectif est de prédire les états futurs d'un système à partir d'observations des états présents et passés (*historique*) ;
2. le *diagnostic* qui consiste à expliquer l'état présent du système en inférant quels sont les états passés qui ont conduit à l'état présent du système ;

3. l'*extraction de connaissances*, dont la visée est de fournir une explication de la dynamique du système en explicitant les relations éventuelles entre les entités le composant.

Nous nous intéressons particulièrement aux tâches de prédiction et d'extraction de connaissances. L'objet de ce chapitre consiste à donner des éléments généraux sur les systèmes dynamiques puis à présenter les modèles existants pour l'extraction de connaissances et la prévision de séries temporelles.

3.2 Généralités sur les systèmes dynamiques

Définition 3.1. *Un* système dynamique *est la donnée :*

- *d'un espace d'états S ;*
- *d'un ensemble T, modélisant le temps ;*
- *d'une règle d'évolution $h : S \to S$ qui détermine les états successeurs à un état $s \in S$ donné.*

Espace d'états. Un état $s \in S$ est une collection de coordonnées ou caractéristiques qui permettent de décrire de manière adéquate le système. L'espace d'états est l'ensemble des états possibles du système. Étant donné l'état courant du système $s \in S$, la règle d'évolution h prédit le ou les prochains états. Le modèle peut aussi dépendre de paramètres (constantes) qu'il faut estimer s'ils ne sont pas donnés *a priori*.

Un espace d'états peut être soit discret soit continu. Un exemple de système à espace d'états discret est donné par le lancer de pièce, modélisé par deux états, *pile* noté `P` ou *face* noté `F` pour lequel tout état à un temps quelconque prend sa valeur dans l'espace d'états $S = \{\texttt{P}, \texttt{F}\}$.

Lorsque l'espace d'états est continu, il s'agit la plupart du temps d'une variété et l'on parle alors d'espace des phases. Par exemple, un pendule simple est une masse ponctuelle fixée à l'extrémité d'une tige rigide de masse nulle pouvant tourner sans frottement dans un plan vertical autour de son extrémité fixe. Sous l'effet de son poids, lorsque le pendule est écarté de sa position d'équilibre (la verticale), ce dernier se met à osciller. Selon les lois de Newton, le connaissance de l'angle θ que fait la tige par rapport à la verticale et de la vitesse angulaire $\nu = d\theta/dt$ est suffisante pour décrire l'état du pendule. Ainsi, l'espace des phases du pendule, constitué de toutes les valeurs possibles de couples (θ, ν) est une variété de dimension deux qui s'avère être un cylindre puisque θ est périodique. En outre, le modèle dépend de deux paramètres : l'intensité de la pesanteur et la longueur de la tige. Un espace des phases peut aussi être de dimension infinie, c'est notamment le cas de systèmes dont la dynamique est modélisée par un ensemble d'équations différentielles

partielles, l'espace des phases associé est alors un espace fonctionnel.

Le temps. À l'instar de l'espace d'états, le temps peut aussi être discret ($T = \mathbb{Z}$) ou continu ($T = \mathbb{R}$).

Règle d'évolution. La règle d'évolution h fournit une prédiction de la valeur des états consécutifs à un état courant. On dit que la règle d'évolution est *déterministe* si on peut faire correspondre à un état donné un unique état successeur et *stochastique* ou aléatoire s'il existe plusieurs états successeurs possibles à un état donné. La trajectoire ou orbite d'un état $s \in S$ est la suite ordonnée d'états qui succèdent à s et obtenus par application de la règle d'évolution. Si l'état initial est noté s_0 et et que le système obéit à une règle déterministe à temps discret, alors il est d'usage de noter la trajectoire résultante $s_0, s_1, s_2, \ldots$. Quand l'espace d'états et le temps sont continus, l'orbite est une courbe $s(t), t > 0$.

Lorsqu'on dispose d'une connaissance experte sur le processus d'intérêt, on peut imposer la forme du modèle h à utiliser, on parle de modèle *paramétrique*. La plupart du temps, cette connaissance fait défaut et l'on ne peut pas *a priori* privilégier une forme de modèle. Dans ces modèles dits *non paramétriques*, le nombre et la nature des paramètres ne sont plus fixés à l'avance mais dépendent des données. Dans les deux cas, comme nous l'avons fait remarquer précédemment, les paramètres du modèle sont souvent des inconnues qu'il convient d'inférer à partir de l'observation empirique d'une partie de la trajectoire résultant d'un état initial $s_0 \in S$.

Dans le reste du mémoire, nous considérons des systèmes dynamiques à temps **discret**, les données observées sont alors en nombre fini et prennent la forme d'une *série temporelle discrète* $s_{0:N} = \{s_0, s_1, \ldots, s_N\}$ correspondant à $N + 1$ observations consécutives du système.

Considérons, à titre d'exemple, la question de l'estimation des paramètres dans les modèles de réseaux de régulation génique. Nous reviendrons plus en détail sur ces systèmes dans la section suivante. Généralement, les systèmes d'équations différentielles ordinaires (ODE pour *Ordinary Differential Equations*) sont un formalisme répandu en biologie computationnelle pour modéliser l'évolution des niveaux d'expression des gènes au sein d'un réseau. Spécifiquement le niveau d'expression $x^i(t)$ d'un gène i est déterminé par une équation de la forme :

$$\frac{d}{dt}x^i(t) = h^i\left(x^1(t), \ldots, x^d(t); \Theta^i\right) \tag{3.1}$$

où d est le nombre de variables (gènes) qui composent le système (réseau) et Θ^i est l'ensemble des paramètres du modèle d'ODE h^i pour la variable i. Le problème de l'estimation de paramètres consiste ici à inférer les d ensembles de paramètres $\Theta^1, \ldots, \Theta^d$ à partir de données de séries temporelles bruitées.

Dynamique d'un système. Soient $\Theta, \mathcal{X}$ et $\mathcal{N}$ des espaces polonais (i.e. des espaces métriques complets qui possèdent un ensemble dense dénombrable, c'est une hypothèse classique en modélisation de systèmes dynamiques), chacun muni de sa σ-algèbre borélienne. L'espace Θ correspond à l'espace des paramètres, le système dynamique sous-jacent évolue dans l'espace d'états $\mathcal{X}$. La dynamique du système est régie par les équations suivantes :

$$\begin{cases} X_0 \sim \mu, \\ X_{t+1} = h(\theta; X_t) + u_{t+1}, \ \theta \in \Theta \end{cases} \tag{3.2}$$

X_0 est une condition initiale tirée suivant μ, une mesure de probabilité sur $\mathcal{X}$, $u_t \in \mathcal{N}$ est une variable aléatoire qui modélise un bruit. La fonction $h : \Theta \times \mathcal{X} \to \mathcal{X}$ détermine l'évolution de la dynamique dans l'espace d'états. Nous définissons ci-dessous quelques caractéristiques des systèmes dynamiques.

Définition 3.2. *On dit que le processus stochastique $(X_t)_t$ est* stationnaire *si pour tous $\ell, m \in \mathbb{N}$ et $n_1, \dots, n_\ell \in \mathbb{N}$, la distribution jointe du ℓ-uplet $(X_{n_1+m}, \dots, X_{n_\ell+m})$ est égale à la distribution jointe de $(X_{n_1}, \dots, X_{n_\ell})$.*

Définition 3.3. *Un processus stochastique stationnaire $(X_t)_t$ à valeurs dans $\mathcal{X}$ est* ergodique *si pour tout $\ell \geq 1$ et pour tout couple de boréliens $(A, B) \in \mathcal{X}^\ell \times \mathcal{X}^\ell$, on a :*

$$\begin{aligned} \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{P}\left((X_1, \dots, X_\ell) \in A, \ (X_{k+1}, \dots, X_{k+\ell}) \in B\right) \\ = \mathbb{P}\left((X_1, \dots, X_\ell) \in A\right) \mathbb{P}\left((X_1, \dots, X_\ell) \in B\right) \end{aligned} \tag{3.3}$$

3.3 Modèles pour la prévision de séries temporelles

3.3.1 Modèles scalaires

Dans cette sous-section, nous considérons des séries temporelles unidimensionnelles.

Modèles classiques : AR, ARMA.

L'approche standard en modélisation de systèmes dynamiques se fonde sur l'hypothèse d'une relation linéaire entre la variable expliquée et ses valeurs passées (Yule, 1927) :

$$x_{t+1} = \sum_{i=1}^{p} \beta_i x_{t+1-i} + u_{t+1} \tag{3.4}$$

Le modèle donné par l'équation (3.4) est un modèle autorégressif linéaire d'ordre p, noté usuellement AR(p). Pour garantir la stationnarité du processus, les racines du polynôme $P(Z) = Z^p - \sum_{i=1}^{p} \beta_i Z^{p-i}$ doivent être dans le cercle unité, i.e. chaque racine de P doit

vérifier $|z_i| < 1$. L'estimation des paramètres β_i peut être réalisée via la méthode des moindres carrés ou une approche par maximum de vraisemblance. Toutefois la méthode consacrée se fonde sur les équations de Yule-Walker (Hamilton, 1994). Ces dernières mettent en jeu les autocovariances et les fonctions d'autocorrélation et permettent une estimation efficace des paramètres du modèle.

Au modèle AR(p) peut s'ajouter un processus dit de *moyenne mobile*. On obtient alors le modèle général suivant (Whitle, 1951) :

$$x_{t+1} = \sum_{i=1}^{p} \beta_i x_{t+1-i} + \sum_{j=1}^{q} \theta_j u_{t+1-j} + u_{t+1} \tag{3.5}$$

Les termes $\theta_j u_{t+1-j}$ viennent de l'idée que l'observation au temps $t+1$ peut aussi s'expliquer par une somme pondérée de chocs passés qui se propagent aux valeurs futures de la série temporelle, ces chocs étant représentés par un bruit blanc de variance finie. Le modèle décrit par l'équation (3.5) est appelé processus autorégressif à moyenne mobile d'ordre (p, q) et noté ARMA(p, q) (*AutoRegressive Moving Average*). Une bonne pratique pour la sélection de modèle (choix des ordres p et q dans les modèles ARMA) et l'estimation des paramètres a été proposée par Box et al. (1970).

Il est toutefois à noter que les modèles ARMA présentent quelques limitations, notamment les hypothèses de stationnarité du processus et de linéarité des dépendances. Ce constat a permis l'émergence d'une littérature autour de l'analyse de séries temporelles non linéaires. Les modèles non linéaires les plus populaires comprennent les modèles TAR (*Threshold AutoRegressive*) (Tong, 1978 ; Tong, 1983), EXPAR (*EXPonential AutoRegressive*) de Haggan et Ozaki (1981), STAR (*Smooth-Transition AutoRegressive*) de Chan et Tong (1986) et de C. W. Granger et Terasvirta (1993), les modèles bilinéaires de C. W. J. Granger et Andersen (1978), Rao (1981) et Rao et Gabr (1984), les modèles à coefficients aléatoires de Nicholls et Quinn (1983), les modèles ARCH (*AutoRegressive Conditional Heteroscedastic*) de Engle (1982) et les modèles GARCH (*Generalized ARCH*) de Bollerslev (1986) et Bera et Higgins (1993).

Modèles issus de l'apprentissage automatique.

La modélisation de séries temporelles a été longtemps le domaine d'étude quasi-exclusif des statisticiens et en particulier des économètres. Cependant, ces dernières décennies, des méthodes d'apprentissage ont été appliquées avec succès sur des problèmes de modélisation de systèmes dynamiques et notamment sur des tâches de prévisions de séries temporelles (Ahmed et al., 2010). En effet, une série temporelle discrète observée $x_{1:N+1} = \{x_1, x_2, \ldots, x_{N+1}\}$ peut être transformée avantageusement en un ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{z}_\ell, x_{\ell+1})\}$ où typiquement $\mathbf{z}_\ell$ est un vecteur qui contient des valeurs

de la série temporelle antérieures à $x_{\ell+1}$. C'est donc un cas particulier de la régression où l'on essaie d'expliquer une variable d'intérêt à partir des valeurs passées de cette même variable. L'on qualifie de *modèle autorégressif* tout modèle de la forme :

$$x_{t+1} = h(\mathbf{z}_t) + u_{t+1} \tag{3.6}$$

En considérant la modélisation de systèmes dynamiques sous l'angle de la régression, l'on peut alors déployer à propos des méthodes issues de l'apprentissage statistique. Nous décrivons ci-après un corpus de modèles utilisés dans la littérature.

Réseaux de neurones. Les réseaux de neurones artificiels (McCulloch et Pitts, 1943) font partie des modèles les plus populaires, que ce soit pour des problèmes de classification ou de régression (Bishop et al., 1995). Un exemple de réseau de neurones est celui du perceptron multi-couches (*Multilayer Perceptron*, MLP) (Rosenblatt, 1958) dont l'architecture se présente de la manière suivante :

- un réseau organisé en plusieurs couches : chaque couche est constituée d'un ensemble de neurones formels sans connexion les uns avec les autres
- chaque couche réalise une transformation vectorielle
- les dimensions d'entrée et de sortie peuvent être différentes
- toute couche distincte de la couche de sortie est appelée couche cachée et les neurones qui la composent sont également appelés neurones cachés
- la structure du réseau est sans cycle (*feed-forward*) : une couche ne peut utiliser que les sorties de la couche précédente

Un neurone formel reçoit n_e entrées $x_1, \ldots, x_{n_e}$ et réalise une opération particulière :

$$f_{\text{act}}\left(w_0 + \sum_{i=1}^{n_e} w_i x_i\right)$$

f_{act} est appelée *fonction d'activation*. Les choix classiques pour la fonction d'activation sont la tangente hyperbolique et la fonction logistique $x \mapsto 1/(1+e^{-x})$. À chaque neurone sont associés $n_e + 1$ paramètres : n_e poids synaptiques $w_1, \ldots, w_{n_e}$ et un biais w_0. Une méthode dédiée pour apprendre ces paramètres est la technique de *rétropropagation du gradient* (Rumelhart et al., 1988) qui consiste à calculer le gradient de l'erreur pour chaque neurone du réseau, de la dernière couche vers la première.

Dans le contexte des systèmes dynamiques, il est souhaitable de relâcher la contrainte de structure sans cycle du MLP. Le réseau de neurones récurrent (*Recurrent Neural Network*, RNN) est une classe de réseaux dont l'architecture autorise les neurones à se connecter de sorte à former au moins un circuit (*feedback loop*), cette caractéristique leur permet de posséder une mémoire interne et rend donc leur application possible dans des tâches de

prévisions de séries temporelles (Connor et al., 1994 ; Dorffner, 1996 ; G. Zhang et al., 1998). Ces modèles ont connu un nouvel essor avec notamment l'émergence du paradigme d'apprentissage profond (*deep learning*) (Bengio, 2009 ; Pascanu et al., 2013).

L'une des qualités majeures des réseaux de neurones qui explique leur popularité réside dans leur puissance d'expressivité, c'est-à-dire la propriété d'approximation universelle (Hornik et al., 1989). Ce résultat, bien que positif, doit attirer la vigilance de l'utilisateur afin d'éviter une sur-paramétrisation du modèle, particulièrement dans le cas de prévisions où n'est disponible qu'un nombre limité de données hautement bruitées. L'on peut contrôler la complexité d'un réseau de neurones en sélectionnant le nombre de couches et de neurones cachés. Teräsvirta et al. (2005) ont apporté quelques idées appréciables dans la manière de construire des modèles de réseaux de neurones parcimonieux.

Modèles non paramétriques précurseurs. Nous nous intéressons à présent à des modèles non paramétriques qui généralisent l'estimateur de Nadaraya-Watson au cas dépendant. Rappelons brièvement la construction de cet estimateur. Soient X une variable aléatoire réelle de densité f_X et $x_1, \dots, x_N$ des réalisations de X. L'estimateur non paramétrique de la densité f_X, aussi appelé estimateur de Parzen–Rosenblatt, est donné par :

$$\hat{f}_X(x) = \frac{1}{N\delta_N} \sum_{i=1}^{N} K\left(\frac{x_i - x}{\delta_N}\right)$$

Ici, K est une fonction noyau, i.e. une densité (fonction positive intégrable qui somme à 1) symétrique, la largeur de bande δ_N détermine la taille du voisinage à considérer pour l'estimation. Ainsi, si la fonction noyau a pour support $[-1, 1]$, l'estimateur utilise uniquement les observations dans l'intervalle $[x - \delta_N, x + \delta_N]$. La largeur de bande δ_N dépend de la taille de l'échantillon N et doit satisfaire les conditions suivantes de manière à garantir la consistance de l'estimation : $\delta_N \to 0$ et $N\delta_N \to \infty$ lorsque $N \to \infty$.

On considère maintenant un couple de variables aléatoires réelles (X, Y) de densité jointe $f_{X,Y}$. La densité marginale de X est notée f_X. En utilisant les estimateurs non paramétriques de $f_{X,Y}$ et f_Y dans l'expression de la fonction de régression : $\mathbb{E}[Y|X = x] = \int y \frac{f_{X,Y}(x,y)}{f_X(x)} dy$, il s'ensuit l'estimateur non paramétrique de la fonction de régression :

$$\hat{h}(x) = \frac{\sum_{i=1}^{N} y_i K\left(\frac{x_i - x}{\delta}\right)}{\sum_{j=1}^{N} K\left(\frac{x_j - x}{\delta}\right)} \tag{3.7}$$

(3.7) est appelé estimateur de Nadaraya-Watson (NW) (Nadaraya, 1964 ; Watson,

1964). Cet estimateur peut être vu comme une moyenne pondérée des sorties observées :

$$\hat{h}(x) = \sum_{i=1}^{N} w_i y_i$$

où les poids $w_i,\ i \in \mathbb{N}_N$ sont donnés par :

$$w_i = \frac{K\left(\frac{x_i - x}{\delta}\right)}{\sum_{j=1}^{N} K\left(\frac{x_j - x}{\delta}\right)}$$

Comme nous l'avons fait remarquer, lorsque l'on applique l'estimateur NW au cas de données dépendantes, par exemple pour des données de séries temporelles, ce dernier est affecté uniquement par la dépendance des observations sur une fenêtre de temps et non pas sur l'ensemble des données, de sorte que la plupart des techniques développées pour des données indépendantes restent valides dans ce cas-là. C'est ce que Hart (1996) appelle le *whitening by windowing principle*.

Considérons un processus stationnaire $(X_t)_t$ d'ordre p, on observe la série temporelle $x_1, \ldots, x_N$ puis on forme l'ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{z}_\ell, x_\ell)\}_{\ell=p+1}^{N} \subseteq \mathbb{R}^p \times \mathbb{R}$ avec $\mathbf{z}_\ell = (x_{\ell-p} \ldots x_{\ell-1})^T \in \mathbb{R}^p$, un estimateur de type NW est alors donné par :

$$\hat{h}(\mathbf{z}_t) = \sum_{\ell=p+1}^{N} w_\ell x_\ell \tag{3.8}$$

avec

$$w_\ell = \frac{K\left(\frac{\mathbf{z}_\ell - \mathbf{z}_t}{\delta}\right)}{\sum_{\ell=p+1}^{N} K\left(\frac{\mathbf{z}_\ell - \mathbf{z}_t}{\delta}\right)} \tag{3.9}$$

Lorsque $\mathbf{z}_t$ correspond au dernier motif observé $\mathbf{z}_{N+1} = (x_{N-p+1} \ldots x_N)^T$, $\hat{h}(\mathbf{z}_{N+1})$ fournit une prédiction pour x_{N+1}, on obtient alors une interprétation très intuitive de cet estimateur : la prédiction est une moyenne pondérée de toutes les valeurs passées de la série temporelle qui succèdent à un motif semblable au dernier motif observé. Les poids w_ℓ reflètent à quel point le motif courant est proche des motifs observés dans le passé.

Heiler (1999) fournit une revue détaillée de ces méthodes non paramétriques pour l'analyse des séries temporelles.

k plus proches voisins. La méthode des k plus proches voisins (k *Nearest Neighbors*, kNN) (Altman, 1992) est une approche non paramétrique qui, comme son nom l'indique, produit une prédiction pour un point donné à partir des sorties des k points les plus proches de ce dernier. Plus précisément, soit un point x donné, on calcule la distance (euclidienne, par exemple) entre ce point et tous les autres dans l'ensemble d'apprentissage

$\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. On détermine alors l'ensemble $\text{knn}(\mathbf{x})$ des indices des k points les plus proches de $\mathbf{x}$ dans $\mathcal{S}_N$. La prédiction pour $\mathbf{x}$ est donnée par :

$$\hat{h}(\mathbf{x}) = \frac{1}{k} \sum_{i \in \text{knn}(\mathbf{x})} y_i \tag{3.10}$$

Le choix du nombre de voisins k est un paramètre crucial de la méthode, une valeur de k élevée conduit à un modèle lisse, peu sensible au bruit et donc de faible variance mais au prix d'un biais important, et vice versa pour une petite valeur de k.

SVR. La régression à vecteurs de support (*Support Vector Regression*, SVR)(B. Schölkopf et A. Smola, 2001 ; A. J. Smola et Bernhard Schölkopf, 2004) est une méthode à noyaux qui construit à partir d'un ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathbb{R}$ une hypothèse $h : \mathcal{X} \to \mathbb{R}$ telle que :

- h est de la forme $h(\mathbf{x}) = \omega^T \phi(\mathbf{x}) + b$, où $b \in \mathbb{R}$, ϕ est une fonction de $\mathcal{X}$ dans un espace de caractéristiques $\mathcal{H}_c$, éventuellement de dimension infinie et $\omega \in \mathcal{H}_c$
- pour $i \in \mathbb{N}_N$, on pénalise chaque prédiction $h(\mathbf{x}_i)$ qui dévie de y_i de plus de ϵ, où $\epsilon > 0$ est un paramètre à fixer
- h est aussi « plat » que possible

Les deux deniers points peuvent être formalisés par le problème d'optimisation suivant :

$$\underset{\omega, b}{\text{argmin}}\, \mathcal{J}(\omega, b) = \frac{1}{2} \langle \omega, \omega \rangle_{\mathcal{H}_c} + C \sum_{i=1}^N \ell_\epsilon(y_i, \omega^T \phi(\mathbf{x}_i) + b) \tag{3.11}$$

où ℓ_ϵ est la fonction de coût ϵ-insensible définie par : $\ell_\epsilon(y, \hat{y}) = \max(0, |y - \hat{y}| - \epsilon)$. En d'autres termes, le modèle ne pénalise pas les erreurs si $|y_i - h(\mathbf{x}_i)| \leq \epsilon$. La constante $C > 0$ est un hyperparamètre qui règle le compromis entre l'objectif d'une adéquation du modèle aux données et celui d'obtenir un modèle plat. Comme pour les SVMs dans le cas de la classification, une manière de résoudre (3.11) est de construire la fonction Lagrangienne associée à la fonction objectif $\mathcal{J}$ puis de dériver le problème dual. On peut prouver que le modèle appris est de la forme :

$$\hat{h}(\mathbf{x}) = \sum_{i=1}^N (\alpha_i^* - \alpha_i)\, k(\mathbf{x}_i, \mathbf{x}) + b$$

où $\{(\alpha_i, \alpha_i^*)\}_{i=1}^N$ sont les multiplicateurs de Lagrange et $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est la fonction noyau associée à ϕ : $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}_c}$. Seules les données d'apprentissage correspondant à des multiplicateurs non nuls jouent un rôle dans le modèle $\hat{h}$, ces vecteurs sont appelés *vecteurs de support* et leur nombre est une mesure de la complexité du modèle.

On trouve les premières applications des SVRs pour la prévision de séries temporelles dans Mukherjee et al. (1997) et Müller et al. (1997), voir également Sapankevych et Sankar (2009) pour un panorama plus complet.

Processus gaussiens. Les processus gaussiens (*Gaussian processes*, GP) sont une extension des lois normales multidimensionnelles en dimension infinie. Formellement, un processus stochastique $(Z_t)_{t\in\mathcal{T}}$ est un GP si et seulement si pour tout $m \in \mathbb{N}$ et pour tout ensemble fini d'indices $t_1, \ldots, t_m \in \mathcal{T}$, $(Z_{t_1}, \ldots, Z_{t_m})$ suit une loi normale multidimensionnelle. Souvent, on suppose, sans perte de généralité, que la moyenne du GP est nulle. La régression par processus gaussiens est une méthode non-paramétrique qui part du postulat que le vecteur des réponses observées dans l'ensemble d'apprentissage $\mathbf{y} = (y_1 \ldots y_N)^T \in \mathbb{R}^N$ est la réalisation d'une variable aléatoire normale multidimensionnelle (ici de dimension N) (Rasmussen, 2004). Ce qui relie une observation à une autre est la *fonction de covariance* k. Un choix populaire pour k est donné par :

$$k(\mathbf{x}, \mathbf{x}') = \sigma_h^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\gamma^2}\right)$$

Le choix d'une fonction de covariance détermine un *a priori* sur la fonction $h \sim \mathcal{GP}(0, k)$ à modéliser :

$$y = h(\mathbf{x}) + u$$

Le bruit u lié aux observations est une variable aléatoire gaussienne de moyenne nulle et de variance σ_n^2.

Soit une donnée de test $\mathbf{x}_*$ de sortie associée y_* inconnue, en partant de l'hypothèse que les données sont la réalisation d'une variable aléatoire normale multidimensionnelle, l'on en déduit :

$$\begin{bmatrix}\mathbf{y}\\ y_*\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}K & K_*^T\\ K_* & K_{**}\end{bmatrix}\right)$$

où $K = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^N \in \mathbb{R}^{N\times N}$ est la matrice de covariance associée aux données d'apprentissage, $K_* = \begin{bmatrix}k(\mathbf{x}_*, \mathbf{x}_1) & \ldots & k(\mathbf{x}_*, \mathbf{x}_N)\end{bmatrix} \in \mathbb{R}^{1\times N}$ et $K_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$. Les sorties de l'ensemble d'apprentissage étant connues, nous nous intéressons à la distribution conditionnelle de y_* sachant $\mathbf{y}$ donnée par :

$$y_* \mid \mathbf{y} \sim \mathcal{N}\left(K_*(K + \sigma_n^2 Id)^{-1}\mathbf{y}, K_{**} - K_*(K + \sigma_n^2 Id)^{-1}K_*^T\right)$$

La prédiction de y_* ainsi obtenue correspond à la moyenne de cette distribution :

$$\hat{y}_* = K_*(K + \sigma_n^2 Id)^{-1}\mathbf{y}$$

et l'incertitude de l'estimation est capturée par la variance $\mathrm{Var}(y_*|\mathbf{y}) = K_{**} - K_*(K + \sigma_n^2 Id)^{-1} K_*^T$.

Modèles de Markov cachés. Un modèle de Markov caché (*Hidden Markov Model*, HMM) (Baum et Petrie, 1966 ; Rabiner, 1989 ; Ljung, 1985 ; Cappé et al., 2005) est un outil pour représenter des distributions de probabilités sur des séquences d'observations. Cette méthode tire son nom de deux propriétés caractéristiques :

1. l'observation au temps t, notée x_t a été générée par un certain processus dont l'état noté s_t est *caché* de l'observateur
2. l'état s_t de ce processus caché vérifie la propriété de Markov à l'ordre 1, c'est-à-dire, étant donné la valeur de l'état s_{t-1}, l'état courant s_t est indépendant des autres états antérieurs à $t-1$. En d'autres termes, cela signifie que l'état à un temps donné *encapsule* toute l'information nécessaire sur l'historique du processus pour prédire le comportement futur de ce dernier.

Les sorties observées x_t satisfont également une propriété de Markov : étant donné un état s_t, l'observation x_t est indépendante des états et des observations à tout autre temps $\ell \neq t$. En combinant ces deux propriétés des HMMs, la distribution jointe d'une suite d'états et d'observations peut se factoriser de la façon suivante :

$$P\left((s_t)_{t=1}^N, (x_t)_{t=1}^N\right) = P(s_1)P(x_1|s_1)\prod_{t=2}^{N} P(s_t|s_{t-1})P(x_t|s_t) \tag{3.12}$$

Cette factorisation de la loi jointe peut être représentée sous forme graphique (voir Figure 3.1).

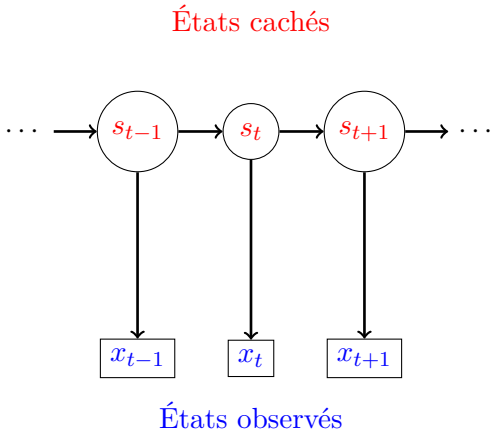


Figure 3.1 – Illustration de la factorisation de la loi jointe d'un modèle de Markov caché.

Une troisième hypothèse sur les HMMs stipule que la variable d'état caché est *discrète* : s_t prend un nombre fini de valeurs (L).

Pour spécifier complètement la distribution sur les suites d'observations, il faut fournir une loi $P(s_1)$ sur l'état initial, la matrice de transition des états de taille $L \times L$ représentant $P(s_t|s_{t-1})$ et le modèle de sortie définissant $P(x_t|s_t)$. Cette dernière distribution peut être modélisée de nombreuses façons, par exemple par une loi normale ou bien un mélange de gaussiennes.

3.3.2 Modèles vectoriels

Dans les modèles scalaires que nous avons décrits, nous fondons la prédiction d'une variable *endogène* sur l'historique de cette dernière. Cependant, de nombreux systèmes dynamiques comprennent plus d'une variable d'état d'intérêt et l'on peut alors souhaiter prédire *simultanément* l'évolution de toutes ces variables. L'on peut alors recourir à un modèle vectoriel. Soit d le nombre de variables d'état qui composent le système pour lequel on suppose avoir observé une série temporelle de $N+1$ vecteurs d'état $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1} \in \mathbb{R}^d$ espacés régulièrement dans le temps. Le problème d'autorégression vectorielle (4.1) consiste en l'estimation d'un modèle $\hat{h} : \mathbb{R}^d \to \mathbb{R}^d$ à partir des données observées $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}$.

Modèles VAR. Le modèle autorégressif vectoriel le plus populaire à ce jour est le modèle VAR (*Vector AutoRegressive*) (Sims, 1980 ; Lütkepohl, 1991) qui est une extension naturelle du modèle AR pour la modélisation de séries temporelles multivariées. Par exemple, le modèle VAR a été appliqué avec succès pour décrire et prévoir le comportement dynamique de séries temporelles économiques et financières pour lesquelles il offre généralement des performances supérieures en prédiction par rapport aux modèles scalaires (Hamilton, 1994).

Le modèle VAR suggère que chaque variable d'état à un temps donné est une combinaison linéaire de ses propres observations passées et de celles des autres variables d'état du système. Plus spécifiquement, un modèle VAR d'ordre (ou de retard) p, noté VAR(p) pour la série temporelle multivariée $\mathbf{x}_t = (x_t^1 \ldots x_t^d)^T \in \mathbb{R}^d$ est de la forme :

$$\mathbf{x}_{t+1} = \sum_{i=1}^{p} B_i \mathbf{x}_{t+1-i} + \mathbf{u}_{t+1} \tag{3.13}$$

où B_i ($i \in \mathbb{N}_p$) est une matrice de taille $d \times d$ et $\mathbf{u}_{t+1}$ est un vecteur vérifiant :

- $\mathbb{E}[\mathbf{u}_t] = 0$;
- $\mathbb{E}[\mathbf{u}_t \mathbf{u}_t^T] = \Sigma_{\mathbf{u}} \in \mathbb{S}_+^d$, la matrice de variance-covariance du bruit est une matrice semi-définie positive ;
- $\mathbb{E}[\mathbf{u}_t \mathbf{u}_s^T] = 0$ pour $s \neq t$: absence d'autocorrélation

Le modèle (3.13) peut s'écrire sous la forme plus condensée suivante :

$$\mathbf{x}_{t+1} = \mathbf{B}\mathbf{z}_t + \mathbf{u}_{t+1} \tag{3.14}$$

où $\mathbf{B} = (B_1 \dots B_p) \in \mathbb{R}^{d \times dp}$ et $\mathbf{z}_t \in \mathbb{R}^{dp}$ est un vecteur où l'on a empilé les vecteurs retardés $(\mathbf{x}_{t+1-i})_{i=1}^{p}$. L'estimation de $\mathbf{B}$ et de $\Sigma_{\mathbf{u}}$ peut alors se faire par la méthode des moindres carrés. Nous présentons ici l'approche par maximum de vraisemblance.

D'après (3.14), la loi conditionnelle de $\mathbf{x}_{t+1}|\mathbf{x}_t, \dots, \mathbf{x}_{t+1-p}$ est une distribution gaussienne multidimensionnelle $\mathcal{N}(\mathbf{B}\mathbf{z}_t, \Sigma_{\mathbf{u}})$. La fonction de vraisemblance est alors donnée par :

$$\prod_{t=1}^{N} P(\mathbf{x}_{t+1}|\mathbf{x}_t, \dots, \mathbf{x}_{t+1-p}) \quad = \quad \prod_{t=1}^{N} \frac{1}{(2\pi)^{d/2}|\Sigma_{\mathbf{u}}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t\right)^T \Sigma_{\mathbf{u}}^{-1} \left(\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t\right)\right\}$$

On obtient donc l'expression de la log-vraisemblance suivante :

$$-\frac{Nd}{2}\log(2\pi) - \frac{N}{2}\log|\Sigma_{\mathbf{u}}| - \frac{1}{2}\sum_{t=1}^{N}\left(\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t\right)^T \Sigma_{\mathbf{u}}^{-1} \left(\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t\right)$$

Après quelques calculs (recherche des points critiques de la log-vraisemblance), la valeur de $\mathbf{B}$ qui maximise la log-vraisemblance est donnée par :

$$\hat{\mathbf{B}} = \left(\sum_{t=1}^{N} \mathbf{x}_{t+1}\mathbf{z}_t^T\right)\left(\sum_{t=1}^{N} \mathbf{z}_t\mathbf{z}_t^T\right)^{-1} \tag{3.15}$$

L'estimateur de $\Sigma_{\mathbf{u}}$ qui s'ensuit est alors donné par :

$$\hat{\Sigma}_{\mathbf{u}} = \frac{1}{N}\sum_{t=1}^{N} \hat{\mathbf{u}}_{t+1}\hat{\mathbf{u}}_{t+1}^T \tag{3.16}$$

où $\hat{\mathbf{u}}_{t+1} = \mathbf{x}_{t+1} - \hat{\mathbf{B}}\mathbf{z}_t,\ t \in \mathbb{N}_N$

Modèles à espace d'états. Les modèles à espace d'états (*State Space Models*, SSM) (Ghahramani, 1998) modélisent une séquence d'observations $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^d$ avec les hypothèses données ci-après :

— à chaque instant t, $\mathbf{x}_t$ a été généré à partir d'une variable d'état cachée $\mathbf{s}_t$ *continue* de dimension D

— la séquence des $\mathbf{s}_t$ définit un processus de Markov d'ordre 1

En définitive, l'on obtient la loi jointe des observations et des états cachés suivante :

$$P\left((\mathbf{s}_t)_{t=1}^N, (\mathbf{x}_t)_{t=1}^N\right) = P(\mathbf{s}_1)P(\mathbf{x}_1|\mathbf{s}_1)\prod_{t=2}^{N} P(\mathbf{s}_t|\mathbf{s}_{t-1})P(\mathbf{x}_t|\mathbf{s}_t) \tag{3.17}$$

Cette factorisation de la loi jointe semble de prime abord identique à celle dans le cadre des HMMs (3.12), à l'exception près que les variables cachées $\mathbf{s}_t$ sont *continues et multidimensionnelles* pour les modèles à espace d'états alors qu'elles sont *discrètes et unidimensionnelles* pour les modèles de Markov cachés. La loi de transition des états $P(\mathbf{s}_t|\mathbf{s}_{t-1})$ peut être décomposée en une composante déterministe et une autre stochastique :

$$\mathbf{s}_t = f_t(\mathbf{s}_{t-1}) + \mathbf{w}_t$$

où f_t est une fonction déterministe qui donne la moyenne de $\mathbf{s}_t$ sachant s_{t-1} et $\mathbf{w}_t$ est un vecteur aléatoire de bruit centré. De la même manière, la probabilité d'observation $P(\mathbf{x}_t|\mathbf{s}_t)$ est donnée par la décomposition suivante :

$$\mathbf{x}_t = h_t(\mathbf{s}_t) + \mathbf{v}_t$$

Si les fonctions de transition et d'observation sont toutes deux linéaires et invariantes par rapport au temps et si de plus, les bruits $\mathbf{w}_t$ et $\mathbf{v}_t$ suivent une loi gaussienne multidimensionnelle, le modèle à espace d'états est dit *linéaire-Gaussien* (*Linear-Gaussian State Space Model*, LGSSM) :

$$\begin{aligned} \mathbf{s}_t &= F\mathbf{s}_{t-1} + \mathbf{w}_t \\ \mathbf{x}_t &= H\mathbf{s}_t + \mathbf{v}_t \end{aligned}$$

où $F \in \mathbb{R}^{D\times D}$ et $H \in \mathbb{R}^{d\times D}$ sont respectivement la matrice de transition et la matrice d'observation. Les LGSSMs sont couramment utilisés dans des domaines tels que le traitement du signal et en théorie du contrôle, où ces modèles sont connus plus généralement sous le vocable de *filtres de Kalman* (Kalman, 1960).

Il est à noter que les SSMs et les HMMs appartiennent à la classe des *réseaux bayésiens dynamiques* (*Dynamic Bayesian Networks*, DBN) qui sont des modèles graphiques orientés appropriés pour des tâches faisant intervenir des séquences ou dans notre cas, des séries temporelles (Ghahramani, 1998).

Modèles vectoriels non linéaires. Dans le cas de systèmes dynamiques non linéaires, l'architecture flexible des réseaux de neurones récurrents justifie leur validité dans une

série d'applications. Une extension non linéaire intéressante des modèles VAR proposée par Ralaivola et d'Alche-Buc (2003) consiste à considérer la série temporelle $(\mathbf{x}_t^{\phi})_{t=1}^{N+1} = \{\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_{N+1})\}$ où ϕ est une fonction de $\mathbb{R}^d$ dans un espace de caractéristiques $\mathcal{H}$ de noyau associé k puis à apprendre les paramètres d'un modèle VAR dans $\mathcal{H}$ et à revenir dans l'espace d'entrée en calculant les préimages de la série temporelle prédite dans H.

En dépit du nombre important et croissant de problèmes impliquant des systèmes dynamiques non linéaires, la littérature sur les modèles autorégressifs vectoriels non linéaires demeure relativement peu fournie en comparaison de leur équivalent scalaire et ce malgré des avancées théoriques dans le domaine. Dans une revue sur les modèles de prévision, De Gooijer et Hyndman (2006) invoquent deux raisons principales : la première est le manque de résultats empiriques prouvant la robustesse des algorithmes de prévision avec des modèles vectoriels et deuxièmement, ils pointent le déficit d'outils logiciels *simples* qui permettraient une plus grande utilisation et une plus grande pénétration de ces modèles dans la communauté. Ensuite, les modèles suggérés, e.g. les modèles VAR, se révèlent difficiles à estimer en pratique en raison du grand nombre de paramètres impliqués.

La table 3.1 dresse un récapitulatif des modèles que nous avons présentés en pointant les qualités et les limites de chacun.

Table 3.1 – Récapitulatif des modèles de prévision de séries temporelles existants : les modèles AutoRégressifs à Moyenne Mobile (ARMA), Réseaux de Neurones Récurrents (RNN), estimateurs non paramétriques de type Nadaraya-Watson (NW), k plus proches voisins (kNN), arbres de décision (Arbres), Régression à Vecteurs de Support (SVR), les Processus Gaussiens (GP), les Modèles de Markov Cachés (HMM), les modèles Vectoriels AutoRégressifs (VAR), les modèles à espace d'états linéaires-Gaussiens (LGSSM). Pour chaque méthode, nous renseignons si elle présente les avantages suivants : le modèle est-il non linéaire ? non paramétrique ? vectoriel ? Peut-on facilement intégrer des connaissances *a priori* ? La méthode passe-t-elle à l'échelle ?

