amU
Aix Marseille Université

NNT : 2025AIXM0078

THÈSE DE DOCTORAT

Soutenue à AMU — Aix-Marseille Université
le 31 mars 2025 par

Mimoun MOHAMED

Algorithmes exploratoires pour la frugalité par la parcimonie

Discipline
Informatique

École doctorale
ED 184 Mathématiques et Informatique

Laboratoires
Institut de Mathématiques de Marseille
Laboratoire d'Informatique et des Systèmes

Partenaire
Euranova

Composition du jury

Laure BLANC-FÉRAUD Directrice de recherche, CNRS	Rapporteure
Gilles GASSO Professeur, INSA Rouen Normandie	Rapporteur
Marianne CLAUSEL Professeure, Université de Lorraine	Examinatrice
Rémi GRIBONVAL Directeur de recherche, INRIA	Examinateur
Hachem KADRI Professeur, AMU	Président du jury
Valentin EMIYA Maître de conférence, AMU	Directeur de thèse
Caroline CHAUX Directrice de recherche, CNRS, AMU	Co-directrice de thèse

Invités

Vincent FRÉOUR
Chercheur, Yamaha Music Europe

François MALGOUYRES
Professeur, Université de Toulouse

INSTITUT de MATHÉMATIQUES de MARSEILLE

LIS LABORATOIRE D'INFORMATIQUE & DES SYSTÈMES

EURA NOVA

Affidavit

Je soussigné, Mimoun MOHAMED, déclare par la présente que le travail présenté dans ce manuscrit est mon propre travail, réalisé sous la direction scientifique de Valentin EMIYA et Caroline CHAUX, dans le respect des principes d'honnêteté, d'intégrité et de responsabilité inhérents à la mission de recherche. Les travaux de recherche et la rédaction de ce manuscrit ont été réalisés dans le respect à la fois de la charte nationale de déontologie des métiers de la recherche et de la charte AMU relative à la lutte contre le plagiat.

Ce travail n'a pas été précédemment soumis en France ou à l'étranger dans une version identique ou similaire à un organisme examinateur.

Fait à Marseille le 27 janvier 2025



Cette œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Pas de Modification 4.0 International.

Liste de publications et participation aux conférences

Liste des publications et/ou brevets réalisées dans le cadre du projet de thèse :

1. Vincent FRÉOUR, Mimoun MOHAMED, Keita ARIMOTO, Valentin EMIYA, Bruno COCHELIN et Christophe VERGEZ : Machine learning applied to the prediction of trumpet bifurcation diagrams : towards a tool for trumpet designers. In *Forum Acusticum - 10th Convention of the European Acoustics Association*, Septembre 2023, Turin, Italie.
2. Mimoun MOHAMED, François MALGOUYRES, Valentin EMIYA et Caroline CHAUX : Straight-Through meets Sparse Recovery : the Support Exploration Algorithm. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, Volume 235, Juillet 2024, Vienne, Autriche.
3. Mimoun MOHAMED, Vincent FRÉOUR, Christophe VERGEZ, Keita ARIMOTO, Valentin EMIYA et Bruno COCHELIN : Prediction of trumpet performance descriptors using machine learning. In *Acta Acustica*, Volume 8, Septembre 2024.
4. Mimoun MOHAMED, Valentin EMIYA et Caroline CHAUX : Learning Permutations in Monarch Factorization. In *ICASSP : IEEE International Conference on Acoustics, Speech and Signal Processing*, Avril 2025, Hyderabad, Inde.

Participation aux conférences et écoles d'été au cours de la période de thèse :

1. JRAF : Journées sur la Recherche en Apprentissage frugal, Grenoble, 24–25 novembre 2022
2. PhDays'2023 : Journée des doctorants du LIS, Marseille, 22 mars 2023 (présentation orale)
3. Workshop on "Imaging inverse problems - regularization, low dimensional models and applications", Bordeaux, 23 mars 2023 (présentation orale)
4. A Multiscale tour of Harmonic Analysis and Machine Learning, Bures–sur–Yvette, 19–21 avril 2023
5. Journée Signal-Image, Marseille, 19 juillet 2023 (présentation orale)

6. Colloque GRETSI 2023, XXIXème Colloque Francophone de Traitement du Signal et des Images, Grenoble, 28 août–1er septembre 2023
7. 10th Convention of the European Acoustics Association, Forum Acusticum 2023, Turin, Italie, 11–15 septembre 2023 (présentation orale)
8. 30 ans de mathématiques pour l'imagerie optique, Marseille, 25–27 septembre 2023 (présentation poster)
9. SynThèse, la rentrée des jeunes chercheurs du LIS, Marseille, 23 octobre 2023 (présentation orale)
10. IA et Sciences Physiques @AMU, Marseille, 22 novembre 2023
11. Workshop DIPOpt, Deep learning, image analysis, inverse problems, and optimization, Lyon 27–30 novembre 2023 (présentation poster)
12. Journées Apprentissage Signal Image du LIS, Aix-en-Provence, 17-18 juin 2024 (présentation orale)
13. CAp (Conférence sur l'Apprentissage automatique) et RFIAP (Reconnaissance des Formes, Image, Apprentissage et Perception), Lille, 1er–3 juillet 2024 (présentation orale et poster)
14. 41st International Conference on Machine Learning (ICML 2024), Vienne, Autriche, 6–11 juillet 2024 (présentation poster)
15. Mathematical foundations of machine learning workshop, Marseille, 16 juillet 2024 (présentation orale)
16. L0-based minimization : from continuous relaxations and non-convex algorithms to global optimization, Paris, 16 septembre 2024 (présentation orale)
17. SIGMA 2024 (Signal - Image - Geometry - Modelling - Approximation), Marseille, 28 octobre–2 novembre (présentation poster)
18. AI and Physical Sciences @AMU workshop, Marseille, 12 novembre 2024
19. Mathematics and Image Analysis (MIA) conference, Paris, 13–15 janvier 2025 (présentation poster)
20. ICASSP 2025 - IEEE International Conference on Acoustics, Speech and Signal Processing, Hyderabad, Inde, 6–11 avril 2025 (présentation à venir)

Résumé et mots clés

La parcimonie, qui se caractérise par un faible nombre de composantes non nulles, est une propriété essentielle dans des domaines tels que le traitement du signal, l'apprentissage automatique, les statistiques et la compression de données. En effet, elle est souvent utilisée pour réduire la complexité des modèles, améliorer la généralisation et faciliter l'interprétation des résultats. Cependant, elle est généralement difficile à obtenir en pratique, car il est complexe de déterminer quelles composantes doivent être nulles. Cela est dû à la non-convexité et non-différentiabilité des problèmes de parcimonie en norme ℓ_0, ce qui rend difficile la recherche de solutions exactes.

Pour contourner cette difficulté, il est courant d'utiliser une relaxation convexe de la norme ℓ_0 en norme ℓ_1 ou de faire appel à des algorithmes gloutons. La première approche permet d'obtenir un problème avec un unique minimum, permettant ainsi l'utilisation d'algorithmes ayant des garanties de convergence. Tandis que la seconde approche converge rapidement, mais vers des minima locaux.

Dans cette thèse, nous avons cherché à développer une méthode offrant des capacités d'exploration plus étendues que celles des méthodes précédemment évoquées, afin de sortir des minima locaux sans recourir à une relaxation. La méthode proposée repose sur l'utilisation de l'estimateur direct (de l'anglais «Straight-Through Estimator» ou «STE»). Elle présente deux avantages. Tout d'abord, elle permet de rendre différentiables les fonctions faisant appel à la norme ℓ_0, sans relaxation, en proposant une approximation du gradient des parties non différentiables. Cela rend possible l'utilisation de toutes les techniques de descente de gradient. Ensuite, elle permet d'explorer les minima locaux, améliorant ainsi la qualité des objets parcimonieux reconstruits.