Modèles	Non linéaire	Non paramétrique	Vectoriel	*A priori*	Passe à l'échelle
ARMA				✓	✓
RNN	✓	✓	✓		
NW	✓	✓			✓
kNN	✓	✓			
Arbres	✓	✓			
SVR	✓	✓		✓	✓
GP	✓	✓		✓	✓
HMM				✓	
VAR			✓	✓	✓
LGSSM			✓	✓	

3.4 Modèles pour l'inférence de réseaux

Dans de nombreux problèmes scientifiques, les données en grande dimension possèdent une structure *causale* sous-jacente et jouent un rôle essentiel dans l'extraction de connaissances (Kolaczyk, 2009). Par exemple, de par les avancées technologiques récentes, les techniques à haut débit ont facilité l'étude conjointe des composants de systèmes biologiques complexes. Ainsi, les chercheurs en biologie moléculaire sont en capacité de mesurer les niveaux d'expression d'un génome entier et une proportion substantielle du protéome et du métabolome sous diverses conditions, ce qui leur permet d'avoir une meilleure compréhension des mécanismes de réponse des organismes à leur environnement. Pour cette raison, la reconstruction de réseaux de régulation génique à partir de données d'expression est devenue un problème canonique en biologie computationnelle (Lawrence et al., 2010). De telles structures de données émergent dans d'autres domaines scientifiques. Notamment, les politologues se sont intéressés aux données de vote par appel nominal dans les parlements, puisque l'analyse de ces dernières apporte des informations quant à la cohésion des partis et la formation de coalitions à travers la reconstruction du réseau sous-jacent (Morton et Williams, 2010). Remarquons par ailleurs que de telles tâches apparaissent également en économie pour l'étude de la solvabilité des entreprises et de l'effet de contagion (Gilchrist et al., 2009) et en climatologie, domaine dans lequel les experts cherchent à expliquer les effets du changement climatique via les dépendances éventuelles entre les variables du climat et les facteurs liés aux activités humaines (Parry et al., 2007 ; Liu et al., 2010).

Deux classes de problèmes d'inférence de réseaux ont émergé de manière concomitante : l'inférence de réseaux d'association qui représentent le couplage entre variables d'intérêt (Meinshausen et Bühlmann, 2006 ; Kramer et al., 2009) et l'inférence de réseaux « causaux » qui décrivent la manière dont les variables s'influencent les unes les autres (Maathuis et al., 2010). Nous nous concentrons sur la dernière tâche citée.

3.4.1 Problématique de l'inférence de réseaux

Considérons un système dynamique qui comporte d variables d'état. La structure sous-jacente à ce dernier peut être codée par un graphe ou *réseau* $G = (V, E)$ défini de la manière suivante :

— chaque sommet $v \in V$ représente une variable d'état
— deux sommets v_i et v_j, $(i, j) \in \mathbb{N}_d^2$, sont connectés par un arc $e = (v_i, v_j) \in E$ s'il existe une *influence causale* de la variable d'état i sur la variable d'état j.

Ce réseau d'interactions est codé de manière équivalente par sa *matrice d'adjacence* $A = (a_{ij})_{i,j=1}^{d} \in \{0,1\}^{d\times d}$, définie avec les conventions suivantes :

$$a_{ij} = \begin{cases} 1 & \text{si la variable d'état } j \text{ influence la variable d'état } i \\ 0 & \text{sinon} \end{cases} \tag{3.18}$$

La tâche d'estimation de structure, aussi appelée *inférence de réseaux* ou encore *rétro-conception* (*reverse-engineering*) consiste à identifier les influences causales, c'est-à dire retrouver les arcs du réseau G ou encore estimer la matrice d'adjacence A à partir de données issues de l'observation du système (données statiques iid ou données de séries temporelles) par des méthodes *automatiques*.

En un peu plus d'une décennie, un nombre substantiel de méthodes statistiques ont été proposées pour estimer de telles structures. Ces méthodes d'inférence de réseaux peuvent être classées en deux grandes familles de modèles. Les approches dites sans modèle (*model-free*) dédiées aux réseaux d'association utilisent des mesures statistiques telles que les corrélations (Stuart et al., 2003) ou d'autres mesures issues de la théorie de l'information, comme l'information mutuelle (Butte et Kohane, 2000 ; Basso et al., 2005 ; Faith et al., 2007 ; Hartemink, 2005 ; Margolin et al., 2006) pour pondérer les arêtes des graphes non orientés associés. À l'opposé, on trouve les approches fondées sur des modèles (*model-driven*). Nous nous focalisons sur cette dernière classe de modèles afin de pouvoir inférer des réseaux causaux.

Deux principaux types de données sont généralement utilisés pour cette tâche :

— les données en régime permanent (*steady states*). Ce sont des données statiques i.i.d. obtenues sur le long terme, en supposant que le système atteint un état d'équilibre

— les données de séries temporelles

3.4.2 Données statiques

Modèles de régression. Dans le cas de données statiques, une approche classique consiste à modéliser chaque variable d'état-cible $i \in \mathbb{N}_d$ par un modèle de régression de la forme :

$$x^i = h^i(x^1, \ldots, x^d) + u^i \tag{3.19}$$

où u^i est un terme de bruit centré. Lorsqu'on considère un modèle de régression linéaire, $h^i,\ i \in \mathbb{N}_d$ s'écrit alors de la manière suivante :

$$h^i(x^1, \ldots, x^d) = \sum_{j\in\mathbb{N}_d\setminus\{i\}} \beta_{ij} x^j \tag{3.20}$$

On conclut à la présence d'une relation d'influence causale de la variable d'état j sur la variable i si la valeur du coefficient β_{ij} est significativement non nulle. La tâche d'inférence de réseaux est dans ce cadre étroitement liée au problème de la sélection de caractéristiques en statistiques avec l'utilisation de méthodes telles que LARS ou le LASSO (Meinshausen et Bühlmann, 2006 ; Haury et al., 2012). Des modèles plus généraux sont envisageables. Aussi Huynh-Thu et al. (2010) modélisent-ils $h^i,\ i \in \mathbb{N}_d$ par le biais de forêts aléatoires (Leo Breiman, 2001).

Réseaux bayésiens. Un autre formalisme largement répandu est celui des réseaux bayésiens (*Bayesian Network*, BN) (Husmeier, 2003a ; Auliac et al., 2008). Un réseau bayésien (Pearl, 1988) est défini par un couple (G, P_G). G est un graphe orienté sans cycle (*Directed Acyclic Graph*, DAG) dont les sommets sont des variables aléatoires et les absences d'arêtes encodent les indépendances conditionnelles entre ces variables. La topologie du graphe définit un ensemble de *parents* pour chaque sommet. Dans notre cas, les sommets correspondent aux variables d'état du système $X = \{x^1, \ldots, x^d\}$ et l'ensemble des parents du sommet $i \in \mathbb{N}_d$ est noté Pa_i. Une arête représente l'influence causale d'une variable parent sur une variable-cible enfant. L'hypothèse markovienne des BNs affirme que chaque sommet est indépendant de ses non-descendants conditionnellement à ses parents. Il s'ensuit la factorisation de la loi jointe :

$$P(x^1, \ldots, x^d) = P_G(X) = \prod_{i=1}^{d} P_G(x^i | Pa_i) \tag{3.21}$$

Un point crucial à signaler sur ce type de modèles est l'impossibilité de représenter des cycles, ce qui constitue une forte restriction.

Modèles graphiques gaussiens Une classe majeure de techniques concerne les modèles graphiques gaussiens (*Gaussian Graphical Models*, GGMs). L'idée sous-jacente est que les dépendances conditionnelles encodent les interactions entre les variables d'état. Sous l'hypothèse que les données observées sont la réalisation d'une loi gaussienne multidimensionnelle de covariance Σ, on peut montrer que les dépendances conditionnelles correspondent aux coefficients non nuls de l'inverse de Σ. Il s'agit alors d'estimer Σ^{-1}. Cette estimation peut s'avérer difficile en grande dimension lorque le nombre de données disponibles N est inférieur au nombre de variables d'états d. Dans ce cas, plusieurs méthodes utilisent par exemple une régularisation parcimonieuse en norme ℓ_1 (voir Schäfer et Strimmer (2005) ainsi que P. Bühlmann et Geer (2011) et références incluses).

3.4.3 Données de séries temporelles

Modèles autorégressifs linéaires. Dans le cas de données temporelles, les modèles autorégressifs linéaires (D'HAESELEER et al., 1999 ; SOMEREN et al., 2006 ; FUJITA et al., 2007 ; SHIMAMURA et al., 2009 ; LOZANO et al., 2009 ; CHARBONNIER et al., 2010) se sont considérablement développés. Pour ces derniers, plusieurs schémas de régularisation *parcimonieuse* (structurée ou non structurée) ont été proposés (BOLSTAD et al. (2011) et CHATTERJEE et al. (2012), voir aussi George MICHAILIDIS et D'ALCHÉ-BUC (2013) et références incluses). Au cœur de ces modèles, le concept de *causalité de Granger* (C. W. GRANGER, 1969) joue un rôle crucial pour capturer des relations de causalité (SHOJAIE et G. MICHAILIDIS, 2010).

Réseaux bayésiens dynamiques. Parallèlement, des approches à base de réseaux bayésiens dynamiques (K. P. MURPHY, 1998 ; HUSMEIER, 2003b ; PERRIN et al., 2003 ; Sunyong KIM et al., 2004 ; N. FRIEDMAN, 2004 ; LÈBRE, 2009 ; C. ZOU et FENG, 2009 ; DONDELINGER et al., 2013) ont connu un véritable essor car ils contournent les limites des BNs en permettant de représenter des cycles. La structure du réseau est en effet *déroulée* à travers le temps. Malgré leur expressivité, l'inconvénient majeur de ces méthodes demeure leur coût computationnel.

ODEs. Un système d'équations différentielles ordinaires (ODE pour *Ordinary Differential Equations*) couplées décrit avec précision l'évolution de l'ensemble des d variables d'état d'un système dynamique. L'état de la variable $i \in \mathbb{N}_d$, notée $x^i(t)$ est régi par une équation de la forme :

$$\frac{d}{dt}x^i(t) = h^i\left(x^1(t), \ldots, x^d(t); \Theta^i\right) \tag{3.22}$$

où Θ^i est l'ensemble des paramètres du modèle d'ODE h^i pour la variable i. Lorsque la forme du modèle h^i est donnée *a priori*, l'estimation des paramètres $\{\Theta^i\}_{i=1}^d$ permet de déterminer la structure du réseau (BANSAL et al., 2006 ; BONNEAU et al., 2006). Cependant, comme la plupart des méthodes paramétriques, les modèles d'ODEs sont souvent spécifiques d'un système particulier, ce qui les rend peu génériques. En l'absence de connaissance experte, ÄIJÖ et LÄHDESMÄKI (2009) ont proposé une manière originale de modéliser h^i, $i \in \mathbb{N}_d$ par le biais de processus gaussiens.

Malgré le nombre et le mérite des contributions sur le sujet, peu de papiers à notre connaissance ont mis l'accent sur l'inférence de réseaux pour des variables continues en présence de dynamiques *non linéaires* alors que la plupart des mécanismes (e.g. de régulation en biologie) en jeu dans les systèmes réels relèvent de cette nature. À ce jour, les approches les plus concluantes incluent

— les méthodes à base d'équations différentielles ordinaires (Chou et Voit, 2009) qui apprennent alternativement la structure du modèle et ses paramètres
— les techniques par apprentissage bayésien (Mazur et al., 2009 ; Äijö et Lähdesmäki, 2009) qui tiennent compte de la nature stochastique des données biologiques tout en incorporant des connaissances *a priori*
— les algorithmes évolutionnaires fondés sur le paradigme de programmation génétique (Iba, 2008) permettant une exploration stochastique de l'espace des structures
— les méthodes d'ensemble (boosting, forêts aléatoires, *extra-trees*) (Geurts et al., 2007 ; Huynh-Thu et al., 2010).

La table 3.2 donne un aperçu des forces et des faiblesses des différents types de méthodes pour l'inférence de réseaux et permet de se rendre compte des défis qui restent encore à relever.

Table 3.2 – Récapitulatif des types de modèles existants pour l'inférence de réseaux à partir de séries temporelles : les modèles autorégressifs linéaires, les approches *model-free* (corrélations, information mutuelle,...), modèles graphiques gaussiens (GGM), équations différentielles ordinaires (ODE), méthodes bayésiennes (réseaux bayésiens dynamiques,...), algorithmes évolutionnaires. Pour chaque méthode, nous renseignons si elle présente les avantages suivants : est-elle générique (nécessite peu de connaissances expertes) ? non linéaire ? Estime-t-elle une structure causale, c'est-à-dire un graphe orienté où les arcs représentent des influences directes des variables les unes sur une autres ? Peut-on facilement intégrer des connaissances *a priori* ? La méthode passe-t-elle à l'échelle ?

Modèles	Générique	Non linéaire	Graphe causal	*A priori*	Passe à l'échelle
Autorégressifs linéaires	✓		✓	✓	✓
Model-free	✓				✓
GGM	✓			✓	✓
ODE		✓	✓	✓	
Bayésiens	✓	✓	✓	✓	
Évolutionnaires	✓	✓	✓	✓	

3.5 Synthèse

Dans ce chapitre, nous avons présenté l'état de l'art de la modélisation de systèmes dynamiques à travers le prisme de deux objectifs : la prévision et l'extraction de la structure causale sous-jacente aux données. Le prochain chapitre introduit une nouvelle classe de modèles autorégressifs non paramétriques que nous présentons comme une extension des

modèles linéaires VAR au cas non linéaire.

Deuxième partie

Une nouvelle famille de modèles autorégressifs vectoriels à base de noyaux à valeur opérateur

Chapitre 4

Modèles autorégressifs vectoriels non paramétriques

Sommaire

4.1 Introduction 75
4.2 Des modèles autorégressifs à noyaux scalaires vers des modèles autorégressifs à noyaux à valeur opérateur 76
4.3 Noyaux à valeur opérateur et espaces fonctionnels associés . . 80
4.3.1 Régularisation dans les RKHS à valeurs vectorielles 84
4.3.2 RKBS et Régularisation parcimonieuse 87
4.4 Une nouvelle famille de modèles autorégressifs non paramétriques : OKVAR 91
4.4.1 La famille de modèles OKVAR 92
4.5 Apprentissage d'un modèle OKVAR 98
4.5.1 Choix de la régularisation 98
4.5.2 Algorithme proximal 100
4.5.3 Résultats 102
4.6 OKVAR-Boost 107
4.6.1 Principe de l'algorithme 107
4.6.2 Résultats 109
4.7 Synthèse 112

4.1 Introduction

Nous avons vu dans le chapitre précédent la diversité des formalismes existants pour modéliser un système dynamique. Nous présentons ici notre première contribution qui consiste en la définition d'une nouvelle famille de modèles autorégressifs vectoriels. En l'absence de connaissance experte qui imposerait une forme spécifique au modèle, nous proposons une modélisation non paramétrique qui constitue une extension des modèles VAR au cas non linéaire. Un réflexe naturel lorsque l'on souhaite transformer un modèle

linéaire en modèle non linéaire est d'utiliser l'astuce du noyau. Or les noyaux ne permettent *a priori* que d'apprendre des fonctions à valeurs scalaires. C'est la raison pour laquelle nous recourons aux noyaux à valeur *opérateur* appropriés pour l'apprentissage de fonctions à valeurs vectorielles. Une étape préliminaire de ce chapitre vise à expliquer l'intérêt de l'apprentissage d'un modèle couplé comparativement à un apprentissage indépendant. Puis, dans un deuxième temps, nous introduisons la théorie générale des noyaux à valeur opérateur pour l'apprentissage d'une fonction à valeurs dans un espace de Hilbert. En revenant à la problématique de l'autorégression vectorielle, nous montrons comment construire des modèles autorégressifs à partir de noyaux à valeur matricielle. Par la suite, se pose la question de l'apprentissage de tels modèles, un problème pour lequel nous proposons un algorithme qui permet d'estimer les paramètres du modèle notamment lorsque l'on incorpore des *a priori* parcimonieux. Des résultats numériques sur des jeux de données synthétiques viennent compléter l'étude.

4.2 Des modèles autorégressifs à noyaux scalaires vers des modèles autorégressifs à noyaux à valeur opérateur

Considérons un système dynamique comportant d variables d'état. On note $\mathbf{x}_t \in \mathbb{R}^d$ le vecteur d'état *observé* au temps t et on suppose qu'on observe le système à $N+1$ temps discrets t_i consécutifs avec un pas de temps constant Δt, c'est-à-dire $t_2 - t_1 = \ldots = t_{N+1} - t_N = \Delta t$. Les vecteurs d'état observés sont donc $\mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_{N+1}} \in \mathbb{R}^d$. Toutefois par souci de simplicité, les vecteurs d'état sont notés $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}$. Le vecteur $\mathbf{x}_1$ est appelé *condition initiale*. Dans ce chapitre, on fait l'hypothèse énoncée ci-dessous.

Hypothèse 4.1. *Un modèle stationnaire markovien d'ordre 1 est adéquat pour modéliser l'évolution temporelle du système dynamique considéré. La dynamique non linéaire sous-jacente au système est capturée par une fonction $h : \mathbb{R}^d \to \mathbb{R}^d$. Le modèle est donc de la forme :*

$$\mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_{t+1} \tag{4.1}$$

où $\mathbf{u}_{t+1} \in \mathbb{R}^d$ est un terme de bruit centré (de moyenne nulle).

Sous l'hypothèse 4.1, la condition initiale $\mathbf{x}_1$ détermine à elle seule l'évolution du système. À partir des données observées $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}$, on forme l'ensemble d'apprentissage correspondant de taille N : $\mathcal{S}_N = \{(\mathbf{x}_\ell, \mathbf{x}_{\ell+1})\}_{\ell=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^d$. Le problème d'autorégression (4.1) consiste à inférer une hypothèse (modèle autorégressif) $\hat{h} : \mathbb{R}^d \to \mathbb{R}^d$ à partir de $\mathcal{S}_N$.

Pour résoudre le problème d'autorégression (4.1), on peut opter pour une décomposition de la tâche d'autorégression vectorielle en d tâches d'autorégression scalaire. Dans cette perspective, au lieu d'apprendre une fonction $h : \mathbb{R}^d \to \mathbb{R}^d$ à sortie vectorielle à partir

de $\mathcal{S}_N = \{(\mathbf{x}_\ell, \mathbf{x}_{\ell+1})\}_{\ell=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^d$, pour chaque variable d'état $i \in \mathbb{N}_d$, on va apprendre une fonction $h^i : \mathbb{R}^d \to \mathbb{R}$ à sortie scalaire à partir de $\mathcal{S}_N^i = \{(\mathbf{x}_\ell, x_{\ell+1}^i)\}_{\ell=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$. On peut alors résoudre chaque tâche i individuellement et indépendamment les unes des autres. La solution ainsi obtenue pour le problème (4.1) est de la forme :

$$\hat{h} = (\hat{h}^1 \dots \hat{h}^d)^T$$

Pour tenir compte des non-linéarités du système, on peut résoudre chaque tâche d'autorégression scalaire i en estimant un modèle non paramétrique non linéaire. Par exemple, les modèles à noyaux scalaires sont des candidats appropriés pour ce type de tâches. Étant donné un noyau semi-défini positif $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, un modèle autorégressif pour la tâche $i \in \mathbb{N}_d$, basé sur le noyau k peut alors s'écrire :

$$\hat{x}_{t+1}^i = h^i(\mathbf{x}_t) = \sum_{\ell=1}^N \alpha_\ell^i k(\mathbf{x}_\ell, \mathbf{x}_t) \tag{4.2}$$

où $\alpha_1^i \dots \alpha_N^i \in \mathbb{R}$ sont les paramètres associés au modèle h^i.

La question qui se pose est de savoir si la décomposition du problème en modèles indépendants est toujours satisfaisante ou bien s'il existe des cas où il est plus intéressant de considérer un unique modèle couplé. Pour cela, considérons un modèle vectoriel $h_\theta : \mathbb{R}^d \to \mathbb{R}^d$ paramétré par $\theta \in \Theta$. Nous supposons que les résidus sont homoscédastiques, c'est-à-dire que la matrice de covariance des erreurs ne varie pas dans le temps et que le bruit $\mathbf{u}_{t+1}, t \in \mathbb{N}_N$ est modélisé par une loi gaussienne multidimensionnelle $\mathcal{N}(0, \Sigma_\mathbf{u})$. La condition initiale est quant à elle tirée selon une loi $\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1)$. La vraisemblance du modèle s'écrit :

$$L(\theta, \Sigma_\mathbf{u}) = p(\mathbf{x}_1, \dots, \mathbf{x}_{N+1} | \theta, \Sigma_\mathbf{u}) = p(\mathbf{x}_1 | \boldsymbol{\mu}_1, \Sigma_1) \prod_{t=1}^N p(\mathbf{x}_{t+1} | \mathbf{x}_t; \theta, \Sigma_\mathbf{u}) \tag{4.3}$$

Les erreurs $\mathbf{u}_{t+1} = \mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)$ suivant une loi normale, on en déduit les lois conditionnelles :

$$\mathbf{x}_{t+1} | \mathbf{x}_t; \theta, \Sigma_\mathbf{u} \sim \mathcal{N}(h_\theta(\mathbf{x}_t), \Sigma_\mathbf{u}) \tag{4.4}$$

Il s'ensuit que la distribution conditionnelle de la $t+1$-ème observation $\mathbf{x}_{t+1}$ connaissant $\mathbf{x}_t$, θ et $\Sigma_\mathbf{u}$ est donnée par :

$$p(\mathbf{x}_{t+1} | \mathbf{x}_t; \theta, \Sigma_\mathbf{u}) = \frac{1}{(2\pi)^{d/2} |\Sigma_\mathbf{u}|^{1/2}} \exp\left\{ -\frac{1}{2} \left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right)^T \Sigma_\mathbf{u}^{-1} \left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right) \right\} \tag{4.5}$$

On obtient alors l'expression de la log-vraisemblance suivante :

$$\begin{aligned}\ln L(\theta, \Sigma_{\mathbf{u}}) &= \ln p(\mathbf{x}_1|\boldsymbol{\mu}_1, \Sigma_1) + \sum_{t=1}^{N} \ln p(\mathbf{x}_{t+1}|\mathbf{x}_t; \theta, \Sigma_{\mathbf{u}}) \\ &= \ln p(\mathbf{x}_1|\boldsymbol{\mu}_1, \Sigma_1) - \frac{Nd}{2}\left(\ln(2\pi) + \frac{1}{d}\ln|\Sigma_{\mathbf{u}}|\right) \\ &\quad -\frac{1}{2}\sum_{t=1}^{N}\left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right)^T \Sigma_{\mathbf{u}}^{-1}\left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right) \end{aligned} \tag{4.6}$$

Par conséquent,

$$\max_{\theta\in\Theta} L(\theta, \Sigma_{\mathbf{u}}) \Leftrightarrow \min_{\theta\in\Theta} \sum_{t=1}^{N}\left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right)^T \Sigma_{\mathbf{u}}^{-1}\left(\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\right) \tag{4.7}$$

Dans le cas où $\mathbf{u}_{t+1}$ est modélisé par une loi gaussienne isotrope $\mathcal{N}(0, \sigma_{\mathbf{u}}^2 Id)$, avec $\sigma_{\mathbf{u}}^2 > 0$. Alors l'équation (4.7) se réduit à :

$$\max_{\theta\in\Theta} L(\theta, \sigma_{\mathbf{u}}^2) \Leftrightarrow \min_{\theta\in\Theta} \sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\|_2^2 \tag{4.8}$$

En d'autres termes, l'estimateur du maximum de vraisemblance sous l'hypothèse d'un bruit gaussien homogène est identique à celui obtenu en minimisant le risque empirique avec la fonction de perte quadratique. Pour minimiser $\sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h_\theta(\mathbf{x}_t)\|_2^2 = \sum_{i=1}^{d}\sum_{t=1}^{N} \left(x_{t+1}^i - h_\theta^i(\mathbf{x}_t)\right)^2$, il peut alors suffire de construire pour chaque composante $i \in \mathbb{N}_d$ un modèle h^i qui minimise $\sum_{t=1}^{N}\left(x_{t+1}^i - h_\theta^i(\mathbf{x}_t)\right)^2$, par exemple h^i peut être un modèle bâti sur un noyau universel.

Lorsque $\Sigma_{\mathbf{u}}$ n'est plus diagonale, la terme général de la covariance du modèle s'exprime alors ainsi :

$$\begin{aligned} cov\left(h(\mathbf{x}_t)^i + u_{t+1}^i, h(\mathbf{x}_t)^j + u_{t+1}^j\right) &= cov\left(h(\mathbf{x}_t)^i, h(\mathbf{x}_t)^j\right) + \underbrace{cov\left(h(\mathbf{x}_t)^i, u_{t+1}^j\right)}_{=0} \\ &\quad + \underbrace{cov\left(u_{t+1}^i, h(\mathbf{x}_t)^j\right)}_{=0} + cov\left(u_{t+1}^i, u_{t+1}^j\right) \\ &= cov\left(h(\mathbf{x}_t)^i, h(\mathbf{x}_t)^j\right) + cov\left(u_{t+1}^i, u_{t+1}^j\right) \end{aligned} \tag{4.9}$$

Intéressons-nous à l'hypothèse où le modèle h^i fait intervenir un couplage entre toutes les composantes du système. Pour fixer les idées, supposons que h est un modèle VAR(1) : $h(\mathbf{x}_t) = B\mathbf{x}_t$ et que le bruit est une variable aléatoire centrée de covariance $\Sigma_{\mathbf{u}}$. L'expression

(4.9) devient

$$\begin{aligned} cov\left(h(\mathbf{x}_t)^i + u^i_{t+1}, h(\mathbf{x}_t)^j + u^j_{t+1}\right) &= cov\left(h(\mathbf{x}_t)^i, h(\mathbf{x}_t)^j\right) + \Sigma_{\mathbf{u},ij} \\ &= cov\left(\sum_{p=1}^{d} B_{ip}x^p_t, \sum_{q=1}^{d} B_{jq}x^q_t\right) + \Sigma_{\mathbf{u},ij} \\ &= \sum_{p=1}^{d}\sum_{q=1}^{d} B_{ip}B_{jq}cov\left(x^p_t, x^q_t\right) + \Sigma_{\mathbf{u},ij} \end{aligned}$$

Par conséquent, on suppute qu'une modélisation vectorielle est adéquate soit en présence d'un bruit de covariance non diagonale ou bien dès lors que la recherche d'un couplage entre les composantes du système sous-tend le problème d'intérêt. Dans cette thèse, la modélisation du bruit n'a pas fait l'objet d'une étude approfondie, nous considérons principalement un bruit gaussien centré isotrope et faisons l'hypothèse qu'il existe un couplage entre les variables d'état. Aussi privilégions-nous par la suite une présentation par moindres carrés plutôt qu'une approche par maximum de vraisemblance.

L'un des objectifs de nos travaux a consisté à étendre les approches de type noyau scalaire au cas vectoriel en définissant une nouvelle famille générale de modèles autorégressifs vectoriels non paramétriques. Pour apprendre des fonctions à valeurs vectorielles, la théorie des RKHS présentée au chapitre 1 pour des fonctions à valeurs scalaires ne s'applique plus. C'est la raison pour laquelle nous avons recours à la théorie des RKHS pour les fonctions à valeurs *vectorielles* et des noyaux à valeur opérateur, issue des travaux précurseurs de Senkene et Tempel'man (1973) au début des années 1970. Il faut attendre les années 2000 pour voir les premières applications en apprentissage, pour des problèmes multi-tâches (Evgeniou et al., 2005 ; C. Micchelli et M. Pontil, 2005). C'est actuellement un domaine de recherche actif (Alvarez et al., 2011) dont les applications comprennent notamment l'apprentissage de champs de vecteurs (Baldassarre et al., 2010 ; Sindhwani et al., 2013), la classification structurée (Dinuzzo et Fukumizu, 2011), la régression fonctionnelle (Kadri et al., 2011) et la prédiction de liens (Brouard et al., 2011). Cependant, à notre connaissance, l'utilisation des noyaux à valeurs opérateurs dans le contexte des séries temporelles est nouveau.

Nous introduisons dans la section suivante les éléments de définitions et les propriétés principales de la théorie des RKHS pour les fonctions à valeurs vectorielles.

4.3 Noyaux à valeur opérateur et espaces fonctionnels associés

Notations. Soit $\mathcal{H}$ un espace de Hilbert, le produit scalaire et la norme associés sont notés respectivement $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ et $\| \cdot \|_{\mathcal{H}}$. Si $\mathcal{H}$ et $\mathcal{H}'$ sont deux espaces de Hilbert, on note $\mathcal{L}(\mathcal{H}, \mathcal{H}')$ l'ensemble des opérateurs linéaires bornés de $\mathcal{H}$ dans $\mathcal{H}'$ ($\mathcal{L}(\mathcal{H}, \mathcal{H})$ est noté plus simplement $\mathcal{L}(\mathcal{H})$). On confère à $\mathcal{L}(\mathcal{H}, \mathcal{H}')$ une structure d'espace de Banach en le munissant de la norme uniforme $\| \cdot \|_{\mathcal{H},\mathcal{H}'}$ dans $\mathcal{L}(\mathcal{H}, \mathcal{H}')$. Soit un opérateur $A \in \mathcal{L}(\mathcal{H}, \mathcal{H}')$, un théorème sur les opérateurs dans les espaces de Hilbert indique qu'il existe un unique opérateur noté $A^* \in \mathcal{L}(\mathcal{H}', \mathcal{H})$ tel que

$$\forall \mathbf{x} \in \mathcal{H}, \forall \mathbf{z} \in \mathcal{H}', \ \langle A\mathbf{x}, \mathbf{z} \rangle_{\mathcal{H}'} = \langle \mathbf{x}, A^*\mathbf{z} \rangle_{\mathcal{H}}$$

L'opérateur A^* est appelé opérateur adjoint de A.

Définition 4.1 (RKHS à valeurs vectorielles (Senkene et Tempel'man, 1973 ; Carmeli, De Vito et Toigo, 2006))**.** *Soient un ensemble $\mathcal{X}$ et un espace de Hilbert $\mathcal{Y}$, on dit que $\mathcal{H}$ est un espace de Hilbert à noyau reproduisant sur $\mathcal{X}$ à valeurs dans $\mathcal{Y}$ si, et seulement si, $\mathcal{H}$ est un espace de Hilbert qui satisfait les propriétés suivantes :*

(1) les éléments de $\mathcal{H}$ sont des fonctions de $\mathcal{X}$ dans $\mathcal{Y}$;

(2) pour tout $\mathbf{x} \in \mathcal{X}$, il existe une constante $C_{\mathbf{x}} > 0$ telle que

$$\forall f \in \mathcal{H}, \ \|f(\mathbf{x})\|_{\mathcal{Y}} \leq C_{\mathbf{x}} \|f\|_{\mathcal{H}}$$

Définition 4.2 (Noyau à valeur opérateur (Senkene et Tempel'man, 1973 ; Carmeli, De Vito et Toigo, 2006))**.** *Un noyau semi-défini positif à valeurs dans $\mathcal{L}(\mathcal{Y})$ est une fonction $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ telle que :*

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathcal{X} \times \mathcal{X}, \ K(\mathbf{x}, \mathbf{z}) = K(\mathbf{z}, \mathbf{x})^*$$

et

$$\forall m \in \mathbb{N}, \forall \{(\mathbf{x}_i, c_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{C}, \forall \mathbf{y} \in \mathcal{Y}, \ \sum_{i,j=1}^{m} c_i \bar{c}_j \langle K(\mathbf{x}_j, \mathbf{x}_i)\mathbf{y}, \mathbf{y} \rangle_{\mathcal{Y}} \geq 0 \tag{4.10}$$

Dans la littérature, on peut trouver plusieurs formulations de la propriété de semi-définie positivité (Eq. 4.10) des noyaux à valeur opérateur :

— par exemple, C. Micchelli et M. Pontil (2005) et Caponnetto, C. A. Micchelli et al. (2008) fournissent l'expression suivante strictement équivalente à

(4.10) :

$$\forall m \in \mathbb{N}, \forall \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathcal{Y}, \ \sum_{i,j=1}^m \langle \mathbf{y}_i, K(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j \rangle_{\mathcal{Y}} \geq 0 \tag{4.11}$$

— une définition alternative de la propriété de semi-définie positivité est donnée par exemple dans H. Zhang, Xu et Q. Zhang (2012) :

$$\forall m \in \mathbb{N}, \forall \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathcal{Y}, \ \sum_{i,j=1}^m \langle K(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_i, \mathbf{y}_j \rangle_{\mathcal{Y}} \geq 0 \tag{4.12}$$

Il est important de noter que les deux définitions conduisent à deux espaces fonctionnels *distincts* :

— un espace de Hilbert $\mathcal{H}_K$ basé sur la propriété de semi-définie positivité (4.10) ou (4.11) est généré à partir d'éléments de la forme :

$$K(\cdot, \mathbf{x})\mathbf{y} \in \mathcal{H}_K \quad \text{avec } \mathbf{x} \in \mathcal{X} \text{ et } \mathbf{y} \in \mathcal{Y} \tag{4.13}$$

— cette propriété telle qu'elle est définie par l'équation (4.12) amène à considérer un espace de Hilbert $\mathcal{H}_K^{\#}$ généré par des fonctions du type :

$$K(\mathbf{x}, \cdot)\mathbf{y} \in \mathcal{H}_K^{\#} \quad \text{avec } \mathbf{x} \in \mathcal{X} \text{ et } \mathbf{y} \in \mathcal{Y} \tag{4.14}$$

Dans le reste de ce chapitre, nous continuons la présentation avec la propriété de semi-définie positivité telle que formulée en (4.10) ou (4.11).

Comme dans le cas scalaire, on peut associer de manière canonique à un RKHS $\mathcal{H}$ à valeurs vectorielles un noyau semi-défini positif à valeur opérateur. En effet, soit $\mathbf{x} \in \mathcal{X}$, d'après la propriété (2) de la définition 4.1, la fonction d'évaluation linéaire en $\mathbf{x}$

$$\begin{array}{rccc} \delta_{\mathbf{x}} : & \mathcal{H} & \to & \mathcal{Y} \\ & f & \mapsto & f(\mathbf{x}) \end{array}$$

est un opérateur borné et le noyau *reproduisant* associé à $\mathcal{H}$ est défini par :

$$\begin{array}{rccc} K_{\mathcal{H}} : & \mathcal{X} \times \mathcal{X} & \to & \mathcal{L}(\mathcal{Y}) \\ & (\mathbf{x}, \mathbf{z}) & \mapsto & \delta_{\mathbf{x}}\delta_{\mathbf{z}}^* \end{array} \tag{4.15}$$

Vérifions que $K_{\mathcal{H}}$ est un noyau semi-défini positif.

Démonstration. Pour $m \in \mathbb{N}$, soient $\{(\mathbf{x}_i, c_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{C}$ et $\mathbf{y} \in \mathcal{Y}$,

$$\begin{aligned}
\sum_{i,j=1}^m c_i \bar{c}_j \langle K_{\mathcal{H}}(\mathbf{x}_j, \mathbf{x}_i)\mathbf{y}, \mathbf{y}\rangle_{\mathcal{Y}} &= \left\langle \sum_{i,j=1}^m c_i \bar{c}_j K_{\mathcal{H}}(\mathbf{x}_j, \mathbf{x}_i)\mathbf{y}, \mathbf{y} \right\rangle_{\mathcal{Y}} \\
&= \left\langle \sum_{i,j=1}^m c_i \bar{c}_j \delta_{\mathbf{x}_j} \delta_{\mathbf{x}_i}^* \mathbf{y}, \mathbf{y} \right\rangle_{\mathcal{Y}} \\
&= \left\langle \sum_{i=1}^m c_i \delta_{\mathbf{x}_i}^* \mathbf{y}, \sum_{j=1}^m c_j \delta_{\mathbf{x}_j}^* \mathbf{y} \right\rangle_{\mathcal{H}} \\
&= \left\| \sum_{i=1}^m c_i \delta_{\mathbf{x}_i}^* \mathbf{y} \right\|_{\mathcal{H}}^2 \geq 0
\end{aligned}$$

□

Nous pouvons désormais introduire la notion de *feature map* pour un noyau à valeur opérateur. Pour cela, on considère un espace de Hilbert $\mathcal{W}$ et $\mathcal{L}(\mathcal{Y}, \mathcal{W})$ l'ensemble des opérateurs linéaires bornés de $\mathcal{Y}$ dans $\mathcal{W}$, une *feature map* associée à $K_{\mathcal{H}}$ est une fonction

$$\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{Y}, \mathcal{W})$$

telle que :

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathcal{X} \times \mathcal{X},\ K_{\mathcal{H}}(\mathbf{x}, \mathbf{z}) = \Phi(\mathbf{x})^* \Phi(\mathbf{z}) \tag{4.16}$$

Une *feature map* naturelle qui découle de (4.15) est définie par : $\forall \mathbf{x} \in \mathcal{X}, \Phi(\mathbf{x}) = \delta_{\mathbf{x}}^* \in \mathcal{L}(\mathcal{Y}, \mathcal{H})$.

Proposition 4.1. *Le noyau à valeur opérateur $K_{\mathcal{H}}$ admet la propriété* reproduisante *:*

$$\forall (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}, \forall f \in \mathcal{H},\ \langle f, K_{\mathcal{H}}(\cdot, \mathbf{x})\mathbf{y}\rangle_{\mathcal{H}} = \langle f(\mathbf{x}), \mathbf{y}\rangle_{\mathcal{Y}} \tag{4.17}$$

Démonstration. Soient $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$ et $f \in \mathcal{H}$,

$$\begin{aligned}
\langle f(\mathbf{x}), \mathbf{y}\rangle_{\mathcal{Y}} &= \langle \delta_{\mathbf{x}}(f), \mathbf{y}\rangle_{\mathcal{Y}} \\
&= \langle f, \delta_{\mathbf{x}}^* \mathbf{y}\rangle_{\mathcal{H}} \\
&= \langle f, K_{\mathcal{H}}(\cdot, \mathbf{x})\mathbf{y}\rangle_{\mathcal{H}}
\end{aligned}$$

□

Le théorème suivant affirme qu'un noyau à valeur opérateur semi-défini positif K définit de manière unique un certain RKHS à valeurs vectorielles.

Théorème 4.1. *(Pedrick, 1957; Senkene et Tempel'man, 1973; C. Micchelli et M. Pontil, 2005) Soit un noyau à valeur opérateur semi-défini positif $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$, il existe un unique RKHS sur $\mathcal{X}$ à valeurs dans $\mathcal{Y}$ qui admet K comme noyau reproduisant.*

Démonstration. Nous donnons des éléments de preuve issus de Schwartz (1964). Soient $\mathbf{x} \in \mathcal{X}$ et $\mathbf{y} \in \mathcal{Y}$, on définit la fonction $K_{\mathbf{x},\mathbf{y}} = K(\cdot, \mathbf{x})\mathbf{y} \in \mathcal{Y}^{\mathcal{X}}$ et le sous-espace vectoriel $\mathcal{H}_0$ de $\mathcal{Y}^{\mathcal{X}}$ par :

$$\mathcal{H}_0 = \text{Vect}\left\{K_{\mathbf{x},\mathbf{y}} \mid \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\right\}$$

Soient $f = \sum_{i=1}^{m} c_i K_{\mathbf{x}_i,\mathbf{y}_i}$ et $g = \sum_{j=1}^{n} d_j K_{\mathbf{z}_j,\mathbf{v}_j}$ deux éléments de $\mathcal{H}_0$. On définit une forme sesquilinéaire sur $\mathcal{H}_0 \times \mathcal{H}_0$ par :

$$\langle f, g\rangle_{\mathcal{H}_0} = \sum_{i=1}^{m}\sum_{j=1}^{n} c_i \bar{d}_j \left\langle K(\mathbf{z}_j, \mathbf{x}_i)\mathbf{y}_i, \mathbf{v}_j\right\rangle_{\mathcal{Y}}$$

On peut montrer aisément que $\langle\cdot,\cdot\rangle_{\mathcal{H}_0}$ est un produit scalaire sur $\mathcal{H}_0$. Ensuite on complète $\mathcal{H}_0$ en lui adjoignant les limites des suites de Cauchy. On vérifie alors que l'espace de Hilbert $(\mathcal{H}_K, \langle\cdot,\cdot\rangle_{\mathcal{H}_0})$ ainsi obtenu admet K comme noyau reproduisant. L'unicité de $\mathcal{H}_K$ découle de l'unicité de la complétion de $\mathcal{H}_0$. □

Considérons le cas où $\mathcal{Y} = \mathbb{R}^d$. L'ensemble des opérateurs linéaires de $\mathbb{R}^d$ dans $\mathbb{R}^d$, $\mathcal{L}(\mathbb{R}^d)$ est l'ensemble des matrices carrées d'ordre d. On dit alors que K est un *noyau à valeurs matricielles*. Pour $\mathbf{x}, \mathbf{z} \in \mathcal{X}$, $K(\mathbf{x}, \mathbf{z})$ est une matrice carrée d'ordre d dont les coefficients peuvent être obtenus via l'équation (4.17) :

$$K(\mathbf{x}, \mathbf{z})_{pq} = \langle K(\mathbf{x}, \mathbf{z})\mathbf{e}_q, \mathbf{e}_p\rangle_{\mathbb{R}^d} = \langle K(\cdot, \mathbf{x})\mathbf{e}_p, K(\cdot, \mathbf{z})\mathbf{e}_q\rangle_{\mathcal{H}} \quad (p, q) \in \mathbb{N}_d^2$$

où $\{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ est la base canonique de $\mathbb{R}^d$.

Nous faisons par la suite l'hypothèse que le noyau K est *continu* relativement à la norme d'opérateur sur $\mathcal{L}(\mathcal{Y})$. On note $\mathcal{C}(\mathcal{Z}, \mathcal{Y})$ l'espace de Banach des fonctions continues à valeurs dans $\mathcal{Y}$ sur un sous-ensemble compact $\mathcal{Z}$ de $\mathcal{X}$, muni de la norme uniforme définie par $\|f\|_{\infty,\mathcal{Z}} = \sup_{\mathbf{x}\in\mathcal{Z}} \|f(\mathbf{x})\|_{\mathcal{Y}}$. Nous notons ensuite, pour tout noyau à valeur opérateur K, le sous-espace de $\mathcal{C}(\mathcal{Z}, \mathcal{Y})$:

$$\mathcal{C}_K(\mathcal{Z}, \mathcal{Y}) = \overline{\text{Vect}\{K(\cdot, \mathbf{x})\mathbf{y} \mid \mathbf{x} \in \mathcal{Z}, \mathbf{y} \in \mathcal{Y}\}}$$

Nous pouvons à présent définir la notion d'universalité pour les noyaux à valeur opérateur.

Définition 4.3 (Noyau à valeur opérateur universel (Caponnetto, C. A. Micchelli et al., 2008))**.** *On dit que le noyau à valeur opérateur $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ est un* noyau universel *si, pour tout sous-ensemble compact $\mathcal{Z}$ de $\mathcal{X}$, $\mathcal{C}_K(\mathcal{Z}, \mathcal{Y}) = \mathcal{C}(\mathcal{Z}, \mathcal{Y})$*

4.3.1 Régularisation dans les RKHS à valeurs vectorielles

Comme dans le cas scalaire, la théorie des RKHS à valeurs vectorielles offre un cadre élégant pour traiter des problèmes d'apprentissage régularisés via notamment l'existence de théorèmes de représentation. La preuve de ces derniers nécessite quelques résultats intermédiaires impliquant la notion d'indépendance linéaire.

Définition 4.4. *Soit un ensemble* $\{\mathbf{x}_i\}_{i=1}^m \subseteq \mathcal{X}$ *de* m *points distincts deux à deux. On dit que les fonctions d'évaluation* $\{\delta_{\mathbf{x}_j}\}_{j=1}^m$*, définies, pour tout* $h \in \mathcal{H}$*, par* $\delta_{\mathbf{x}_j}(h) = h(\mathbf{x}_j)$*,* $(j \in \mathbb{N}_m)$ *sont* linéairement indépendantes *si, et seulement si, pour tout* $h \in \mathcal{H}$ *et tout* $\{\mathbf{c}_i\}_{i=1}^m \subseteq \mathcal{Y}$

$$\sum_{i=1}^{m} \langle \mathbf{c}_i, h(\mathbf{x}_i) \rangle_{\mathcal{Y}} = 0 \Rightarrow \forall i \in \mathbb{N}_m, \mathbf{c}_i = 0 \tag{4.18}$$

Lemme 4.2. *Les fonctionnelles* $\{\delta_{\mathbf{x}_j}\}_{j=1}^m$ *sont linéairement indépendantes si et seulement si pour tout* $\{\mathbf{y}_j\}_{j=1}^m \subseteq \mathcal{Y}$*, il existe un ensemble de coefficients* $\{\mathbf{c}_j\}_{j=1}^m \subseteq \mathcal{Y}$ *tel que :*

$$\sum_{\ell=1}^{m} K(\mathbf{x}_j, \mathbf{x}_\ell)\mathbf{c}_\ell = \mathbf{y}_j, \ \ j \in \mathbb{N}_m \tag{4.19}$$

Démonstration. On confère au produit cartésien $\mathcal{Y}^m$ une structure d'espace de Hilbert en le munissant du produit scalaire défini pour tout $\mathbf{a} = (\mathbf{a}_1^T \dots \mathbf{a}_m^T)^T \in \mathcal{Y}^m$ et $\mathbf{b} = (\mathbf{b}_1^T \dots \mathbf{b}_m^T)^T \in \mathcal{Y}^m$ par :

$$\langle \mathbf{a}, \mathbf{b} \rangle_{\mathcal{Y}^m} = \sum_{j=1}^{m} \langle \mathbf{a}_j, \mathbf{b}_j \rangle_{\mathcal{Y}}$$

Considérons l'opérateur linéaire borné $\mathbf{R} : \mathcal{H} \to \mathcal{Y}^m$ défini pour tout $h \in \mathcal{H}$ par :

$$\mathbf{R}h = (h(\mathbf{x}_1)^T \dots h(\mathbf{x}_m)^T)^T \tag{4.20}$$

Soit $\mathbf{c} = (\mathbf{c}_1^T \dots \mathbf{c}_m^T)^T \in \mathcal{Y}^m$, on peut identifier l'opérateur adjoint $\mathbf{R}^*$ grâce à la propriété reproduisante (4.17) :

$$\langle \mathbf{R}^*\mathbf{c}, h \rangle_{\mathcal{H}} = \langle \mathbf{c}, \mathbf{R}h \rangle_{\mathcal{Y}^m} = \sum_{\ell=1}^{m} \langle \mathbf{c}_\ell, h(\mathbf{x}_\ell) \rangle_{\mathcal{Y}} = \sum_{\ell=1}^{m} \langle K(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell, h \rangle_{\mathcal{H}}$$

Nous venons ainsi d'établir que : $\mathbf{R}^*\mathbf{c} = \sum_{\ell=1}^m K(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell$. Il vient alors que l'opérateur symétrique $\mathbf{R}\mathbf{R}^* : \mathcal{Y}^m \to \mathcal{Y}^m$ est défini par :

$$\mathbf{R}\mathbf{R}^*\mathbf{c} = \left(\left(\sum_{\ell=1}^{m} K(\mathbf{x}_1, \mathbf{x}_\ell)\mathbf{c}_\ell \right)^T \dots \left(\sum_{\ell=1}^{m} K(\mathbf{x}_m, \mathbf{x}_\ell)\mathbf{c}_\ell \right)^T \right)^T$$

Par suite, les assertions suivantes sont équivalentes :

$$\begin{aligned}
& \text{les fonctionnelles } \{\delta_{\mathbf{x}_j}\}_{j=1}^m \text{ sont linéairement indépendantes} \\
\Leftrightarrow\ & \forall h \in \mathcal{H}, \forall \mathbf{c} \in \mathcal{Y}^m, \langle \mathbf{c}, \mathbf{R}h \rangle_{\mathcal{Y}^m} = 0 \Rightarrow \mathbf{c} = 0 \\
\Leftrightarrow\ & \ker(\mathbf{R}^*) = \{0\} \\
\Leftrightarrow\ & \ker(\mathbf{R}\mathbf{R}^*) = \{0\} \\
\Leftrightarrow\ & \mathrm{Im}(\mathbf{R}\mathbf{R}^*) = \mathcal{Y}^m \\
\Leftrightarrow\ & \forall \mathbf{y} \in \mathcal{Y}^m, \exists \mathbf{c} \in \mathcal{Y}^m, \mathbf{R}\mathbf{R}^*\mathbf{c} = \mathbf{y}
\end{aligned}$$

La dernière assertion dans la suite d'équivalences sus-citée est exactement (4.19). □

Théorème 4.3. *Soient* $\{\mathbf{x}_i\}_{i=1}^m \subseteq \mathcal{X}$ *et* $\{\mathbf{y}_i\}_{i=1}^m \subseteq \mathcal{Y}$. *Si les fonctionnelles d'évaluation* $\{\delta_{\mathbf{x}_j}\}_{j=1}^m$ *sont linéairement indépendantes alors le problème*

$$\min_{h \in \mathcal{H}} \left\{ \|h\|_{\mathcal{H}}^2 \mid h(\mathbf{x}_i) = \mathbf{y}_i,\ i \in \mathbb{N}_m \right\} \tag{4.21}$$

admet une unique solution de la forme :

$$\hat{h} = \sum_{i=1}^m K(\cdot, \mathbf{x}_i)\mathbf{c}_i \tag{4.22}$$

où les paramètres $\{\mathbf{c}_i\}_{i=1}^m \subseteq \mathcal{Y}$ *sont les solutions du système linéaire d'équations*

$$\sum_{\ell=1}^m K(\mathbf{x}_j, \mathbf{x}_\ell)\mathbf{c}_\ell = \mathbf{y}_j,\ j \in \mathbb{N}_m \tag{4.23}$$

Démonstration. Soit $h \in \mathcal{H}$ tel que $h(\mathbf{x}_i) = \mathbf{y}_i,\ i \in \mathbb{N}_m$. Une telle fonction existe puisque nous avons montré que l'opérateur $\mathbf{R}$ défini en (4.20) était surjectif. En effet, $\ker(\mathbf{R}^*) = \{0\}$, or $\ker(\mathbf{R}^*) = \mathrm{Im}(\mathbf{R})^\perp$, donc $\mathrm{Im}(\mathbf{R}) = \mathcal{Y}^m$. Considérons à présent la fonction g définie par : $g = h - \hat{h}$, on a alors :

$$\|h\|_{\mathcal{H}}^2 = \|g + \hat{h}\|_{\mathcal{H}}^2 = \|g\|_{\mathcal{H}}^2 + 2\left\langle \hat{h}, g \right\rangle_{\mathcal{H}} \|\hat{h}\|_{\mathcal{H}}^2$$

Or $g(\mathbf{x}_j) = 0,\ j \in \mathbb{N}_m$, on obtient ainsi :

$$\left\langle \hat{h}, g \right\rangle_{\mathcal{H}} = \sum_{i=1}^m \langle K(\cdot, \mathbf{x}_i)\mathbf{c}_i, g \rangle_{\mathcal{H}} = \sum_{i=1}^m \langle \mathbf{c}_i, g(\mathbf{x}_i) \rangle_{\mathcal{Y}} = 0$$

Il s'ensuit :

$$\|h\|_{\mathcal{H}}^2 = \|g\|_{\mathcal{H}}^2 + \|\hat{h}\|_{\mathcal{H}}^2 \geq \|\hat{h}\|_{\mathcal{H}}^2 \tag{4.24}$$

Et on en conclut que $\hat{h}$ est l'unique solution de (4.21). □

Nous disposons désormais des éléments nécessaires à la démonstration du théorème du représentant qui s'énonce comme suit.

Théorème 4.4. *(C. Micchelli et M. Pontil, 2005) Soit $\mathcal{H}$ un RKHS sur $\mathcal{X}$ à valeurs dans $\mathcal{Y}$ qui admet $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ comme noyau reproduisant. Soient m points $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathcal{X}$. Soit une fonction $V : \mathcal{Y}^m \times \mathbb{R}_+ \to \mathbb{R}$. Pour $\mathbf{y} \in \mathcal{Y}^m$, si $V_\mathbf{y} : \mathbb{R}_+ \to \mathbb{R}_+$, définie pour $z \in \mathbb{R}_+$ par $V_\mathbf{y}(z) = V(\mathbf{y}, z)$, est une fonction strictement croissante, alors toute solution du problème :*

$$\operatorname*{argmin}_{h \in \mathcal{H}} V\left(h(\mathbf{x}_1), \ldots, h(\mathbf{x}_m), \|h\|_\mathcal{H}^2\right) \tag{4.25}$$

admet une représentation de la forme :

$$\hat{h} = \sum_{i=1}^{m} K(\cdot, \mathbf{x}_i)\mathbf{c}_i$$

où les paramètres $\mathbf{c}_1, \ldots, \mathbf{c}_m$ sont des vecteurs, éléments de $\mathcal{Y}$. De plus, si V est strictement convexe, alors le problème (4.25) admet une unique solution.

Démonstration. Soit $h \in \mathcal{H}$ une fonction telle que $h(\mathbf{x}_j) = \hat{h}(\mathbf{x}_j),\ j \in \mathbb{N}_m$. On pose $\mathbf{h}_{1:m} = \left(h(\mathbf{x}_1)^T \ldots h(\mathbf{x}_m)^T\right)^T \in \mathcal{Y}^m$. Par hypothèse, $\hat{h}$ vérifie l'inégalité :

$$V\left(\mathbf{h}_{1:m}, \|\hat{h}\|_\mathcal{H}^2\right) \leq V\left(\mathbf{h}_{1:m}, \|h\|_\mathcal{H}^2\right)$$

Par conséquent,

$$\|\hat{h}\|_\mathcal{H} = \min_{h \in \mathcal{H}} \left\{ \|h\|_\mathcal{H} \mid h(\mathbf{x}_i) = \hat{h}(\mathbf{x}_i),\ i \in \mathbb{N}_m \right\}$$

Le résultat découle alors de l'application du théorème 4.3. L'unicité de la solution vient de la stricte convexité de V. □

Le théorème 4.4 s'applique notamment dans le cas de problèmes supervisés à sorties vectorielles. Considérons un ensemble d'apprentissage de taille N, $\mathcal{S}_N = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$, un cas particulier du problème (4.25) est :

$$\operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{N} \ell\left(\mathbf{y}_i, h(\mathbf{x}_i)\right) + f\left(\|h\|_\mathcal{H}^2\right) \tag{4.26}$$

où $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ est une fonction de coût et $f : \mathbb{R}_+ \to \mathbb{R}_+$ est une fonction strictement croissante.

Kernel Ridge Regression. Nous pouvons étendre le cas de la Kernel Ridge Regression étudiée dans le cas scalaire au cas vectoriel. Il s'agit d'une instanciation particulière du problème (4.26) où

— ℓ est la fonction de perte quadratique : $\ell(\mathbf{y}, \mathbf{y}') = \|\mathbf{y} - \mathbf{y}'\|_2^2$ avec $\mathbf{y}, \mathbf{y}' \in \mathcal{Y}$

— $f(z) = \lambda z$ avec $\lambda > 0$ et $z \in \mathbb{R}_+$

De plus, comme dans le cas scalaire, les vecteurs de paramètres $\mathbf{c}_i$ s'obtiennent en forme close, comme l'énonce le théorème suivant.

Théorème 4.5. *(C. Micchelli et M. Pontil, 2005) Soit $\mathcal{H}$ un RKHS sur $\mathcal{X}$ à valeurs dans $\mathcal{Y}$ qui admet $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ comme noyau reproduisant. Étant donné un ensemble $\mathcal{S}_N = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$, d'après le théorème 4.4, le problème de minimisation :*

$$\operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{N} \|\mathbf{y}_i - h(\mathbf{x}_i)\|_2^2 + \lambda \|h\|_{\mathcal{H}}^2 \tag{4.27}$$

avec $\lambda > 0$, admet au plus une solution de la forme

$$\hat{h} = \sum_{i=1}^{N} K(\cdot, \mathbf{x}_i)\mathbf{c}_i$$

où les paramètres $\mathbf{c}_1, \ldots, \mathbf{c}_N \in \mathcal{Y}$ sont solutions du système d'équations linéaires :

$$\sum_{\ell=1}^{N} \left(K(\mathbf{x}_j, \mathbf{x}_\ell) + \lambda \delta_{j\ell}\right) \mathbf{c}_\ell = \mathbf{y}_j, \quad j \in \mathbb{N}_N \tag{4.28}$$

où δ est le symbole de Kronecker.

Dans le cas où $\mathcal{Y} = \mathbb{R}^d$, on a l'écriture matricielle suivante pour les paramètres $\{\mathbf{c}_j\}_{j=1}^N \subseteq \mathbb{R}^d$:

$$\mathbf{c} = (\mathbf{K} + \lambda Id)^{-1} \mathbf{y} \tag{4.29}$$

où $\mathbf{c} = \begin{pmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_N \end{pmatrix} \in \mathbb{R}^{Nd}$, $\mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{pmatrix} \in \mathbb{R}^{Nd}$, $\mathbf{K} = \begin{pmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \cdots & K(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & & \vdots \\ K(\mathbf{x}_N, \mathbf{x}_1) & \cdots & K(\mathbf{x}_N, \mathbf{x}_N) \end{pmatrix} \in \mathbb{R}^{Nd \times Nd}$ est une matrice par blocs où le bloc (i, j) correspond à la matrice $K(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{R}^{d \times d}$ et Id est la matrice identité de dimension $Nd \times Nd$.

4.3.2 RKBS et Régularisation parcimonieuse

En outre, de la même manière que pour le LASSO (cf. section 2.3.2), on peut souhaiter pénaliser *a priori* la complexité du modèle afin d'obtenir une solution parcimonieuse, c'est-à-dire que la plupart des paramètres $\mathbf{c}_i$ du modèle appris $\hat{h} = \sum_i K(\cdot, \mathbf{x}_i)\mathbf{c}_i$ soient nuls. Une

première tentative en ce sens a été proposée avec une approche utilisant la programmation linéaire (B. Schölkopf et A. Smola, 2001). Cependant la méthode manque d'arguments théoriques. Pour établir un fondement approprié dans le cas d'une régularisation de type ℓ_1 avec des noyaux, la notion d'espace de *Banach* à noyau reproduisant (Reproducing Kernel Banach Space, RKBS) (H. Zhang, Xu et J. Zhang, 2009) a été introduite pour les fonctions à valeurs scalaires. En travaillant dans ces espaces plus généraux, il est alors possible de prouver des théorèmes de représentation avec des régularisations parcimonieuses (Song et al., 2013).

Nous proposons de donner ici quelques éléments de construction des RKBS telle qu'elle est présentée dans Song et al. (2013). Nous invitons le lecteur intéressé à lire l'article cité précédemment, notamment pour les démonstrations des propositions données ci-après.

Définition 4.5 (Pré-RKBS)**.** *Soit $\mathcal{B} \subseteq \mathbb{C}^{\mathcal{X}}$ un espace de Banach de fonctions sur $\mathcal{X}$, muni d'une norme $\|\cdot\|_{\mathcal{B}}$. On dit que $\mathcal{B}$ est un pré-RKBS si toutes les fonctions d'évaluation sont des opérateurs linéaires continus sur $\mathcal{B}$, i.e. pour tout $\mathbf{x} \in \mathcal{X}$, il existe une constante $C_{\mathbf{x}}$ telle que :*

$$\forall f \in \mathcal{B},\ |\delta_{\mathbf{x}}(f)| = |f(\mathbf{x})| \leq C_{\mathbf{x}} \|f\|_{\mathcal{B}}$$

Définition 4.6 (Norme consistante)**.** *On dit qu'un espace vectoriel normé $\mathcal{E} \subseteq \mathbb{C}^{\mathcal{X}}$ de fonctions sur $\mathcal{X}$, muni d'une norme $\|\cdot\|_{\mathcal{E}}$, est de* norme consistante *si pour toute suite de Cauchy $(f_n)_{n\in\mathbb{N}}$ dans $\mathcal{E}$, on a :*

$$\forall \mathbf{x} \in \mathcal{X},\ \lim_{n\to\infty} f_n(\mathbf{x}) = 0 \Rightarrow \lim_{n\to\infty} \|f_n\|_{\mathcal{E}} = 0 \tag{4.30}$$

Considérons à présent, une fonction $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$. Dans ce cas et contrairement aux RKHS, Γ n'est plus nécessairement ni symétrique ni semi-définie positive. Nous introduisons l'espace vectoriel $\mathcal{B}_0$ défini par :

$$\mathcal{B}_0 = \mathrm{Vect}\{\Gamma(\mathbf{x}, \cdot) \mid \mathbf{x} \in \mathcal{X}\} \tag{4.31}$$

On suppose que $\mathcal{B}_0$ est muni d'une norme $\|\cdot\|_{\mathcal{B}_0}$ de telle sorte que toutes les fonctions d'évaluation sont des opérateurs linéaires continus sur $\mathcal{B}_0$. Puis on complète $\mathcal{B}_0$ par les limites des suites de Cauchy. L'espace $\mathcal{B}_\Gamma$ ainsi obtenu est alors un espace de Banach de fonctions sur $\mathcal{X}$.

Proposition 4.2. *L'espace de Banach $\mathcal{B}_\Gamma$ est un pré-RKBS si et seulement si $\|\cdot\|_{\mathcal{B}_0}$ est une norme consistante pour $\mathcal{B}_0$.*

On suppose dans la suite de cette section que $(\mathcal{B}_0, \|\cdot\|_{\mathcal{B}_0})$ est de norme consistante. Le

but est de définir la notion de noyau reproduisant pour $\mathcal{B}$. À cette fin, nous posons :

$$\mathcal{B}_0^{\#} = \text{Vect}\{\Gamma(\cdot, \mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\} \tag{4.32}$$

Nous définissons ensuite une forme bilinéaire $\langle \cdot, \cdot \rangle_\Gamma$ sur $\mathcal{B}_0 \times \mathcal{B}_0^{\#}$ par :

$$\begin{aligned} \forall m, n \in \mathbb{N}, \forall \{(a_i, \mathbf{r}_i)\}_{i=1}^n \subseteq \mathbb{C} \times \mathcal{X}, \forall \{(b_j, \mathbf{s}_j)\}_{j=1}^m \subseteq \mathbb{C} \times \mathcal{X}, \\ \left\langle \sum_{i=1}^n a_i \Gamma(\mathbf{r}_i, \cdot), \sum_{j=1}^m b_j \Gamma(\cdot, \mathbf{s}_j) \right\rangle_\Gamma = \sum_{i=1}^n \sum_{j=1}^m a_i b_j \Gamma(\mathbf{r}_i, \mathbf{s}_j) \end{aligned} \tag{4.33}$$

On a immédiatement les deux propriétés suivantes :

$$\forall f \in \mathcal{B}_0, \forall \mathbf{s} \in \mathcal{X}, \langle f, \Gamma(\cdot, \mathbf{s}) \rangle_\Gamma = f(\mathbf{s}) \tag{4.34}$$

$$\forall g \in \mathcal{B}_0^{\#}, \forall \mathbf{r} \in \mathcal{X}, \langle \Gamma(\mathbf{r}, \cdot), g \rangle_\Gamma = g(\mathbf{r}) \tag{4.35}$$

Nous définissons une norme $\| \cdot \|_{\mathcal{B}_0^{\#}}$ sur $\mathcal{B}_0^{\#}$:

$$\forall g \in \mathcal{B}_0^{\#}, \quad \|g\|_{\mathcal{B}_0^{\#}} = \sup_{f \in \mathcal{B}_0, f \neq 0} \frac{|\langle f, g \rangle_\Gamma|}{\|f\|_{\mathcal{B}_0}} \tag{4.36}$$

De la même manière que pour $\mathcal{B}_0$, on obtient un espace de Banach de fonctions sur $\mathcal{X}$, noté $\mathcal{B}_\Gamma^{\#}$ en complétant $\mathcal{B}_0^{\#}$ avec les limites des suites de Cauchy. On a également une proposition semblable à celle pour $\mathcal{B}$.

Proposition 4.3. *L'espace de Banach $\mathcal{B}_\Gamma^{\#}$ est un pré-RKBS si et seulement si $\| \cdot \|_{\mathcal{B}_0^{\#}}$ est une norme consistante pour $\mathcal{B}_0^{\#}$.*

Si $(\mathcal{B}_0^{\#}, \|\cdot\|_{\mathcal{B}_0^{\#}})$ est de norme consistante, on peut alors appliquer le théorème d'extension de Hahn-Banach deux fois ; ainsi la forme bilinéaire $\langle \cdot, \cdot \rangle_\Gamma$ définie originellement sur $\mathcal{B}_0 \times \mathcal{B}_0^{\#}$ peut être étendue sur $\mathcal{B}_\Gamma \times \mathcal{B}_\Gamma^{\#}$ de manière unique de sorte à vérifier l'inégalité suivante de type Cauchy-Schwarz :

$$\forall (f, g) \in \mathcal{B}_\Gamma \times \mathcal{B}_\Gamma^{\#}, \quad |\langle f, g \rangle_\Gamma| \leq \|f\|_{\mathcal{B}_\Gamma} \|g\|_{\mathcal{B}_\Gamma^{\#}} \tag{4.37}$$

Nous sommes prêts à donner la définition d'un RKBS.

Théorème 4.6 (RKBS)**.** *Soit une fonction $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$. Soient $\mathcal{B}_0$ et $\mathcal{B}_0^{\#}$ définis respectivement par (4.31) et (4.32) deux espaces vectoriels normés munis respectivement des normes consistantes $\| \cdot \|_{\mathcal{B}_0}$ et $\| \cdot \|_{\mathcal{B}_0^{\#}}$. Alors les espaces complétés respectifs $\mathcal{B}_\Gamma$ et $\mathcal{B}_\Gamma^{\#}$ sont des pré-RKBS et Γ admet la propriété reproduisante pour la forme bilinéaire $\langle \cdot, \cdot \rangle_\Gamma$*

définie par (4.33) et étendue via (4.37), i.e.

$$\forall f \in \mathcal{B}_\Gamma, \forall \mathbf{s} \in \mathcal{X}, \langle f, \Gamma(\cdot, \mathbf{s})\rangle_\Gamma = f(\mathbf{s}) \tag{4.38}$$

et

$$\forall g \in \mathcal{B}_\Gamma^\#, \forall \mathbf{r} \in \mathcal{X}, \langle \Gamma(\mathbf{r}, \cdot), g\rangle_\Gamma = g(\mathbf{r}) \tag{4.39}$$

On dit alors que $\mathcal{B}_\Gamma$ et $\mathcal{B}_\Gamma^\#$ sont des espaces de Banach à noyau reproduisant (RKBS) de noyau reproduisant Γ.

Dès lors, on souhaite pouvoir définir sur ces espaces fonctionnels une norme ℓ_1. Song et al. (2013) montrent que pour que de tels espaces possèdent une norme ℓ_1, le noyau reproduisant Γ doit appartenir à une certaine classe de fonctions. À cette fin, nous introduisons l'espace de Banach $\ell^1(E)$ des fonctions sur un ensemble E intégrables par rapport à la mesure de comptage sur E :

$$\ell^1(E) = \left\{ \mathbf{c} = (c_i \in \mathbb{C},\ i \in E) \mid \sum_{i \in E} |c_i| < +\infty \right\}$$

Pour $\mathbf{c} \in \ell^1(E)$, $\text{supp}\ \mathbf{c} = \{i \in E \mid c_i \neq 0\}$ est dénombrable et on note $\|\mathbf{c}\|_{\ell^1(E)} = \sum_{i \in E} |c_i|$.

Définition 4.7 (Noyau admissible)**.** *Une fonction $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ est un noyau* admissible *pour la construction de RKBS sur $\mathcal{X}$ avec une norme ℓ_1 si Γ vérifie les propriétés suivantes :*

(A) pour tout ensemble de points distincts deux à deux $\mathbf{x} = \{\mathbf{x}_j\}_{j=1}^n \subseteq \mathcal{X}$, la matrice

$$\Gamma[\mathbf{x}] = (\Gamma(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^n \in \mathbb{C}^{n \times n}$$

est non singulière

(B) Γ est bornée, i.e. il existe une constante C telle que :

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathcal{X} \times \mathcal{X}, \quad |\Gamma(\mathbf{x}, \mathbf{z})| \leq C$$

(C) pour toute suite de points distincts deux à deux $(\mathbf{x}_n)_{n \in \mathbb{N}}$ et pour $\mathbf{c} \in \ell^1(\mathbb{N})$, on a :

$$\forall \mathbf{x} \in \mathcal{X}, \ \sum_{j=1}^{\infty} c_j \Gamma(\mathbf{x}_j, \mathbf{x}) = 0 \Rightarrow \mathbf{c} = 0$$

(D) pour tout ensemble de points distincts deux à deux $\mathbf{x} = \{\mathbf{x}_j\}_{j=1}^{n+1} \subseteq \mathcal{X}$,

$$\left\| (\Gamma[\mathbf{x}])^{-1} \Gamma_{\mathbf{x}}[\mathbf{x}_{n+1}] \right\|_{\ell^1(\mathbb{N}_n)} \leq 1$$

où pour $\mathbf{z} \in \mathcal{X}$, $\Gamma_{\mathbf{x}}(\mathbf{z}) = (\Gamma(\mathbf{z}, \mathbf{x}_1) \dots \Gamma(\mathbf{z}, \mathbf{x}_n))^T \in \mathbb{C}^n$.

Les RKBS construits à partir de noyaux admissibles jouissent alors d'un théorème du représentant semblable à celui existant pour les RKHS.

Théorème 4.7 (Théorème du représentant pour les RKBS à noyau reproduisant admissible)**.** *Soit* $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ *un noyau admissible. Alors les espaces de fonctions*

$$\mathcal{B}_\Gamma = \left\{ \sum_{\mathbf{z} \in supp\ \mathbf{c}} c_{\mathbf{z}} \Gamma(\mathbf{z}, \cdot) \mid \mathbf{c} \in \ell^1(\mathcal{X}) \right\} \text{ muni de la norme } \left\| \sum_{\mathbf{z} \in supp\ \mathbf{c}} c_{\mathbf{z}} \Gamma(\mathbf{z}, \cdot) \right\|_{\mathcal{B}_\Gamma} = \|\mathbf{c}\|_{\ell^1(\mathcal{X})}$$

et $\mathcal{B}_\Gamma^\#$ *le complété de l'espace vectoriel Vect*$\{\Gamma(\cdot, \mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\}$ *muni de la norme supremum*

$$\left\| \sum_{j=1}^n c_j \Gamma(\cdot, \mathbf{x}_j) \right\|_{\mathcal{B}_\Gamma^\#} = \sup_{\mathbf{x} \in \mathcal{X}} \left| \sum_{j=1}^n c_j \Gamma(\mathbf{x}, \mathbf{x}_j) \right|$$

sont tous deux des RKBS de noyau reproduisant Γ*. De plus, pour toute solution du problème de la forme :*

$$\inf_{h \in \mathcal{B}_\Gamma} V\left(h(\mathbf{x}_1), \dots, h(\mathbf{x}_m)\right) + \lambda \phi(\|h\|_{\mathcal{B}_\Gamma}) \tag{4.40}$$

où $\lambda > 0$ *est un hyperparamètre de régularisation,* $V : \mathbb{C}^m \to \mathbb{R}_+$, $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ *sont deux fonctions continues avec* $\lim_{s \to +\infty} \phi(s) = +\infty$*, il existe un minimiseur* h_0 *de la forme :*

$$h_0 = \sum_{j=1}^m c_j \Gamma(\mathbf{x}_j, \cdot) \tag{4.41}$$

avec $\{c_j\}_{j=1}^m \subseteq \mathbb{C}$.

Une extension de cette théorie au cas vectoriel a été proposée par H. Zhang et J. Zhang (2013).

4.4 Une nouvelle famille de modèles autorégressifs non paramétriques : OKVAR

Revenons au problème d'autorégression vectorielle (4.1) :

$$\mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_{t+1}, \quad t \in \mathbb{N}_N$$

On rappelle que l'objectif est d'estimer un modèle autorégressif $\hat{h} : \mathbb{R}^d \to \mathbb{R}^d$ à partir d'un ensemble d'apprentissage $\mathcal{S}_N = \{(\mathbf{x}_\ell, \mathbf{x}_{\ell+1})\}_{\ell=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^d$ issu de l'observation d'une série temporelle $\mathbf{x}_1, \dots, \mathbf{x}_{N+1} \in \mathbb{R}^d$. Dans ce cadre, l'ensemble d'entrée $\mathcal{X}$ est $\mathbb{R}^d$ et l'espace de

Hilbert de sortie $\mathcal{Y}$ est aussi $\mathbb{R}^d$ muni du produit scalaire canonique $\langle \cdot, \cdot \rangle_{\mathbb{R}^d}$. Étant donné un noyau à valeur matricielle $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$, une manière d'estimer h est de suivre une approche dictée par le paradigme de la régularisation et de résoudre le problème de minimisation :

$$\underset{h \in \mathcal{H}_K}{\operatorname{argmin}} \sum_{t=1}^{N} \ell\left(\mathbf{x}_{t+1}, h(\mathbf{x}_t)\right) + \Omega(h)$$

où $\mathcal{H}_K$ est un sous-ensemble de l'espace des fonctions de $\mathbb{R}^d$ dans $\mathbb{R}^d$, $\ell : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ une fonction de coût et $\Omega : \mathcal{H}_K \to \mathbb{R}_+$ une fonction de régularisation. Nous nous intéressons au cas particulier où ℓ est la fonction de perte quadratique :

$$\underset{h \in \mathcal{H}_K}{\operatorname{argmin}} \sum_{\ell=1}^{N} \|\mathbf{x}_{\ell+1} - h(\mathbf{x}_\ell)\|_2^2 + \Omega(h) \tag{4.42}$$

On rappelle que l'introduction de la fonction de perte quadratique est issue de l'estimation par maximum de vraisemblance en présence de résidus homoscédastiques distribués selon une loi gaussienne $\mathcal{N}(0, \sigma_{\mathbf{u}}^2 Id)$ (cf. Équation (4.8)). Nous avons vu que dans certaines conditions, (Théorèmes 4.4 pour les RKHS et 4.7 pour les RKBS), la solution du problème (4.42) admet une représentation de la forme :

$$\hat{h} = \sum_{\ell=1}^{N} K(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell$$

et où les paramètres $\{\mathbf{c}_\ell\}_{\ell=1}^N \subseteq \mathbb{R}^d$ sont à estimer.

Dans cette thèse, nous étudions les modèles autorégressifs vectoriels non paramétriques définis par :

$$h = \sum_{\ell=1}^{N} K(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell \tag{4.43}$$

où $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\} \subseteq \mathbb{R}^d$ est une série temporelle observée, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$ est un noyau à valeur matricielle et $\{\mathbf{c}_\ell\}_{\ell=1}^N \subseteq \mathbb{R}^d$ sont des paramètres à estimer. Nous appelons modèle Vectoriel Autorégressif à base de Noyau à valeur Opérateur (en anglais Operator-valued Kernel-based Vector Autoregressive, OKVAR) tout modèle vectoriel autorégressif de la forme donnée par l'équation (4.43) (Néhémy Lim et al., 2014). Par la suite, nous examinons différents noyaux matriciels et les propriétés des modèles correspondants.

4.4.1 La famille de modèles OKVAR

À l'instar du cas scalaire où toute constante positive est un noyau scalaire, un premier exemple très simple de noyau à valeur matricielle (Carmeli, De Vito, Toigo et

Umanitá, 2010) est donné par :

$$\forall(\mathbf{x},\mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K(\mathbf{x},\mathbf{z}) = B \tag{4.44}$$

où $B \in \mathbb{S}_+^d$ est une matrice symétrique semi-définie positive.

Démonstration. Trivialement, $K(\mathbf{x},\mathbf{z}) = K(\mathbf{z},\mathbf{x})^T = B$. Vérifions la propriété de semi-définie positivité : soit $m \in \mathbb{N}$ et soit $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^m \subseteq \mathbb{R}^d \times \mathbb{R}^d$, on a :
$\sum_{i,j=1}^m \langle \mathbf{y}_i, K(\mathbf{x}_i,\mathbf{x}_j)\mathbf{y}_j\rangle_{\mathbb{R}^d} = \sum_{i,j=1}^m \langle B\mathbf{y}_i, \mathbf{y}_j\rangle_{\mathbb{R}^d} \geq 0$ □

Rappelons que dans le cas scalaire, parmi les noyaux les plus simples, on trouve les noyaux linéaires de la forme $k(\mathbf{x},\mathbf{z}) = \mathbf{x}^T Q\mathbf{z}$, $(\mathbf{x},\mathbf{z} \in \mathbb{R}^d)$ où $Q \in \mathbb{S}_+^d$ est une matrice symétrique semi-définie positive. La première question qui vient à l'esprit est de savoir quel est leur équivalent dans le cas vectoriel. Quelle est la forme d'un noyau linéaire à valeur matricielle $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$?

C. A. Micchelli et Massimiliano Pontil (2004) affirment que lorsque $\mathcal{Y} = \mathbb{R}^d$, les noyaux matriciels linéaires $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ sont de la forme

$$K(\mathbf{x},\mathbf{z})_{pq} = \langle A_p\mathbf{x}, A_q\mathbf{z}\rangle_{\mathbb{R}^m}, \quad \mathbf{x},\mathbf{z} \in \mathbb{R}^d \tag{4.45}$$

où A_p $(p \in \mathbb{N}_d)$ est une matrice de taille $n \times d$ pour un certain $n \in \mathbb{N}$. Pour le voir, il suffit de passer par une représentation en *feature map* du noyau K :

$$\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{Y},\mathcal{W})$$

telle que : pour $\forall(\mathbf{x},\mathbf{z}) \in \mathcal{X} \times \mathcal{X}$

$$K(\mathbf{x},\mathbf{z}) = \Phi(\mathbf{x})^*\Phi(\mathbf{z})$$

Dans notre cas, $\mathcal{X} = \mathcal{Y} = \mathbb{R}^d$, en choisissant $\mathcal{W} = \mathbb{R}^n$ $(n \in \mathbb{N})$, on peut identifier $\mathcal{L}(\mathbb{R}^d,\mathbb{R}^n)$ à l'ensemble des matrices de taille $n \times d$. Pour $(p,q) \in \mathbb{N}_d^2$, on a alors :

$$K(\mathbf{x},\mathbf{z})_{pq} = \sum_{r=1}^n \Phi(\mathbf{x})_{rp}\Phi(\mathbf{z})_{rq} \tag{4.46}$$

Une fonction de redescription linéaire de $\mathbf{x}$ consiste alors à choisir $\Phi(\mathbf{x})$ telle que, pour $p \in \mathbb{N}_d$, $(\Phi(\mathbf{x})_{rp})_{r=1}^n = A_p\mathbf{x}$, avec $A_p \in \mathbb{R}^{n\times d}$ et on retrouve bien la forme donnée par l'équation (4.45). Voici quelques choix spéciaux de matrices A_p :

— Lorsque A_p, $p \in \mathbb{N}_d$, est la matrice identité de taille $d \times d$, alors le noyau matriciel se réduit à :

$$K(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z}\rangle_{\mathbb{R}^d} Id \tag{4.47}$$

On retrouve ainsi un noyau linéaire scalaire appliqué à chaque dimension.

— Lorsque pour $p \in \mathbb{N}_d$, $A_p = \mathbf{e}_p^T$ où $\mathbf{e}_p$ est le p-ème vecteur de la base canonique de $\mathbb{R}^d$, on a alors $\mathbf{e}_p^T \mathbf{x} = x^p$, la p-ème coordonnée de $\mathbf{x}$. Le terme général du noyau matriciel linéaire correspondant devient :

$$K(\mathbf{x}, \mathbf{z})_{pq} = x^p z^q \tag{4.48}$$

Ce noyau compare alors les coordonnées de $\mathbf{x}$ et de $\mathbf{z}$.