Cette nouvelle stratégie d'exploration a été exploitée dans le cadre de la reconstruction de supports parcimonieux dans le cas linéaire (MOHAMED, MALGOUYRES et al. 2024), ainsi que sur une extension de cette approche dans le cadre de la factorisation de matrices en produit de matrices creuses (MOHAMED, EMIYA et al. 2025). Dans le premier contexte, nous avons construit un algorithme surpassant l'état de l'art pour le cas difficile des dictionnaires redondants, avec des garanties théoriques reposant sur la propriété d'isométrie restreinte du dictionnaire. Dans le second contexte, nous avons conçu un algorithme capable d'estimer un modèle plus expressif que celui des matrices de type «Monarch». Ce modèle peut être utilisé pour remplacer les matrices denses dans les réseaux de neurones, réduisant ainsi les complexités spatiale et temporelle. Cette propriété de parcimonie est finalement exploitée dans une dernière étude effectuée dans le cadre d'une mission de doctorant expert (FRÉOUR, MOHAMED et al. 2023 ; MOHAMED, FRÉOUR et al. 2024) pour la résolution d'un problème industriel (pour Yamaha).

Mots clés : estimateur direct, parcimonie, reconstruction de supports, apprentissage automatique, factorisation de matrices, trompette

Abstract and keywords

Sparsity, characterized by a small number of non-zero components, is an essential property in fields such as signal processing, machine learning, statistics, and data compression. Indeed, it is often used to reduce model complexity, improve generalization, and facilitate the interpretation of results. However, it is generally difficult to achieve in practice, as determining which components should be null is complex. This is due to the non-convexity and non-differentiability of sparse problems in ℓ_0, making it difficult to find exact solutions.

To overcome this challenge, it is common to use a convex relaxation of the ℓ_0-norm into the ℓ_1-norm or employ greedy algorithms. The first approach leads to a problem with a unique minimum, enabling the use of algorithms with convergence guarantees. In contrast, the second approach converges quickly but to local minima.

In this thesis, we sought to develop a method offering broader exploration capabilities than those of the previously mentioned methods, to escape local minima without relying on relaxation. The proposed method is based on the use of the Straight-Through Estimator (STE). It has two advantages. First, it allows for differentiability of functions involving the ℓ_0-norm, without relaxation, by approximating the gradient of the non-differentiable parts. This makes it possible to use all gradient descent techniques. Second, it enables the exploration of local minima, thereby improving the quality of the reconstructed sparse objects.

This new exploration strategy has been applied in the context of reconstructing sparse supports in the linear case (Mohamed, Malgouyres, et al. 2024), as well as an extension of this approach in the context of matrix factorization as a product of sparse matrices (Mohamed, Emiya, et al. 2025). In the first context, we developed an algorithm surpassing the state-of-the-art for the challenging case of redundant dictionaries, with theoretical guarantees based on the restricted isometry property (RIP) of the dictionary. In the second context, we designed an algorithm capable of estimating a more expressive model than that of "Monarch" matrices. This model can be used to replace dense matrices in neural networks, thus reducing spatial and temporal complexities. Finally, sparsity is exploited in a final study carried out within the framework of a french expert doctoral mission (Fréour, Mohamed, et al. 2023; Mohamed, Fréour, et al. 2024) to solve an industrial problem (for Yamaha).

Keywords: straight-through estimator, sparsity, support reconstruction, machine learning, matrix factorization, trumpet

Remerciements

Je tiens à remercier toutes les personnes qui m'ont permis de mener à bien ce travail de thèse, en commençant par la région Sud ainsi qu'Euranova pour leur soutien financier.

Merci à mes directeurs de thèse, Valentin EMIYA et Caroline CHAUX, pour leur temps, leur énergie, ainsi que pour leur confiance, leurs conseils et surtout leur patience. Grâce à Valentin, je pense être capable d'expliquer presque n'importe quoi à n'importe qui, et grâce à Caroline, le faire en gardant la rigueur nécessaire. Merci aussi à François MALGOUYRES pour notre collaboration qui nous a mené loin.

Je remercie tous les membres du jury pour avoir accepté d'évaluer mon travail de recherche ainsi que pour leurs retours constructifs, leurs intérêts pour mes travaux et les échanges que nous avons eus.

Je remercie également mes deux laboratoires, le LIS et l'I2M ainsi que leurs équipes administratives, pour leur soutien précieux et leur aide dans les démarches administratives. Je tiens particulièrement à remercier Sarah DARFEUILLE et Kaïna TIGHILT du côté de l'I2M, ainsi que Nelly SAMMUT au LIS. Nelly, merci encore d'avoir géré, anticipé et facilité toutes mes missions farfelues qui ont, pour certaines, eu jusqu'à cinq sources de financements différentes. Merci à Manuel BERTRAND et Sébastien BOUTELIER pour leur gestion de l'infrastructure du LIS, ainsi que Guillaume CHAGNARD et Olivier CHABROL à l'I2M. Merci aux doctorants pour tous les moments passés ensemble : Guillaume, Chiheb, Anatole, Adel, Yenni, Clément, Joffrey, Joscha, Julius, Roméo, Nathan, Raphaël, Félix, Ulysse, Elie, Eunice, Mikhael, Mira, François, Ioana, Jules, Marjorie, Nino, Valentin, Eliot et Alice.

Je remercie aussi l'école doctorale et sa direction. Merci à Sonia ASSEUM d'avoir facilité mes démarches et répondu à toutes mes questions administratives.

Je remercie l'IPAL et le CNRS@CREATE d'avoir accepté de m'accueillir trois mois dans leurs locaux à Singapour. Je n'oublierai jamais cette expérience qui m'a permis d'élargir mes horizons. Merci à tous les membres de CREATE qui m'ont permis de profiter pleinement de cette mobilité : Swee Yee, Dominique, Chamira, Nicolas, Izzati et Celia. Merci également à tous les jeunes chercheurs pour tous les moments passés ensemble : Maxime, Yanis, Marie, Mario, Kamalika, Marie Herminie, Abijhit, Beatriz, Emiel, Clément, Stiven, Pierre, Joséphin, Damien, Hydar et Perla.

Merci à tous les permanents de mes deux équipes (QARMA et Signal-Image) pour leur aide et leurs conseils qui m'ont été très précieux : Thierry, Cécile, Ronan, François-Xavier, Hachem, Thomas, Rafika, Stéphane, Paul, Sandrine, Clothilde, Bruno, Arthur, Kaï et Frédéric. Merci aussi à Dominique BENIELLI. pour ses conseils. Merci à tous les jeunes chercheurs de mes deux équipes pour leur soutien et les expériences partagées : Ahmed, Loris, Felipe, Swetali, Hanwei, Joachim, Constance, Raphaël, Charly,

Emmanuelle, Chandrasekar, Alexandre, Etienne, Ambre, Kais, Malek, Abilasha, Christian, Mathias, Lam, Thibault, Pierre et Philip. Merci aux stagiaires que j'ai encadrés : Mamadou, Lenna et Thibault.

Merci aux équipes enseignantes d'avoir allégé mes enseignements aux moments où j'en avais le plus besoin.

Merci à tous les membres de Terminus Jeux pour leur soutien.

Je dédie la fin de ses remerciements à ma famille et mes amis, qui ont su me soutenir comme toujours pendant ses années de doctorats.