Afin de faciliter la présentation des noyaux à valeur matricielle, nous utilisons une généralisation du produit de Schur des noyaux scalaires aux noyaux à valeur matricielle. Dans le cas où $\mathcal{Y} = \mathbb{R}^d$, la propriété de semi-définie positivité des noyaux à valeur opérateur (Eq. (4.11)) devient :

$$\forall m \in \mathbb{N}, \forall \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{R}^d, \sum_{i,j=1}^{m} \sum_{p,q=1}^{d} y_i^p K(\mathbf{x}_i, \mathbf{x}_j)_{pq} y_j^q \geq 0 \tag{4.49}$$

On conclut de l'équation (4.49) que K est un noyau à valeur matricielle si, et seulement si, la matrice $(K(\mathbf{x}_i, \mathbf{x}_j)_{p,q})$ d'indice ligne $(p, i) \in \mathbb{N}_d \times \mathbb{N}_m$ et d'indice colonne $(q, j) \in \mathbb{N}_d \times \mathbb{N}_m$ est semi-définie positive. Ainsi, dès lors que $\mathcal{Y}$ est de dimension finie, cette observation permet de prouver le caractère semi-défini positif d'une fonction à valeur opérateur en recourant à des astuces analogues à celles utilisées pour les noyaux à valeur scalaire. Nous illustrons ce principe par la proposition suivante.

Proposition 4.4 (Caponnetto, C. A. Micchelli et al. (2008))**.** *Soient K_1 et K_2 deux noyaux à valeur matricielle. Alors le produit terme à terme ou produit de Hadamard de K_1 et de K_2, $K_1 \circ K_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^d$ défini, pour tout $(\mathbf{x}, \mathbf{z}) \in \mathcal{X}^2$ et pour tout $(p, q) \in \mathbb{N}_d^2$, par*

$$(K_1 \circ K_2(\mathbf{x}, \mathbf{z}))_{pq} = K_1(\mathbf{x}, \mathbf{z})_{pq} K_2(\mathbf{x}, \mathbf{z})_{pq} \tag{4.50}$$

est un noyau à valeur matricielle.

Démonstration. Soit $m \in \mathbb{N}$ et un ensemble de m couples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{R}^d$, on a :

$$\sum_{i,j=1}^{m} \langle \mathbf{y}_i, K_1 \circ K_2(\mathbf{x}_i, \mathbf{x}_j) \mathbf{y}_j \rangle_{\mathbb{R}^d} = \sum_{p,i} \sum_{q,j} y_i^p y_j^q K_1(\mathbf{x}_i, \mathbf{x}_j)_{pq} K_2(\mathbf{x}_i, \mathbf{x}_j)_{pq} \tag{4.51}$$

K_1 et K_2 étant des noyaux à valeur matricielle, les matrices $(K_1(\mathbf{x}_i, \mathbf{x}_j)_{p,q})$ et $(K_2(\mathbf{x}_i, \mathbf{x}_j)_{p,q})$ d'indice ligne $(p, i) \in \mathbb{N}_d \times \mathbb{N}_m$ et d'indice colonne $(q, j) \in \mathbb{N}_d \times \mathbb{N}_m$ sont semi-définies positives. Le lemme de Schur (Aronszajn, 1950), qui affirme que le produit de Hadamard de deux matrices semi-définies positives est également une matrice semi-définie positive,

garantit la positivité de l'expression donnée par l'équation (4.51), ce qui achève la démonstration.

□

Proposition 4.5. *Soient $k_1, \dots, k_d$ des noyaux scalaires sur $\mathbb{R}^d$. Alors la fonction $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ définie par :*

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K(\mathbf{x}, \mathbf{z}) = diag(k_1(\mathbf{x}, \mathbf{z}), \dots, k_d(\mathbf{x}, \mathbf{z})) = \begin{pmatrix} k_1(\mathbf{x}, \mathbf{z}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & k_d(\mathbf{x}, \mathbf{z}) \end{pmatrix} \tag{4.52}$$

est un noyau matriciel.

Démonstration. On a immédiatement : pour tout $(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d$, $K(\mathbf{x}, \mathbf{z}) = K(\mathbf{z}, \mathbf{x})^T$. Pour $m \in \mathbb{N}$ et étant donné un ensemble $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{R}^d$,

$$\sum_{i,j=1}^{m} \sum_{p,q=1}^{d} y_i^p K(\mathbf{x}_i, \mathbf{x}_j)_{pq} y_j^q = \sum_{i,j=1}^{m} \sum_{p=1}^{d} y_i^p y_j^p k_p(\mathbf{x}_i, \mathbf{x}_j) \tag{4.53}$$

Or, pour $p_0 \in \mathbb{N}_d$, la propriété de semi-définie positivité du noyau scalaire k_{p_0} implique : $\sum_{i,j=1}^m y_i^{p_0} y_j^{p_0} k_{p_0}(\mathbf{x}_i, \mathbf{x}_j) \geq 0$. Ceci prouve la positivité de l'expression (4.53). □

Le noyau diagonal (4.52) n'est rien d'autre que l'application d'un noyau scalaire à chaque dimension de sortie, ce qui revient à traiter chaque tâche de manière indépendante.

Un autre exemple est issu de C. Micchelli et M. Pontil (2005).

Proposition 4.6. *(C. Micchelli et M. Pontil, 2005 ; Caponnetto, C. A. Micchelli et al., 2008) Si $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ est un noyau scalaire et $B \in \mathbb{S}_+^d$ est une matrice semi-définie positive de $\mathbb{R}^{d\times d}$, alors la fonction $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ définie par :*

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K(\mathbf{x}, \mathbf{z}) = k(\mathbf{x}, \mathbf{z})B \tag{4.54}$$

est un noyau matriciel.

Démonstration. Soient $\mathbf{x}, \mathbf{z} \in \mathcal{X}$ on vérifie aisément : $K(\mathbf{x}, \mathbf{z})^T = K(\mathbf{z}, \mathbf{x})$. En effet, $K(\mathbf{x}, \mathbf{z})^T = k(\mathbf{x}, \mathbf{z})B^T = k(\mathbf{z}, \mathbf{x})B = K(\mathbf{z}, \mathbf{x})$.

Par suite, soit $m \in \mathbb{N}$ et un ensemble de m couples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{R}^d$, k étant un noyau scalaire, la matrice de Gram $(k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^m$ est semi-définie positive. Il en est de même de la matrice $(\langle B\mathbf{y}_i, \mathbf{y}_j \rangle_{\mathbb{R}^d})_{i,j=1}^m$ de par la nature de B. En appliquant le lemme de Schur, la matrice $(k(\mathbf{x}_i, \mathbf{x}_j) \langle B\mathbf{y}_i, \mathbf{y}_j \rangle_{\mathbb{R}^d})_{i,j=1}^m$ est semi-définie positive. On en conclut le caractère semi-défini positif de K. □

De manière générale, on a

Proposition 4.7. *(Caponnetto, C. A. Micchelli et al., 2008) Soit $m \in \mathbb{N}$, si pour $j \in \mathbb{N}_m$, $k_j : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ est un noyau scalaire et $B_j \in \mathbb{S}_+^d$ est une matrice semi-définie positive de $\mathbb{R}^{d\times d}$, alors la fonction $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ définie par :*

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d, \ K(\mathbf{x}, \mathbf{z}) = \sum_{j=1}^{m} k_j(\mathbf{x}, \mathbf{z}) B_j \tag{4.55}$$

est un noyau matriciel.

Les noyaux de la forme (4.55) sont appelés noyaux *décomposables* ou *séparables* et ont notamment été utilisés dans le contexte de l'apprentissage multi-tâches (Evgeniou et al., 2005). Les matrices B_j encodent les dépendances entre les tâches ou sorties. Le vocable de « décomposable » est ainsi justifié puisque le traitement des entrées et des sorties est *découplé* avec d'une part des noyaux k_j agissant sur les entrées et d'autre part des opérateurs B_j qui encodent les interactions entre sorties sans référence aux entrées. L'exemple le plus simple consiste à choisir $m = 1$, $B = Id$ la matrice identité de taille $d \times d$. Dans ce cas, $K(\mathbf{x}, \mathbf{z})_{pq} = k(\mathbf{x}, \mathbf{z})\delta_{pq}$, on considère alors que toutes les tâches à apprendre sont très différentes les unes des autres et toutes les sorties sont en conséquence traitées de manière indépendante.

Comme nous faisons l'hypothèse que les séries temporelles d'intérêt sont non linéaires, nous nous concentrons sur des noyaux non linéaires. Nous notons $k_{\text{Gauss}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ le noyau gaussien scalaire dont nous rappelons la définition : $k(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|_2^2)$ pour $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$ avec $\gamma > 0$. Nous définissons ensuite un type particulier de noyaux décomposables non linéaires.

Définition 4.8 (Noyau décomposable gaussien)**.** *On appelle* noyau décomposable gaussien *et on note $K_{dec} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ tout noyau matriciel décomposable de la forme*

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d, \ K_{dec}(\mathbf{x}, \mathbf{z}) = k_{Gauss}(\mathbf{x}, \mathbf{z}) B \tag{4.56}$$

avec $B \in \mathbb{S}_+^d$

Définition 4.9 (Modèle OKVAR décomposable gaussien)**.** *Soit une série temporelle $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\} \subseteq \mathbb{R}^d$, le modèle OKVAR décomposable gaussien noté h_{dec} associé au noyau décomposable gaussien est défini par :*

$$h_{dec} = \sum_{\ell=1}^{N} K_{dec}(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell \tag{4.57}$$

Une autre classe de noyaux est celle des noyaux transformables.

Proposition 4.8 (Noyau transformable)**.** *(*Caponnetto*, C. A.* Micchelli *et al., 2008) Soient $\mathcal{X}_0$ un espace de Hausdorff (ou espace séparé) et $k : \mathcal{X}_0 \times \mathcal{X}_0 \to \mathbb{R}$ un noyau scalaire et pour $p \in \mathbb{N}_d$, T_p est une fonction de $\mathbb{R}^d$ dans $\mathcal{X}_0$ alors la fonction $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ définie par :*

$$\forall(\mathbf{x},\mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K(\mathbf{x},\mathbf{z})_{pq} = k\left(T_p(\mathbf{x}), T_q(\mathbf{z})\right),\quad (p,q) \in \mathbb{N}_d^2 \tag{4.58}$$

est un noyau matriciel.

Démonstration. Soient $\mathbf{x},\mathbf{z} \in \mathcal{X}$, vérifions qu'on a : $K(\mathbf{x},\mathbf{z})^T = K(\mathbf{z},\mathbf{x})$. Pour tout $(p,q) \in \mathbb{N}_d^2$,

$$\begin{aligned} K(\mathbf{x},\mathbf{z})_{qp} &= k\left(T_q(\mathbf{x}), T_p(\mathbf{z})\right) \\ &= k\left(T_p(\mathbf{z}), T_q(\mathbf{x})\right) \\ &= K(\mathbf{z},\mathbf{x})_{pq} \end{aligned}$$

Ensuite, soit $m \in \mathbb{N}$ et un ensemble de m couples $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathbb{R}^d$

$$\sum_{i,j=1}^{m} \langle \mathbf{y}_i, K(\mathbf{x}_i,\mathbf{x}_j)\mathbf{y}_j\rangle_{\mathbb{R}^d} = \sum_{p,i}\sum_{q,j} y_i^p y_j^q k(T_p(\mathbf{x}_i), T_q(\mathbf{x}_j)) \tag{4.59}$$

La positivité de l'expression ci-dessus provient du caractère semi-défini positif du noyau scalaire k sur $\mathcal{X}_0$. Ce qui donne le résultat. □

Si on choisit T_p comme les projections-coordonnées, i.e. $T_p(\mathbf{x}) = \mathbf{x}^T\mathbf{e}_p = x^p,\ p \in \mathbb{N}_d$ et le noyau gaussien pour k, on obtient le noyau transformable suivant :

Définition 4.10 (Noyau transformable gaussien)**.** *On appelle* noyau transformable gaussien *et on note $K_{Gauss} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ le noyau matriciel transformable de la forme*

$$\forall(\mathbf{x},\mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K_{Gauss}(\mathbf{x},\mathbf{z})_{pq} = k_{Gauss}(x^p, z^q),\quad (p,q) \in \mathbb{N}_d^2 \tag{4.60}$$

On peut voir le noyau transformable gaussien comme une extension naturelle du noyau gaussien scalaire au cas matriciel. Il est intéressant de noter que le coefficient (p,q) de la matrice $K_{\text{Gauss}}(\mathbf{x},\mathbf{z})$ compare la p-ème coordonnée de $\mathbf{x}$ à la q-ème coordonnée de $\mathbf{z}$, ce qui permet une comparaison plus riche entre $\mathbf{x}$ et $\mathbf{z}$.

Définition 4.11 (Modèle OKVAR transformable gaussien)**.** *Soit une série temporelle $\{\mathbf{x}_1,\ldots,\mathbf{x}_{N+1}\} \subseteq \mathbb{R}^d$, le modèle OKVAR transformable gaussien noté h_{Gauss} associé au*

noyau transformable gaussien est défini par :

$$h_{Gauss} = \sum_{\ell=1}^{N} K_{Gauss}(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell \tag{4.61}$$

Une caractéristique remarquable du modèle OKVAR transformable gaussien est que la i-ème coordonnée du vecteur $h_{\text{Gauss}}(\mathbf{x}_t)$ s'exprime comme une combinaison linéaire de fonctions non linéaires des variables $j \in \mathbb{N}_d$:

$$h_{\text{Gauss}}(\mathbf{x}_t)^i = \sum_{\ell=1}^{N}\sum_{j=1}^{d} \exp\left(-\gamma(x_t^i - x_\ell^j)^2\right) c_\ell^j$$

4.5 Apprentissage d'un modèle OKVAR

4.5.1 Choix de la régularisation

Dans un certain nombre d'applications, le noyau à valeur matricielle K peut avoir été spécifié au préalable. Par exemple, le noyau transformable gaussien dépend uniquement d'un hyperparamètre γ qui peut avoir été fixé ou appris. Pour un noyau décomposable gaussien, la matrice B peut être fournie *a priori* comme une donnée du problème. En conséquence, apprendre le modèle OKVAR basé sur l'un ou l'autre de ces noyaux revient à apprendre les paramètres $\{\mathbf{c}_\ell\}_{\ell=1}^N \subseteq \mathbb{R}^d$ du modèle. Dans la suite de l'exposé, nous notons $C \in \mathbb{R}^{d\times N}$ la matrice qui contient les N vecteurs $\mathbf{c}_\ell$ rangés en colonnes. Ainsi, nous notons le modèle correspondant h_C pour mettre en exergue cette dépendance. Pour estimer C, nous cherchons donc à minimiser la fonction de coût régularisée suivante :

$$\mathcal{J}(C) = \sum_{t=1}^{N} ||h_C(\mathbf{x}_t) - \mathbf{x}_{t+1}||^2 + \Omega(h_C) \tag{4.62}$$

où on rappelle que h_C est un modèle OKVAR de la forme donnée par l'équation (4.43) :

$$h_C = \sum_{\ell=1}^{N} K(\cdot, \mathbf{x}_\ell)\mathbf{c}_\ell$$

et $\Omega(h_C)$ est la somme de deux termes :

$$\Omega(h_C) = \lambda_h \|h_C\|_{\mathcal{H}_K}^2 + \Omega_C(C)$$

$\|h_C\|^2_{\mathcal{H}_K}$ est définie par :

$$\|h_C\|^2_{\mathcal{H}_K} = \sum_{i,j=1}^{N} \mathbf{c}_i^T K(\mathbf{x}_i, \mathbf{x}_j)\mathbf{c}_j \tag{4.63}$$

Cette norme joue le rôle d'une norme ℓ_2 sur C pondérée par les matrices $K(\mathbf{x}_i, \mathbf{x}_j)$ et correspond au terme classique de régularisation utilisé pour éviter le surapprentissage. $\Omega_C : \mathbb{R}^{d\times N} \to \mathbb{R}_+ \cup \{+\infty\}$ est un terme de régularisation supplémentaire que nous analysons par la suite. Lorsque $\Omega_C(C) = 0$, minimiser (4.62) revient à résoudre le problème de la Kernel Ridge Regression. Nous rappelons que dans ce cas, la matrice de paramètres C peut être calculée en forme close :

$$\mathbf{c} = (\mathbf{K} + \lambda_h Id)^{-1}\mathbf{x}_{2:N+1} \tag{4.64}$$

où $\mathbf{c} \in \mathbb{R}^{Nd}$ est la forme vectorisée de la matrice C obtenue en empilant les vecteurs $\mathbf{c}_\ell$, $\mathbf{K} = (K(\mathbf{x}_\ell, \mathbf{x}_t))_{\ell,t=1}^N \in \mathbb{R}^{Nd\times Nd}$ est la matrice de Gram par blocs où pour $(\ell, t) \in \mathbb{N}_N^2$, le bloc (ℓ, t) est la matrice $K(\mathbf{x}_\ell, \mathbf{x}_t) \in \mathbb{R}^{d\times d}$, Id est la matrice identité de taille $Nd \times Nd$ et $\mathbf{x}_{2:N+1} \in \mathbb{R}^{Nd}$ est le vecteur résultant de la concaténation des observations $\{\mathbf{x}_\ell\}_{\ell=2}^{N+1}$.

Cependant, la solution donnée par (4.64) n'est pas parcimonieuse. Afin d'induire de la parcimonie dans le modèle, on peut introduire une pénalité en norme ℓ_1 sur C en posant $\Omega_{\ell_1}(C) = \lambda_C\|C\|_1$ avec $\lambda_C > 0$ et où $\|\cdot\|_1$ désigne à la fois la norme ℓ_1 d'un vecteur et celle de la forme vectorisée d'une matrice. La fonction de coût $\mathcal{J}(C)$ devient alors analogue à celle utilisée dans les modèles régularisés par une pénalité de type *elastic net* dans le cas scalaire.

Dans les approches non paramétriques, une problématique-clé est celle du contrôle de la complexité. Une manière de l'aborder consiste à utiliser un nombre limité de paramètres $\mathbf{c}_\ell$, ce qui signifie que seule une poignée de données est réellement impliquée dans le modèle. En analogie aux SVMs, nous utilisons le vocable de *vecteurs de support* pour désigner les données correspondant à des vecteurs $\mathbf{c}_\ell$ non nuls. Une régularisation par la norme ℓ_1 pénalise de manière uniforme toutes les coordonnées des vecteurs $\mathbf{c}_\ell$ mais ne permet pas d'annuler tous les coefficients d'un vecteur $\mathbf{c}_\ell$ donné. Pour y parvenir, une stratégie fondée sur une parcimonie *structurée* est plus appropriée en considérant les colonnes de C, c'est-à-dire les vecteurs $\mathbf{c}_\ell$, comme une partition des coefficients de la matrice. Une contrainte de ce type qu'on note Ω_{ℓ_1/ℓ_2} prend la forme suivante :

$$\Omega_{\ell_1/\ell_2}(C) = \lambda_C \sum_{\ell=1}^{N} w_\ell\|\mathbf{c}_\ell\|_2 \tag{4.65}$$

Comme elle est définie dans (4.65), Ω_{ℓ_1/ℓ_2} est la norme mixte ℓ_1/ℓ_2. Nous avons vu (sous-

section 2.3.3) que cette norme possédait plusieurs caractéristiques intéressantes : elle agit comme une norme ℓ_1 au niveau des groupes, i.e. les vecteurs $\mathbf{c}_\ell$ tandis qu'au sein de chaque vecteur $\mathbf{c}_\ell$ les coefficients sont sujets à une contrainte en norme ℓ_2. Les valeurs des poids $\{w_\ell\}_{\ell=1}^N \subseteq \mathbb{R}_+$ dépendent de l'application.

4.5.2 Algorithme proximal

Dans le cas où Ω_C est Ω_{ℓ_1} ou Ω_{ℓ_1/ℓ_2}, (4.62) est une fonction de coût convexe qui est la somme de deux termes :

$$\mathcal{J}(C) = f_C(\mathbf{c}) + g_C(\mathbf{c}) \tag{4.66}$$

où

— f_C est définie par :

$$\begin{aligned} f_C(\mathbf{c}) &= \sum_{t=1}^{N} ||h_C(\mathbf{x}_t) - \mathbf{x}_{t+1}||^2 + \lambda_h \|h_C\|^2_{\mathcal{H}_K} \\ &= \sum_{t=1}^{N} \left\| \sum_{\ell=1}^{N} K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell - \mathbf{x}_{t+1} \right\|_2^2 + \lambda_h \sum_{t,\ell=1}^{N} \mathbf{c}_t^T K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell \end{aligned}$$

— $g_C(\mathbf{c}) = \Omega(C)$

On remarque que f_C et g_C sont deux fonctions convexes de C, en particulier f_C est différentiable alors que g_C est non lisse mais est sous-différentiable. La décomposition (4.66) de la fonction de coût $\mathcal{J}(C)$ justifie l'emploi d'un algorithme à base de gradients proximaux (section 2.4). L'Algorithme 4.1 repose sur les éléments suivants :

— L_C est une constante de Lipschitz de $\nabla_C f_C$, le gradient de f_C par rapport à la variable C

— les variables intermédiaires $t^{(m)}$ et $\mathbf{y}^{(m)}$ introduites respectivement aux étapes 2 et 3 permettent d'accélérer la méthode (A. Beck et M. Teboulle, 2010)

Proposition 4.9. *Une constante de Lipschitz de $\nabla_C f_C$ est :*

$$L_C = 2\rho(\mathbf{K}^2 + \lambda_h \mathbf{K})$$

où $\mathbf{K}$ est la matrice de Gram par blocs calculée sur les couples $(\mathbf{x}_\ell, \mathbf{x}_t)$, $(\ell, t) \in \mathbb{N}_N^2$ et $\rho(\mathbf{K}^2 + \lambda_h \mathbf{K})$ est la plus grande valeur propre de $\mathbf{K}^2 + \lambda_h \mathbf{K}$.

Démonstration. On remarque dans un premier temps que $f_C(C)$ peut se réécrire de la façon suivante :

$$f_C(\mathbf{c}) = \|\mathbf{K}\mathbf{c} - \mathbf{x}_{2:N+1}\|_2^2 + \lambda_h \mathbf{c}^T \mathbf{K}\mathbf{c}$$

On a alors :

$$\begin{aligned}\nabla_C f_C(\mathbf{c}) &= 2\mathbf{K}\left(\mathbf{K}\mathbf{c} - \mathbf{x}_{2:N+1}\right) + 2\lambda_h \mathbf{K}\mathbf{c} \\ &= 2\mathbf{K}([\mathbf{K} + \lambda_h Id]\mathbf{c} - \mathbf{x}_{2:N+1})\end{aligned}$$

Par suite, soient $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^{Nd}$,

$$\begin{aligned}\|\nabla_C f_C(\mathbf{c}_1) - \nabla_C f_C(\mathbf{c}_2)\|_2 &= \|2\mathbf{K}([\mathbf{K} + \lambda_h Id]\mathbf{c}_1 - \mathbf{x}_{2:N+1}) - 2\mathbf{K}([\mathbf{K} + \lambda_h Id]\mathbf{c}_2 - \mathbf{x}_{2:N+1})\|_2 \\ &= \left\|2\mathbf{K}^2\mathbf{c}_1 + 2\lambda_h\mathbf{K}\mathbf{c}_1 - 2\mathbf{K}^2\mathbf{c}_2 - 2\lambda_h\mathbf{K}\mathbf{c}_2\right\|_2 \\ &= \left\|(2\mathbf{K}^2 + 2\lambda_h\mathbf{K})(\mathbf{c}_1 - \mathbf{c}_2)\right\|_2 \\ &\leq 2\rho\left(\mathbf{K}^2 + \lambda_h\mathbf{K}\right)\|\mathbf{c}_1 - \mathbf{c}_2\|_2\end{aligned}$$

□

Algorithme 4.1 Minimiser (4.62)

Entrées : $\mathbf{c}^{(0)} \in \mathbb{R}^{Nd}$; M ; $\epsilon_{\mathbf{c}}$; L_C
Initialisation : $m = 0; \mathbf{y}^{(1)} = \mathbf{c}^{(0)}; t^{(1)} = 1$; `stop` :=`faux`
tant que $m < M$ et `stop`=`faux` **faire**
  0. : $m := m + 1$
  1. : $\mathbf{c}^{(m)} = \text{prox}_{\frac{1}{L_C}g_C}\left(\mathbf{y}^{(m)} - \frac{1}{L_C}\nabla_{\mathbf{y}^{(m)}} f_C(\mathbf{y}^{(m)})\right)$
  si $||\mathbf{c}^{(m)} - \mathbf{c}^{(m-1)}|| \leq \epsilon_{\mathbf{c}}$ **alors**
    `stop` :=`vrai`
  sinon
    2. : $t^{(m+1)} = \frac{1+\sqrt{1+4t^{(m)^2}}}{2}$
    3. : $\mathbf{y}^{(m)} = \mathbf{c}^{(m)} + \frac{t^{(m)}-1}{t^{(m+1)}}\left(\mathbf{c}^{(m)} - \mathbf{c}^{(m-1)}\right)$
  fin si
fin tant que
Sortie : $\mathbf{c}^{(m)}$

On rappelle alors que l'étape de gradient proximal à la m-ème itération est donnée par :

$$\text{prox}_{\frac{1}{L_C}g_C}\left(\mathbf{y}^{(m)} - \frac{1}{L_C}\nabla_{\mathbf{y}^{(m)}} f_C(\mathbf{y}^{(m)})\right)_{I_\ell} = \mathcal{T}_{s_\ell}\left(\mathbf{y}^{(m)} - \frac{1}{L_C}\nabla_{\mathbf{y}^{(m)}} f_C(\mathbf{y}^{(m)})\right)_{I_\ell}$$

où pour $s \geq 0$, $\mathcal{T}_s$ désigne l'opérateur de seuillage doux (2.33). Spécifiquement, lorsque $g_C = \Omega_{\ell_1/\ell_2}$, $s_\ell = \frac{\lambda_C w_\ell}{L_C}$ et $I_\ell,\ \ell \in \mathbb{N}_N$ est le sous-ensemble des indices correspondant à la ℓ-ème colonne de la matrice C. Dans le cas où $g_C = \Omega_{\ell_1}$, $s_\ell = \frac{\lambda_C}{L_C}$ et $I_\ell,\ \ell \in \mathbb{N}_{Nd}$ est réduit à un singleton correspondant à un coefficient donné de C.

Au lieu de mettre à jour tous les coefficients de la matrice des paramètres C *simultanément*, une stratégie alternative consiste à se focaliser uniquement sur un sous-ensemble ou groupe de coefficients indicés par J : $\mathbf{c}^J = \{c^j \mid j \in J\}$ au cours d'une itération et à minimiser la fonction de coût (4.62) par rapport aux variables $\mathbf{c}^J$, les autres coefficients $c_i, i \notin J$ étant fixés. Cette approche d'optimisation connue sous le nom de *descente de coordonnées par blocs* (*Block Coordinate Descent*, BCD) prend une connotation singulière lorsque l'ensemble J choisi à l'itération ℓ correspond aux indices du vecteur $\mathbf{c}_\ell$, i.e. la donnée d'une série temporelle au temps t_ℓ, celle-ci s'apparente alors à une version *en ligne* de l'algorithme 4.1 où on pourrait envisager que les données sont vues une à une par l'algorithme qui modifie le modèle au fur et à mesure qu'elles apparaissent.

4.5.3 Résultats

Génération des données

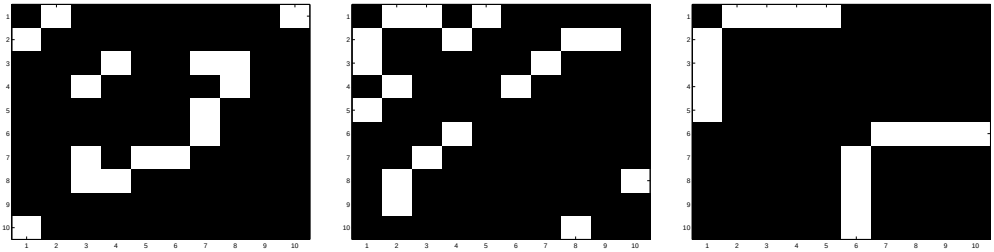
Nous présentons ici les performances de la famille de modèles OKVAR (Table 4.1) et du modèle VAR(1) à travers plusieurs expériences numériques. Pour cela, nous avons considéré trois patterns de structure distincts : « random », « scale-free », « hub » (Figure 4.1). Ces graphes ont été obtenus via le package `flare` de `R`. La génération des données obéit au protocole décrit ci-après. La matrice d'adjacence A du graphe est d'abord transformée de telle sorte à vérifier : $\|A\|_2 = 0.5$, cela garantit la stationnarité d'un processus VAR(1). Une série multivariée non linéaire de dimension d est alors générée selon le modèle :

$$\begin{cases} \mathbf{x}_1 \sim \mathcal{N}(0, \Sigma_{\mathbf{x}}) \\ \mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_{t+1} \end{cases} \tag{4.67}$$

où $h(x_t) = A\left(\psi(x_t^1) \dots \psi(x_t^d)\right)^T$ et les résidus sont homoscédastiques distribués selon une loi $\mathbf{u}_{t+1} \sim \mathcal{N}(0, \Sigma_{\mathbf{u}})$. Nous avons choisi d'étudier un bruit de deux natures. Une première configuration correspond à un bruit de covariance multiple de l'identité : $\Sigma_{\mathbf{u}} = \sigma_{\mathbf{u}}^2 Id$ avec $\sigma_{\mathbf{u}} > 0$. Le deuxième cas consiste à prendre un bruit de covariance non diagonale. En l'occurrence, $\Sigma_{\mathbf{u}}$ est ici une matrice de Toeplitz définie par : $\Sigma_{\mathbf{u},pq} = \nu^{|p-q|}$ pour un certain ν dans $(0, 1)$. La fonction $\psi : \mathbb{R} \to \mathbb{R}$ est non linéaire. Typiquement, $\psi(z) = \exp(-\gamma z^2)$. Via le modèle spécifié par (4.67) et instancié avec les valeurs de paramètres $\sigma_{\mathbf{u}}^2 = 1$, $\nu = 0.7$ et $\gamma = 0.1$, nous avons simulé six séries temporelles multivariées de dimension 10 avec 50 points de temps avec les combinaisons de structures (A) et de covariances ($\Sigma_{\mathbf{u}}$) suivantes :

- pattern « random » et bruit de covariance diagonale
- pattern « random » et bruit de covariance Toeplitz
- pattern « scale-free » et bruit de covariance diagonale
- pattern « scale-free » et bruit de covariance Toeplitz

— pattern « hub » et bruit de covariance diagonale
— pattern « hub » et bruit de covariance Toeplitz



(a) (b) (c)

FIGURE 4.1 – Patterns de structure générés pour l'évaluation des modèles OKVAR en prévision. (a) Random taille 10 (b) Scale-free taille 10 (c) Hub taille 10. Les pixels blancs correspondent aux coefficients non nuls de la matrice d'adjacence.

Sélection de modèle

Nous nous penchons ici sur le problème de sélection de modèle. En effet, l'estimation des modèles OKVAR fait intervenir deux hyperparamètres λ_h et λ_C. Nous présentons ici deux stratégies pour la sélection des hyperparamètres : le *critère BIC* et la *validation croisée séquentielle*.

BIC. Un critère de sélection de modèle largement employé est le critère d'information bayésien (*Bayesian Information Criterion*, BIC) aussi appelé critère de Schwarz (SCHWARZ, 1978), défini en toute généralité par :

$$\mathrm{BIC} = -2 \ln L(\hat{\theta}_k) + k \ln N \tag{4.68}$$

où :

— N est la taille de l'échantillon
— k est le nombre de paramètres *libres* à estimer
— $\hat{\theta}_k \in \Theta$ est le paramètre du modèle qui maximise la fonction de vraisemblance
— $L(\hat{\theta}_k)$ est la valeur maximale de la vraisemblance

Dérivons les calculs du critère BIC dans le cas d'un modèle OKVAR : $\mathbf{x}_{t+1} = h_C(\mathbf{x}_t) + \mathbf{u}_{t+1}$ où $h_C(\mathbf{x}_t) = \sum_{\ell=1}^{N} K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell$ et les vecteurs $\mathbf{c}_\ell$ sont les paramètres à estimer. Nous supposons que le bruit $\mathbf{u}_{t+1},\ t \in \mathbb{N}_N$ est modélisé par une loi gaussienne isotrope $\mathcal{N}(0, \sigma^2_{\mathbf{u}} Id)$,

avec $\sigma^2_{\mathbf{u}} > 0$. D'après l'équation (4.6), la log-vraisemblance du modèle s'écrit :

$$\begin{aligned}\ln L(C, \sigma^2_{\mathbf{u}}) &= \ln p(\mathbf{x}_1|\mu_1, \Sigma_1) + \sum_{t=1}^{N} \log p(\mathbf{x}_{t+1}|\mathbf{x}_t; C, \sigma^2_{\mathbf{u}}) \\ &= \log p(\mathbf{x}_1|\mu_1, \Sigma_1) - \frac{Nd}{2}\left(\ln(2\pi) + \ln \sigma^2_{\mathbf{u}}\right) \\ &\quad -\frac{1}{2\sigma^2_{\mathbf{u}}} \sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h_C(\mathbf{x}_t)\|_2^2\end{aligned}$$

Par conséquent,

$$\max_C L(C, \sigma^2_{\mathbf{u}}) \Leftrightarrow \min_C \sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h_C(\mathbf{x}_t)\|_2^2 \tag{4.69}$$

Sous l'hypothèse d'homoscédasticité des résidus distribués selon une loi $\mathcal{N}(0, \sigma^2_{\mathbf{u}} Id)$, la recherche de l'estimateur du maximum de vraisemblance de C revient donc à minimiser le risque empirique pour une fonction de perte quadratique. La variance du bruit est également estimée par maximum de vraisemblance et est donnée par :

$$\hat{\sigma}^2_{\mathbf{u}} = \frac{1}{Nd} \sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h_{\hat{C}}(\mathbf{x}_t)\|_2^2$$

On obtient, par suite, l'expression du critère BIC ci-après :

$$BIC = -2 \log p(\mathbf{x}_1|\mu_1, \Sigma_1) + Nd\left(\ln(2\pi) + \ln \hat{\sigma}^2_{\mathbf{u}}\right) + Nd(1 + \ln N)$$

Validation croisée séquentielle. Si nous avions affaire à des données i.i.d., on mettrait classiquement en œuvre une procédure de validation croisée. Dans le cas de données de séries temporelles, cette procédure doit être quelque peu amendée de sorte que l'erreur en validation en un point donné soit calculée en utilisant *uniquement* les données de la série qui lui sont antérieures. Nous avons alors recours à une procédure dite de *validation croisée séquentielle* décrite ci-après (Algorithme 4.2).

Dans cette section, la sélection de modèle est réalisée en utilisant la validation croisée séquentielle. Les résultats de la table 4.2 indiquent que les modèles à noyaux présentent des performances en prédiction supérieures à celles de VAR(1). Bien que les modèles transformables gaussiens exhibent de plus faibles erreurs que les autres sur quatre bases de données, les écarts ne semblent pas suffisamment significatifs pour déterminer de manière certaine quel modèle à noyaux est le plus performant. D'autre part, les modèles $h_{k \cdot Id}$ qui correspondent en réalité à d modèles à noyaux scalaires gaussiens indépendants obtiennent des performances similaires aux modèles décomposables et transformables. Un éclairage

peut être apporté à ces résultats à l'aune de la discussion menée à la section 4.2.

TABLE 4.1 – Récapitulatif des modèles OKVAR étudiés pour la tâche de prévision.

	Modèles OKVAR							
	$h^{\text{Ridge}}_{k\cdot Id}$	$h^{\ell_1}_{k\cdot Id}$	$h^{\text{Ridge}}_{\text{Gauss}}$	$h^{\ell_1}_{\text{Gauss}}$	$h^{\ell_1/\ell_2}_{\text{Gauss}}$	$h^{\text{Ridge}}_{\text{dec}}$	$h^{\ell_1}_{\text{dec}}$	$h^{\ell_1/\ell_2}_{\text{dec}}$
Noyau	$k_{\text{Gauss}} \times Id$		Transformable gaussien			Décomposable gaussien		
Coût	Eq. (4.62)							
Ω_C	0	Ω_{ℓ_1}	0	Ω_{ℓ_1}	Ω_{ℓ_1/ℓ_2}	0	Ω_{ℓ_1}	Ω_{ℓ_1/ℓ_2}

Algorithme 4.2 Validation croisée séquentielle pour la sélection de λ_h et λ_C pour un modèle OKVAR

Entrées :
— Vecteurs d'etat observés : $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1} \in \mathbb{R}^d$
— Grille de paramètres : $(\lambda_h, \lambda_C) \in \Lambda_h \times \Lambda_C$
— Taille de la fenêtre de temps : $0 < N_w < N$
Initialisation : $\texttt{err} := \mathbf{0} \in \mathbb{R}^{|\Lambda_h| \times |\Lambda_C|}$
pour $\lambda_h \in \Lambda_h$ **faire**
  pour $\lambda_C \in \Lambda_C$ **faire**
    pour $t := N_w$ à N **faire**
      1. : Estimer $\hat{C}$ à partir de $\mathbf{x}_{t-N_w+1}, \ldots, \mathbf{x}_t$
      2. : Obtenir une prédiction : $\hat{\mathbf{x}}_{t+1} := h_{\hat{C}}(\mathbf{x}_t)$
    fin pour
    3. : Calculer l'erreur moyenne :

$$\texttt{err}(\lambda_h, \lambda_C) := \frac{1}{d(N - N_w + 1)} \sum_{t=N_w}^{N} \|\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1}\|_2^2$$

  fin pour
fin pour
Sortie : $(\lambda_h^\star, \lambda_C^\star) := \text{argmin}_{(\lambda_h, \lambda_C) \in \Lambda_h \times \Lambda_C} \texttt{err}(\lambda_h, \lambda_C)$

TABLE 4.2 – Meilleures erreurs moyennes de validation croisée séquentielle des modèles OKVAR, VAR(1) sur les bases de données synthétiques de dimension $d = 10$. Les écarts-types sont donnés entre parenthèses. Taille de la fenêtre utilisée : $N_w = 25$. Les nombres en **gras** sont les plus petites valeurs de chaque colonne.

Structure	Random		Scale-free		Hub	
Bruit	diag.	Toeplitz	diag.	Toeplitz	diag.	Toeplitz
Modèles						
$h^{\text{Ridge}}_{k\cdot Id}$	1.0673	1.0218	1.0831	1.3875	1.1048	1.0783
	(0.4946)	(0.5475)	(0.5415)	(1.0587)	(0.4597)	(0.6670)
$h^{\ell_1}_{k\cdot Id}$	1.0472	1.0033	1.0624	1.3118	1.0563	**1.0310**
	(0.4856)	(0.5389)	(0.5196)	(1.0718)	(0.4509)	(0.6772)
$h^{\text{Ridge}}_{\text{dec}}$	1.0499	1.0441	1.0593	1.3900	1.0527	1.0762
	(0.4834)	(0.5806)	(0.5305)	(1.0173)	(0.4387)	(0.6785)
$h^{\ell_1}_{\text{dec}}$	**1.0326**	1.0229	1.0567	1.3455	1.0493	1.0539
	(0.4849)	(0.5749)	(0.5266)	(1.0870)	(0.4490)	(0.6879)
$h^{\ell_1/\ell_2}_{\text{dec}}$	1.0483	1.0386	1.0579	1.3800	1.0486	1.0654
	(0.4841)	(0.5791)	(0.5304)	(1.0305)	(0.4413)	(0.6734)
$h^{\text{Ridge}}_{\text{Gauss}}$	1.0466	0.9982	1.0433	**1.2625**	**1.0420**	1.0531
	(0.5221)	(0.6297)	(0.5169)	(1.0498)	(0.4322)	(0.7347)
$h^{\ell_1}_{\text{Gauss}}$	1.0466	**0.9978**	**1.0431**	**1.2625**	**1.0420**	1.0531
	(0.5221)	(0.6278)	(0.5171)	(1.0498)	(0.4322)	(0.7347)
$h^{\ell_1/\ell_2}_{\text{Gauss}}$	1.0466	0.9982	1.0433	**1.2625**	**1.0420**	1.0531
	(0.5222)	(0.6298)	(0.5169)	(1.0498)	(0.4322)	(0.7349)
VAR(1)	1.6324	1.5403	1.6316	1.8113	1.7039	2.0463
	(0.7431)	(1.0474)	(0.9994)	(1.7120)	(0.9698)	(1.8402)

4.6 OKVAR-Boost

La question de l'estimation d'un modèle OKVAR se pose de façon prégnante en grande dimension, lorsque le nombre de variables d'états d d'un système excède la longueur des séries temporelles. Dans l'état de l'art, on note l'utilisation de méthodes d'ensemble pour résoudre de manière découplée d problèmes de régression dans le cas de données i.i.d. . Ainsi, Anjum et al. (2009) proposent comme alternative aux approches de type LASSO de construire une combinaison de modèles autorégressifs linéaires dans l'esprit du L_2-boosting (J. Friedman, 2001). En modélisation non paramétrique, les forêts aléatoires et leurs variantes, les *extra-trees* ont fait leurs preuves, notamment dans des applications biologiques (Huynh-Thu et al., 2010). Nous proposons dans ce chapitre une *méthode d'ensemble* pour les modèles OKVAR afin de pallier le problème de la dimension.