À ma famille, à mes amis

Table des matières

Affidavit 2

Liste de publications et participation aux conférences 3

Résumé et mots clés 5

Abstract and keywords 7

Remerciements 8

Table des matières 10

Table des figures 13

Liste des tableaux 16

Liste des algorithmes 16

Introduction 17
Contexte de la thèse . . . 17
Contenu du manuscrit . . . 19

1 Généralités 22
1.1 Parcimonie . . . 22
1.2 Problèmes d'optimisation . . . 26
1.2.1 Le cas linéaire . . . 26
1.2.2 Le cas matriciel . . . 27
1.3 Méthodes de résolutions actuelles . . . 28
1.4 Limites . . . 30
1.5 Optimisation de structures non convexes et non différentiables en apprentissage automatique . . . 34

2 Une application de la parcimonie à l'acoustique musicale 39
2.1 Introduction . . . 40
2.2 Du modèle physique de la trompette et du musicien aux descripteurs de performance . . . 41
2.3 Analyse des descripteurs provenant de trompettes haut de gamme . . . 45
2.3.1 Classification des trompettes . . . 46
2.3.2 Interprétabilité des résultats de la classification . . . 47

2.4 Prédiction des descripteurs sur des trompettes virtuelles 49
2.4.1 Construction du jeu de données 49
2.4.2 Étude comparative d'algorithmes d'apprentissage automatique 52
2.5 Zoom sur le LASSO-LARS 54
2.5.1 Formulation du problème d'optimisation 54
2.5.2 Interprétabilité du modèle 55
2.6 Outil pour les concepteurs de trompettes 57
2.7 Conclusions 58

3 SEA : Algorithme exploratoire pour le modèle parcimonieux linéaire 60
3.1 Introduction 60
3.1.1 Problème de reconstruction de supports parcimonieux 61
3.1.2 Estimateur direct (STE) 61
3.1.3 Contribution 62
3.2 Reformulation du problème de reconstruction de supports parcimonieux 63
3.3 Application de l'estimateur direct : l'algorithme SEA 65
3.3.1 Principe de l'algorithme SEA 65
3.3.2 Comparaison à l'état de l'art 67
3.4 Complexité de l'exploration 70
3.4.1 Implémentation efficace de SEA 70
3.4.2 Mesure de la complexité de l'exploration 71
3.5 Mise en évidence de l'efficacité de l'exploration 73
3.6 Conclusions 74

4 Une preuve théorique du succès de SEA 77
4.1 Introduction 78
4.2 Analyse d'un oracle pour l'exploration 82
4.2.1 Principe de l'oracle et énoncé du théorème de reconstruction . 82
4.2.2 Préliminaires 84
4.2.3 Preuve du théorème 4.2.1 : reconstruction par l'oracle 85
4.3 Analyse de SEA dans le cas général et garanties de reconstructions . . . 86
4.3.1 Énoncé du théorème de reconstruction dans le cas général . . . 87
4.3.2 Préliminaires 88
4.3.3 Preuve du théorème 4.3.1 : reconstruction dans le cas général . 92
4.4 Corollaire : Le cas des matrices orthogonales 94
4.5 Analyse de SEA dans le cas RIP et garanties de reconstructions 95
4.5.1 Énoncé du théorème de reconstruction et de son corollaire . . . 96
4.5.2 Propriétés issues du RIP 98
4.5.3 Préliminaires 99
4.5.4 Preuve du théorème 4.5.3 : reconstruction dans le cas RIP 102
4.5.5 Preuve du corollaire 4.5.4 : le cas RIP non bruité 104
4.5.6 Discussion 104
4.6 Conclusions 107

5 Applications de SEA **108**
5.1 Introduction . . . 108
5.2 Déconvolution de signaux . . . 109
5.2.1 Contexte . . . 109
5.2.2 Étude d'un exemple . . . 111
5.2.3 Généralisation des résultats . . . 116
5.2.4 Étude d'autres configurations . . . 120
5.3 Apprentissage supervisé . . . 124
5.3.1 Contexte . . . 124
5.3.2 Tâches de régression . . . 125
5.3.3 Tâches de classification binaire par régression logistique . . . 127
5.4 Conclusions . . . 128

6 STEAM : Algorithme exploratoire pour l'estimation de permutation **130**
6.1 Introduction . . . 130
6.1.1 Factorisation et matrices de Monarque . . . 131
6.1.2 Permutations et algorithmes d'affectation . . . 132
6.2 Principe de l'algorithme STEAM . . . 133
6.3 Étude expérimentale de STEAM . . . 137
6.3.1 Factorisation de matrices . . . 137
6.3.2 Apprentissage supervisé . . . 138
6.4 Conclusion . . . 141

Conclusions et perspectives **142**

Bibliographie **146**

Table des figures

1.1 Microtubules observés au microscope et reconstructions par divers algorithmes . . . 23
1.2 Représentation temporelle et fréquentielle d'un Si♭4 joué à la trompette 24
1.3 Exemple de compression JPEG d'une photo de loutre . . . 25
1.4 Exemple de représentation matricielle d'une permutation de $\mathbb{R}^{10}$ . . . 25
1.5 Diagramme de phase théorique de la relaxation en norme ℓ_1 . . . 33
1.6 Couche d'un réseau de neurones à poids binaires . . . 35
1.7 Réseau de neurones à poids binaires . . . 35
1.8 Exemple d'application de l'estimateur direct (STE) pour différencier la fonction d'arrondi . . . 37

2.1 Schéma d'une trompette . . . 42
2.2 Diagramme de bifurcation de la trompette pour un Si♭4 . . . 44
2.3 Objectif du projet : prédire les descripteurs de performance à partir d'un modèle d'apprentissage automatique . . . 46
2.4 Matrices de confusion de la classification des instruments avec XGBoost 48
2.5 Gain de chaque caractéristique après entraînement de XGBoost . . . 48
2.6 Projections sur les axes de l'ACP possédant les gains les plus élevés lors de la classification avec XGBoost . . . 49
2.7 Amplitude normalisée et angle de phase des impédances des 200 instruments virtuels générés . . . 51
2.8 Zoom autour du quatrième et du cinquième pic des impédances des 200 instruments virtuels générés . . . 52
2.9 Paramètres des lèvres des 60 musiciens virtuels de référence générés . 53
2.10 Approximation affine par morceaux des diagrammes de bifurcation par LASSO-LARS . . . 56
2.11 Moyenne et écart-type de l'importance normalisée des 25 paramètres modaux les plus importants pour la prédiction du descripteur $P_{\min 1}$. 57
2.12 Diagrammes en boîte des descripteurs prédits pour les 60 musiciens virtuels de référence sur 3 instruments test . . . 58

3.1 Graphe de calcul de $\bar{F}(H_q(\mathscr{W}))$. . . . 62
3.2 Graphe de calcul de $F(H(\mathscr{X}))$. . . . 64
3.3 Représentation de $\ell_{2,\text{rel_loss}}(x^t)$ et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t)})$ en fonction des itérations sur un problème de déconvolution . . . 67
3.4 Moyenne du nombre de supports explorés par les algorithmes . . . 72
3.5 Diagramme de transition de phase . . . 75

3.6 Diagramme de transition de phase dans le cas non bruité 75

4.1 Résumé des résultats théoriques du chapitre 82
4.2 Représentation visuelle des principaux ensembles d'indices rencontrés dans l'article. 83