4.6.1 Principe de l'algorithme

On suppose qu'on observe une série temporelle multivariée $\mathbf{x}_{1:N+1} = \{\mathbf{x}_1, \dots, \mathbf{x}_{N+1}\} \subseteq \mathbb{R}^d$. Soit M le nombre de modèles OKVAR de base qui composent l'ensemble. Le modèle autorégressif final qui résulte de notre algorithme est noté H et admet la forme suivante :

$$\widehat{\mathbf{x}}_{t+1} = H(\mathbf{x}_t) = \sum_{m=1}^{M} \rho_m h(\mathbf{x}_t; \mathcal{E}_m) \tag{4.70}$$

où, pour $m \in \mathbb{N}_M$, $\mathcal{E}_m \subseteq \mathbb{N}_d$ désigne un sous-ensemble des d variables d'état. Le modèle de base $h(\cdot; \mathcal{E}_m)$ est défini, pour $i \in \mathbb{N}_d$ et $\mathbf{x} \in \mathbb{R}^d$, par :

$$h(\mathbf{x}; \mathcal{E}_m)^i = \begin{cases} h_m(\mathbf{x}^{(m)})^i & \text{si } i \in \mathcal{E}_m \\ 0 & \text{sinon} \end{cases} \tag{4.71}$$

avec $\mathbf{x}^{(m)} = (x^i,\ i \in \mathcal{E}_m) \in \mathbb{R}^{|\mathcal{E}_m|}$. $h_m \in \mathcal{H}_{K^{(m)}}$ est un modèle OKVAR bâti sur un noyau à valeurs matricielles $K^{(m)} : \mathbb{R}^{|\mathcal{E}_m|} \times \mathbb{R}^{|\mathcal{E}_m|} \to \mathbb{R}^{|\mathcal{E}_m| \times |\mathcal{E}_m|}$ qui opère sur le sous-espace de $\mathbb{R}^d$ induit par $\mathcal{E}_m$. Pour $\mathbf{z} \in \mathbb{R}^{|\mathcal{E}_m|}$,

$$h_m(\mathbf{z}) = \sum_{\ell=1}^{N} K^{(m)}(\mathbf{z}, \mathbf{x}_\ell^{(m)}) \mathbf{c}_\ell^{(m)} \tag{4.72}$$

Il existe de multiples façons de construire un ensemble de modèles (cf. section 1.5), cela inclut notamment le bagging, le boosting ou encore les forêts aléatoires (Dietterich, 2000 ; Leo Breiman, 2001 ; J. Friedman, 2001). Dans notre travail, nous employons un algorithme analogue au L_2-boosting adéquat pour des problèmes de régression (J. Friedman, 2001 ; Peter Bühlmann et Yu, 2003) auquel on a greffé une composante

aléatoire avec la sélection d'un sous-ensemble $\mathcal{E}_m$ à chaque itération. Lorsque $\mathcal{E}_m = \mathbb{N}_d$, l'algorithme est identique au L_2-boosting appliqué au problème de l'autorégression. Pour traiter les grandes dimensions, le sous-ensemble $\mathcal{E}_m$ est choisi en général de telle sorte que $|\mathcal{E}_m| \ll d$. L'ajout de la composante aléatoire est justifié par plusieurs travaux qui montrent la robustesse de techniques combinant des caractéristiques des forêts aléatoires et des algorithmes de boosting (Geurts et al., 2007).

Nous décrivons ci-dessous l'algorithme 4.3 que nous avons appelé *OKVAR-Boost*.

On commence par initialiser le modèle d'ensemble $H := H_0$ avec les valeurs moyennes des vecteurs d'état observés, puis à l'itération m de l'algorithme, on réalise les étapes suivantes :

— **Étape 1** : on calcule les résidus $\mathbf{u}_{t+1}^{(m)} = \mathbf{x}_{t+1} - H_{m-1}(\mathbf{x}_t)$, $t \in \mathbb{N}_N$. Cette étape confère à OKVAR-Boost sa nature de L_2-boosting.

— **Étape 2** : on note $\mathcal{I}_m^\epsilon$ l'ensemble des variables d'état dont la norme des résidus est au-delà d'un seuil $\epsilon > 0$ pré-défini, i.e. $\mathcal{I}_m^\epsilon = \left\{ j \in \mathbb{N}_d;\ \left\| \mathbf{u}^{j\,(m)} \right\|_2 > \epsilon \right\}$. Un arrêt précoce (*early-stopping*) de l'algorithme est décidé lorsque $\mathcal{I}_m^\epsilon = \emptyset$. Cette règle d'arrêt a pour but d'éviter le sur-apprentissage.

— **Étape 3** : on choisit $q \leq d$ variables d'état dans $\mathcal{I}_m^\epsilon$, on obtient alors un sous-ensemble aléatoire $\mathcal{E}_m$ de taille q, ce qui correspond à un tirage de q éléments dans $\mathcal{I}_m^\epsilon$ selon une loi uniforme.

— **Étape 4** : l'estimation de la matrice des paramètres C_m du modèle h_m est réalisée en utilisant les résidus courants dans le sous-ensemble $\mathcal{E}_m$, notés $\left(\tilde{\mathbf{u}}_{t+1}^{(m)}\right)_{t=1}^N$ et définis par :

$$\tilde{\mathbf{u}}_{t+1}^{(m)} = P^{(m)} \mathbf{u}_{t+1}^{(m)}, \quad t \in \mathbb{N}_N$$

où $P^{(m)}$ est une matrice diagonale de taille $d \times d$ telle que $P_{ii}^{(m)} = 1$, $i \in \mathbb{N}_d$ si $i \in \mathcal{E}_m$ et 0 sinon.

(a) La matrice C_m est choisie de manière à minimiser le coût régularisé :

$$\mathcal{J}(C) = \sum_{t=1}^{N} \left\| \tilde{\mathbf{u}}_{t+1}^{(m)} - h(\tilde{\mathbf{u}}_t^{(m)}; \mathcal{E}_m) \right\|_2^2 + \lambda_h \|h_m\|_{\mathcal{H}_{K^{(m)}}}^2 + \lambda_C \|C\|_1 \tag{4.73}$$

Cette régularisation de type *filet élastique* combine deux pénalités, une norme ℓ_1 et une norme ℓ_2. Elle induit de la parcimonie sur les paramètres comme le LASSO tout en encourageant des effets de groupe en permettant à un sous-ensemble de variables fortement corrélées d'être sélectionnées.

(b) Enfin, le paramètre ρ_m est sélectionné par une recherche linéaire :

$$\rho_m = \underset{\rho \in \mathbb{R}}{\operatorname{argmin}} \sum_{t=1}^{N} \left\| \tilde{\mathbf{u}}_{t+1}^{(m)} - \rho h(\tilde{\mathbf{u}}_t^{(m)}; \mathcal{E}_m) \right\|_2^2 \tag{4.74}$$

— **Étape 5** : le modèle d'ensemble H est mis à jour avec les nouveaux paramètres estimés C_m et ρ_m :

$$H_m := H_{m-1} + \rho_m h(\cdot; \mathcal{E}_m) \tag{4.75}$$

— **Étape 6** : si on n'a pas atteint le nombre maximum d'itérations M ou si la règle d'arrêt précoce n'est pas satisfaite, on retourne à l'étape 1 de l'algorithme.

Algorithme 4.3 OKVAR-Boost pour la prévision

Entrées :
— Vecteurs d'etat observés : $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1} \in \mathbb{R}^d$
— Seuil d'arrêt précoce : $\epsilon > 0$
— Taille des sous-ensembles : $q \in \mathbb{N}_d$
Initialisation :
— $\forall t \in \mathbb{N}_N$, $H_0(\mathbf{x}_t) := (\bar{\mathbf{x}}^1, \ldots, \bar{\mathbf{x}}^d)^T$
— $m := 0$, `stop` :=`faux`
tant que $m < M$ et `stop`=`faux` **faire**
0. $m := m + 1$
1. $\forall t \in \mathbb{N}_N$, $\mathbf{u}_{t+1}^{(m)} := \mathbf{x}_{t+1} - H_{m-1}(\mathbf{x}_t)$
2. $\mathcal{I}_m^\epsilon = \left\{ j \in \mathbb{N}_d;\ \left\| \mathbf{u}^{j(m)} \right\|_2 > \epsilon \right\}$; `stop` :=« $\mathcal{I}_m^\epsilon = \emptyset$? »
si `stop`=`faux` **alors**
3. Sélectionner $\mathcal{E}_m$, un sous-ensemble aléatoire des variables d'état, de taille q dans $\mathcal{I}_m^\epsilon$
4. À partir de $\mathbf{u}_2^{(m)}, \ldots, \mathbf{u}_{N+1}^{(m)}$,(a) estimer la matrice des paramètres du modèle C_m et (b) estimer ρ_m par une recherche linéaire
5. $H_m := H_{m-1} + \rho_m h(\cdot; \mathcal{E}_m)$
fin si
fin tant que
$m_{\text{stop}} := m$
Sortie : $H_{m_{\text{stop}}}$

4.6.2 Résultats

La figure 4.2 montre le comportement de OKVAR-Boost en fonction du nombre de modèles ajoutés dans la combinaison (paramètre M) en termes de MSE $= \frac{1}{Nd} \sum_{t=1}^{N} \|H(\mathbf{x}_t) - \mathbf{x}_{t+1}\|_2^2$. Nous avons choisi d'appliquer OKVAR-Boost instancié avec le noyau K_{dec} sur la base de données synthétiques de taille 10 « Scale-free » (bruit de covariance diagonale). Nous avons testé l'impact de la composante aléatoire en choisissant trois tailles de sous-ensembles : $|\mathcal{E}_m| = 3, 6$ ou 10. Le dernier cas, $|\mathcal{E}_m| = 10 = d$, ne présente en réalité aucun aléa et se réduit à l'application du L_2-boosting. On remarque que les MSE diminuent rapidement et que l'algorithme a tendance à surapprendre après 10 itérations pour $|\mathcal{E}_m| \in \{3, 6\}$ (courbes bleue et rouge, respectivement). Lorsque $|\mathcal{E}_m| = 10 = d$, on note

une faible amélioration de l'erreur mais cette dernière ne diminue plus après 4 itérations, ce qui suggère un véritable intérêt de la randomisation dans OKVAR-Boost.

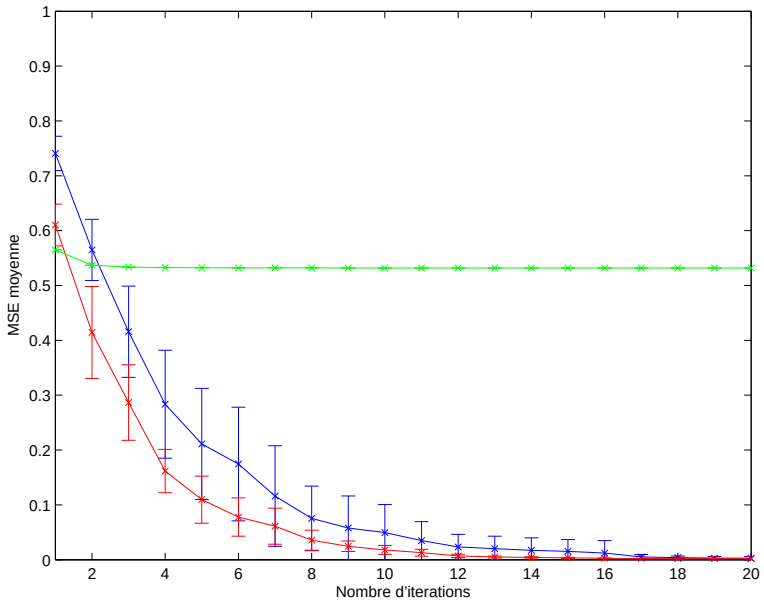


FIGURE 4.2 – MSE moyenne de OKVAR-Boost en fonction du nombre de modèles agérégés (itérations) pour diverses tailles de sous-ensembles aléatoires sur la base de données synthétiques de taille 10 « Scale-free » (bruit de covariance diagonale). Les courbes bleue, rouge et verte correspondent respectivement à des exécutions d'OKVAR-Boost avec des tailles de sous-ensembles $|\mathcal{E}_m| \in \{3, 6, 10\}$. Les barres d'erreur représentent les écarts-types des MSE pour 10 exécutions d'OKVAR-Boost. Le cas $|\mathcal{E}_m| = 10 = d$ (courbe verte) est déterministe (écart-type nul) et correspond au L_2-boosting.

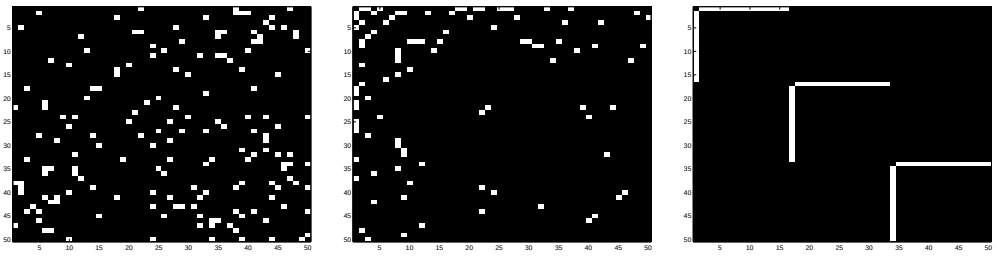
Les tables 4.3 et 4.4 répertorient les performances en prévision d'OKVAR-Boost obtenus respectivement sur les bases de données synthétiques de taille $d = 10$ (Figure 4.1) qui ont servi à tester les modèles de base OKVAR ainsi que sur des bases de données additionnelles de taille $d = 50$ générées selon le même protocole détaillé au début de la section 4.5.3. La figure 4.3 exhibe les patterns des réseaux de taille 50.

En ce qui concerne les jeux de données de taille 10 (Table 4.3), on note de prime abord qu'OKVAR-Boost exhibe des erreurs inférieures à VAR(1), excepté pour le noyau transformable K_{Gauss}. Comparativement à OKVAR (Table 4.2), OKVAR-Boost ne permet pas d'améliorer les performances. Cela suggère qu'OKVAR-Boost n'est peut-être pas approprié dans un cas où l'estimation d'un unique modèle OKVAR peut être réalisée dans des conditions favorables : petite dimension $d = 10$, longueur de la série temporelle $N = 50 > d$.

TABLE 4.3 – Meilleures erreurs moyennes de validation croisée séquentielle pour OKVAR-Boost instancié avec les noyaux $k_{\text{Gauss}} \times Id$, K_{dec} et K_{Gauss} sur les bases de données synthétiques de dimension $d = 10$. Les écarts-types sont donnés entre parenthèses. Taille de la fenêtre utilisée : $N_w = 25$. Taille des sous-ensembles : $|\mathcal{E}_m| = 6$. Seuil d'arrêt : $\epsilon = 10^{-4}$. Les nombres en **gras** sont les plus petites valeurs de chaque colonne.

Structure	Random		Scale-free		Hub	
Bruit	diag.	Toeplitz	diag.	Toeplitz	diag.	Toeplitz
Modèles						
$k_{\text{Gauss}} \times Id$	1.1154	1.0845	**1.0555**	**1.3983**	1.1158	1.0956
	(0.5442)	(0.5901)	(0.5202)	(1.1335)	(0.4508)	(0.7257)
K_{dec}	**1.0805**	**1.0135**	1.1023	1.4498	**1.0759**	**0.9528**
	(0.5426)	(0.6833)	(0.4903)	(1.0890)	(0.4313)	(0.5803)
K_{Gauss}	2.4224	1.8370	2.1401	2.7205	1.9283	2.3603
	(1.3200)	(1.0347)	(1.5007)	(3.1450)	(0.9640)	(1.4803)
VAR(1)	1.6324	1.5403	1.6316	1.8113	1.7039	2.0463
	(0.7431)	(1.0474)	(0.9994)	(1.7120)	(0.9698)	(1.8402)

Pour les bases de données de taille 50, la table 4.4 atteste une fois de plus des bonnes performances d'OKVAR-Boost par rapport au modèle VAR(1). Cela laisse penser qu'une stratégie d'ensemble est pertinente pour des cas moins favorables (ici, $d = N = 50$) où le nombre de données N est de l'ordre ou inférieur à la dimension d. Par ailleurs, les



(a) (b) (c)

FIGURE 4.3 – Patterns de structure générés pour l'évaluation de OKVAR-Boost en prévision. (a) Random taille 50 (b) Scale-free taille 50 (c) Hub taille 50. Les pixels blancs correspondent aux coefficients non nuls de la matrice d'adjacence.

mauvais résultats d'OKVAR-Boost instancié avec le noyau transformable K_{Gauss} sur les structures hubs indiquent que ce noyau n'est peut-être pas adéquat pour ce type de topologie. Toutefois, il est à signaler que nous avons présenté les résultats avec une configuration particulière d'OKVAR-Boost et qu'il faut probablement jouer sur d'autres paramètres de l'algorithme tels que la taille des sous-ensembles ou bien le seuil d'arrêt précoce ϵ pour obtenir de meilleures performances.

TABLE 4.4 – Meilleures erreurs moyennes de validation croisée séquentielle pour VAR(1) et OKVAR-Boost instancié avec les noyaux $k_{\text{Gauss}} \times Id$, K_{dec} et K_{Gauss} sur les bases de données synthétiques de dimension $d = 50$. Les écarts-types sont donnés entre parenthèses. Taille de la fenêtre utilisée : $N_w = 25$. Taille des sous-ensembles : $|\mathcal{E}_m| = 7$. Seuil d'arrêt : $\epsilon = 10^{-4}$. Les nombres en **gras** sont les plus petites valeurs de chaque colonne.

Structure	Random		Scale-free		Hub	
Bruit	diag.	Toeplitz	diag.	Toeplitz	diag.	Toeplitz
Modèles						
$k_{\text{Gauss}} \cdot Id$	1.1186	1.0903	**1.0666**	1.1788	**1.1505**	**1.2118**
	(0.2450)	(0.3778)	(0.2779)	(0.4456)	(0.2520)	(0.3471)
K_{dec}	**1.1109**	**1.0878**	1.0673	**1.1636**	1.1636	1.2285
	(0.2410)	(0.3882)	(0.2760)	(0.4293)	(0.2321)	(0.3855)
K_{Gauss}	1.3059	1.2560	1.3068	1.3401	3.5561	4.0837
	(0.2687)	(0.3676)	(0.3209)	(0.4744)	(1.7830)	(2.6384)
VAR(1)	1.5465	1.6817	1.5744	1.5250	1.5796	1.8415
	(0.3402)	(0.6788)	(0.3582)	(0.5203)	(0.3340)	(0.5593)

4.7 Synthèse

Dans ce chapitre, nous avons présenté les fondements d'une nouvelle famille de modèles autorégressifs vectoriels non paramétriques. Ces modèles reposent sur la théorie des noyaux à valeur opérateur qui offre un cadre élégant pour l'apprentissage de fonctions à valeurs vectorielles. Une fois le noyau choisi et fixé, nous avons développé un algorithme proximal permettant d'apprendre les paramètres du modèle, sous des contraintes de parcimonie. Par ailleurs, au-delà de la problématique de l'autorégression, tous les outils développés, de par leur grande généricité, peuvent être appliqués à des tâches de régression vectorielle ou plus généralement à des problèmes de prédiction de sorties structurées.

Nous avons également élaboré une méthode d'ensemble OKVAR-Boost qui est une alternative aux approches de régularisation de type LASSO et s'inspire du L_2-boosting. Il s'en distingue néanmoins puisque l'algorithme travaille sur des sous-espaces sélectionnés aléatoirement à chaque itération. Cette stratégie modulaire rend la méthode attractive pour des systèmes de grande taille.

Chapitre 5

Extraction d'influences causales à partir de modèles autorégressifs vectoriels

Sommaire

5.1 Introduction 115
5.2 Apprentissage de structure d'un système dynamique à partir de modèles autorégressifs vectoriels 116
5.3 Choix des modèles à noyaux pour la tâche d'apprentissage de structure 118
5.4 Apprentissage des paramètres du modèle OKVAR Hadamard 121
5.5 OKVAR-Boost 124
5.6 Applications 127
5.6.1 Inférence de réseaux de régulation génique 127
5.6.2 Données climat 139
5.7 Synthèse 142

5.1 Introduction

Dans le chapitre précédent, nous avons défini une famille de modèles autorégressifs vectoriels à base de noyaux à valeur matricielle (OKVAR) pour laquelle nous avons mis en place une panoplie d'outils pour apprendre soit un modèle de base, soit un ensemble de modèles (OKVAR-Boost). La flexibilité due à la richesse de cette famille permet d'adresser des problèmes divers qui vont au-delà de la tâche naturelle de prévision. En l'occurrence, ce chapitre décrit une méthodologie originale pour l'extraction de relations d'influence *causale* entre les variables d'état d'un système à partir de modèles autorégressifs vectoriels. À cette fin, nous introduisons la notion de jacobienne et montrons comment dans le cas spécifique d'un modèle OKVAR, cette dernière guide le choix des noyaux à valeur matricielle à

utiliser. Des résultats sur l'inférence de réseaux de régulation génique et sur des données réelles du climat corroborent l'intérêt de l'approche proposée.

5.2 Apprentissage de structure d'un système dynamique à partir de modèles autorégressifs vectoriels

Considérons un système dynamique composé de d variables. Nous nous intéressons à l'extraction de liens de causalités entre les variables d'état, c'est-à-dire l'inférence d'influences d'une variable d'état j sur d'autres variables $i \neq j$, $(i,j) \in \mathbb{N}_d^2$ à partir de données de séries temporelles. L'ensemble des influences croisées entre les variables d'états est codé par une *matrice d'adjacence* $A = (a_{ij})_{i,j=1}^d \in \{0,1\}^{d\times d}$ telle que $a_{ij} = 1$ si la variable d'état j influence la variable d'état i, et 0 sinon. Nous cherchons à montrer comment la tâche d'inférence de réseaux causaux peut être liée à l'estimation de modèles autorégressifs vectoriels.

Supposons qu'on observe le système à $N+1$ temps discrets régulièrement espacés, les vecteurs d'état observés sont notés $\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}$. On conserve l'hypothèse 4.1 de stationnarité du processus sous-jacent et de modèle d'ordre 1. La dynamique du système est supposée non linéaire et est capturée par une fonction $h : \mathbb{R}^d \to \mathbb{R}^d$ de la forme (4.1) :

$$\mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_{t+1}$$

où $\mathbf{u}_{t+1} \in \mathbb{R}^d$ est un terme de bruit.

Dans le cas linéaire, C. W. Granger (1969) définit un concept de causalité de la manière suivante : une variable d'état j a une influence causale sur une variable d'état i si les valeurs passées de la série temporelle $\mathbf{x}^j$ améliorent substantiellement la prédiction des valeurs de la série $\mathbf{x}^i$. Le concept se traduit formellement comme suit.

Définition 5.1 (C. W. Granger (1969))**.** *Sous l'hypothèse d'un modèle d'ordre 1, soit le modèle de régression pour la variable d'état $i \in \mathbb{N}_d$:*

$$x_{t+1}^i = b_{ii} x_t^i + b_{ij} x_t^j + u_{t+1}^i$$

On dit que la variable d'état j est causale au sens de Granger pour la variable d'état i si le coefficient b_{ij} est significativement non nul.

La notion de causalité proposée par Granger fait ainsi le lien entre causalité et prédiction. Les méthodes linéaires telles que le modèle VAR(1) et les modèles graphiques (voir par exemple Arnold et al. (2007) et Shojaie et G. Michailidis (2010) et références incluses) sont des modèles paramétriques qui font apparaître explicitement une matrice

qui peut être interprétée comme une matrice d'adjacence. Très souvent, une connaissance experte donnée *a priori* plaide pour une structure causale sous-jacente creuse et l'estimation de la matrice d'adjacence A, sous contrainte de parcimonie, peut alors se réaliser de manière directe. Cependant, les systèmes réels rencontrés présentent généralement des interactions non linéaires. À notre connaissance, il n'existe pas de méthode systématique d'estimation de la matrice d'adjacence dans le cas des modèles non linéaires. Nous proposons ici une approche originale pour l'estimation de A pour les modèles non linéaires. Notre stratégie consiste d'abord à apprendre un modèle autorégressif h à partir des données de série temporelle puis d'estimer A en utilisant les valeurs de la *matrice jacobienne instantanée* du modèle appris $\hat{h}$, mesurée en chaque point $t \in \mathbb{N}_N$. Le principe proposé repose sur l'idée que les dérivées partielles $\frac{\partial \hat{h}(\mathbf{x}_t)^i}{\partial x_t^j}$ reflètent l'influence de la variable explicative j au temps t sur la valeur de la i-ème composante du modèle, $\hat{h}(\mathbf{x}_t)^i$. Si la valeur absolue de $\frac{\partial \hat{h}(\mathbf{x}_t)^i}{\partial x_t^j}$ est élevée, on conclut à l'existence d'un lien d'influence causale de la variable j sur la variable i. Plusieurs estimateurs de la matrice d'adjacence A peuvent être construits à partir de ces dérivées partielles :

$$s_N\left(\frac{\partial \hat{h}(\mathbf{x}_1)^i}{\partial x_1^j}, \ldots, \frac{\partial \hat{h}(\mathbf{x}_N)^i}{\partial x_N^j}\right) \tag{5.1}$$

Un estimateur naturel de A consiste à prendre la moyenne empirique des matrices jacobiennes instantanées, i.e. $s_N(z_1, \ldots, z_N) = \frac{1}{N}\sum_{t=1}^{N} z_t$. Pour $(i,j) \in \mathbb{N}_d^2$, un estimateur $\hat{J}(h)$ de la matrice jacobienne $\nabla h = J(h)$ est alors donné par :

$$\hat{J}(h)_{ij} = \frac{1}{N}\sum_{t=1}^{N} \frac{\partial h(\mathbf{x}_t)^i}{\partial x_t^j} \tag{5.2}$$

Cependant, d'autres statistiques peuvent être envisagées telles que le plus grand élément en valeur absolue ou encore la moyenne des valeurs absolues des matrices jacobiennes instantanées. Une étude sur ces différents estimateurs est réalisée dans la partie consacrée aux résultats (cf. section 5.6.1).

Dans la suite du manuscrit, nous notons $\hat{J}(h)_{ij}$ le coefficient (i,j) de la moyenne empirique de la jacobienne de h et $\hat{J}(h)_{ij}(\mathbf{x}_t)$ sa valeur pour le vecteur d'état au temps t. Chaque coefficient $\hat{J}(h)_{ij}$ en valeur absolue peut être interprété comme un score sur l'influence potentielle de la variable j sur la variable i. Pour obtenir un estimateur de la matrice d'adjacence A, ces coefficients sont ordonnés puis un seuil est sélectionné de sorte que $\hat{a}_{ij} = 1$ si le coefficient (i,j) correspondant est au-delà de ce seuil.

La qualité de l'estimation de la matrice d'adjacence est dépendante de la classe de modèles h choisie. Il faut pour cela s'assurer de pouvoir contrôler la jacobienne du modèle h pendant la phase d'apprentissage de sorte à obtenir de bons approximateurs continus de

A. C'est la raison pour laquelle nous avons mis en exergue une nouvelle classe de modèles vectoriels autorégressifs non paramétriques, les modèles OKVAR qui jouissent d'une telle propriété. Nous allons voir dans la section suivante comment spécifier des modèles OKVAR pour la tâche d'extraction de causalités.

5.3 Choix des modèles à noyaux pour la tâche d'apprentissage de structure

En vue d'utiliser la jacobienne comme estimateur de la matrice d'adjacence encodant la structure causale sous-jacente, nous devons questionner le choix des modèles h à considérer. Lors de la présentation de la théorie des RKHS des fonctions à valeurs vectorielles, nous avons fait remarquer qu'à partir d'un noyau à valeur matricielle K, on peut définir deux espaces fonctionnels distincts $\mathcal{H}_K = \overline{\text{Vect}\{K(\cdot,\mathbf{x})\mathbf{y} \mid \mathbf{x},\mathbf{y} \in \mathbb{R}^d\}}$ (e.g. C. Micchelli et M. Pontil (2005)) et $\mathcal{H}_K^{\#} = \overline{\text{Vect}\{K(\mathbf{x},\cdot)\mathbf{y} \mid \mathbf{x},\mathbf{y} \in \mathbb{R}^d\}}$ (e.g. H. Zhang, Xu et Q. Zhang (2012)). Le choix de travailler dans l'un ou l'autre n'est pas anodin et a un impact déterminant dans le calcul de la jacobienne. Cela conduit aux expressions des matrices jacobiennes instantanées suivantes. Pour tout modèle OKVAR $h \in \mathcal{H}_K$ construit à partir des vecteurs observés $\mathbf{x}_1, \ldots, \mathbf{x}_N$ et pour toute variable d'état-cible $i \in \mathbb{N}_d$, on a :

$$\forall j \in \mathbb{N}_d, \forall \mathbf{z} \in \mathbb{R}^d, \quad \hat{J}(h)_{ij}(\mathbf{z}) = \frac{\partial h(\mathbf{z})^i}{\partial z^j} = \sum_{\ell=1}^{N} \frac{\partial (K(\mathbf{z},\mathbf{x}_\ell)\mathbf{c}_\ell)^i}{\partial z^j} \tag{5.3}$$

Dans le cas où $h \in \mathcal{H}_K^{\#}$, on obtient l'expression suivante :

$$\forall j \in \mathbb{N}_d, \forall \mathbf{z} \in \mathbb{R}^d, \quad \hat{J}(h)_{ij}(\mathbf{z}) = \frac{\partial h(\mathbf{z})^i}{\partial z^j} = \sum_{\ell=1}^{N} \frac{\partial (K(\mathbf{x}_\ell,\mathbf{z})\mathbf{c}_\ell)^i}{\partial z^j} \tag{5.4}$$

D'autre part, ces modèles non paramétriques ont ceci de particulier que les données font partie des paramètres du modèle. Pour la tâche d'inférence qui nous intéresse, les jacobiennes instantanées sont calculées uniquement aux points de temps observés $t \in \mathbb{N}_N$ de sorte qu'on peut considérer le modèle $\tilde{h}(\mathbf{x}_t) = h(\mathbf{x}_t; \mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_N)$ où l'argument $\mathbf{x}_t$ figure parmi les paramètres du modèle. Le coefficient (i,j) de la jacobienne instantanée du modèle $\tilde{h}$ est alors donnée par le calcul de la dérivée suivante :

$$\forall j \in \mathbb{N}_d, \forall t \in \mathbb{N}_N, \quad \hat{J}(\tilde{h})_{ij}(\mathbf{x}_t) = \frac{\partial \tilde{h}(\mathbf{x}_t)^i}{\partial x_t^j} = \sum_{\ell=1}^{N} \frac{\partial (K(\mathbf{x}_t,\mathbf{x}_\ell)\mathbf{c}_\ell)^i}{\partial x_t^j} \tag{5.5}$$

Dans la suite de l'exposé, nous considérons les modèles à noyaux matriciels h dans $\mathcal{H}_K$ (argument à « gauche » du noyau) et nous calculons les jacobiennes instantanées de ces

modèles en utilisant la dérivée définie en (5.5) (Néhémy Lim et al., 2014). Par souci de simplicité, $\tilde{h}$ est notée h et nous utilisons par abus de langage le vocable de jacobienne pour désigner la dérivée (5.5).

Afin que les coefficients de la jacobienne reflètent la dépendance d'une composante en certaines variables d'état, chaque composante de sortie du modèle h doit être fonction des variables d'état du système. Étant donné notre hypothèse de non linéarité de la dynamique du système sous-jacent, le noyau considéré doit contenir des fonctions non linéaires des variables d'état. De plus, un modèle à noyau à valeur matricielle h pertinent doit permettre un contrôle de la parcimonie de la jacobienne à travers l'apprentissage de ses paramètres. Nous donnons ci-après les expressions des jacobiennes instantanées des modèles à noyaux présentés précédemment en section 4.4.

Modèle OKVAR transformable gaussien. Le coefficient (i, j) de la matrice jacobienne instantanée pour le modèle OKVAR transformable gaussien h_{Gauss} (4.61) est donné par :

$$\hat{J}(h_{\text{Gauss}})_{ij}(\mathbf{x}_t) = 2\gamma(x_t^i - x_t^j)\exp\left(-\gamma(x_t^i - x_t^j)^2\right)c_t^j, \tag{5.6}$$

Les coefficients c_t^j ont tous le même impact, quelle que soit la variable-cible $i \in \mathbb{N}_d$. Par conséquent, il devient impossible de contrôler ces paramètres pour la tâche d'inférence de réseaux.

Modèle OKVAR décomposable gaussien. Si nous considérons le modèle OKVAR décomposable gaussien, h_{dec}, défini dans (4.57), le coefficient (i, j) correspondant de la matrice jacobienne est donné par :

$$\hat{J}(h_{\text{dec}})_{ij}(\mathbf{x}_t) = \sum_{\ell=1}^{N} \frac{\partial \exp(-\gamma\|\mathbf{x}_t - \mathbf{x}_\ell\|^2)}{\partial x_t^j}(B\mathbf{c}_\ell)^i \tag{5.7}$$

Ici, le terme non linéaire apparaissant dans le noyau à valeur matricielle ne diffère pas d'un couple (i, j) à un autre. De ce fait, il est impossible de contrôler une valeur spécifique de la matrice jacobienne à travers B ou les vecteurs $\mathbf{c}_\ell$.

Nous nous apercevons donc que le noyau décomposable gaussien et le noyau transformable gaussien ne permettent pas à eux seuls de réaliser la tâche d'inférence de réseaux. Considérons à présent un noyau décomposable gaussien K_{dec} ayant pour paramètres une largeur de bande $\gamma_1 > 0$ et une matrice semi-définie positive B et K_{Gauss} un noyau transformable gaussien de largeur de bande $\gamma_2 > 0$. Les noyaux à valeur opérateur peuvent être combinés astucieusement pour donner d'autres noyaux semi-définis positifs à travers différentes opérations (Caponnetto, C. A. Micchelli et al., 2008). Ici, nous proposons de combiner les noyaux K_{dec} et K_{Gauss} par le produit de Hadamard, i.e. le produit matriciel

terme à terme (N. Lim et al., 2013).

Définition 5.2 (Noyau Hadamard)**.** *Nous appelons* noyau Hadamard *la fonction* $K_{Hadamard}$ *définie par :*

$$\forall(\mathbf{x},\mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d,\ K_{Hadamard}(\mathbf{x},\mathbf{z}) = K_{dec}(\mathbf{x},\mathbf{z}) \circ K_{Gauss}(\mathbf{x},\mathbf{z}) \tag{5.8}$$

où $\circ$ *représente le produit de Hadamard pour les matrices.*

Proposition 5.1. *La proposition 4.4 assure que le fonction* $K_{Hadamard}$ *définie par (5.8) est un noyau à valeur matricielle.*

Définition 5.3 (Modèle OKVAR Hadamard)**.** *Soit une série temporelle* $\{\mathbf{x}_1,\ldots,\mathbf{x}_{N+1}\} \subseteq \mathbb{R}^d$*, le modèle OKVAR Hadamard noté* $h_{Hadamard}$ *est défini par :*

$$h_{Hadamard}(\mathbf{x}_t) = \sum_{\ell=1}^{N} \exp(-\gamma_1 \|\mathbf{x}_t - \mathbf{x}_\ell\|_2^2) B \circ K_{Gauss}(\mathbf{x}_t,\mathbf{x}_\ell)\mathbf{c}_\ell \tag{5.9}$$

La suite de notre exposé vise à démontrer que nous obtenons une classe plus riche de jacobiennes, plus appropriées pour la tâche d'inférence de structure, si nous utilisons le modèle OKVAR Hadamard (5.9). Le coefficient (i,j) de la matrice jacobienne instantanée $\hat{J}(h_{Hadamard})_{ij}(\mathbf{x}_t)$, notée par souci de simplicité $\hat{J}_{ij}(\mathbf{x}_t)$, est donné par :

$$\hat{J}_{ij}(\mathbf{x}_t) = \sum_{\ell=1}^{N} \frac{\partial \exp(-\gamma_1 ||\mathbf{x}_t - \mathbf{x}_\ell||^2)}{\partial x_t^j} (B \circ K_{Gauss}(\mathbf{x}_t,\mathbf{x}_\ell)\mathbf{c}_\ell)^i + \exp(-\gamma_1 ||\mathbf{x}_t - \mathbf{x}_\ell||^2) \frac{\partial (B \circ K_{Gauss}(\mathbf{x}_t,\mathbf{x}_\ell)\mathbf{c}_\ell)^i}{\partial x_t^j}$$

On obtient, après quelques calculs intermédiaires, le résultat suivant :

$$\begin{aligned} \hat{J}_{ij}(\mathbf{x}_t) \quad = \quad & 2\gamma_2 b_{ij}(x_t^i - x_t^j) \exp\left(-\gamma_2 (x_t^i - x_t^j)^2\right) c_t^j \\ & -2\gamma_1 \sum_{\ell \neq t} \exp(-\gamma_1 \|\mathbf{x}_t - \mathbf{x}_\ell\|^2)(x_t^j - x_\ell^j) \sum_{p=1}^{d} b_{ip} \exp\left(-\gamma_2 (x_t^i - x_\ell^p)^2\right) c_\ell^p \end{aligned} \tag{5.10}$$

Cette expression de la jacobienne nous permet de mettre en lumière plusieurs caractéristiques intéressantes : si la largeur de bande γ_1 avoisine 0, alors $k_{Gauss}(\mathbf{x}_t,\mathbf{x}_\ell)$ vaut à peu près 1 pour tout couple $(\mathbf{x}_t,\mathbf{x}_\ell)$ et le second terme dans (5.10) est approximativement égal à 0. La valeur du coefficient (i,j) de la jacobienne est alors contrôlée par b_{ij}, le coefficient (i,j) de la matrice B. En effet, pour un couple de variables d'état (i,j), si $b_{ij} = 0$, alors quelles que soient les valeurs des paramètres $c_\ell^j,\ \ell \in \mathbb{N}_N$, le coefficient $\hat{J}_{ij}(\mathbf{x}_t)$ est quasi-nul

et on conclut que la variable j n'influence pas la variable i. La matrice B est donc en mesure d'imposer une structure dans le modèle, on dit que B joue le rôle de *masque* pour la structure du réseau sous-jacent. Inversement, un coefficient b_{ij} non nul n'implique pas nécessairement qu'il existe une influence de la variable j sur la variable i puisque les paramètres $\mathbf{c}_\ell$ peuvent encore annuler le coefficient (i, j) de la jacobienne. Ainsi, nous nous apercevons du rôle *conjoint* des deux types de paramètres, la matrice B et les paramètres $\mathbf{c}_\ell$. Enfin, remarquons que les vecteurs $\mathbf{c}_\ell$ et les différences croisées entre les coordonnées dans l'équation (5.10) permettent d'obtenir des matrices jacobiennes non symétriques, une caractéristique souhaitable pour la reconstruction de structures orientées.