5.1 Exemple d'instance d'un signal x^*, k-parcimonieux, de taille $n = 500$ avec $k = 20$. 110
5.2 Exemple d'instance d'observation y de taille $n = 500$ résultant de la convolution de x^* par un filtre gaussien A. 110
5.3 Exemple d'instance d'observation bruitée y de taille $n = 500$ résultant de la convolution de x^* par un filtre gaussien A. 111
5.4 Représentation de l'instance de x^* et y de la figure 5.3 avec les solutions fournies par les algorithmes. La solution d'OMP (identique à celle de HTP_{OMP}, IHT_{OMP} et OMPR) diffère de celle d'ELS (identique à HTP_{ELS} et IHT_{ELS}) par 3 pics. Afin de ne pas surcharger la figure, ces différences sont regroupées sous un même symbole. Cela ne compromet pas l'interprétation des résultats. 112
5.5 Agrandissement de la région en pointillés de la figure 5.4 avec les résultats de tous les algorithmes analysés 113
5.6 Représentation d'une instance de x^* et y avec les solutions fournies par les algorithmes lorsque $k = 20$ et $n = 500$ dans le cas sans bruit. 113
5.7 Découpe de la région en pointillés de la figure 5.6 113
5.8 Représentation de $\ell_{2,\text{rel_loss}}(x^t)$ et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t)})$ pour chaque itération, pour l'expérience de la figure 5.3. 114
5.9 Représentation de $\ell_{2,\text{rel_loss}}(x^{t(s)})$ et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t(s))})$ pour chaque nouveau support exploré, pour l'expérience de la figure 5.3. 114
5.10 Représentation de $\ell_{2,\text{rel_loss}}(x^{t(s)})$ et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t(s))})$ pour chaque nouveau support exploré, pour l'expérience de la figure 5.6. 116
5.11 Distance moyenne entre S^* et les supports renvoyés par les algorithmes sur 200 problèmes en fonction de k 117
5.12 Moyenne de $\ell_{2,\text{rel_loss}}(x)$ pour les sorties des algorithmes sur les 200 problèmes en fonction de k 118
5.13 Moyenne de la distance de Wasserstein-1 pour les sorties des algorithmes sur les 200 problèmes en fonction de k 118
5.14 Moyenne du nombre de supports explorés par les algorithmes dans le cas bruité en fonction de k 119
5.15 Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et le support des solutions fournies par plusieurs algorithmes en fonction de k avec différents pas 121
5.16 Distance moyenne $\text{dist}_{\text{supp},k'}$ entre S^* et les supports renvoyés par plusieurs algorithmes en fonction de k 122
5.17 Distance moyenne $\text{dist}_{\text{supp,largest}}$ entre S^* et les supports renvoyés par plusieurs algorithmes en fonction de k 123

5.18 Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes en fonction de k lorsque le bruit est plus important 123
5.19 Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes en fonction de k lorsqu'il est approximé 125
5.20 Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes en fonction de k lorsque $\|x^*\|_\infty$ est plus élevé 125
5.21 Performances sur le jeu de données de régression `comp-activ-harder` ($m = 8191$ exemples, $n = 12$ caractéristiques). 126
5.22 Performances sur le jeu de données de régression `year` ($m = 463715$ exemples, $n = 90$ caractéristiques). 126
5.23 Performances sur le jeu de données de régression `cal_housing` ($m = 20639$ exemples, $n = 8$ caractéristiques). 127
5.24 Performances sur le jeu de données de régression `slice` ($m = 53500$ exemples, $n = 384$ caractéristiques). 127
5.25 Performances sur le jeu de données de classification `letter` ($m = 20000$ exemples, $n = 16$ caractéristiques). 128
5.26 Performances sur le jeu de données de classification `ijcnn1` ($m = 24995$ exemples, $n = 22$ caractéristiques). 128

6.1 Graphe de calcul de $F(H(\mathscr{X}_0, \mathscr{X}_2))$ 135
6.2 Erreur de Frobenius relative entre A et la factorisation courante en fonction de l'itération de STEAM quand $N = 49$ 137
6.3 Erreur de Frobenius relative pour diverses factorisations en fonction de la taille N de la matrice A 139
6.4 Graphe de calcul de la factorisation remplaçant la matrice dense de la couche cachée du Perceptron. 139
6.5 Valeurs prises par la fonction de coût sur le jeu de test de MNIST en fonction du temps pendant l'entraînement d'un Perceptron 140

Liste des tableaux

2.1 Définition des descripteurs de performance de jeu extraits du diagramme de bifurcation de la figure 2.2 45
2.2 MAPE et StdAPE en test dans la prédiction des valeurs des descripteurs 54

6.1 Précision sur le jeu de données de MNIST en utilisant les optimiseurs AdamW et Adadelta. 140

Liste des algorithmes

1 Support Exploration Algorithm (SEA) 66
2 SEA (copie de l'algorithme 1) 68
3 HTP (FOUCART 2011) 68
4 SEA (copie de l'algorithme 1) 69
5 IHT (BLUMENSATH et Mike E. DAVIES 2009) 69
6 Support Exploration Algorithm (SEA) : implémentation efficace 71

7 SEA utilisant la règle de mise à jour de l'oracle 83

8 STEAM 136

Introduction

Contexte de la thèse

Aujourd'hui, l'apprentissage profond est devenu un outil incontournable dans de nombreux domaines, tels que la reconnaissance d'images, la traduction automatique, la recommandation de produits, la détection de fraudes, la conduite autonome, etc. Il est de plus en plus utilisé dans l'ensemble des domaines scientifiques et techniques. De plus, il fait partie des technologies clés des quatrième et cinquième révolutions industrielles, visant à automatiser, optimiser et personnaliser les processus de production et de gestion. Enfin, depuis que des interfaces permettant de manipuler des modèles génératifs ont été mises à disposition du grand public, l'apprentissage profond est devenu un outil très populaire pour la génération et la modification de contenu multimédia (images, vidéos, sons, textes).

Cependant, cet intérêt croissant vers l'apprentissage profond n'est pas sans conséquences. Justement, dans la lignée de cet engouement, les chercheurs et surtout les entreprises ont tendance à utiliser des modèles toujours plus volumineux, complexes et coûteux en ressources (LUCCIONI et al. 2024). Cette approche a un coût humain, matériel, énergétique et environnemental considérable qui ne cesse de croître (VAROQUAUX et al. 2024). Les modèles les plus performants nécessitent des millions, voire des milliards de paramètres à estimer, ainsi que des semaines, voire des mois de calculs sur des centaines de GPU (processeurs graphiques). Parallèlement à cette tendance d'augmentation de la taille des modèles, beaucoup d'efforts sont également consacrés à réduire leur taille (c'est-à-dire le nombre de paramètres) pendant et après leur entraînement. L'objectif est d'obtenir des modèles frugaux, c'est-à-dire des modèles utilisant un minimum de ressources pour résoudre les problèmes posés (BERTHELOT et al. 2024).

Cette recherche de frugalité est motivée par la sur-paramétrisation des modèles d'apprentissage profond. En effet, il a été démontré que les réseaux de neurones profonds peuvent être compressés sans perte significative de performance, en supprimant une grande partie de leurs paramètres (MALACH et al. 2020). Cette frugalité est atteignable de plusieurs manières. Par l'entraînement d'un modèle plus petit à partir du plus grand, via des méthodes de distillation (G. HINTON 2015) ou de factorisation de rang faible (HU et al. 2021 ; AUDIBERT et al. 2023). Toutefois, cette méthode n'est applicable qu'après l'entrainement du modèle. Une autre approche consiste à quantifier les paramètres du modèle, c'est-à-dire à les représenter avec moins de bits. Cette technique peut être mise en œuvre pendant l'entraînement, mais nécessite de modifier les algorithmes d'optimisation en utilisant l'estimateur direct (STE) (COURBARIAUX

et al. 2015).

Cette frugalité est également atteignable en utilisant des modèles parcimonieux, autrement dit des modèles avec un faible nombre de paramètres non nuls. Une méthode simple pour atteindre la parcimonie a posteriori consiste à éliminer les paramètres les plus petits, c'est ce qu'on appelle l'élagage (pruning) (HAN et al. 2015). Cependant, cette méthode est coûteuse en temps de calcul et en ressources, car elle nécessite de ré-entrainer le modèle après chaque suppression de paramètres. Une autre méthode consiste à factoriser les matrices de paramètres des réseaux de neurones en produits de matrices creuses, autrement dit des matrices avec un faible nombre de valeurs non nulles (DAO, GU et al. 2019 ; DAO, SOHONI et al. 2020 ; DAO, B. CHEN et al. 2022 ; QIU et al. 2024).