5.4 Apprentissage des paramètres du modèle OKVAR Hadamard

Nous avons détaillé l'apprentissage d'un modèle OKVAR (section 4.5) lorsque le noyau K est entièrement spécifié *a priori*. Pour la tâche d'inférence de réseaux, si l'on choisit un noyau décomposable gaussien K_{dec} ou un noyau Hadamard K_{Hadamard}, une fois les paramètres de largeur de bande γ_1 et γ_2 fixés, il reste à apprendre la matrice semi-définie positive B qui joue un rôle crucial. Cela conduit au problème plus délicat d'apprendre conjointement la matrice des paramètres du modèle $C \in \mathbb{R}^{d \times N}$ et B. Ces matrices sont estimées de manière à minimiser la fonction de coût suivante :

$$\mathcal{J}(B, C) = \sum_{t=1}^{N} ||h_{B,C}(\mathbf{x}_t) - \mathbf{x}_{t+1}||^2 + \Omega(h_{B,C}) \tag{5.11}$$

où $h_{B,C}$ est un modèle OKVAR décomposable gaussien ou Hadamard et $\Omega(h_{B,C})$ incorpore des *a priori* sur le modèle :

$$\Omega(h_{B,C}) = \lambda_h \left\|h_{B,C}\right\|^2_{\mathcal{H}_K} + \Omega_C(C) + \Omega_B(B)$$

où on rappelle $\left\|h_{B,C}\right\|^2_{\mathcal{H}_K}$ est définie par :

$$\left\|h_{B,C}\right\|^2_{\mathcal{H}_K} = \sum_{i,j=1}^{N} \mathbf{c}_i^T K(\mathbf{x}_i, \mathbf{x}_j)\mathbf{c}_j \tag{5.12}$$

Ω_C est une norme (Ω_{ℓ_1} ou Ω_{ℓ_1/ℓ_2}) qui induit de la parcimonie et $\Omega_B = \lambda_B \|\cdot\|_1 + \iota_{\mathbb{S}^d_+}$ est une régularisation parcimonieuse à laquelle on a rajouté l'indicatrice du cône des matrices semi-définies positives de taille $d \times d$ pour tenir compte de la contrainte de semi-définie positivité de B. Lorsque $\hat{C}$ est fixé, la norme $\left\|h_{B,\hat{C}}\right\|^2_{\mathcal{H}_K}$ impose une contrainte de lissage sur B tandis que la norme ℓ_1 de B sert au contrôle de la parcimonie du modèle et de sa

jacobienne. Par ailleurs, à $\hat{B}$ fixé, la fonction de coût $\mathcal{J}(\hat{B}, C)$ se réduit à :

$$\mathcal{J}(\hat{B}, C) = \sum_{t=1}^{N} \|h_{\hat{B},C}(\mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \lambda_h \|h_{B,C}\|^2_{\mathcal{H}_K} + \Omega_C(C), \tag{5.13}$$

tandis que, à $\hat{C}$ fixé, la fonction de coût $\mathcal{J}(B, \hat{C})$ devient :

$$\mathcal{J}(B, \hat{C}) = \sum_{t=1}^{N} \|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \lambda_h \|h_{B,C}\|^2_{\mathcal{H}_K} + \lambda_B ||B||_1 + \iota_{\mathbb{S}^d_+}(B) \tag{5.14}$$

On note que, à $\hat{B}$ fixé, $\mathcal{J}(\hat{B}, C)$ est convexe en C et de même, à $\hat{C}$ fixé, $\mathcal{J}(B, \hat{C})$ est convexe en la variable B. Nous proposons donc une méthode d'optimisation alternée pour minimiser la fonction de coût globale $\mathcal{J}(B, C)$. De plus, étant donné que les deux fonctions de coût $\mathcal{J}(\hat{B}, C)$ et $\mathcal{J}(B, \hat{C})$ peuvent s'écrire comme une somme de termes différentiables et non-lisses, nous recourons aux méthodes de gradient proximal pour minimiser chacune d'elles. Les grandes lignes de la méthode sont données dans l'Algorithme 5.1.

Algorithme 5.1 Minimiser (5.11)

Entrées : $B_0 \in \mathbb{S}^d_+$; M ; $\epsilon_B > 0$; $\epsilon_C > 0$
Initialisation : $m := 0$; `stop` :=`faux`
tant que $m < M$ et `stop`=`faux` **faire**
1. Étant donné B_m, minimiser la fonction de coût (5.13) et obtenir C_m
2. Étant donné C_m, minimiser la fonction de coût (5.14) et obtenir B_{m+1}
3.
si $m > 0$ **alors**
`stop` :=$\|B_{m+1} - B_m\|_F \leq \epsilon_B$ et $\|C_m - C_{m-1}\|_F \leq \epsilon_C$
fin si
4. $m := m + 1$
fin tant que

À l'itération m de l'algorithme, B_m est fixé, et le noyau K (décomposable gaussien ou Hadamard) est bien défini. Par conséquent, l'estimation de C_m à l'étape 1 revient simplement à appliquer l'Algorithm 4.1 pour minimiser (5.13).

Apprendre la matrice B à C fixé

Lorsque la matrice de paramètres C est fixée, la fonction de coût $\mathcal{J}(B, \hat{C})$ (5.14) peut être décomposée de la façon suivante :

$$\mathcal{J}(B, \hat{C}) = f_B(B) + g_{B,1}(B) + g_{B,2}(B) \tag{5.15}$$

avec :

— $f_B(B) = \sum_{t=1}^{N} \|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \lambda_h \left\|h_{B,\hat{C}}\right\|^2_{\mathcal{H}_K}$
— $g_{B,1}(B) = \lambda_B ||B||_1$
— $g_{B,2}(B) = \iota_{\mathbb{S}^d_+}(B)$

La fonction f_B est différentiable par rapport à B, tandis que les fonctions $g_{B,1}$ et $g_{B,2}$ sont toutes deux non lisses, mais convexes et sous-différentiables par rapport à B. Comme deux fonctions non lisses apparaissent dans la fonction de coût à minimiser, nous adoptons la stratégie d'éclatement explicite-implicite généralisé proposée par Raguet et al. (2013) (section 2.4). Notre algorithme 5.2 est présenté ci-dessous.

Algorithme 5.2 Minimiser (5.14)

Entrées : M ; ϵ_B ; $Z_1^{(0)}, Z_2^{(0)} \in \mathbb{S}^d_+$; $\alpha \in (0,1)$; $\forall m \in \mathbb{N}, \eta_m \in (0, 2/L_B)$; $\forall m \in \mathbb{N}, \mu_m \in I_\mu$
Initialisation : $m := 0$; $B_0 := \alpha Z_1^{(0)} + (1-\alpha) Z_2^{(0)}$; `stop` :=`faux`
tant que $m < M$ et `stop`=`faux` **faire**
1.1. $Z_1^{(m+1)} = Z_1^{(m)} + \mu_m \left(\mathrm{prox}_{\frac{\eta_m}{\alpha}}(g_{B,1}) \left(2B_m - Z_1^{(m)} - \eta_m \nabla_B f_B(B_m)\right) - B_m\right)$
1.2. $Z_2^{(m+1)} = Z_2^{(m)} + \mu_m \left(\mathrm{prox}(g_{B,2}) \left(2B_m - Z_2^{(m)} - \eta_m \nabla_B f_B(B_m)\right) - B_m\right)$
2. $B_{m+1} = \alpha Z_1^{(m+1)} + (1-\alpha) Z_2^{(m+1)}$
3. `stop` :=$||B_{m+1} - B_m||_F \leq \epsilon_B$
4. $m := m + 1$
fin tant que

L'algorithme fait apparaître deux opérateurs proximaux : celui correspondant à $g_{B,1}$ est l'opérateur de seuillage doux (2.33) et l'opérateur proximal de l'indicatrice $\iota_{\mathbb{S}^d_+}$ est la projection euclidienne $\Pi_{\mathbb{S}^d_+}$ sur le cône des matrices semi-définies positives de taille $d \times d$. La convergence de la suite $(B_m)_m$ est garantie si les conditions suivantes sont vérifiées (Théorème 2.1 dans Raguet et al. (2013)) : On note $\overline{\lim}\eta_m = \overline{\eta}$,

(A) (i) $0 < \underline{\lim}\mu_m \leq \overline{\lim}\mu_m < \min\left(\frac{3}{2}, \frac{1+2/(L_B\overline{\eta})}{2}\right)$

(ii) $\sum_{m=0}^{+\infty} \|u_{2,m}\| < +\infty$, et pour $i \in \{1,2\}$, $\sum_{m=0}^{+\infty} \|u_{1,m,i}\| < +\infty$

(B) (i) $0 < \underline{\lim}\eta_m \leq \overline{\eta} < \frac{2}{L_B}$

(ii) $I_\mu = (0,1]$

où, pour $i \in \{1,2\}$, $u_{1,m,i}$ désigne l'erreur commise à l'itération m lorsqu'on calcule l'opérateur proximal $\mathrm{prox}(g_{B,i})$ et $u_{2,m}$ est l'erreur quand on applique $\nabla_B f_B$ à son argument. Ces conditions sont satisfaites en sélectionnant précautionneusement les paramètres α, η_m et μ_m, comme nous le montrons dans la section consacrée aux résultats.

5.5 OKVAR-Boost

Nous avons fait remarquer que le problème de l'apprentissage conjoint de la matrice des paramètres du modèle C et de la matrice B est une tâche difficile qui implique la résolution d'un problème non convexe. Si l'algorithme 5.1 permet d'obtenir des solutions en un temps raisonnable pour des problèmes de dimension moyenne (d de l'ordre de la centaine), il devient inenvisageable d'appliquer la méthode en l'état en très grande dimension. Nous proposons d'appliquer OKVAR-Boost afin de pallier ce problème. Pour cela, l'algorithme 4.3 utilisé pour la prévision doit être en partie modifié car les paramètres des noyaux ne sont plus fixés mais doivent être appris. Les principaux changements figurent à l'étape 4 du nouvel algorithme 5.3. À l'itération m de OKVAR-Boost, les paramètres attachés au modèle OKVAR Hadamard de base $h(\cdot, \mathcal{E}_m)$ sont notés B_m et C_m où la matrice B_m est ici définie comme le Laplacien d'un graphe non orienté, de matrice d'adjacence W_m, de manière à assurer le caractère semi-défini positif de B_m. Apprendre B_m revient donc à apprendre la matrice W_m. Dans ce cadre, nous substituons au problème délicat de l'apprentissage conjoint de B_m et C_m l'apprentissage *découplé* de la matrice W_m et de la matrice des paramètres du modèle C_m en estimant d'abord W_m puis C_m. L'estimation de ces paramètres se fait en utilisant les résidus courants $\left(\tilde{\mathbf{u}}_{l+1}^{(m)}\right)_{t=1}^{N}$ dans le sous-ensemble $\mathcal{E}_m$.

La matrice W_m est le résultat d'un apprenant faible de graphes. En l'occurrence, nous avons fait le choix d'une estimation par corrélations partielles, un test statistique d'indépendance conditionnelle détaillé ci-après. Pour la clarté de l'exposé, l'indice m est omis à propos. Pour chaque couple de variables $(i, j) \in \mathbb{N}_d^2$, un coefficient de corrélation partielle empirique r_{ij} est calculé à partir des données projetées sur $\mathcal{E}$ conditionnellement aux autres variables dans $\mathcal{E} \setminus \{i, j\}$. Les coefficients r_{ij} sont alors obtenus via la formule suivante :

$$r_{ij} = \frac{-\hat{\Sigma}_{ij}^{-1}}{\sqrt{\hat{\Sigma}_{ii}^{-1}\hat{\Sigma}_{jj}^{-1}}} \tag{5.16}$$

où $\hat{\Sigma}$ est la matrice de covariance empirique. En faisant l'hypothèse que les variables suivent une loi normale multidimensionnelle, r_{ij} vaut zéro si, et seulement, si les variables d'état i et j sont indépendantes conditionnellement aux autres variables. Le test d'hypothèses statistique suivant est ensuite réalisé :

$$\begin{aligned} H_0 :&\quad r_{ij} = 0 \quad \text{(hypothèse nulle)} \\ H_1 :&\quad r_{ij} \neq 0 \quad \text{(hypothèse alternative)} \end{aligned}$$

La statistique de test fait apparaître la transformée z de Fisher de la corrélation partielle : $z(r_{ij}) = \frac{1}{2}\ln\left(\frac{1+r_{ij}}{1-r_{ij}}\right)$. L'hypothèse nulle H_0 est rejetée significativement au niveau de

confiance $1-\alpha$ ($\alpha \in (0,1)$) si

$$\sqrt{N-(q-2)-3}\cdot|z(r_{ij})| > \Phi^{-1}\left(1-\frac{\alpha}{2}\right)$$

où Φ est la fonction de répartition d'une loi normale centrée réduite $\mathcal{N}(0,1)$. On obtient alors la matrice W par la règle : $W_{ij}=1$ si H_0 est rejetée, 0 sinon.

Algorithme 5.3 OKVAR-Boost pour l'inférence de réseaux

Entrées :
— Vecteurs d'etat observés : $\mathbf{x}_1,\ldots,\mathbf{x}_{N+1}\in\mathbb{R}^d$
— Seuil d'arrêt précoce : $\epsilon>0$
— Taille des sous-ensembles : $q\in\mathbb{N}_d$
Initialisation :
— $\forall t\in\mathbb{N}_N$, $H_0(\mathbf{x}_t):=(\bar{\mathbf{x}}^1,\ldots,\bar{\mathbf{x}}^d)^T$
— $m:=0$, `stop` :=`faux`
tant que $m<M$ et `stop`=`faux` **faire**
0. $m:=m+1$
1. $\forall t\in\mathbb{N}_N$, $\mathbf{u}_{t+1}^{(m)}:=\mathbf{x}_{t+1}-H_{m-1}(\mathbf{x}_t)$
2. $\mathcal{I}_m^\epsilon=\left\{j\in\mathbb{N}_d;\ \left\|\mathbf{u}^{j(m)}\right\|_2>\epsilon\right\}$; `stop` :=« $\mathcal{I}_m^\epsilon=\emptyset$? »
si `stop`=`faux` **alors**
3. Sélectionner $\mathcal{E}_m$, un sous-ensemble aléatoire des variables d'état, de taille q dans $\mathcal{I}_m^\epsilon$
4. À partir de $\mathbf{u}_2^{(m)},\ldots,\mathbf{u}_{N+1}^{(m)}$, (a) estimer $W_m\in\{0,1\}^{q\times q}$ et calculer B_m comme le Laplacien du graphe associé à W_m, (b) estimer la matrice des paramètres du modèle C_m et (c) estimer ρ_m par une recherche linéaire
5. $H_m:=H_{m-1}+\rho_m h(\cdot;\mathcal{E}_m)$
fin si
fin tant que
$m_{\text{stop}}:=m$
6. Calculer la matrice jacobienne empirique $\hat{J}(H_{m_{\text{stop}}})$ puis la seuiller pour obtenir un estimateur de la matrice d'adjacence $\hat{A}$.
Sortie : $\hat{A}$

Jacobienne utilisée pour OKVAR-Boost

Nous proposons de détailler ici le calcul de la jacobienne du modèle d'ensemble H. Les modèles OKVAR de base $h_m\in\mathcal{H}_{K^{(m)}_{\text{Hadamard}}}$ étudiés sont de la forme :

$$\forall\mathbf{x}\in\mathbb{R}^{|\mathcal{E}_m|},\quad h_m(\mathbf{x})=\sum_{\ell=1}^{N}K^{(m)}_{\text{Hadamard}}(\mathbf{x}_\ell^{(m)},\mathbf{x})\mathbf{c}_\ell^{(m)} \tag{5.17}$$

où $\mathbf{x}_\ell^{(m)}=(x_\ell^i,i\in\mathcal{E}_m)\in\mathbb{R}^{|\mathcal{E}_m|}$.

La matrice jacobienne empirique $J(H)$ du modèle d'ensemble H est calculée à partir des jacobiennes empiriques des modèles de base $J(h_m),\ m \in \mathbb{N}_{m_{\text{stop}}}$:

$$\forall (i,j) \in \mathbb{N}_d \times \mathbb{N}_d, J_{ij}(H) = \sum_{m=1}^{m_{\text{stop}}} \rho_m J_{ij}(h\left(\cdot; \mathcal{E}_m\right))$$

où

$$J_{ij}(h\left(\cdot; \mathcal{E}_m\right)) = \begin{cases} \frac{1}{N} \sum_{t=1}^{N} J_{ij}(h_m)(\mathbf{x}_t^{(m)}) & \text{si } (i,j) \in \mathcal{E}_m \times \mathcal{E}_m \\ 0 & \text{sinon} \end{cases} \tag{5.18}$$

Par souci de clarté dans la présentation des calculs, l'indice m est omis délibérément. Le coefficient $(i,j) \in \mathcal{E}_m \times \mathcal{E}_m$ de la jacobienne instantanée, $J_{ij}(h_m)(\mathbf{x}_t)$, est donné par :

$$\begin{aligned}
J_{ij}(h_m)(\mathbf{x}_t) &= \frac{\partial h_m(\mathbf{x}_t)^i}{\partial x_t^j} \\
&= \sum_{\ell=1}^{N} \sum_{p=1}^{d} c_\ell^p \frac{\partial K_{\text{Hadamard}}(\mathbf{x}_\ell, \mathbf{x}_t)_{ip}}{\partial x_t^j} \\
&= \sum_{\ell=1}^{N} \sum_{p=1}^{d} c_\ell^p \frac{\partial}{\partial x_t^j} \left\{ b_{ip} \exp(-\gamma_1 \left\|\mathbf{x}_\ell - \mathbf{x}_t\right\|_2^2)\ \exp\left(-\gamma_2 (x_\ell^i - x_t^p)^2\right) \right\} \\
&= \sum_{\ell=1}^{N} \sum_{p=1}^{d} b_{ip} c_\ell^p \left\{ \exp\left(-\gamma_1 \left\|\mathbf{x}_\ell - \mathbf{x}_t\right\|_2^2\right) \frac{\partial}{\partial x_t^j}\ \exp\left(-\gamma_2 (x_\ell^i - x_t^p)^2\right) \right. \\
&\qquad \left. + \exp\left(-\gamma_2 (x_\ell^i - x_t^p)^2\right) \frac{\partial}{\partial x_t^j} \exp\left(-\gamma_1 \left\|\mathbf{x}_\ell - \mathbf{x}_t\right\|_2^2\right) \right\}
\end{aligned}$$

On obtient alors l'expression suivante pour $J_{ij}(h_m)(\mathbf{x}_t)$:

$$\begin{aligned}
& J_{ij}(h_m)(\mathbf{x}_t) \\
= \ & 2 b_{ij} \gamma_2 \sum_{\ell=1}^{N} c_\ell^j \exp\left(-\gamma_1 \left\|\mathbf{x}_\ell - \mathbf{x}_t\right\|_2^2\right) \left(x_\ell^i - x_t^j\right) \exp\left(-\gamma_2 (x_\ell^i - x_t^j)^2\right) \\
& + 2\gamma_1 \sum_{\ell=1}^{N} \sum_{p=1}^{d} c_\ell^p b_{ip} \left(x_\ell^j - x_t^j\right) \exp\left(-\gamma_2 (x_\ell^i - x_t^p)^2\right) \exp\left(-\gamma_1 \left\|\mathbf{x}_\ell - \mathbf{x}_t\right\|_2^2\right)
\end{aligned}$$

Lorsque γ_1 est proche de 0, une approximation du coefficient de la jacobienne admet pour expression : $J_{ij}(h_m)(\mathbf{x}_t) \approx 2 b_{ij} \gamma_2 \sum_{\ell=1}^{N} c_\ell^j \left(x_\ell^i - x_t^j\right) \exp\left(-\gamma_2 (x_\ell^i - x_t^j)^2\right)$.

5.6 Applications

Nous avons mené l'évaluation de la performance de la famille OKVAR et des algorithmes d'optimisation proposés pour la tâche d'extraction de causalités à travers l'étude de deux jeux de données : des données simulées à partir de systèmes biologiques (bases de données issues de la compétition DREAM3) et des données réelles du climat.

5.6.1 Inférence de réseaux de régulation génique

Réseaux de régulation génique

Commençons par définir le cadre du problème biologique qui nous intéresse en reprenant ici les termes de Segal et al. (2003) : « Les fonctions complexes d'une cellule vivante sont réalisées grâce à l'activité concertée de plusieurs gènes [...]. Cette activité est souvent coordonnée par l'organisation du génome en modules de régulation, ou ensembles de gènes co-régulés [...] ». Les gènes codent pour des protéines et les protéines elles-mêmes servent de facteurs de transcription pour d'autres gènes. L'un des objectifs de la biologie moléculaire moderne consiste à identifier les interactions entre les gènes (via des protéines) afin de comprendre le fonctionnement d'organismes vivants dont on a préalablement séquencé le génome. Dans ce contexte, on peut simplifier la complexité d'un réseau de régulation génique (*Gene Regulatory Network*, GRN) en le représentant grossièrement par un graphe dans lequel les sommets sont les gènes et les arcs entre les sommets codent pour des influences régulatrices : on dit qu'un gène i régule l'expression d'un gène j si l'expression du gène i à un temps t influence l'expression du gène j au temps $t+1$.

Les données utilisées pour la tâche d'inférence de réseaux proviennent généralement de puces à ADN qui permettent de mesurer les concentrations en ARN-messager de plusieurs milliers de gènes de manière simultanée. On obtient ainsi des données de grande dimension de séries temporelles multivariées d'expression de gènes. Ces séries temporelles ont la capacité de nous fournir une image détaillée du comportement dynamique des GRNs et constituent des données de choix dans la tâche d'inférence de réseaux. Malheureusement, les séries temporelles obtenues sont généralement courtes (peu de points de temps) et ces rares données sont, de plus, hautement bruitées. Dans un article de revue, Jaeger et Monk (2010) affirment que la tâche de rétro-conception en présence d'une faible quantité de mesures entachées d'erreurs et soumises à des fluctuations aléatoires dues à son environnement, est un problème intrinsèquement difficile, en raison principalement de la pénurie de données comparée au nombre total possible de connexions entre les gènes.

Les données DREAM3

Les bases de données sur lesquelles portent notre étude proviennent de la compétition DREAM3 (Prill et al., 2010). DREAM est un acronyme pour *Dialogue for Reverse Engineering Assessments and Methods* (http://dreamchallenges.org/). Il s'agit d'un consortium scientifique qui organise chaque année des défis autour des problèmes que posent la biologie des systèmes et la médecine translationnelle qui incluent notamment la tâche d'inférence de réseaux de régulation génique. Le projet DREAM3 fournit des données simulées réalistes inspirées de réseaux issus de différents organismes (e.g. la bactérie *E. coli* et la levure *S. cerevisiae* ou *Yeast*) de tailles et de complexités topologiques variées. Notre analyse porte plus particulièrement sur l'étude des réseaux de tailles 10, 50 et 100 générés pour les défis *in silico* de DREAM3. Ces derniers correspondent à des sous-graphes des réseaux de régulation génique, généralement admis, de la bactérie *E. coli* et de la levure *S. cerevisiae*, présentant divers schémas de parcimonie et de structures topologiques. Dans la suite de l'exposé, nous faisons référence à ces réseaux par les sigles E1, E2, Y1, Y2, Y3 accompagnés d'une indication sur leur taille (e.g. Y2-50 correspond à un sous-réseau de *S. cerevisiae* de taille 50). Les données obtenues ont été générées à partir d'un modèle thermodynamique d'expression de gènes auquel s'ajoute un bruit gaussien (Prill et al., 2010). Chacune des 4, 23 et 46 séries temporelles multivariées mises à diposition des participants est constituée de 21 points de temps pour les réseaux de tailles 10, 50 et 100 respectivement. Pour certaines expériences, nous avons jugé pertinent de générer des données supplémentaires (extension des séries temporelles à 50 points de temps ou simulation de séries temporelles additionnelles) pour étudier plus finement le comportement de nos algorithmes dans diverses conditions. À cette fin, nous avons utilisé le même outil, le logiciel GeneNetWeaver (Schaffter et al., 2011), qui a servi à générer les séries temporelles pour la compétition DREAM3.

Protocole d'évaluation

Dans toutes les expériences que nous avons réalisées, nous évaluons la performance de nos modèles au moyen de l'aire sous la courbe ROC (*Receiver Operating Characteristic*) (AUROC) et sous la courbe de précision-rappel (AUPR) qui sont deux mesures classiques pour illustrer la performance d'un classifieur binaire (la classe positive correspond à la présence d'un arc et la classe négative à l'absence d'arc).

Lorsque nous comparons notre méthode à celle d'autres participants de la compétition DREAM3, nous ne donnons que les performances des équipes qui utilisent *seulement* des données de séries temporelles, sachant que d'autres types de données (e.g. données à l'état stationnaire et données de perturbation (*knock-out/knock-down*)) étaient également disponibles pour ce défi, mais celles-ci ne sont pas prises en compte dans les résultats que

nous donnons par la suite.

Cas de multiples séries temporelles. En outre, pour chaque réseau, plusieurs séries temporelles multivariées sont à disposition, celles-ci correspondant à différentes conditions initiales et/ou réplicats techniques. Si l'on note L le nombre de séries temporelles, nous appliquons la procédure suivante. Pour $\ell \in \mathbb{N}_L$

- apprendre un modèle OKVAR $h^{(\ell)}$ sur chaque série temporelle multivariée
- former une statistique à partir des matrices jacobiennes instantanées et obtenir l'estimateur $\hat{J}\left(h^{(\ell)}\right)$,
- comme décrit à la section 5.2, ranger les éléments de $\hat{J}\left(h^{(\ell)}\right)$ par ordre décroissant et construire la matrice $R_\ell = (r_{pq}^{(\ell)})_{p,q=1}^{d}$ où le coefficient $r_{pq}^{(\ell)}$ correspond au rang de l'élément $\hat{J}_{pq}\left(h^{(\ell)}\right)$
- moyenner les matrices de rang $\bar{R} = \frac{1}{L}\sum_{\ell=1}^{L} R_\ell$.

L'estimateur final $\hat{A}$ de la matrice d'adjacence est obtenu en seuillant $\bar{R}$. Le graphe associé à $\hat{A}$ est appelé réseau *consensus*.

Critère de stabilité pour la sélection de modèle

Pour la tâche d'inférence de réseaux, on peut recourir à un critère de sélection de modèle faisant intervenir le concept de *stabilité*. Il s'agit d'un critère en échantillon fini utilisé dans des tâches spécifiques telles que le clustering ou bien la sélection de caractéristiques en régression (Meinshausen et Peter Bühlmann, 2010). Un tel critère amène à choisir les hyperparamètres qui fournissent les résultats les plus stables lorsqu'on sous-échantillonne aléatoirement les données. Nous proposons un critère fondé sur le bloc-bootstrap. Cette technique consiste à rééchantillonner une série temporelle en blocs consécutifs en s'assurant que chaque bloc d'observations dans une série temporelle stationnaire puisse être considéré comme *échangeable* (Politis et al., 1999). Nous définissons l'instabilité par bloc (*Block-instability*) notée BIS qui mesure combien les matrices jacobiennes de deux modèles appris à partir de deux sous-échantillons distincts diffèrent en moyenne. Soit m_B le nombre de paires de sous-échantillons blocs-bootstraps, le critère BIS pour une série temporelle $\mathbf{x}_{1:N+1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\}$ est défini par :

$$BIS(\mathbf{x}_{1:N+1}) = \frac{1}{m_B}\sum_{b=1}^{m_B}\left\|\hat{J}(h_{b,1}) - \hat{J}(h_{b,2})\right\|_F^2 \tag{5.19}$$

où $\hat{J}(h_{b,1})$ et $\hat{J}(h_{b,2})$ sont respectivement les estimateurs de jacobiennes des modèles autorégressifs $h_{b,1}$ et $h_{b,2}$ appris à partir de la b-ème paire de sous-échantillons blocs-bootstraps $\{b,1;b,2\}$ tirés de la série temporelle $\mathbf{x}_{1:N+1}$.

Lorsque L séries temporelles sont disponibles, le critère devient :

$$\overline{BIS}\left(\{\mathbf{x}_{1:N+1}^{(\ell)}\}_{\ell=1}^{L}\right) = \frac{1}{L}\sum_{\ell=1}^{L} BIS(\mathbf{x}_{1:N+1}^{(\ell)}) \tag{5.20}$$

Comparaison de plusieurs modèles de la famille OKVAR

Nous proposons dans un premier temps de comparer les modèles OKVAR décrits dans la table 5.1 afin d'évaluer leur pertinence pour la tâche d'inférence de réseaux. Nous testons trois estimateurs de la matrice d'adjacence obtenus à partir de différentes statistiques S_N des coefficients des jacobiennes instantanées :

$$s_N\left(\frac{\partial\hat{h}(\mathbf{x}_1)^i}{\partial x_1^j},\ldots,\frac{\partial\hat{h}(\mathbf{x}_N)^i}{\partial x_N^j}\right)$$

où s_N est soit :

— la moyenne $s_{N,1}(z_1,\ldots,z_N) = \frac{1}{N}\sum_{t=1}^{N} z_t$

— le plus grand élément en valeur absolue $s_{N,2}(z_1,\ldots,z_N) = \max_{1\leq t\leq N}|z_t|$

— la moyenne des valeurs absolues $s_{N,3}(z_1,\ldots,z_N) = \frac{1}{N}\sum_{t=1}^{N}|z_t|$

Les estimateurs qui en résultent sont notés respectivement $\hat{J}_1, \hat{J}_2$ et $\hat{J}_3$.

TABLE 5.1 – Récapitulatif des modèles OKVAR étudiés pour la tâche d'inférence de réseaux.

	Modèles OKVAR								
	$h^{\text{Ridge}}_{k\cdot Id}$	$h^{\ell_1}_{k\cdot Id}$	$h^{\text{Ridge}}_{\text{Gauss}}$	$h^{\ell_1}_{\text{Gauss}}$	$h^{\ell_1/\ell_2}_{\text{Gauss}}$	$h^{\ell_1}_{\text{dec}}$	$h^{\ell_1/\ell_2}_{\text{dec}}$	$h^{\ell_1}_{\text{Hadamard}}$	$h^{\ell_1/\ell_2}_{\text{Hadamard}}$
Noyau	$k_{\text{Gauss}}\times Id$		Transformable gaussien			Décomposable gaussien		Hadamard	
Coût	Eq. (4.62)					Eq. (5.11)			
Ω_C	0	Ω_{ℓ_1}	0	Ω_{ℓ_1}	Ω_{ℓ_1/ℓ_2}	Ω_{ℓ_1}	Ω_{ℓ_1/ℓ_2}	Ω_{ℓ_1}	Ω_{ℓ_1/ℓ_2}

Dans le cas d'une régularisation via une norme mixte ℓ_1/ℓ_2, nous avons à définir les valeurs des poids w_ℓ figurant dans la pénalité Ω_{ℓ_1/ℓ_2} (4.65). Comme les données de séries temporelles observées correspondent ici à la réponse d'un système dynamique à une condition initiale, w_ℓ a été choisi comme une fonction croissante de ℓ, ce qui signifie que nous mettons l'accent sur les premiers points de temps et proposons, par conséquent, de définir w_ℓ comme suit : $w_\ell = 1 - \exp(-(\ell-1))$.

Les tables 5.2, 5.3 et 5.4 montrent que les noyaux $k_{\text{Gauss}}\times Id$, transformable gaussien et décomposable gaussien permettent d'atteindre des niveaux de performance satisfaisants pour certains réseaux, sans pour autant qu'aucun des modèles fondés sur ces trois noyaux ne permettent à lui seul de reconstruire fidèlement tous les réseaux, quelle que soit la ré-

TABLE 5.2 – AUROC et AUPR consensus (données en %) pour les réseaux DREAM3 de taille 10 en utilisant les données originelles de la compétition (4 séries temporelles multivariées de 21 points). Ici, l'estimateur de la matrice d'adjacence choisi est $\hat{J}_1$, la *moyenne* des jacobiennes instantanées. Les modèles étudiés sont donnés dans la table 5.1. Les nombres en **gras** sont les plus grandes valeurs de chaque colonne.

	AUROC					AUPR				
Modèles OKVAR	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
$h^{\text{Ridge}}_{k\cdot Id}$	68.2	58.7	50.8	59.3	56.7	20.6	18.4	9.3	37.0	24.2
$h^{\ell_1}_{k\cdot Id}$	69.2	54.3	42.2	52.6	43.4	21.7	15.0	7.9	23.9	17.8
$h^{\text{Ridge}}_{\text{Gauss}}$	68.8	37.7	62.1	68.6	66.7	15.6	11.2	15.5	46.9	32.9
$h^{\ell_1}_{\text{Gauss}}$	69.3	38.0	61.9	69.3	66.7	15.7	11.3	15.2	47.4	32.8
$h^{\ell_1/\ell_2}_{\text{Gauss}}$	68.7	37.1	62.4	68.6	66.7	15.5	11.1	15.6	47.5	32.6
$h^{\ell_1}_{\text{dec}}$	67.0	68.5	38.2	45.4	38.3	23.6	20.8	7.4	21.1	16.8
$h^{\ell_1/\ell_2}_{\text{dec}}$	65.9	47.8	45.3	56.6	38.5	23.1	14.0	8.3	28.5	16.8
$h^{\ell_1}_{\text{Hadamard}}$	81.2	46.2	47.7	**76.2**	70.5	23.5	12.7	8.7	**50.1**	39.5
$h^{\ell_1/\ell_2}_{\text{Hadamard}}$	**81.5**	**78.7**	**76.5**	70.3	**75.1**	**32.1**	**50.1**	**35.4**	37.4	**39.7**

gularisation employée. Notons que les performances du modèle OKVAR Hadamard appris avec une régularisation en norme mixte ℓ_1/ℓ_2 surpassent uniformément celles des autres modèles OKVAR. Dans l'ensemble, ces résultats tendent à corroborer la discussion de la section 5.3.

TABLE 5.3 – AUROC et AUPR consensus (données en %) pour les réseaux DREAM3 de taille 10 en utilisant les données originelles de la compétition (4 séries temporelles multivariées de 21 points). Ici, l'estimateur de la matrice d'adjacence choisi est $\hat{J}_2$, le *plus grand élément en valeur absolue* des jacobiennes instantanées. Les modèles étudiés sont donnés dans la table 5.1. Les nombres en **gras** sont les plus grandes valeurs de chaque colonne.

	AUROC					AUPR				
Modèles OKVAR	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
$h^{\text{Ridge}}_{k\cdot Id}$	53.7	47.0	42.8	56.8	41.1	11.1	15.7	8.0	27.2	17.4
$h^{\ell_1}_{k\cdot Id}$	52.9	44.3	42.4	50.4	39.5	10.7	19.6	7.9	22.9	17.0
$h^{\text{Ridge}}_{\text{Gauss}}$	64.6	41.2	58.7	65.2	66.8	14.2	11.6	**15.2**	44.3	27.9
$h^{\ell_1}_{\text{Gauss}}$	65.3	40.1	**58.9**	65.8	66.6	14.3	11.5	14.7	44.9	27.8
$h^{\ell_1/\ell_2}_{\text{Gauss}}$	65.1	41.3	58.4	65.5	66.5	14.3	11.7	14.7	44.0	27.6
$h^{\ell_1}_{\text{dec}}$	60.1	41.6	41.7	49.6	37.9	**21.0**	12.3	7.9	22.7	16.6
$h^{\ell_1/\ell_2}_{\text{dec}}$	60.0	46.4	42.9	51.1	33.0	**21.0**	13.5	8.0	22.9	15.5
$h^{\ell_1}_{\text{Hadamard}}$	76.1	39.3	42.1	**73.7**	69.2	19.2	11.4	7.9	**46.9**	36.2
$h^{\ell_1/\ell_2}_{\text{Hadamard}}$	**76.4**	**79.6**	55.4	71.2	**76.2**	19.6	**47.4**	13.8	44.9	**45.2**

TABLE 5.4 – AUROC et AUPR consensus (données en %) pour les réseaux DREAM3 de taille 10 en utilisant les données originelles de la compétition (4 séries temporelles multivariées de 21 points). Ici, l'estimateur de la matrice d'adjacence choisi est $\hat{J}_3$, la *moyenne des valeurs absolues* des jacobiennes instantanées. Les modèles étudiés sont donnés dans la table 5.1. Les nombres en **gras** sont les plus grandes valeurs de chaque colonne.

	AUROC					AUPR				
Modèles OKVAR	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
$h^{\text{Ridge}}_{k\cdot Id}$	67.7	50.7	44.1	59.0	43.6	20.4	20.8	8.2	35.5	18.4
$h^{\ell_1}_{k\cdot Id}$	65.4	43.7	39.0	49.4	37.9	18.2	12.4	7.5	22.3	16.7
$h^{\text{Ridge}}_{\text{Gauss}}$	66.0	41.5	56.8	65.2	67.0	14.6	11.7	14.2	43.2	31.8
$h^{\ell_1}_{\text{Gauss}}$	67.0	39.9	**57.6**	66.1	67.5	14.9	11.5	**14.4**	44.6	33.0
$h^{\ell_1/\ell_2}_{\text{Gauss}}$	66.1	40.9	56.9	65.5	66.8	14.7	11.6	14.2	43.5	31.5
$h^{\ell_1}_{\text{dec}}$	60.7	44.6	38.0	47.5	32.3	**21.7**	12.7	7.4	21.7	15.4
$h^{\ell_1/\ell_2}_{\text{dec}}$	60.3	45.9	43.8	50.5	31.6	21.6	13.6	8.1	22.7	15.3
$h^{\ell_1}_{\text{Hadamard}}$	71.6	39.7	42.8	**72.4**	70.5	17.3	11.4	8.0	**45.7**	37.6
$h^{\ell_1/\ell_2}_{\text{Hadamard}}$	**72.5**	**73.5**	55.1	69.7	**76.9**	18.5	**33.4**	13.5	43.8	**49.1**

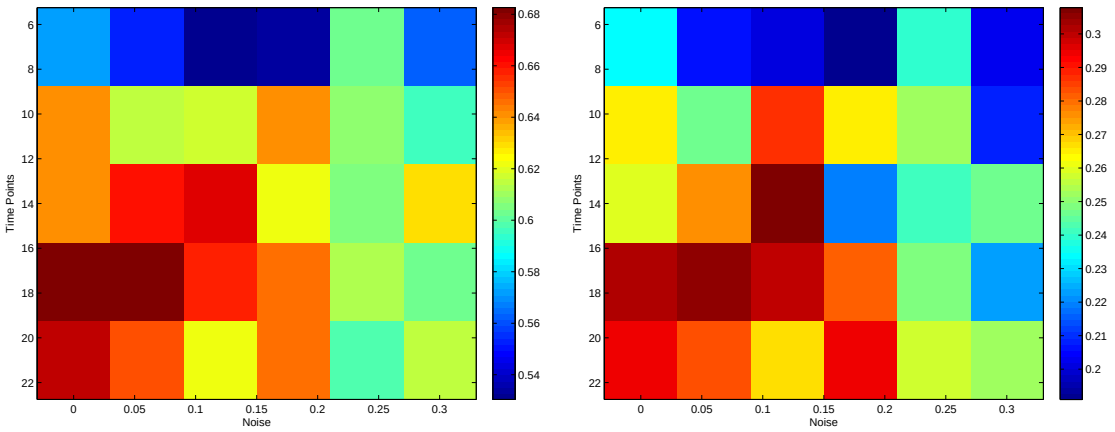
À la lumière de l'ensemble de ces résultats, nous nous focalisons sur les modèles OKVAR Hadamard qu'on note *OKVAR-Prox* et nous utilisons la moyenne des matrices jacobiennes instantanées comme estimateur de la matrice jacobienne.

Effets des hyperparamètres, du bruit, du nombre de données et de la taille de réseau

Nous étudions à présent l'impact de différentes combinaisons de paramètres qui vont de la taille de l'échantillon de la base de données (nombre de points de temps) au nombre de séries temporelles, en passant par le niveau de bruit et la valeur des hyperparamètres λ_C and λ_B.

Taille de l'échantillon et bruit. Premièrement, nous analysons l'effet conjoint du bruit et de la longueur de la série temporelle sur la capacité de notre algorithme à reconstruire les réseaux de taille 10 de DREAM3. Pour cela, un modèle OKVAR-Prox est appris sur les séries tronquées originelles de longueurs $N = 7, 11, 14, 17$ et 21 points de temps, auxquelles est ajouté un bruit gaussien centré d'écarts-types $\sigma = 0, 0.05, 0.1, 0.15, 0.2$ et 0.3. Comme attendu, les performances se détériorent pour un niveau de bruit croissant et une nombre décroissante de points de temps (Figure 5.1).