Les modèles parcimonieux sont particulièrement intéressants pour deux raisons. Tout d'abord, ils sont plus faciles à interpréter et à stocker, car ils sont plus simples et contiennent moins de paramètres. Ensuite, ils sont plus efficaces, car les calculs sur des objets parcimonieux sont généralement plus rapides que les calculs sur des objets non parcimonieux. Pour ces raisons, la parcimonie est une propriété souvent recherchée. Historiquement, elle provient du domaine du traitement du signal, notamment dans des problèmes linéaires et matriciels. Par exemple, elle est utilisée en acquisition comprimée (D. L. DONOHO 2006 ; CANDÈS et al. 2006) pour reconstruire des signaux à partir d'un nombre d'échantillons bien inférieur à celui prédit par le théorème de Shannon-Nyquist. La parcimonie est également appliquée à la factorisation de matrices en produits de matrices creuses, afin de réduire la complexité spatiale et temporelle des calculs matriciels (Y. LI et al. 2015 ; DAO, B. CHEN et al. 2022). Depuis, elle est devenue une propriété centrale en traitement du signal, en apprentissage automatique, en statistiques et en compression de données.

Cependant, la parcimonie est généralement difficile à obtenir en pratique, car il est complexe de déterminer quels paramètres doivent être nuls ou non. En effet, la recherche de la position des zéros (ou, de manière équivalente, des non-zéros) dans un objet mathématique, appelée également recherche du support, est un problème combinatoire NP-difficile (DAVIS et al. 1997). Par conséquent, son obtention engendre souvent des calculs coûteux en temps et en ressources, ce qui va à l'encontre de la recherche de frugalité. En effet, la plupart des techniques d'optimisation modernes reposent sur des algorithmes de descente de gradient. Or, ces derniers ne peuvent pas être directement appliqués aux problèmes de parcimonie, car ces problèmes sont non convexes et non différentiables. Face à cette difficulté, il est courant de recourir à une relaxation convexe de la norme ℓ_0 en norme ℓ_1, ou à des algorithmes gloutons qui convergent rapidement. Les premières méthodes n'aboutissent pas toujours à des solutions exactes, tandis que les secondes convergent rapidement, mais risquent de rester piégées dans des minima locaux.

Durant cette thèse, dans une démarche de recherche de frugalité par la parcimonie, nous avons développé une méthode offrant des capacités d'exploration plus étendues que celles des méthodes précédemment évoquées, le but étant de sortir des minima locaux sans recourir à une relaxation. Ces capacités d'exploration découlent de la stratégie du « tout différentiable » largement utilisée aujourd'hui en

apprentissage automatique. En s'inspirant des travaux réalisés en quantification, nous avons utilisé l'estimateur direct (G. E. HINTON 2012 ; BENGIO et al. 2013) (de l'anglais «Straight-Through Estimator» ou «STE») pour appliquer des techniques de descente de gradient à des fonctions non différentiables. Nous avons exploité cette nouvelle stratégie d'exploration dans le cadre de la reconstruction de supports parcimonieux dans le cas linéaire et dans le cadre de la factorisation de matrices en produit de matrices creuses. Ainsi, nous avons exploité une approche issue de l'apprentissage profond pour résoudre des problèmes de parcimonie en traitement du signal.

Enfin, cette thèse a aussi été l'occasion de collaborer avec Yamaha et le Laboratoire de Mécanique et d'Acoustique de Marseille (LMA), dans le cadre d'une mission de doctorant expert. Nous avons montré que les méthodes parcimonieuses peuvent être appliquées à des problèmes industriels, en l'occurrence le développement d'un outil d'aide à la conception de trompettes.

Contenu du manuscrit

Ce manuscrit s'articule autour de six chapitres. Il commence par cette introduction. Le chapitre 1 introduit les éléments de contexte et les problèmes abordés dans cette thèse. Le chapitre 2 décrit une application de la parcimonie pour la conception de trompettes, basée sur les travaux de (FRÉOUR, MOHAMED et al. 2023 ; MOHAMED, FRÉOUR et al. 2024). Les chapitres 3 à 5 présentent un nouvel algorithme d'exploration de support, son cadre théorique ainsi qu'une validation expérimentale, basés sur les travaux de (MOHAMED, MALGOUYRES et al. 2024). Le chapitre 6 présente un algorithme pour la recherche de permutations pour la factorisation de matrices en produit de matrices creuses, basé les travaux de (MOHAMED, EMIYA et al. 2025). Le manuscrit se termine par les conclusions et perspectives générales.

Chapitre 1 : Généralités

Dans le chapitre 1, nous introduisons la notion de parcimonie et soulignons son importance en traitement du signal, en apprentissage automatique, en statistiques et en compression de données. Nous formulons ensuite les problèmes d'optimisation abordés dans cette thèse, à savoir la reconstruction de supports parcimonieux dans le cas linéaire et la factorisation de matrices en produit de matrices creuses. Nous passons en revue les méthodes de résolution actuelles, en mettant en évidence leurs limites. Enfin, nous traçons un parallèle entre la parcimonie et l'apprentissage profond, en montrant que les difficultés rencontrées en parcimonie sont similaires à celles rencontrées en apprentissage profond. Nous concluons ce chapitre en présentant la technique de l'estimateur direct (STE), employée en apprentissage profond pour faire face à ces difficultés.

Chapitre 2 : Une application de la parcimonie à l'acoustique musicale

Dans le chapitre 2, nous commençons par présenter le contexte et les enjeux de la modélisation des instruments à vent en acoustique musicale. Après avoir décrit

le modèle physique de la trompette et du musicien, nous détaillons les descripteurs de performance employés pour caractériser le comportement d'une trompette lorsqu'elle est jouée par un musicien. Nous analysons ensuite ces descripteurs pour des trompettes haut de gamme et montrons qu'ils peuvent être utilisés pour classer les trompettes. Enfin, nous introduisons un outil d'aide à la conception de trompettes, fondé sur une méthode parcimonieuse, qui permet de prédire les descripteurs de performance de prototypes virtuels de trompettes jouées par des musiciens de référence. Cet outil est utilisé par les ingénieurs de Yamaha pour concevoir rapidement des prototypes de trompette, sans nécessiter d'expertise en dynamique non linéaire ni de simulations coûteuses.

Chapitre 3 : SEA : Algorithme exploratoire pour le modèle parcimonieux linéaire

Dans le chapitre 3, nous présentons l'algorithme SEA, conçu pour résoudre le problème de reconstruction de supports parcimonieux dans le cas linéaire. Nous rappelons tout d'abord la formulation classique de ce problème, qui est un problème d'optimisation avec une contrainte de parcimonie, ainsi que la méthode de l'estimateur direct. Nous montrons ensuite une reformulation de ce problème, sous la forme d'un problème d'optimisation sans contrainte, pour lequel il est possible d'utiliser des algorithmes de descente de gradient par le biais de l'estimateur direct (STE). Cette stratégie nous permet de proposer un algorithme d'exploration des minima locaux, SEA (pour « Support Exploration Algorithm »), qui se distingue par l'utilisation d'une variable d'exploration de supports. Cet algorithme est enfin comparé à l'état de l'art en termes de pseudo-code, de capacité d'exploration, de complexité de calcul et de qualité de reconstruction de supports sur des problèmes synthétiques.