Taille de l'échantillon pour les réseaux de DREAM3 de taille 100. Nous nous intéressons à présent à la façon dont se comporte l'algorithme dans le cas moins favorable



(a) (b)

FIGURE 5.1 – Moyenne des AUROC (a) et AUPR (b) consensus pour les 5 réseaux de taille 10 de DREAM3 en utilisant $N = 7, 11, 14, 17$ et 21 premiers points des 4 séries temporelles originelles pour différents niveaux de bruit (écarts-types $\sigma = 0, 0.05, 0.1, 0.15, 0.2$ et 0.3).

de l'inférence de grands réseaux. Le premier constat, qu'on peut apporter à la lumière de la figure 5.2, est la nette baisse de performance de la méthode pour les réseaux de DREAM3 de taille 100 en comparaison des résultats obtenus pour les réseaux de taille 10. Cette chute substantielle est probablement due à un contexte plus difficile où le ratio nombre de données $N = 21$ sur taille de réseaux $d = 100$ est plus défavorable par rapport aux bases de données de taille 10 (ratio de $N/d = 21/10$). Pour cette tâche plus ardue, le logiciel GeneNetWeaver nous a permis de générer des points de temps additionnels et d'étendre ainsi la longueur des séries temporelles originelles. Le fait d'apporter au modèle des séries temporelles plus longues améliore quelque peu la tâche d'inférence de réseaux, particulièrement en termes d'AUROC (Figure 5.2). Toutefois, en raison de la stabilité des systèmes dynamiques générés par DREAM3, l'ajout de points de temps n'a pas un impact significatif ici, notamment en termes d'AUPR.

Taille de réseaux et bruit. Afin de déterminer la robustesse de notre méthode, le modèle OKVAR-Prox est évalué sur différentes tailles de réseaux et plusieurs niveaux de bruit. Puis sa performance est comparée à une approche de type LASSO, i.e. autorégression linéaire par moindres carrés pénalisée par une norme ℓ_1 réalisée dimension par dimension (gène). La table 5.5 indique en premier lieu que les deux algorithmes subissent une détérioration de leurs performances pour des tailles de réseaux croissantes. Deuxièmement, OKVAR-Prox surpasse nettement le LASSO, aussi bien en termes d'AUROC et d'AUPR,

et ce quelles que soient les configurations de niveaux de bruit et de tailles de réseaux. Il est enfin à noter que, à taille de réseau fixée, l'impact du bruit est plus prononcé sur les réseaux de taille de 10 que sur ceux de tailles 50 et 100. Pour expliquer cela, on rappelle que l'on ne dispose que de 4 séries temporelles multivariées pour les réseaux de taille 10 tandis que les bases de données pour les réseaux de tailles 50 et 100 sont plus fournies avec respectivement 23 et 46 séries temporelles. Aussi peut-on penser que la technique du consensus de nombreuses séries temporelles permet d'atténuer l'effet du bruit.

Taille de réseaux et nombre de séries temporelles. Pour évaluer la pertinence de la technique du consensus, nous avons généré 50 séries temporelles en utilisant GeneNetWeaver. Les figures 5.3a et 5.3b montrent que l'apport d'un grand nombre de séries temporelles à l'algorithme aide pour la tâche d'inférence de réseaux. L'amélioration est d'autant plus remarquable pour les réseaux de taille 10. Ici, les écarts-types observés sont dus à la variabilité des combinaisons de séries temporelles.

Hyperparamètres. Nous avons également étudié l'impact des hyperparamètres (Table 5.6). Nous remarquons que les performances peuvent fortement varier pour des ordres de grandeur d'hyperparamètres très différents. Pour des changements de faible amplitude des valeurs des hyperparamètres, les valeurs d'AUROC et d'AUPR demeurent relativement

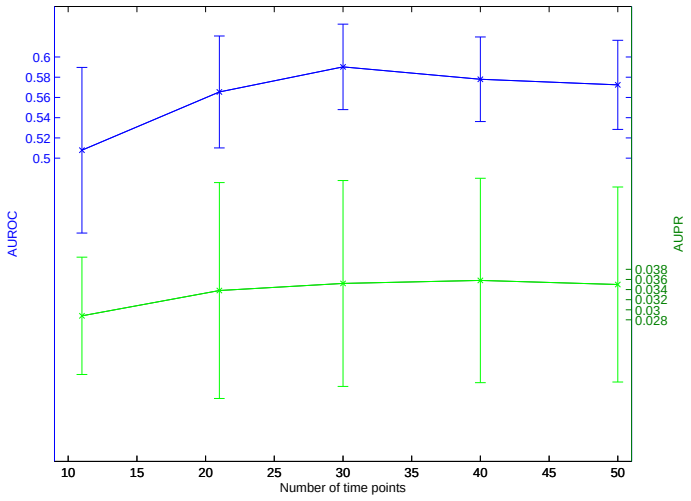


FIGURE 5.2 – Moyenne des AUROC (ligne bleue) et AUPR (ligne verte) consensus pour les 5 réseaux de taille 100 en utilisant les bases de données générées par GeneNetWeaver (20 séries temporelles) pour différents nombres de points de temps ($N = 11, 21, 30, 40$ et 50).

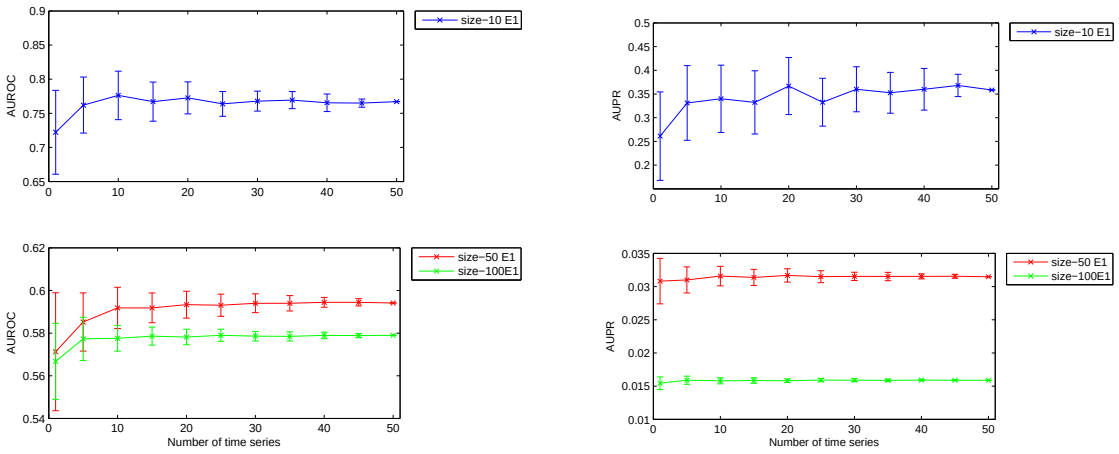
TABLE 5.5 – AUROC et AUPR consensus (données en %) obtenues par OKVAR-Prox et le LASSO pour les réseaux E1 de DREAM3 en utilisant les bases de données originelles (4, 23 et 46 séries temporelles de 21 points pour les tailles 10, 50 et 100 respectivement) et pour différents niveaux de bruit (écarts-types $\sigma = 0, 0.05, 0.1, 0.15, 0.2$ et 0.3). Les nombres en **gras** sont les plus grandes valeurs de chaque colonne.

E1-10	AUROC						AUPR					
Bruit (σ)	0	0.05	0.1	0.15	0.2	0.3	0	0.05	0.1	0.15	0.2	0.3
OKVAR-Prox	**81.5**	**67.9**	**72.0**	**75.2**	**72.4**	**74.4**	**32.1**	**20.7**	**21.2**	**24.3**	**22.7**	**21.4**
LASSO	69.5	64.2	61.7	43.0	50.7	48.8	17.0	13.9	17.8	9.0	10.1	10.7
E1-50												
OKVAR-Prox	**66.4**	**67.3**	**68.6**	**69.2**	**69.8**	**70.9**	**4.1**	**4.3**	**5.0**	**5.9**	**6.6**	**6.9**
LASSO	52.8	54.5	46.0	54.9	45.9	50.7	2.9	2.8	2.1	3.5	2.2	2.6
E1-100												
OKVAR-Prox	**65.4**	**56.1**	**56.5**	**56.4**	**57.2**	**58.0**	**4.6**	**1.7**	**1.7**	**1.7**	**1.7**	**1.8**
LASSO	52.2	54.3	53.3	47.2	51.3	51.5	1.4	1.6	1.4	1.2	1.3	1.3

peu sensibles. Une recherche en grille permet alors de déterminer un ordre de grandeur approprié pour les valeurs d'hyperparamètres.

Choix de la régularisation

Afin de mesurer l'impact du type de régularisation employé, la performance de OKVAR-Prox est évaluée sur trois problèmes de différentes tailles : les réseaux E1 de DREAM3 de



(a) (b)

FIGURE 5.3 – AUROC (a) et AUPR (b) consensus pour les réseaux E1 de DREAM3 de taille 10 (ligne bleue), taille 50 (ligne rouge) et taille 100 (ligne verte) en utilisant 1 à 50 séries temporelles de 50 points. Les écarts-types (barres d'erreur) correspondent à différentes combinaisons de séries temporelles.

TABLE 5.6 – AUROC et AUPR consensus (données en %) pour le réseau E1 de taille 10 de DREAM3 en utilisant les bases de données originelles de la compétition (4 séries temporelles de 21 points) pour différentes combinaisons de valeurs d'hyperparamètres (λ_C, λ_B). $\lambda_h = 1$

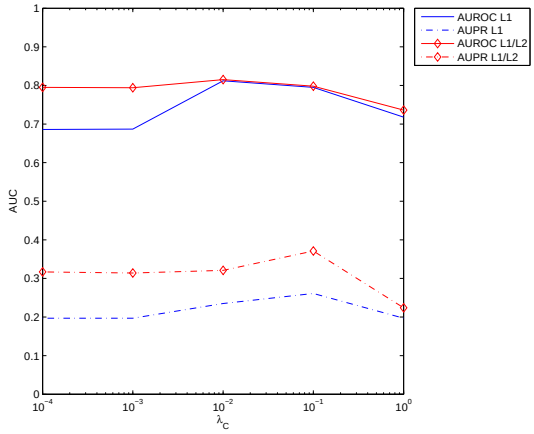
		λ_B		
		10^{-2}	10^{-1}	1
λ_C	10^{-2}	79.1/31.5	81.5/32.1	73.5/21.2
	10^{-1}	79.7/36.5	76.9/21.6	66.7/25.9
	1	78.1/25.9	71.0/20.7	61.4/16.0

tailles 10, 50 et 100. Les figures 5.4(a)–(c) mettent en évidence le fait que le choix de la régularisation dépend de la taille du problème considéré. Spécifiquement, une régularisation impliquant une norme mixte ℓ_1/ℓ_2 est utile pour la tâche d'inférence lorsque le ratio entre le nombre de données disponibles et taille du réseau est favorable. Cela s'avère être le cas pour les réseaux de taille 10 de DREAM3. *A contrario*, il demeure plus compliqué de décider laquelle des pénalités est la plus appropriée entre une norme ℓ_1 et une norme mixte ℓ_1/ℓ_2 pour de grands réseaux pour lesquels on dispose de peu de données, e.g. les réseaux de tailles 50 et 100 de DREAM3.

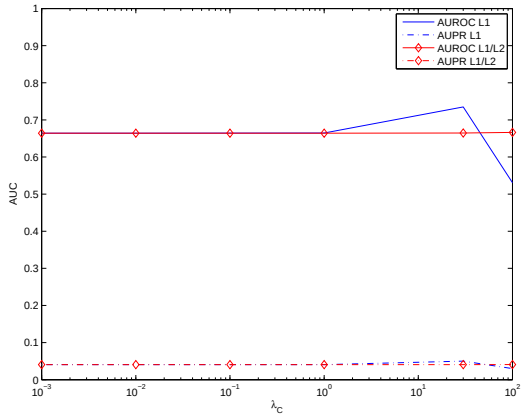
Comparaison avec les méthodes de l'état de l'art

Pour mettre en évidence la pertinence de notre approche pour la tâche d'apprentissage de structure, nous nous comparons à d'autres techniques d'inférences de réseaux sur les bases de données de DREAM3 pour les réseaux de tailles 10 et 100. Les résultats sont consignés dans les Tables 5.7 et 5.8 pour les problèmes de tailles 10 et 100 respectivement. Les lignes des tableaux font référence aux méthodes suivantes : (i) OKVAR + *B vrai* correspond à un modèle OKVAR-Prox appris, le vérité terrain étant connue : le *vrai B* est obtenu en projetant la matrice d'adjacence du réseau sur le cône des matrices semi-définies positives, puis on apprend *uniquement* les paramètres du modèle, à B fixé. Les résultats obtenus pour cette ligne fournissent une borne sur la meilleure performance possible atteinte par le modèle. (ii) Le modèle OKVAR-Prox a été appris en utilisant une contrainte en norme mixte ℓ_1/ℓ_2 sur les paramètres C du modèle pour les bases de données de taille 10 et en norme ℓ_1 sur celles de taille 100. (iii) Le LASSO. (iv) G1DBN est un algorithme fondé sur l'inférence de réseaux bayésiens dynamiques en estimant les dépendances conditionnelles du premier ordre (LÈBRE, 2009). (v) GPODE est une méthode d'inférence de structure qui met en jeu une modélisation non-paramétrique par processus gaussiens et une estimation des paramètres d'équations différentielles ordinaires (ÄIJÖ et LÄHDESMÄKI, 2009). (vi,vii) Enfin, les deux dernières lignes présentent les résultats des

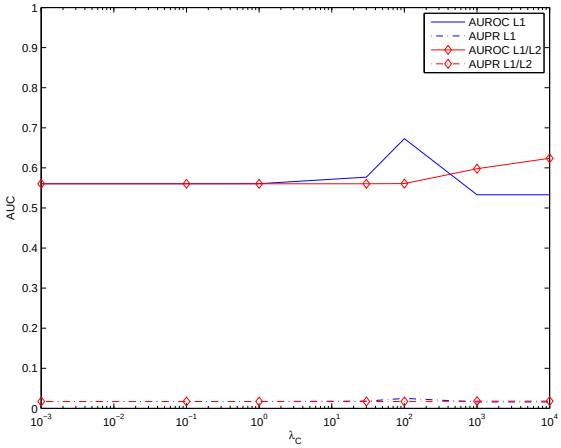
deux équipes ayant participé à la compétition et ayant obtenu les meilleures performances à partir *uniquement* des données de séries temporelles. Bien qu'aucune information sur le type d'algorithme de l'équipe 236 n'ait été fournie, ses auteurs ont répondu à l'enquête



(a) E1 taille 10



(b) E1 taille 50



(c) E1 taille 100

FIGURE 5.4 – AUROC et AUPR consensus pour les réseaux de DREAM3 de (a) taille 10, (b) taille 50 et (c) taille 100 en utilisant les bases de données originelles de DREAM3 (4, 23 et 46 séries temporelles de 21 points pour les réseaux E1 de tailles 10, 50 et 100 respectivement) pour une régularisation en norme ℓ_1 (ligne bleue) et une norme mixte ℓ_1/ℓ_2 (ligne rouge).

post-compétition en indiquant que leur méthode employait des modèles bayésiens intégrant une contrainte sur le degré entrant (Prill et al., 2010). On apprend, dans cette même enquête, que l'équipe 190 utilisait également une méthode bayésienne traitant plus particulièrement l'aspect non linéaire des dynamiques et incorporant une optimisation locale.

Table 5.7 – AUROC et AUPR consensus (données en %) pour les modèles OKVAR + *B vrai*, OKVAR-Prox, LASSO, GPODE, G1DBN, Équipe 236 et Équipe 190 (équipes ayant participé à DREAM3) appliqués sur les bases de données originelles des réseaux de taille 10 de DREAM3 (4 séries temporelles de 21 points). Les nombres en **gras** sont les plus grandes valeurs de chaque colonne, hormis les résultats de OKVAR + *B vrai* qui fournissent une borne sur les performances des modèles OKVAR-Prox.

	AUROC					AUPR				
Taille 10	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
OKVAR + *B vrai*	96.2	86.9	89.2	75.6	86.6	75.2	67.7	47.3	52.3	58.6
OKVAR-Prox	**81.5**	**78.7**	**76.5**	**70.3**	**75.1**	**32.1**	**50.1**	**35.4**	37.4	**39.7**
LASSO	69.5	57.2	46.6	62.0	54.5	17.0	16.9	8.5	32.9	23.2
GPODE	60.7	51.6	49.4	61.3	57.1	18.0	14.6	8.9	**37.7**	34.1
G1DBN	63.4	77.4	60.9	50.3	62.4	16.5	36.4	11.6	23.2	26.3
Équipe 236	62.1	65.0	64.6	43.8	48.8	19.7	37.8	19.4	23.6	23.9
Équipe 190	57.3	51.5	63.1	57.7	60.3	15.2	18.1	16.7	37.1	37.3

Les valeurs d'AUROC et d'AUPR obtenues pour les réseaux de taille 10 (Table 5.7) indiquent clairement qu'OKVAR-Prox surpasse les modèles de l'état de l'art et les équipes qui ont utilisé *uniquement* les données des séries temporelles fournies par la compétition DREAM3, exception faite de Y2-10 (AUPR à peine inférieure à la méthode GPODE). Les résultats révèlent en particulier que notre méthode fait montre d'excellentes performances en termes d'AUPR par rapport aux approches concurrentes.

Une comparaison des algorithmes dans le cas des réseaux de taille 100 (Table 5.8) fait de nouveau ressortir que la méthode OKVAR atteint des résultats en AUROC supérieurs à ceux de l'équipe 236, même si notre méthode exhibe une performance quelque peu en retrait pour les réseaux Y1 et Y3 de taille 100 en termes d'AUPR. L'équipe 236 est la seule équipe à utiliser exclusivement des données de séries temporelles pour le défi sur les réseaux de taille 100, l'équipe 190 n'ayant soumis aucun résultat pour les problèmes de cette taille. Aucun résultat n'est donné non plus pour la méthode GPODE sur les réseaux de taille 100 car l'algorithme nécessite une recherche combinatoire complète lorsqu'aucune connaissance *a priori* n'est disponible, ce qui est calculatoirement prohibitif pour de grands réseaux. La méthode OKVAR est devancée par G1DBN pour E2-100 en termes d'AUPR et pour Y2-100 avec des valeurs d'AUROC comparables. Dans l'ensemble, notons que toutes

les valeurs d'AUPR, quelle que soit la méthode employée, sont plutôt faibles (inférieures à 10%) comparées à celles obtenues pour les réseaux de taille 10. Par conséquent, nous avons appliqué une contrainte en norme ℓ_1 sur les paramètres du modèle, plutôt qu'une régularisation en norme mixte ℓ_1/ℓ_2 qui aurait induit une parcimonie trop stricte, étant donné la rareté des données disponibles. Enfin, nous tenons à faire remarquer qu'OKVAR-Prox aurait figuré parmi les cinq et les dix premiers de la compétition, respectivement pour les défis de taille 10 et de taille 100, sachant que les meilleures équipes ont fait usage des données de perturbation (*knock-out/knock-down*) en sus des données de séries temporelles.

TABLE 5.8 – AUROC et AUPR consensus (données en %) pour les modèles OKVAR + *B vrai*, OKVAR-Prox, LASSO, G1DBN, Équipe 236 (équipe ayant participé à DREAM3) appliqués sur les bases de données originelles des réseaux de taille 100 de DREAM3 (46 séries temporelles de 21 points). Les nombres en **gras** sont les plus grandes valeurs de chaque colonne, hormis les résultats de OKVAR + *B vrai* qui fournissent une borne sur les performances des modèles OKVAR-Prox.

	AUROC					AUPR				
Taille 100	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
OKVAR + *B vrai*	96.2	97.1	95.8	90.6	89.7	43.2	51.6	27.9	40.7	36.4
OKVAR-Prox	**65.4**	**64.0**	**54.9**	56.8	**53.5**	**4.6**	2.6	2.3	**5.0**	6.3
LASSO	52.2	55.0	53.2	52.4	52.3	1.4	1.3	1.8	4.3	6.1
G1DBN	53.4	55.8	47.0	**58.1**	43.4	1.6	**6.3**	2.2	4.6	4.4
Team 236	52.7	54.6	53.2	50.8	50.8	1.9	4.2	**3.5**	4.6	**6.5**

Comparaison avec OKVAR-Boost

Dans la Table 5.9, nous comparons les performances du modèle de base OKVAR-Prox et celles de la stratégie d'ensemble d'OKVAR-Boost (Algorithme 5.3). OKVAR-Prox affiche de meilleures valeurs d'AUROC qu'OKVAR-Boost. Pour les réseaux de taille 10, excepté pour le réseau E1, alors qu'aucune méthode ne l'emporte clairement sur l'autre en termes d'AUPR. Sur les tâches d'inférence de taille 100, OKVAR-Boost bénéficie des projections sur des sous-espaces aléatoires et ses performances surpassent allègrement celles d'OKVAR-Prox qui agit directement dans l'espace de dimension 100 avec une quantité limitée de données.

5.6.2 Données climat

La deuxième application à laquelle nous nous sommes intéressés concerne les données climat, présentes initialement dans LIU et al. (2010). Ces dernières sont obtenues par agrégation de mesures d'agents de forçage du climat et de mécanismes de rétroaction,

TABLE 5.9 – AUROC et AUPR consensus (données en %) pour OKVAR-Prox et OKVAR-Boost appliqués sur les bases de données originelles de DREAM3 (4 et 46 séries temporelles de 21 points pour les réseaux de tailles 10 et 100 respectivement). Les nombres en **gras** sont les plus grandes valeurs de chaque colonne.

	AUROC					AUPR				
Taille 10	E1	E2	Y1	Y2	Y3	E1	E2	Y1	Y2	Y3
OKVAR-Prox	81.5	**78.7**	**76.5**	**70.3**	**75.1**	32.1	50.1	**35.4**	**37.4**	39.7
OKVAR-Boost	**85.3**	74.9	68.9	65.3	69.5	**58.3**	**53.6**	28.3	26.8	**44.3**
Taille 100										
OKVAR-Prox	65.4	64.0	54.9	56.8	53.5	**4.6**	2.6	2.3	5.0	6.3
OKVAR-Boost	**71.8**	**77.2**	**72.9**	**65.0**	**64.3**	3.6	**10.7**	**4.2**	**7.3**	**6.9**

issues de différentes bases de données. Nous avons extrait les mesures mensuelles de douze variables sur l'année 2002 (i.e. 12 points de temps) : la température (TMP), niveau de précipitation (PRE), concentration en vapeur d'eau (VAP), nombre de jours avec une couverture nuageuse (CLD), nombre de jours de pluie (WET), nombre de jours où il a gelé (FRS), concentration en méthane (CH4), concentration en dioxyde de carbone (CO2), concentration en dihydrogène (H2), concentration en monoxyde de carbone (CO), radiation solaire (SOL) et concentration en aérosols (AER). Ces mesures proviennent de 125 stations météorologiques espacées régulièrement à travers tout le territoire des États-Unis.

Un modèle OKVAR-Prox régularisé par une norme ℓ_1 est utilisé pour identifier et explorer les dépendances entre les facteurs naturels et anthropogènes (liés à l'activité humaine). À cette fin, un modèle causal est appris pour chaque série temporelles multivariée correspondant à une des 125 zones spécifiques des États-Unis. Aussi, pour la clarté de l'exposé, commençons-nous par considérer une seule zone située au nord du Texas.

TABLE 5.10 – Moyenne±écart-type du critère BIC pour les données climat pour une zone située au nord du Texas. L'algorithme est appliqué 10 fois pour chaque couple d'hyperparamètres (λ_C, λ_B). $\lambda_h = 1$.

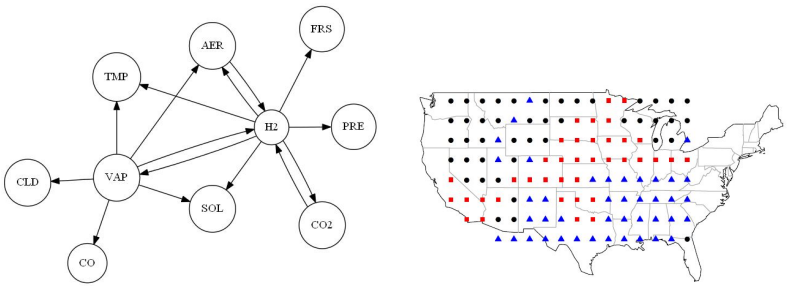
		λ_B		
		10^{-2}	10^{-1}	1
λ_C	10^{-2}	279.89 ± 8.20	224.09 ± 5.76	129.76 ± 4.23
	10^{-1}	311.24 ± 305.32	115.52 ± 8.66	60.66 ± 0.38
	1	$5.63 \times 10^5 \pm 1.78 \times 10^6$	$5.89 \times 10^6 \pm 1.86 \times 10^7$	$\mathbf{51.70 \pm 0.79}$

De manière analogue aux expériences décrites dans la section 5.6.1, nous avons utilisé

une grille pour fixer les hyperparamètres du modèle en recherchant les valeurs de λ_C et λ_B qui minimise un critère d'information, de type BIC (*Bayesian Information Criterion*) défini par :

$$BIC(\lambda_C, \lambda_B) = N \ln\left(\frac{1}{N}\sum_{t=1}^{N} \|\mathbf{x}_{t+1} - h^{\lambda_C,\lambda_B}_{\hat{B},\hat{C}}(\mathbf{x}_t)\|_2^2\right) + \left(\|\hat{C}\|_1 + \frac{1}{2}\|\hat{B}\|_1\right)\ln(N) \quad (5.21)$$

Ce critère est calculé à chacune des dix exécutions de l'algorithme sur les données de la zone du nord Texas sélectionnée, cela correspond à dix initialisations aléatoires de la matrice B_0. Comme on peut le constater dans la Table 5.10, la valeur 1 a été sélectionnée à la fois pour λ_C et λ_B. Par suite, nous pouvons appliquer l'algorithme 5.1 sur les séries temporelles multivariées des 124 autres zones. Un graphe consensus a été construit pour chaque zone en conservant un nombre prédéfini d'arcs. En effet, de par la complexité extrême du système étudié, des relations de dépendance peuvent être trouvées entre la plupart des variables. C'est la raison pour laquelle nous avons considéré une sélection stricte des arcs pour faciliter l'interprétation du graphe extrait. Le niveau de parcimonie (15 arcs sur les 132 possibles) a été fixé après consultation auprès d'experts du climat. Les résultats de notre première expérience sur la base de données du nord Texas aboutissent au graphe orienté parcimonieux de la Figure 5.5a.



(a) (b)

FIGURE 5.5 – (a) Graphe consensus obtenu après 10 applications indépendantes de OKVAR-Prox à la base de données du nord Texas. Les variables WET et CH4 n'apparaissent pas car aucun arc retenu dans le graphe consensus ne les relie au reste du graphe. (b) Partition des réseaux causaux appris par OKVAR-Prox sur les 125 zones des États-Unis via un algorithme de clustering spectral.

La plupart des arcs identifiés par OKVAR-Prox paraissent raisonnables et sont appuyés par des connaissances externes sur les interactions des variables sous-jacentes : spécifiquement, VAP influence CLD puisque la probabilité de voir apparaître une couverture nuageuse (CLD) augmente avec la concentration en vapeur d'eau (VAP). La vapeur est aussi le principal gaz à effet de serre sur terre, ce qui corrobore son impact sur la température (TMP). Les aérosols (AER) interagissent avec le dihydrogène (H2) au travers de réactions

chimiques ayant lieu dans l'atmosphère qui font baisser la concentration en vapeur (VAP) en favorisant la condensation de l'eau. Bien sûr, certains liens causaux manquent dans le réseau obtenu et on aurait pu s'attendre à voir un impact de concentration en dioxyde de carbone (CO2) ou de méthane (CH4) sur la température (TMP). Cependant, nous tenons à préciser que tous ces arcs apparaissent dans le graphe consensus initial et auraient été retenus dans le réseau causal final si un seuil de sélection moins strict avait été choisi.

Par ailleurs, puisque les processus physiques et chimiques en jeu dans l'atmosphère ne changent pas drastiquement d'une zone géographique voisine à une autre, cela suggère que les graphes causaux appris sur l'ensemble du territoire américain devraient révéler un certain degré de similarité. Inversement, les réseaux causaux correspondant à des zones qui sont éloignées exhibent probablement des différences topologiques dues à des spécificités régionales en termes de climat et d'activité humaine. Pour mettre en exergue ces différences et ces similitudes, on définit une similarité s entre deux graphes G_1 et G_2 fondée sur la distance de Hamming entre les matrices d'adjacence correspondantes A_1 et A_2 : $s(G_1, G_2) = 1 - \frac{1}{d^2} \sum_{i,j=1}^{d} |A_{1ij} - A_{2ij}|$. Un algorithme de clustering spectral utilisant cette similarité est alors appliqué avec un nombre de classes fixé à trois. Ce nombre sélectionné *a priori* provient du nombre de variables cachées considérées dans le modèle à variables latentes de Liu et al. (2010) portant sur les interactions spatiales. La figure 5.5b montre les étiquettes des graphes et leur situation sur une carte des États-Unis. Il émerge une segmentation très claire d'ensembles géographiques qui possèdent des structures de réseaux similaires : une première zone, en noir, englobe des localités de l'ouest et de la moitié nord du pays sur lesquelles domine un climat continental humide ; une autre zone, en bleu, couvre des régions industrialisées du sud et de l'est marquées par de hauts taux de $\mathtt{CO}_2$ tandis que la zone rouge qui s'étend principalement sur le centre des États-Unis correspond à des endroits faiblement peuplés où les facteurs anthropogènes sont moins dominants dans notre modèle.

5.7 Synthèse

L'inférence de réseaux à partir de données de séries temporelles multivariées est un problème clé dans de nombreux domaines scientifiques. Nous avons abordé ce problème en introduisant la notion de jacobienne d'un modèle autorégressif vectoriel comme un estimateur de la matrice d'adjacence du réseau causal sous-jacent. Le postulat d'une structure cible creuse et par conséquent d'une jacobienne parcimonieuse a guidé la conception d'un noyau à valeur matricielle et le schéma de régularisation à adopter. La fonction de coût associée fait intervenir des pénalités non lisses et nous avons développé à propos une méthode d'optimisation alternée fondée sur des algorithmes proximaux pour l'estimation

des paramètres du modèle et celle du paramètre du noyau. Alternativement, nous avons adapté OKVAR-Boost, approprié en grande dimension, via une approche modulaire qui se focalise sur des sous-espaces tirés aléatoirement. Dans ce cas-ci, on réalise un apprentissage découplé du paramètre du noyau et des paramètres du modèles. La qualité des performances sur des réseaux benchmarks issus de la compétition biologique DREAM3 ainsi que les résultats obtenus sur les données de climat légitiment notre démarche.

Conclusion et perspectives

Synthèse

Les travaux de cette thèse ont posé les fondements d'une nouvelle famille de modèles autorégressifs vectoriels non paramétriques, les modèles OKVAR (Néhémy Lim et al., 2014). Ces modèles reposent sur la théorie des RKHS des fonctions à valeurs vectorielles (Senkene et Tempel'man, 1973 ; C. Micchelli et M. Pontil, 2005), qui demeure encore relativement peu connue (Alvarez et al., 2011). Ces modèles sont particulièrement adéquats lorsque le problème d'intérêt présuppose un couplage entre les composantes d'un système non linéaire. La richesse de la famille de modèles OKVAR provient essentiellement de la capacité à pouvoir concevoir un noyau matriciel approprié selon la tâche à traiter. Une fois le noyau K fixé, nous nous sommes intéressés au problème de la régularisation en considérant deux pénalités parcimonieuses sur les paramètres du modèle, une contrainte en norme ℓ_1 et une autre en norme mixte ℓ_1/ℓ_2. La norme ℓ_1/ℓ_2 induit une parcimonie structurée qui permet d'annuler ou de sélectionner simultanément un groupe de variables. La partition des coefficients que nous avons choisie autorise un nombre limité de vecteurs de paramètres $\mathbf{c}_\ell$ à être non nuls, ce qui signifie que seuls quelques vecteurs d'état $\mathbf{x}_\ell$ observés à certains points de temps $t_\ell, \ell \in \mathbb{N}_N$ interviennent dans le modèle. Dans le même esprit que les SVMs et les SVRs, les vecteurs $\mathbf{x}_\ell$ non nuls sont assimilables à des vecteurs de support. Nous avons conçu un algorithme proximal qui réalise l'estimation de la matrice C des paramètres du modèle sous les deux types de contraintes. Le calcul des opérateurs proximaux des normes ℓ_1 et ℓ_1/ℓ_2 est, quant à lui, effectué de manière efficace via l'opérateur de seuillage doux.

Le problème de la grande dimension a motivé l'élaboration d'une méthode d'ensemble OKVAR-Boost (N. Lim et al., 2013) dont le principe consiste à chaque itération de l'algorithme à travailler dans un sous-espace aléatoire de faible dimension ($|\mathcal{E}_m| \ll d$), les résidus des variables sélectionnées ayant une norme supérieure à un certain seuil.

Nous avons principalement illustré les performances de la famille de modèles OKVAR sur la tâche d'inférence de réseaux. Pour cela, nous avons proposé un cadre méthodologique général dans lequel la matrice d'adjacence qui encode la structure causale sous-jacente est estimée par une statistique des matrices jacobiennes instantanées d'un modèle autorégressif vectoriel. Lorsque l'on choisit un modèle OKVAR pour réaliser ce type de tâche, c'est alors la forme de la jacobienne qui guide la conception du noyau à valeurs matricielles

à utiliser. Le noyau qui résulte de notre étude est le produit de Hadamard d'un noyau gaussien transformable et d'un noyau gaussien décomposable qui fait apparaître une matrice symétrique semi-définie positive B. Cet hyperparamètre joue un rôle central car il encode les indépendances entre les variables d'état. En faisant l'hypothèse d'une structure causale sous-jacente creuse, l'apprentissage des deux types de paramètres (C et B) s'avère délicat et implique un problème non convexe. Nous avons alors proposé de résoudre ce dernier par un schéma d'optimisation alternée. L'estimation de la matrice B, sous la double contrainte de parcimonie (norme ℓ_1) et de semi-définie positivité, nous a conduit à élaborer un deuxième algorithme proximal.

Alternativement, si l'on décide d'utiliser OKVAR-Boost pour la tâche d'inférence de réseaux, une stratégie consiste à découpler l'apprentissage des deux types de paramètres. À chaque itération m, on estime d'abord une matrice d'adjacence W_m qui reflète les indépendances conditionnelles entre les variables d'état du sous-ensemble $\mathcal{E}_m$ sélectionné. B_m est alors défini comme le Laplacien du graphe associé à W_m de manière à assurer le caractère semi-défini positif de la matrice. Enfin, on procède à l'estimation de C_m sous une contrainte de type filet élastique.

Nous avons évalué l'ensemble des méthodes proposées sur deux jeux de données : des données de réseaux de régulation génique issues de la compétition DREAM3 et des données de climat. Pour les données DREAM3, nous avons mené une série substantielle d'expériences afin de démontrer l'intérêt de nos modèles sous diverses configurations (bruit, longueur des séries temporelles, nombre de séries temporelles, taille des réseaux, impact des hyperparamètres). Les bonnes performances obtenues sur les réseaux benchmarks de DREAM3 de tailles et de complexités topologiques variées illustrent le bien-fondé de nos méthodes. L'application à des données réelles du climat a quant à elle permis d'identifier des relations d'influences entre des facteurs naturels et anthropogènes qui ont pu être validées par des experts.

Perspectives

À l'issue de cette thèse, nous proposons un ensemble de perspectives qui sont d'ordres applicatif et théorique.

Perspectives applicatives

Concernant l'inférence de réseaux de régulation génique, nos travaux illustrent (Table 5.8) les difficultés à inférer des réseaux de grande taille (100 gènes) à partir de courtes séries temporelles (une vingtaine de points). L'on peut alors envisager une méthode d'apprentissage actif où l'utilisateur choisirait de perturber le système d'intérêt, soit en réduisant

l'expression d'un ou plusieurs gènes (*knock-down*) ou bien en les rendant totalement inactifs (*knockout*). Le frein majeur à l'utilisation de ces données dites d'intervention réside dans le coût nécessaire à leur obtention de sorte que relativement peu de méthodes ont été proposées pour inférer des réseaux de régulation génique à partir de ce type de données (Gupta et al., 2011 ; Yip et al., 2010 ; Llamosi et al., 2014). Une alternative consiste à combiner différentes sources de données et d'élaborer une technique qui offre la possibilité de réaliser l'inférence de réseaux en utilisant des données en régime permanent (*steady states*) en sus des données de séries temporelles.

Au-delà de la problématique de l'inférence de réseaux, les modèles OKVAR peuvent être utilisés à propos dans de nombreux champs applicatifs. Par exemple, l'émergence puis l'essor rapide des réseaux de capteurs dans une multitude de domaines (l'industrie, l'environnement, la santé) répond à un besoin croissant de surveillance et de contrôle à des fins de prévention (signalement de dépassement d'un seuil de pollution, détection des fuites de produits toxiques sur des sites industriels, monitoring de patients). Toutes ces problématiques requièrent la mise en œuvre d'outils de modélisation automatiques pour des systèmes dynamiques non linéaires multivariés, que ce soit à des fins de prévision, d'inférence de causalités ou de détection de comportements. Cette dernière fonctionnalité peut notamment être conçue comme un problème multitâches pour lequel les outils que nous avons développés sont également pertinents. Dans ce contexte, chaque tâche (comportement) est modélisée par une fonction à valeurs scalaires et l'on suppose que ces tâches sont interdépendantes dans le sens où elles partagent une structure sous-jacente commune. Plusieurs travaux ont montré que la prise en compte de cette structure par une modélisation conjointe des tâches permet d'atteindre de meilleures performances qu'un apprentissage indépendant tâche par tâche (Evgeniou et al., 2005 ; C. Micchelli et M. Pontil, 2005).

Perspectives théoriques

Ordre p. Nous avons travaillé jusqu'à présent avec l'hypothèse d'un processus markovien sous-jacent d'ordre 1. Un tel postulat peut ne pas être vérifié et cette limitation peut être à l'origine des difficultés dans la tâche de prévision. Afin d'étendre les modèles OKVAR à l'ordre $p > 1$, on peut s'inspirer des travaux de Heiler (1999) qui adapte l'estimateur non paramétrique de Nadaraya-Watson en pondérant les observations passées en fonction de la similarité du motif des p derniers points observés à tous les autres motifs de p points consécutifs dans la série temporelle.

Cadre bayésien. Durant cette thèse, nous avons élaboré nos modèles sans nous préoccuper de la modélisation du bruit. La nature hétéroscédastique des résidus dans une série

d'applications justifie de revisiter les modèles OKVAR à travers une approche par vraisemblance pénalisée, voire d'étudier ces modèles dans un cadre purement bayésien sous l'angle d'un modèle graphique probabiliste. L'intérêt est de pouvoir intégrer de manière naturelle des *a priori* sur les hyperparamètres des noyaux. On peut par exemple spécifier la distribution des coefficients de la matrice B du noyau Hadamard.

Régularisation. Sur les réseaux de régulation génique de taille 100 de DREAM3 (Table 5.8), on remarque qu'il existe encore une différence notable entre les résultats obtenus par nos modèles lorsqu'on doit apprendre la matrice B et ceux d'un modèle OKVAR appris quand le « vrai » B est donné et fixé. Cela suppose que l'apprentissage de B demeure perfectible et que la simple contrainte en norme ℓ_1 n'est peut-être pas suffisante, particulièrement pour des réseaux biologiques qui exhibent diverses propriétés topologiques (modules, *hubs*). Comme dans le cadre bayésien, on aimerait pouvoir intégrer des connaissances *a priori*. Cela est possible dans le cadre de la régularisation en incorporant ces *a priori* sous la forme de contraintes supplémentaires dans la fonction de coût. Par exemple, si le réseau comporte un petit nombre de modules, on peut faire l'hypothèse que la matrice B est de faible rang. La contrainte de faible rang peut être avantageusement relâchée en considérant à la place une pénalité sur la norme nucléaire de B (Richard et al., 2012).

Plus généralement, le choix du noyau à valeurs matricielles K implique une régularisation particulière sur le modèle via la norme RKHS $\|\cdot\|_{\mathcal{H}_K}$ (Baldassarre et al., 2010). C'est donc grâce à une sélection judicieuse de K que l'on peut espérer imposer des contraintes sur la nature du système dynamique étudié.