Chapitre 4 : Une preuve théorique du succès de SEA

Dans le chapitre 4, nous proposons une preuve théorique du succès de SEA pour la reconstruction de supports parcimonieux dans le cas linéaire. Nous commençons par introduire un algorithme oracle pour l'exploration, capable de résoudre de manière exacte et systématique le problème de reconstruction de support parcimonieux. Ensuite, en nous appuyant sur une borne de la distance entre les mises à jour de l'algorithme oracle et de SEA, nous montrons que SEA reconstruit le support exact du signal parcimonieux, sous certaines conditions sur le dictionnaire et le signal. Enfin, nous spécifions ces résultats à des dictionnaires orthogonaux et à des dictionnaires vérifiant la propriété RIP (Restricted Isometry Property) dans le cas bruité.

Chapitre 5 : Applications de SEA

Dans le chapitre 5, nous présentons les applications de SEA à la reconstruction de supports parcimonieux dans le cas linéaire dans le contexte de la déconvolution de signaux et de l'apprentissage supervisé. Dans le contexte de la déconvolution, nous étudions d'abord extensivement le comportement de SEA et des algorithmes de l'état de l'art sur une instance du problème. Ensuite, nous révélons, par des expériences

intensives et l'étude de plusieurs mesures de performances sous de nombreuses configurations, que SEA surpasse l'état de l'art dans ce domaine, particulièrement dans des scénarios complexes où le dictionnaire est redondant. Dans le contexte de l'apprentissage supervisé, nous montrons que SEA est efficace pour résoudre des problèmes de régression linéaire et logistique avec des contraintes de budget sur les caractéristiques, en utilisant des jeux de données publiques.

Chapitre 6 : STEAM : Algorithme exploratoire pour l'estimation de permutation

Dans le chapitre 6, nous présentons l'algorithme STEAM, qui permet de résoudre le problème de reconstruction de supports parcimonieux dans le cas d'estimation de permutations pour la factorisation de matrices en produit de matrices creuses. Dans ce chapitre, nous étendons le modèle de factorisation de « Monarque », un modèle composé de matrices de permutations fixes et de matrices bloc-diagonales variables. Nous proposons un modèle plus expressif, intégrant des matrices de permutations variables. En adaptant la stratégie utilisée pour concevoir SEA à la recherche de matrices de permutation (un problème d'estimation de support), nous proposons STEAM (pour « STE Alternate Minimization »), un algorithme d'exploration utilisant l'estimateur direct et une descente de gradient alternée. Nous comparons notre modèle et STEAM à l'état de l'art dans deux contextes : la factorisation de matrices denses et une tâche d'apprentissage supervisé où les matrices denses d'un réseau de neurones sont remplacées par des factorisations en produits de matrices creuses. Dans ces deux contextes, nous montrons que notre modèle et STEAM permettent d'améliorer l'approximation des matrices denses ainsi que les capacités d'apprentissage des réseaux de neurones, tout en conservant une complexité spatiale et temporelle réduite.

1 Généralités

Sommaire

1.1 Parcimonie 22
1.2 Problèmes d'optimisation 26
1.2.1 Le cas linéaire 26
1.2.2 Le cas matriciel 27
1.3 Méthodes de résolutions actuelles 28
1.4 Limites 30
1.5 Optimisation de structures non convexes et non différentiables en apprentissage automatique 34

1.1 Parcimonie

La parcimonie, qui se caractérise par un faible nombre de composantes non nulles (c'est-à-dire qu'une grande partie des éléments d'un vecteur ou d'une matrice sont égaux à zéro), est une propriété essentielle dans des domaines tels que le traitement du signal, l'apprentissage automatique, les statistiques et la compression de données. Elle peut être naturellement présente dans les données de certaines disciplines telles que l'astronomie ou la biologie. Cependant, les appareils de mesures actuels ne permettent pas toujours de capturer cette propriété. C'est pour cela qu'il est parfois nécessaire de traiter les données pour retrouver leur parcimonie. La figure 1.1 montre un exemple d'image parcimonieuse prise par un microscope à fluorescence. Sur cette figure, on voit qu'il est nécessaire de traiter l'image, à l'aide d'algorithmes favorisants la parcimonie, pour retrouver la structure parcimonieuse des microtubules perdue lors de la prise de vue.

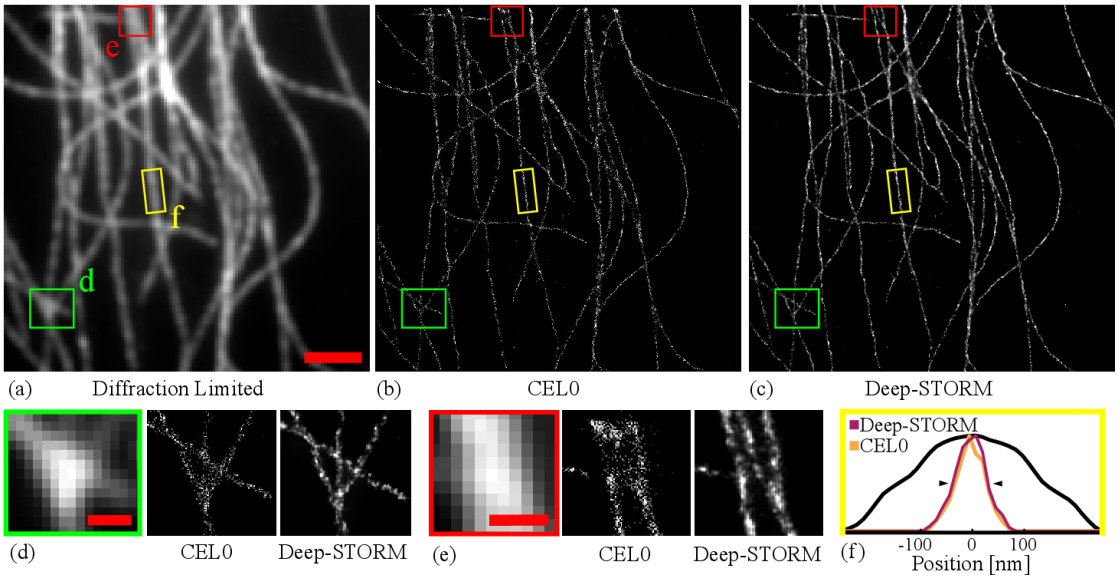

FIGURE 1.1 – Microtubules (structures filamenteuses du cytosquelette de certaines cellules vivantes) observés au microscope. Leur dynamique est étudiée en biologie, car toute altération peut conduire à des pathologies telles que le cancer (SCHUKKEN et al. 2020) ou des maladies neurodégénératives (KOUNAKIS et al. 2019). (a) Image retournée par le microscope. La barre d'échelle est de 2 μm. (b) Reconstruction par la méthode CEL0 (GAZAGNES et al. 2017). (c) Reconstruction par la méthode Deep-STORM (NEHME et al. 2018). (d), (e) Vues agrandies de deux régions sélectionnées. Les barres d'échelle sont de 0,5 μm. (f) Projection en largeur de la région dans le rectangle jaune. On y voit la largeur de la région mesurée. La ligne noire montre la largeur de la diffraction liée au microscope. Image reprise de (NEHME et al. 2018).

Cependant, la parcimonie est rarement naturellement présente dans les signaux du quotidien. Pour la retrouver, il est nécessaire de trouver une transformation qui permette de passer d'une représentation dense à une représentation parcimonieuse. Ce problème est résolu dans le cas des signaux sonores par la transformée de Fourier, qui permet de passer d'une représentation temporelle dense à une représentation fréquentielle parcimonieuse. La figure 1.2 montre un exemple de représentation temporelle et fréquentielle d'un son joué à la trompette.

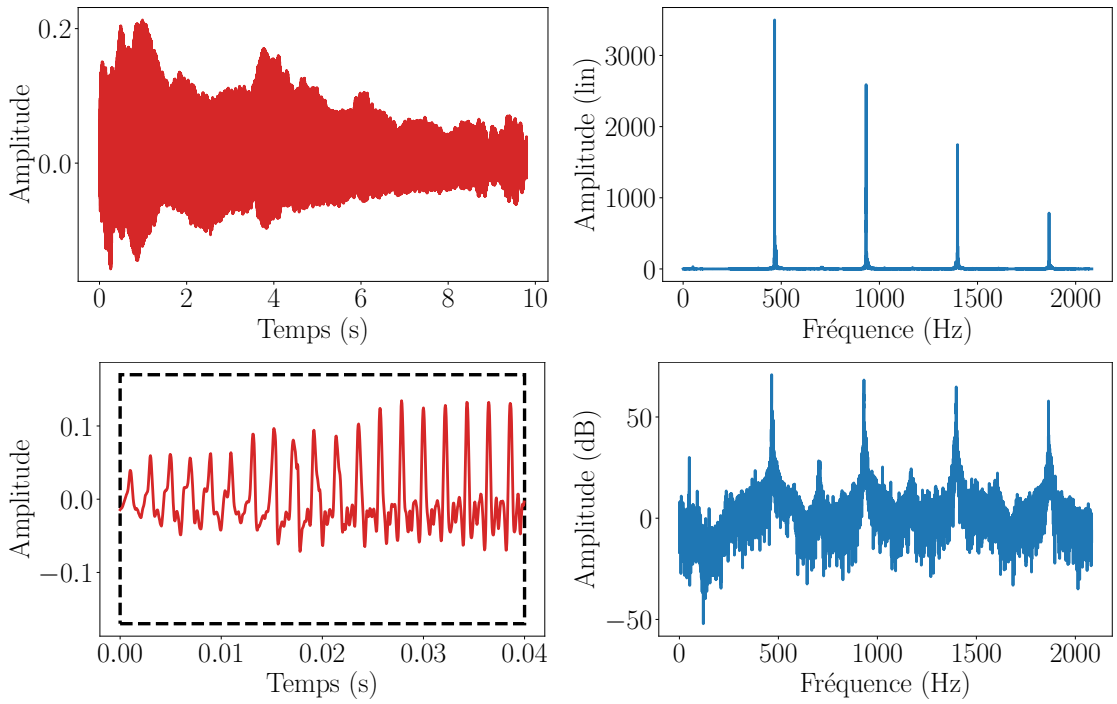

FIGURE 1.2 – Si♭4 joué à la trompette. Gauche : Représentation temporelle (dense) de 10 secondes du son (haut) et d'un zoom de 0.04 seconde correspondant à la région en pointillés (bas). Droite : Représentation fréquentielle (parcimonieuse) du son sur $[0, 2\text{kHz}]$ de manière linéaire (haut) et en décibels (bas). Le son est librement disponible à l'adresse : https://pixabay.com/sound-effects/trumpet-32070/.

Le problème est également résolu dans le cas des images naturelles (images prises par un téléphone portable ou un appareil photographique par exemple) par la transformée en cosinus discrète ou en ondelettes, qui permet de passer d'une représentation pixel par pixel dense à une représentation parcimonieuse. Cela permet de compresser les images avec peu de perte de qualité visuelle. La figure 1.3 montre un exemple de compression JPEG d'une photo de loutre.



FIGURE 1.3 – Exemple de compression JPEG d'une photo de loutre prise à une résolution de 1084 × 625 pixels. L'image de gauche est au format PNG, elle occupe 1.2MB de mémoire. L'image de droite est au format JPG, elle occupe 45kB de mémoire (soit plus de 96% de moins).
Crédit photo : Royal Society of Biology

Cette propriété existe non seulement pour les signaux, mais aussi pour les transformations. La figure 1.4 montre un exemple avec une permutation (transformation consistant à échanger de place les coefficients d'un vecteur). La représentation matricielle dans la base canonique d'une matrice de permutation est parcimonieuse. On dit également qu'une matrice de permutation est creuse.

$$\begin{bmatrix} 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \end{bmatrix}$$

FIGURE 1.4 – Exemple de représentation matricielle d'une permutation de $\mathbb{R}^{10}$. Une matrice de permutation est parcimonieuse, elle comporte un coefficient non nul par ligne et par colonne.

Aujourd'hui, il existe beaucoup de transformations permettant de passer d'une représentation dense à une représentation parcimonieuse. Cependant, elles sont souvent spécifiques à une catégorie de signaux. Ainsi, même si dans le cas général la parcimonie n'est pas garantie, elle reste très recherchée. En effet, en apprentissage automatique comme en apprentissage statistique, on cherche généralement à apprendre des estimateurs parcimonieux pour obtenir un modèle simple, peu coûteux et facile à interpréter. En apprentissage automatique, les problèmes de ce type

les plus courants sont appelés «régression parcimonieuse», «sélection de variable» ou «sélection d'ensemble» (BERTSIMAS et al. 2016). En apprentissage statistique, on peut citer le problème du «sélecteur discret de Dantzig» (MAZUMDER et al. 2017). En traitement du signal, on pourrait citer l'ensemble du domaine de l'acquisition comprimée (D. L. DONOHO 2006 ; CANDÈS et al. 2006). Cependant, nous allons principalement nous intéresser au problème de reconstruction de supports parcimonieux dans le cas linéaire et dans le cadre de la factorisation de matrices en produit de matrices creuses. En termes d'applications, le cas linéaire est utilisé pour la compression de signaux, la transmission d'information, les systèmes de communication et les systèmes de détection et de reconnaissance. Même si ces applications sont essentiellement connues dans le domaine de la santé et en astronomie, elles sont utilisées dans tous les domaines utilisant des signaux. La revue de la littérature (Andreas M TILLMANN et al. 2024) mentionne les principaux enjeux qui motivent ces applications. En ce qui concerne la factorisation de matrices en produit de matrices creuses, la principale application réside dans la réduction du coût de stockage et d'utilisation des matrices dans les algorithmes, notamment ceux d'apprentissage automatique. En effet, le coût d'utilisation et de stockage des réseaux de neurones développés de nos jours est directement lié au coût du produit matrice/vecteur et à la représentation des matrices en mémoire (C. YUAN et al. 2023).

1.2 Problèmes d'optimisation

Nous allons maintenant introduire formellement les problèmes qui nous intéressent.

1.2.1 Le cas linéaire

Tout d'abord, intéressons-nous au cas linéaire.

Définition 1.2.1. (*Support*) Soit x un vecteur de $\mathbb{R}^n$. On appelle support de x la quantité suivante :

$$\operatorname{supp}(x) \coloneqq \{i \in [\![1, n]\!] \mid x_i \neq 0\}$$

Il s'agit de l'ensemble des indices (ou positions) des coefficients non nuls de x.

Définition 1.2.2. («*Norme*» ℓ_0) Soit x un vecteur de $\mathbb{R}^n$. La «norme» ℓ_0 de x est définie comme

$$\begin{aligned} &\mathbb{R}^n \to \mathbb{N} \\ &x \quad \mapsto \|x\|_0 \coloneqq |\operatorname{supp}(x)| = \operatorname{Card}(\operatorname{supp}(x)). \end{aligned}$$

Il s'agit du nombre de coefficients non-nuls de x (ou de la taille de son support). On dit également que x est k-parcimonieux si $\|x\|_0 = k$.

Remarque. (*Norme ℓ_0*) L'opérateur $\|\cdot\|_0$ n'est pas une norme, car elle n'est pas absolument homogène ($\|\lambda x\|_0 = \|x\| \neq |\lambda|\|x\|_0,\ \lambda \in \mathbb{R}^*,\ x \neq 0$). Il est malgré tout courant d'utiliser la notation des normes ℓ_p en appelant et représentant cet opérateur en tant que norme ℓ_0.