Consistance. Afin d'apporter des garanties statistiques sur la performance de nos algorithmes, il pourrait être intéressant de donner des bornes sur l'erreur en généralisation de nos modèles. Deux directions sont à explorer. Du côté modélisation de séries temporelles, McDonald et al. (2011) ont dérivé des bornes pour des modèles autorégressifs univariés. D'autre part, Caponnetto et De Vito (2007) et Caponnetto, C. A. Micchelli et al. (2008) ont étudié la consistance des méthodes basées sur des noyaux à valeur opérateur universels pour le cas de données i.i.d. Cette question de consistance se pose également pour l'estimation de la jacobienne d'un modèle OKVAR pour une série temporelle donnée.

Temps continu et ODEs. Enfin, si l'on envisage un temps continu plutôt qu'un temps discret, le choix le plus populaire à ce jour consiste à modéliser l'évolution d'un système dynamique par un ensemble d'ODEs couplées. On peut alors se poser la question de la pertinence des outils que nous avons développés dans ce contexte. Récemment, (Heinonen et d'Alché-Buc, 2014) ont introduit un cadre original pour la modélisation non paramétrique d'ODEs qui bénéficie de la flexibilité d'une approche par régression pénalisée dans

les RKHS à valeurs vectorielles.

Annexe A

Détails des calculs pour l'Algorithme 5.2

À l'itération m, l'algorithme 5.2 requiert le calcul du gradient de la fonction f_B suivante :

$$f_B(B) = \sum_{t=1}^{N} \|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \lambda_h \left\|h_{B,\hat{C}}\right\|^2_{\mathcal{H}_K}$$

où la matrice des paramètres du modèle $\hat{C}$ est fixée. Le calcul de $\nabla_B f_B$ revient donc à expliciter les deux termes suivants :

— $\nabla_B \left\|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\right\|^2$
— $\nabla_B \left\|h_{B,\hat{C}}\right\|^2_{\mathcal{H}_K}$

Nous détaillons dans cette annexe les calculs de ces gradients dans le cas du noyau Hadamard K_{Hadamard}, sachant que les calculs exhibés demeurent valides pour le noyau décomposable Gaussien en posant $K_{\text{Gauss}} = 1$. L'on a alors :

— pour $\frac{\partial}{\partial b_{ij}} \left\|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\right\|^2$

$$\begin{aligned}
& \frac{\partial}{\partial b_{ij}} \left\|h_{B,\hat{C}}(\mathbf{x}_t) - \mathbf{x}_{t+1}\right\|^2 \\
= \; & \frac{\partial}{\partial b_{ij}} \sum_{p=1}^{d} \left(h_{B,\hat{C}}(\mathbf{x}_t)^p - x^p_{t+1}\right)^2 \\
= \; & 2\left(\sum_{\ell=1}^{N} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ij} c^j_\ell\right)\left(h_{B,\hat{C}}(\mathbf{x}_t)^i - x^i_{t+1}\right) \\
& +2\left(\sum_{\ell=1}^{N} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ji} c^i_\ell\right)\left(h_{B,\hat{C}}(\mathbf{x}_t)^j - x^j_{t+1}\right)
\end{aligned}$$

— pour $\frac{\partial}{\partial b_{ij}} \left\| h_{B,\hat{C}} \right\|^2_{\mathcal{H}_K}$

$$\begin{aligned}\frac{\partial}{\partial b_{ij}} ||h_{B,\hat{C}}||^2_{\mathcal{H}} &= \frac{\partial}{\partial b_{ij}} \sum_{t=1}^{N}\sum_{\ell=1}^{N} \mathbf{c}_t^T K_{Hadamard}(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell \\ &= \sum_{t=1}^{N}\sum_{\ell=1}^{N} c_t^i k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ij} c_\ell^j \\ &\quad + c_t^j k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ji} c_\ell^i\end{aligned}$$

Une constante de Lipschitz L_B du gradient $\nabla_B f_B(B)$ doit également être fournie à l'algorithme. L'on peut établir, après quelques calculs intermédiaires, l'inégalité donnée ci-après : Pour $B_1, B_2 \in \mathbb{S}_+^d$,

$$\|\nabla_B f_B(B_1) - \nabla_B f_B(B_2)\|_F^2 \leq L_B^2 \|B_1 - B_2\|_F^2$$

où $\|\cdot\|_F$ désigne la norme de Frobenius d'une matrice et

$$\begin{aligned}&L_B^2 \\ =\ & 4 \sum_{i,j=1}^{d} \left(\sum_{t=1}^{N} \left(\sum_{\ell=1}^{N} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ij} c_\ell^j \right) \left(\sum_{\ell=1}^{N}\sum_{q=1}^{d} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{iq} c_\ell^q \right) \right. \\ & \left. + \sum_{t=1}^{N} \left(\sum_{\ell=1}^{N} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{ji} c_\ell^i \right) \left(\sum_{\ell=1}^{N}\sum_{q=1}^{d} k_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell) K_{\text{Gauss}}(\mathbf{x}_t, \mathbf{x}_\ell)_{jq} c_\ell^q \right) \right)^2\end{aligned}$$

Bibliographie

Bibliographie

Ahmed, Nesreen K et al., « An empirical comparison of machine learning models for time series forecasting », in : *Econometric Reviews* **29**.*5-6* (2010), page(s): 594–621.

Äijö, T. et H. Lähdesmäki, « Learning gene regulatory networks from gene expression measurements using non-parametric molecular kinetics », in : *Bioinformatics* **25** (2009), page(s): 2937–2944.

Aizerman, A, Emmanuel M Braverman et LI Rozoner, « Theoretical foundations of the potential function method in pattern recognition learning », in : *Automation and remote control* **25** (1964), page(s): 821–837.

Altman, Naomi S, « An introduction to kernel and nearest-neighbor nonparametric regression », in : *The American Statistician* **46**.*3* (1992), page(s): 175–185.

Alvarez, M.A., L. Rosasco et N.D. Lawrence, *Kernels for Vector-Valued Functions : a Review*, rapp. tech., 2011.

Anjum, S., A. Doucet et C.C. Holmes, « A boosting approach to structure learning of graphs with and without prior knowledge », in : *Bioinformatics* **25**.*22* (2009), page(s): 2929, issn : 1367-4803.

Arnold, Andrew, Yan Liu et Naoki Abe, « Temporal causal modeling with graphical Granger methods », in : *Proceedings of the 13th ACM SIGKDD international Conference on Knowledge Discovery and Data mining*, ACM, 2007, page(s): 66–75.

Aronszajn, N., « Theory of reproducing kernels », in : *Transactions of the American mathematical society* **68**.*3* (1950), page(s): 337–404.

Auliac, Cédric et al., « Evolutionary approaches for the reverse-engineering of gene regulatory networks : a study on a biologically realistic dataset. », in : *BMC Bioinformatics* **9**.*1* (2008), page(s): 91.

Bach, Francis R, « Exploring large feature spaces with hierarchical multiple kernel learning », in : (2009), page(s): 105–112.

Baldassarre, L. et al., « Vector Field Learning via Spectral Filtering », in : *Machine Learning and Knowledge Discovery in Databases*, sous la dir. de J. Balcázar et al., t. 6321, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2010, page(s): 56–71.

Bansal, M., G.D. Gatta et D. Di Bernardo, « Inference of gene regulatory networks and compound mode of action from time course gene expression profiles », in : *Bioinformatics* **22**.*7* (2006), page(s): 815.

Basso, Katia et al., « Reverse engineering of regulatory networks in human B cells », in : *Nature genetics* **37**.*4* (2005), page(s): 382–390.

Baum, Leonard E et Ted Petrie, « Statistical inference for probabilistic functions of finite state Markov chains », in : *The Annals of Mathematical Statistics* **37**.*6* (1966), page(s): 1554–1563.

Beck, A. et M. Teboulle, « Gradient-based algorithms with applications to signal recovery problems », in : *Convex Optimization in Signal Processing and Communications*, sous la dir. de DP Palomar et YC Eldar, Cambridge press, 2010, page(s): 42–88.

Beck, Amir et Marc Teboulle, « A fast iterative shrinkage-thresholding algorithm for linear inverse problems », in : *SIAM Journal on Imaging Sciences* **2**.*1* (2009), page(s): 183–202.

Bengio, Yoshua, « Learning deep architectures for AI », in : *Foundations and trends® in Machine Learning* **2**.*1* (2009), page(s): 1–127.

Bera, Anil K et Matthew L Higgins, « ARCH models : properties, estimation and testing », in : *Journal of economic surveys* **7**.*4* (1993), page(s): 305–366.

Bickel, Peter J, Ya'acov Ritov et Alexandre B Tsybakov, « Simultaneous analysis of Lasso and Dantzig selector », in : *The Annals of Statistics* (2009), page(s): 1705–1732.

Bishop, Christopher M et al., « Neural networks for pattern recognition », in : (1995).

Bollerslev, Tim, « Generalized autoregressive conditional heteroskedasticity », in : *Journal of econometrics* **31**.*3* (1986), page(s): 307–327.

Bolstad, A., B. Van Veen et R. Nowak, « Causal Network Inference via Group Sparsity Regularization », in : *IEEE Trans Signal Process* **59(6)** (2011), page(s): 2628–2641.

Bonneau, Richard et al., « The Inferelator : an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo », in : *Genome biology* **7**.*5* (2006), page(s): R36.

Boser, Bernhard E, Isabelle M Guyon et Vladimir N Vapnik, « A training algorithm for optimal margin classifiers », in : *Proceedings of the fifth annual workshop on Computational learning theory*, ACM, 1992, page(s): 144–152.

Box, George EP, Gwilym M Jenkins et Gregory C Reinsel, *Time series analysis : forecasting and control*, John Wiley & Sons, 1970.

Boyd, Stephen et Lieven Vandenberghe, *Convex optimization*, Cambridge university press, 2009.

Breiman, Leo, « Bagging predictors », in : *Machine learning* **24**.*2* (1996), page(s): 123–140.

— « Random forests », in : *Machine learning* **45**.*1* (2001), page(s): 5–32.

Breiman, L. et al., *Classification and Regression Trees*, Monterey, CA : Wadsworth et Brooks, 1984.

Brouard, C., F. d'Alché–Buc et M. Szafranski, « Semi-supervised Penalized Output Kernel Regression for Link Prediction », in : *ICML-2011*, 2011, page(s): 593–600.

Bühlmann, P. et S. van de Geer, *Statistics for High-Dimensional Data : Methods, Theory and Applications*, Springer, 2011.

Bühlmann, Peter et Bin Yu, « Boosting with the L_2 loss : regression and classification », in : *Journal of the American Statistical Association* **98**.*462* (2003), page(s): 324–339.

Butte, Atul J et Isaac S Kohane, « Mutual information relevance networks : functional genomic clustering using pairwise entropy measurements », in : *Pac Symp Biocomput*, t. 5, 2000, page(s): 418–429.

Caponnetto, Andrea et Ernesto De Vito, « Optimal rates for the regularized least-squares algorithm », in : *Foundations of Computational Mathematics* **7**.*3* (2007), page(s): 331–368.

Caponnetto, Andrea, Charles A Micchelli et al., « Universal multi-task kernels », in : *The Journal of Machine Learning Research* **9** (2008), page(s): 1615–1646.

Cappé, Olivier, Éric Moulines et Tobias Ryden, *Inference in Hidden Markov Models*, Springer-Verlag New York, 2005.

Carmeli, Claudio, Ernesto De Vito et Alessandro Toigo, « Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem », in : *Analysis and Applications* **4**.*04* (2006), page(s): 377–408.

Carmeli, Claudio, Ernesto De Vito, Alessandro Toigo et Veronica Umanitá, « Vector valued reproducing kernel Hilbert spaces and universality », in : *Analysis and Applications* **8**.*01* (2010), page(s): 19–61.

Chan, Kung Sik et Howell Tong, « On estimating thresholds in autoregressive models », in : *Journal of time series analysis* **7**.*3* (1986), page(s): 179–190.

Chapelle, Olivier, « Training a support vector machine in the primal », in : *Neural Computation* **19**.*5* (2007), page(s): 1155–1178.

Charbonnier, Camille, Julien Chiquet et Christophe Ambroise, « Weighted-LASSO for structured network inference from time course data », in : *Statistical applications in genetics and molecular biology* **9**.*1* (2010).

Chatterjee, Soumyadeep et al., « Sparse Group Lasso : Consistency and Climate Applications », in : *SDM*, 2012, page(s): 47–58.

Chen, Scott Shaobing, David L Donoho et Michael A Saunders, « Atomic decomposition by basis pursuit », in : *SIAM journal on scientific computing* **20**.*1* (1998), page(s): 33–61.

Chou, I.C. et E. O. Voit, « Recent developments in parameter estimation and structure identification of biochemical and genomic systems », in : *Math Biosci.* **219**.*2* (2009), page(s): 57–83.

COMBETTES, Patrick L et Jean-Christophe PESQUET, « Proximal splitting methods in signal processing », in : *Fixed-point algorithms for inverse problems in science and engineering*, Springer, 2011, page(s): 185–212.

CONNOR, Jerome T, R Douglas MARTIN et Les E ATLAS, « Recurrent neural networks and robust time series prediction », in : *Neural Networks, IEEE Transactions on* **5**.*2* (1994), page(s): 240–254.

CORTES, Corinna et Vladimir VAPNIK, « Support-vector networks », in : *Machine learning* **20**.*3* (1995), page(s): 273–297.

DE GOOIJER, Jan G et Rob J HYNDMAN, « 25 years of time series forecasting », in : *International journal of forecasting* **22**.*3* (2006), page(s): 443–473.

DEMIRIZ, A., K.P. BENNETT et J. SHAWE-TAYLOR, « Linear programming boosting via column generation », in : *Machine Learning* **46**.*1* (2002), page(s): 225–254.

DENNIS JR, John E et Robert B SCHNABEL, *Numerical methods for unconstrained optimization and nonlinear equations*, t. 16, Siam, 1996.

D'HAESELEER, Patrik et al., « Linear modeling of mRNA expression levels during CNS development and injury. », in : *Pacific symposium on biocomputing*, t. 4, 1, 1999, page(s): 41–52.

DIETTERICH, Thomas G, « Ensemble methods in machine learning », in : *Multiple classifier systems*, Springer, 2000, page(s): 1–15.

DINUZZO, F. et K. FUKUMIZU, « Learning low-rank output kernels », in : *Proceedings of the 3rd Asian Conference on Machine Learning*, t. 20, JMLR : Workshop and Conference Proceedings, nov. 2011.

DONDELINGER, F., S. LÈBRE et D. HUSMEIER, « Non-homogeneous dynamic Bayesian networks with Bayesian regularization for inferring gene regulatory networks with gradually time-varying structure », in : *Machine Learning Journal* **90(2)** (juil. 2013), page(s): 191–230.

DORFFNER, Georg, « Neural Networks for Time Series Processing », in : *Neural Network World* **6** (1996), page(s): 447–468.

EFRON, Bradley et al., « Least angle regression », in : *The Annals of statistics* **32**.*2* (2004), page(s): 407–499.

ENGLE, Robert F, « Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation », in : *Econometrica : Journal of the Econometric Society* (1982), page(s): 987–1007.

EVGENIOU, Theodoros, Charles A MICCHELLI et Massimiliano PONTIL, « Learning multiple tasks with kernel methods », in : *Journal of Machine Learning Research*, 2005, page(s): 615–637.

Faith, Jeremiah J et al., « Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles », in : *PLoS biology* **5**.*1* (2007), page(s): e8.

Freund, Y. et R. Schapire, « A desicion-theoretic generalization of on-line learning and an application to boosting », in : *Computational learning theory*, Springer, 1995, page(s): 23–37.

Freund, Y. et R.E. Schapire, « Game theory, on-line prediction and boosting », in : *Proceedings of the ninth annual conference on Computational learning theory*, ACM, 1996, page(s): 325–332.

Friedman, J.H., « Greedy function approximation : a gradient boosting machine », in : *The Annals of Statistics* **29**.*5* (2001), page(s): 1189–1232, issn : 0090-5364.

Friedman, Nir, « Inferring Cellular Networks Using Probabilistic Graphical Models », in : *Science* **303**.*5659* (2004), page(s): 799–805.

Fu, W.J., « Penalized regressions : the bridge versus the lasso », in : *Journal of computational and graphical statistics* (1998), page(s): 397–416.

Fujita, André et al., « Modeling gene expression regulatory networks with the sparse vector autoregressive model », in : *BMC Systems Biology* **1**.*1* (2007), page(s): 39.

Geurts, P., L. Wehenkel et F. D'Alché-Buc, « Gradient boosting for kernelized output spaces », in : *Proceedings of the 24th international conference on Machine learning*, ACM, 2007, page(s): 289–296.

Ghahramani, Zoubin, « Learning dynamic Bayesian networks », in : *Adaptive processing of sequences and data structures*, Springer, 1998, page(s): 168–197.

Gilchrist, Simon, Vladimir Yankov et Egon Zakrajšek, « Credit market shocks and economic fluctuations : Evidence from corporate bond and stock markets », in : *Journal of Monetary Economics* **56**.*4* (2009), page(s): 471–493, issn : 0304-3932.

Granger, Clive William John et Allan Paul Andersen, *An introduction to bilinear time series models*, Vandenhoeck und Ruprecht Göttingen, 1978.

Granger, Clive WJ, « Investigating causal relations by econometric models and cross-spectral methods », in : *Econometrica : Journal of the Econometric Society* (1969), page(s): 424–438.

Granger, Clive WJ et Timo Terasvirta, « Modelling non-linear economic relationships », in : *OUP Catalogue* (1993).

Gupta, R. et al., « A computational framework for gene regulatory network inference that combines multiple methods and datasets », in : *BMC Systems Biology* **5 :52** (2011).

Haggan, Valérie et Tohru Ozaki, « Modelling nonlinear random vibrations using an amplitude-dependent autoregressive time series model », in : *Biometrika* **68**.*1* (1981), page(s): 189–196.

HAMILTON, James Douglas, *Time series analysis*, t. 2, Princeton university press Princeton, 1994.

HANSEN, Lars Kai et Peter SALAMON, « Neural network ensembles », in : *IEEE transactions on pattern analysis and machine intelligence* **12**.*10* (1990), page(s): 993–1001.

HART, Jeffrey D, « Some automated methods of smoothing time-dependent data », in : *Journal of Nonparametric Statistics* **6**.*2-3* (1996), page(s): 115–142.

HARTEMINK, Alexander, « Reverse engineering gene regulatory networks », in : *Nat. Biotechnol.* **23**.*5* (2005), page(s): 554–555.

HAURY, Anne-Claire et al., « TIGRESS : trustful inference of gene regulation using stability selection », in : *BMC systems biology* **6**.*1* (2012), page(s): 145.

HEILER, Siegfried, *A survey on nonparametric time series analysis*, rapp. tech., Universität Konstanz, Fakultät für Wirtschaftswissenschaften und Statistik, 1999.

HEINONEN, Markus et Florence D'ALCHÉ-BUC, « Learning nonparametric differential equations with operator-valued kernels and gradient matching », in : *CoRR* **abs/1411.5172** (2014), URL : http://arxiv.org/abs/1411.5172.

HORNIK, Kurt, Maxwell STINCHCOMBE et White HALBERT, « Multilayer feedforward networks are universal approximators », in : *Neural Networks* **2** (1989), page(s): 359–366.

HUANG, Junzhou, Tong ZHANG et al., « The benefit of group sparsity », in : *The Annals of Statistics* **38**.*4* (2010), page(s): 1978–2004.

HUSMEIER, Dirk, « Reverse engineering of genetic networks with Bayesian networks », in : *Biochemical Society Transactions* **31**.*6* (2003), page(s): 1516–1518.

— « Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks », in : *Bioinformatics* **19**.*17* (2003), page(s): 2271–2282.

HUYNH-THU, Van Anh et al., « Inferring Regulatory Networks from Expression Data Using Tree-Based Methods », in : *PLoS ONE* **5**.*9* (2010), page(s): e12776.

IBA, Hitoshi, « Inference of Differential Equation Models by Genetic Programming », in : *Inf. Sci.* **178**.*23* (déc. 2008), page(s): 4453–4468, ISSN : 0020-0255.

JACOB, Laurent, Guillaume OBOZINSKI et Jean-Philippe VERT, « Group lasso with overlap and graph lasso », in : (2009), page(s): 433–440.

JAEGER, Johannes et Nicholas AM MONK, « Reverse engineering of gene regulatory networks », in : *Learning and Inference in Computational Systems Biology* (2010), page(s): 9–34.

JENATTON, Rodolphe, Jean-Yves AUDIBERT et Francis BACH, « Structured variable selection with sparsity-inducing norms », in : *The Journal of Machine Learning Research* **12** (2011), page(s): 2777–2824.

KADRI, H. et al., « Functional Regularized Least Squares Classication with Operator-valued Kernels », in : *ICML-2011*, 2011, page(s): 993–1000.

KALMAN, Rudolph Emil, « A new approach to linear filtering and prediction problems », in : *Journal of Fluids Engineering* **82**.*1* (1960), page(s): 35–45.

KANTOROVICH, Leonid Vital'evich, « Functional analysis and applied mathematics », in : *Uspekhi Matematicheskikh Nauk* **3**.*6* (1948), page(s): 89–185.

KIM, Seyoung et Eric P XING, « Tree-guided group lasso for multi-task regression with structured sparsity », in : (2010), page(s): 543–550.

KIM, Sunyong, Seiya IMOTO et Satoru MIYANO, « Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data », in : *Biosystems* **75**.*1* (2004), page(s): 57–65.

KIMELDORF, George et Grace WAHBA, « Some results on Tchebycheffian spline functions », in : *Journal of Mathematical Analysis and Applications* **33**.*1* (1971), page(s): 82–95.

KOLACZYK, Eric D, *Statistical Analysis of Network Data : Methods and Models*, Springer : series in Statistics, 2009.

KRAMER, Mark A. et al., « Network inference with confidence from multivariate time series », in : *Physical Review E* **79**.*6* (juin 2009), page(s): 061916+.

LAWRENCE, N. et al., éds., *Learning and Inference in Computational Systems Biology*, MIT Press, 2010.

LÈBRE, Sophie, « Inferring dynamic genetic networks with low order independencies », in : *Statistical applications in genetics and molecular biology* **8**.*1* (2009), page(s): 1–38.

LIM, Néhémy et al., « Operator-valued kernel-based vector autoregressive models for network inference », in : *Machine Learning* (2014), page(s): 1–25, ISSN : 0885-6125, DOI : 10.1007/s10994-014-5479-3, URL : http://dx.doi.org/10.1007/s10994-014-5479-3.

LIM, N. et al., « OKVAR-Boost : a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks », in : *Bioinformatics* **29**.*11* (2013), page(s): 1416–1423.

LIU, Y., A. NICULESCU-MIZIL et A. LOZANO, « Learning Temporal Causal Graphs for Relational Time-Series Analysis », in : *ICML-2010*, sous la dir. de Johannes FÜRNKRANZ et Thorsten JOACHIMS, 2010.

LJUNG, Lennart, *System identification*, English, sous la dir. de Jürgen ACKERMANN, t. 70, Lecture Notes in Control and Information Sciences, Springer Berlin Heidelberg, 1985, page(s): 48–83, ISBN : 978-3-540-15533-1, DOI : 10.1007/BFb0007280, URL : http://dx.doi.org/10.1007/BFb0007280.

LLAMOSI, Artémis et al., « Experimental Design in Dynamical System Identification : A Bandit-Based Active Learning Approach », English, in : *Machine Learning and Knowledge Discovery in Databases*, sous la dir. de Toon CALDERS et al., t. 8725, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2014, page(s): 306–321, ISBN : 978-3-662-44850-2, DOI : 10.1007/978-3-662-44851-9_20, URL : http://dx.doi.org/10.1007/978-3-662-44851-9_20.

LOUNICI, Karim et al., « Sup-norm convergence rate and sign concentration property of Lasso and Dantzig estimators », in : *Electronic Journal of statistics* **2** (2008), page(s): 90–102.

LOUNICI, Karim et al., « Taking advantage of sparsity in multi-task learning », in : *arXiv preprint arXiv :0903.1468* (2009).

LOZANO, Aurélie C et al., « Grouped graphical Granger modeling for gene expression regulatory networks discovery », in : *Bioinformatics* **25**.*12* (2009), page(s): i110–i118.

LÜTKEPOHL, Helmut, *Introduction to multiple time series analysis*, Springer, Berlin, 1991.

MAATHUIS, M.H. et al., « Predicting causal effects in large-scale systems from observational data », in : *Nature Methods* **7** (2010), page(s): 247–248.

MANN, W Robert, « Mean value methods in iteration », in : *Proceedings of the American Mathematical Society* **4**.*3* (1953), page(s): 506–510.

MARGOLIN A.and Nemenman, I. et al., « ARACNE : an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context », in : *BMC Bioinf* **7**.*Suppl 1* (2006), page(s): S7.

MARKOWITZ, Harry, « The optimization of a quadratic function subject to linear constraints », in : *Naval research logistics Quarterly* **3**.*1-2* (1956), page(s): 111–133.

MAZUR, Johanna et al., « Reconstructing nonlinear dynamic models of gene regulation using stochastic sampling », in : *BMC Bioinformatics* **10**.*1* (2009), page(s): 448.

MCCULLOCH, Warren S et Walter PITTS, « A logical calculus of the ideas immanent in nervous activity », in : *The bulletin of mathematical biophysics* **5**.*4* (1943), page(s): 115–133.

MCDONALD, Daniel J, Cosma Rohilla SHALIZI et Mark SCHERVISH, « Generalization error bounds for stationary autoregressive models », in : *arXiv preprint arXiv :1103.0942* (2011).

MEINSHAUSEN, Nicolai et Peter BÜHLMANN, « High dimensional graphs and variable selection with the Lasso », in : *Annals of Statistics* **34** (2006), page(s): 1436–1462.

MEINSHAUSEN, Nicolai et Peter BÜHLMANN, « Stability selection », in : *Journal of the Royal Statistical Society : Series B (Statistical Methodology)* **72**.*4* (2010), page(s): 417–473.

MEINSHAUSEN, Nicolai et Bin YU, « Lasso-type recovery of sparse representations for high-dimensional data », in : *The Annals of Statistics* (2009), page(s): 246–270.

MICCHELLI, C.A. et M. PONTIL, « On learning vector-valued functions », in : *Neural Computation* **17**.*1* (2005), page(s): 177–204.

MICCHELLI, Charles A et Massimiliano PONTIL, « Kernels for Multi–task Learning », in : (2004), page(s): 921–928.

MICHAILIDIS, George et Florence D'ALCHÉ-BUC, « Autoregressive models for gene regulatory network inference : Sparsity, stability and causality issues », in : *Math. Biosci.* **246**.*2* (2013), page(s): 326–334.

MITCHELL, Tom M, « Machine learning. 1997 », in : *Burr Ridge, IL : McGraw Hill* **45** (1997).

MOREAU, Jean-Jacques, « Fonctions convexes duales et points proximaux dans un espace hilbertien », in : *Comptes-Rendus de l'Académie des Sciences de Paris, Série A, Mathématiques* **255** (1962), page(s): 2897–2899.

MORTON, R.B. et K. C. WILLIAMS, *Experimental Political Science and the Study of Causality*, Cambridge University Press, 2010.

MUKHERJEE, Sayan, Edgar OSUNA et Federico GIROSI, « Nonlinear prediction of chaotic time series using support vector machines », in : *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, IEEE, 1997, page(s): 511–520.

MÜLLER, K-R et al., « Predicting time series with support vector machines », in : *Artificial Neural Networks—ICANN'97*, Springer, 1997, page(s): 999–1004.

MURPHY, K. P., « Dynamic Bayesian Networks : representation, inference and learning », thèse de doct., Computer Science, University of Berkeley, CA, USA, 1998.

NADARAYA, Elizbar A, « On estimating regression », in : *Theory of Probability & Its Applications* **9**.*1* (1964), page(s): 141–142.

NATARAJAN, Balas Kausik, « Sparse approximate solutions to linear systems », in : *SIAM journal on computing* **24**.*2* (1995), page(s): 227–234.

NESTEROV, Yurii, « A method of solving a convex programming problem with convergence rate $O(1/k^2)$ », in : **27**.*2* (1983), page(s): 372–376.

— *Gradient methods for minimizing composite objective function*, rapp. tech., 2007.

NICHOLLS, Des F et Barry G QUINN, « Random coefficient autoregressive models : an introduction », in : *Lecture Notes in Statistics* **11** (1983).

OBOZINSKI, Guillaume, Ben TASKAR et Michael I JORDAN, « Joint covariate selection and joint subspace selection for multiple classification problems », in : *Statistics and Computing* **20**.*2* (2010), page(s): 231–252.

ORTEGA, James M et Werner C RHEINBOLDT, *Iterative solution of nonlinear equations in several variables*, t. 30, Siam, 2000.

PARRY, M. et al., *Climate change 2007 : impacts, adaptation and vulnerability*, Intergovernmental Panel on Climate Change, 2007.

PASCANU, Razvan et al., « How to Construct Deep Recurrent Neural Networks », in : *arXiv preprint arXiv :1312.6026* (2013).

PEARL, Judea, *Probabilistic reasoning in intelligent systems : networks of plausible inference*, sous la dir. de CA SAN FRANCISCO, Morgan Kaufmann Publishers, Inc., 1988.

PEDRICK, GB, *Theory of reproducing kernels in Hilbert spaces of vector-valued functions*, rapp. tech., University of Kansas Department of Mathematics, Lawrence, Kansas, 1957.

PERRIN, B.E. et al., « Gene networks inference using dynamic Bayesian networks », in : *Bioinformatics* **19**.*suppl 2* (2003), page(s): ii138.

POLITIS, D.N., J.P. ROMANO et M. WOLF, *Subsampling*, Springer New York, 1999.

PRILL, Robert J. et al., *Towards a Rigorous Assessment of Systems Biology Models : The DREAM3 Challenges*, t. 5, 2, Public Library of Science, fév. 2010, page(s): e9202, DOI : 10.1371/journal.pone.0009202, URL : http://dx.doi.org/10.1371/journal.pone.0009202.

RABINER, Lawrence, « A tutorial on hidden Markov models and selected applications in speech recognition », in : *Proceedings of the IEEE* **77**.*2* (1989), page(s): 257–286.

RAGUET, Hugo, Jalal FADILI et Gabriel PEYRÉ, « A generalized forward-backward splitting », in : *SIAM Journal on Imaging Sciences* **6**.*3* (2013), page(s): 1199–1226.

RALAIVOLA, Liva et Florence D'ALCHE-BUC, « Dynamical modeling with kernels for nonlinear time series prediction », in : *Advances in Neural Information Processing Systems*, t. 16, MIT Press, 2003, page(s): 129.

RAO, T Subba, « On the theory of bilinear time series models », in : *Journal of the Royal Statistical Society. Series B (Methodological)* (1981), page(s): 244–255.

RAO, T Subba et MM GABR, *An introduction to bispectral analysis and bilinear time series models*, Springer, 1984.

RASMUSSEN, Carl Edward, « Gaussian processes in machine learning », in : *Advanced Lectures on Machine Learning*, Springer, 2004, page(s): 63–71.

RICHARD, Emile, Pierre-Andre SAVALLE et Nicolas VAYATIS, « Estimation of Simultaneously Sparse and Low Rank Matrices », in : *ICML-2012*, sous la dir. de John LANGFORD et Joelle PINEAU, Edinburgh, Scotland, GB : Omnipress, juil. 2012, page(s): 1351–1358, ISBN : 978-1-4503-1285-1.

ROSENBLATT, Frank, « The perceptron : a probabilistic model for information storage and organization in the brain. », in : *Psychological review* **65**.*6* (1958), page(s): 386.

ROTH, Volker et Bernd FISCHER, « The group-lasso for generalized linear models : uniqueness of solutions and efficient algorithms », in : (2008), page(s): 848–855.

Rumelhart, David E, Geoffrey E Hinton et Ronald J Williams, « Learning representations by back-propagating errors », in : *Cognitive modeling* (1988).

Sapankevych, Nicholas I et Ravi Sankar, « Time series prediction using support vector machines : a survey », in : *Computational Intelligence Magazine, IEEE* **4**.*2* (2009), page(s): 24–38.

Schäfer, Juliane et Korbinian Strimmer, « A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics », in : *Statistical applications in genetics and molecular biology* **4**.*1* (2005).

Schaffter, Thomas, Daniel Marbach et Dario Floreano, « GeneNetWeaver : in silico benchmark generation and performance profiling of network inference methods », in : *Bioinformatics* **27**.*16* (2011), page(s): 2263–2270.

Schmidt, Mark W et Kevin P Murphy, « Convex structure learning in log-linear models : Beyond pairwise potentials », in : (2010), page(s): 709–716.

Schölkopf, B. et A.J. Smola, *Learning with kernels : Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, 2001.

Schölkopf, Bernhard, Ralf Herbrich et Alex J. Smola, « A Generalized Representer Theorem », in : *Computational Learning Theory*, sous la dir. de David Helmbold et Bob Williamson, t. 2111, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001, page(s): 416–426.

Schölkopf, Bernhard, Koji Tsuda et Jean-Philippe Vert, *Kernel methods in computational biology*, MIT press, 2004.

Schwartz, Laurent, « Sous-espaces hilbertiens d'espaces vectoriels topologiques et noyaux associés (noyaux reproduisants) », in : *Journal d'analyse mathématique* **13**.*1* (1964), page(s): 115–256.

Schwarz, Gideon, « Estimating the dimension of a model », in : *The annals of statistics* **6**.*2* (1978), page(s): 461–464.

Segal, Eran et al., « Module networks : identifying regulatory modules and their condition-specific regulators from gene expression data », in : *Nature genetics* **34**.*2* (2003), page(s): 166–176.

Senkene, É. et A. Tempel'man, « Hilbert spaces of operator-valued functions », in : *Lithuanian Mathematical Journal* **13**.*4* (1973), page(s): 665–670.

Shimamura, Teppei et al., « Recursive regularization for inferring gene networks from time-course gene expression profiles », in : *BMC Systems Biology* **3**.*1* (2009), page(s): 41.

Shojaie, A. et G. Michailidis, « Discovering Graphical Granger Causality Using a Truncating Lasso Penalty », in : *Bioinformatics* **26(18)** (2010), page(s): i517–i523.

Sims, Christopher A, « Macroeconomics and reality », in : *Econometrica : Journal of the Econometric Society* (1980), page(s): 1–48.

Sindhwani, Vikas, Ha Quang Minh et Aurélie C. Lozano, « Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and Granger causality », in : *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, 2013.

Smola, Alex J et Bernhard Schölkopf, « A tutorial on support vector regression », in : *Statistics and computing* **14**.*3* (2004), page(s): 199–222.

Someren, Eugene P van et al., « Least absolute regression network analysis of the murine osteoblast differentiation network », in : *Bioinformatics* **22**.*4* (2006), page(s): 477–484.

Song, Guohui, Haizhang Zhang et Fred J Hickernell, « Reproducing kernel Banach spaces with the ℓ_1 norm », in : *Applied and Computational Harmonic Analysis* **34**.*1* (2013), page(s): 96–116.

Steinwart, I., « On the influence of the kernel on the consistency of support vector machines », in : *The Journal of Machine Learning Research* **2** (2002), page(s): 67–93.

Stuart, Joshua M et al., « A gene-coexpression network for global discovery of conserved genetic modules », in : *science* **302**.*5643* (2003), page(s): 249–255.

Teräsvirta, Timo, Dick Van Dijk et Marcelo C Medeiros, « Linear models, smooth transition autoregressions, and neural networks for forecasting macroeconomic time series : A re-examination », in : *International Journal of Forecasting* **21**.*4* (2005), page(s): 755–774.

Tibshirani, Robert, « Regression shrinkage and selection via the lasso », in : *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), page(s): 267–288.

Tikhonov, Andrej Nikolaevich et Vasiliy Yakovlevich Arsenin, *Solutions of ill-posed problems*, Winston, 1977.

Todd, Michael J, « The many facets of linear programming », in : *Mathematical Programming* **91**.*3* (2002), page(s): 417–436.

Tong, Howell, *On a threshold model*, 29, Sijthoff & Noordhoff, 1978.

— *Threshold models in non-linear time series analysis. Lecture notes in statistics, No. 21*, Springer-Verlag, 1983.

Tropp, Joel A, « Greed is good : Algorithmic results for sparse approximation », in : *Information Theory, IEEE Transactions on* **50**.*10* (2004), page(s): 2231–2242.

Tseng, P., « On accelerated proximal gradient methods for convex-concave optimization », in : *SIAM Journal on Optimization* (2008).

Turlach, Berwin A, William N Venables et Stephen J Wright, « Simultaneous variable selection », in : *Technometrics* **47**.*3* (2005), page(s): 349–363.

VANDENBERGHE, L., *Fast proximal gradient methods*, 2010, URL : http://www.seas.ucla.edu/~vandenbe/236C/lectures/fgrad.pdf.

VAPNIK, Vladimir N, *The nature of statistical learning theory*, Springer-Verlag New York, Inc., 1995.

WAINWRIGHT, MJ, « Sharp thresholds for noisy and high-dimensional recovery of sparsity using ℓ_1-constrained quadratic programming (lasso) », in : *IEEE Transactions on Information Theory* **55**.*5* (2009), page(s): 2183–2202.

WATSON, Geoffrey S, « Smooth regression analysis », in : *Sankhy : The Indian Journal of Statistics, Series A* (1964), page(s): 359–372.

WHITLE, Peter, *Hypothesis testing in time series analysis*, t. 4, Almqvist & Wiksells, 1951.

WRIGHT, Stephen J, Robert D NOWAK et Mário AT FIGUEIREDO, « Sparse reconstruction by separable approximation », in : *Signal Processing, IEEE Transactions on* **57**.*7* (2009), page(s): 2479–2493.

WU, Tong Tong et Kenneth LANGE, « Coordinate descent algorithms for lasso penalized regression », in : *The Annals of Applied Statistics* (2008), page(s): 224–244.

YIP, K.Y. et al., « Improved reconstruction of in silico gene regulatory networks by integrating knockout and perturbation data », in : *PLoS One* **5(1) :e8121** (2010).

YUAN, Ming et Yi LIN, « Model selection and estimation in regression with grouped variables », in : *Journal of the Royal Statistical Society : Series B* **68**.*1* (2006), page(s): 49–67.

YULE, G Udny, « On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers », in : *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* (1927), page(s): 267–298.

ZHANG, Guoqiang, B EDDY PATUWO et Michael Y HU, « Forecasting with artificial neural networks : The state of the art », in : *International journal of forecasting* **14**.*1* (1998), page(s): 35–62.

ZHANG, Haizhang, Yuesheng XU et Jun ZHANG, « Reproducing kernel Banach spaces for machine learning », in : *The Journal of Machine Learning Research* **10** (2009), page(s): 2741–2775.

ZHANG, Haizhang, Yuesheng XU et Qinghui ZHANG, « Refinement of operator-valued reproducing kernels », in : *The Journal of Machine Learning Research* **13**.*1* (2012), page(s): 91–136.

ZHANG, Haizhang et Jun ZHANG, « Vector-valued reproducing kernel Banach spaces with applications to multi-task learning », in : *Journal of Complexity* **29**.*2* (2013), page(s): 195–215.

Zhao, Peng, Guilherme Rocha et Bin Yu, « The composite absolute penalties family for grouped and hierarchical variable selection », in : *The Annals of Statistics* (2009), page(s): 3468–3497.

Zhao, Peng et Bin Yu, « On model selection consistency of Lasso », in : *The Journal of Machine Learning Research* **7** (2006), page(s): 2541–2563.

Zou, Cunlu et Jianfeng Feng, « Granger causality vs. dynamic Bayesian network inference : a comparative study », in : *BMC Bioinformatics* **10**.*1* (2009), page(s): 122, issn : 1471-2105.

Zou, H. et T. Hastie, « Regularization and variable selection via the elastic net », in : *Journal of the Royal Statistical Society : Series B (Statistical Methodology)* **67**.*2* (2005), page(s): 301–320.

Zou, Hui, « The adaptive lasso and its oracle properties », in : *Journal of the American statistical association* **101**.*476* (2006), page(s): 1418–1429.