Prenons n et m deux entiers positifs et un signal dense y de $\mathbb{R}^m$ que l'on appellera une observation. Dans de nombreux contextes on suppose que cette observation y (ex : représentation temporelle d'une note à la trompette) est la résultante d'un signal parcimonieux $x^* \in \mathbb{R}^n$ (ex : représentation fréquentielle d'une note à la trompette) de parcimonie $k \in \mathbb{N}$ qui aurait été transformé par un opérateur $A \in \mathbb{R}^{m\times n}$ (ex : transformée de Fourrier) et contaminé par un bruit additif $e \in \mathbb{R}^m$ lié à la mesure. Si l'on veut retrouver ce signal x^* à partir de y, il faut résoudre le problème inverse correspondant au modèle direct :

$$y = Ax^* + e \quad \text{avec } \|x^*\|_0 \leq k \text{ et } e \text{ inconnu.}$$

Afin de trouver un $\tilde{x}$ qui approche au mieux x^* au sens de l'erreur quadratique, il faut résoudre le problème d'optimisation suivant :

$$\tilde{x} \in \underset{x\in\mathbb{R}^n,\ \|x\|_0\leq k}{\operatorname{argmin}} \frac{1}{2}\|Ax - y\|_2^2. \tag{SPARSE}$$

Aujourd'hui, il existe de nombreuses méthodes numériques permettant de résoudre des problèmes de régression (non contraints) du type $\min_{x\in\mathbb{R}^n} \|Ax - y\|_2^2$. De plus dans des cas précis comme celui des sons ou celui des images naturelles, un opérateur A permettant de passer de la forme dense à la forme parcimonieuse et inversement est connu. En audio, il s'agit de la transformée de Fourier ; en image, il s'agit de la transformée en cosinus discrets et de la transformée en ondelettes. Cependant, dans le cas général, la contrainte en norme ℓ_0 rend le problème non convexe, non continu et non différentiable. Ainsi, la principale difficulté de ce problème n'est pas d'estimer avec précision les coefficients de x^* ; mais plutôt d'estimer son support, c'est-à-dire la position de ses coefficients non nuls. En effet, la combinatoire des $\binom{n}{k}$ supports possibles rend le problème d'optimisation (SPARSE) NP-difficile (Davis et al. 1997).

1.2.2 Le cas matriciel

Intéressons-nous maintenant à une matrice $A \in \mathbb{R}^{m\times n}$. Nous voulons l'approximer avec un produit de p matrices parcimonieuses $(A_i)_{i\in[\![1,p]\!]}$, chacune respectant une contrainte de parcimonie $(C_i)_{i\in[\![1,p]\!]}$ (Exemples : Avoir k coefficients non-nuls par ligne et/ou par colonne, avoir moins de k coefficients non-nuls...). Pour cela, il faut

résoudre le problème d'optimisation suivant (Le Magoarou et al. 2016) :

$$(\tilde{\lambda}, \tilde{A}_1, \dots, \tilde{A}_p) \in \underset{\substack{\lambda, A_1, \dots, A_p \\ \lambda \in \mathbb{R},\ \forall i \in [\![1,p]\!] \\ \mathrm{supp}(A_i) \in C_i, \|A_i\|_F^2 = 1}}{\operatorname{argmin}} \|\lambda \prod_{i=1}^{p} A_i - A\|_F^2 \tag{1.1}$$

Ici, on se heurte à trois difficultés. Tout d'abord, la norme d'un produit de matrices n'est pas convexe. Néanmoins, elle est séparément convexe (convexe en chacune des matrices). Cette propriété, favorisant les approches de minimisation alternée (Bolte et al. 2014), ne permet pas de garantir l'obtention d'un minimum global dans le cas général. Ensuite, pour chacun des facteurs A_i dont la contrainte C_i n'est pas une contrainte de support fixe[1], mais une contrainte d'appartenance à un ensemble de supports (ex : $\mathrm{Card}(\mathrm{supp}(A_i)) = k$ avec $k \in [\![1, p^2]\!]$), on retrouve la difficulté de l'estimation du support qui est NP-difficile, comme expliqué précédemment. Enfin, même si l'on se restreint à des problèmes à deux facteurs utilisant uniquement des contraintes de support fixe, non seulement ces problèmes sont déjà NP-difficiles, mais on ne peut pas non plus garantir l'existence d'un minimum global (Le, Riccietti et al. 2023).

1.3 Méthodes de résolutions actuelles

Les algorithmes d'optimisation les plus utilisés aujourd'hui sont des algorithmes dits de descente, le plus connu étant celui de descente de gradient. Il suffit que la fonction à optimiser soit différentiable afin que la méthode garantisse une convergence locale et que la fonction soit convexe pour garantir une convergence globale. Cependant, la norme ℓ_0 rend les problèmes d'optimisation non convexes et non différentiables. Ainsi, les techniques d'optimisation modernes consistent à relaxer la norme ℓ_0 en norme ℓ_1 pour rendre le problème convexe et différentiable partout, sauf sur un nombre restreint de points. Ce problème, connu sous le nom de LASSO (Least Absolute Shrinkage and Selection Operator) (Tibshirani 1996) ou BP (Basis Pursuit) (S. S. Chen et al. 2001) donne la formulation régularisée suivante :

$$\tilde{x}_{\ell_1} \in \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2}\|Ax - y\|_2^2 + \lambda \|x\|_1, \quad \lambda \in \mathbb{R}^+.$$

Dans cette équation λ est un hyperparamètre qui permet de contrôler le niveau de parcimonie de la solution. Plus il est élevé, plus la solution est parcimonieuse. Cette formulation permet d'utiliser des algorithmes tels que le LARS (Least Angle Regression) (Efron et al. 2004), qui vont permettre de construire la solution de manière incrémentale, ou de descente proximaux tels que FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) (Beck et al. 2009), qui vont permettre de construire la solution de manière itérative par pas de gradient et de seuillage. Il est également possible

1. C'est-à-dire que le support est connu à l'avance.

de remplacer la pénalisation ℓ_1 par une pénalisation non convexe et d'utiliser la programmation par Différence de Convexe (DC), comme proposé dans (GASSO et al. 2009 ; GOTOH et al. 2018).

Une deuxième méthode consiste à utiliser des approches gloutonnes pour rapidement trouver une estimation de l'objet à reconstruire (MARQUES et al. 2018). Nous distinguons quatre familles d'algorithmes. La première contient les algorithmes de poursuite tels que MP (Matching Pursuit) (MALLAT et al. 1993) et OMP (Orthogonal Matching Pursuit) (PATI et al. 1993) et toutes leurs variantes. Ces algorithmes démarrent avec un support vide et vont construire $\tilde{x}$ en ajoutant itérativement une composante au support courant et en optimisant les coefficients associés. La deuxième est celle des algorithmes de seuillage tels que Iterative Hard Thresholding (IHT) (BLUMENSATH et Mike E. DAVIES 2009), NIHT (Normalized Iterative Hard Thresholding) (BLUMENSATH et Mike E DAVIES 2010), HTP (Hard Thresholding Pursuit) (FOUCART 2011) et leurs variantes. Ces algorithmes alternent entre des étapes de seuillage et de projection sur la contrainte de parcimonie. Ensuite, les algorithmes dits « à remplacement » comme OMPR (Orthogonal Matching Pursuit with Replacement) (JAIN et al. 2011), ELS (Exhaustive Local Search) (AXIOTIS et al. 2020) et leurs variantes, vont modifier $\tilde{x}$ en remplaçant itérativement une composante du support courant par une composante externe à ce support. Enfin, on peut considérer une seconde famille d'algorithme de poursuite avec CoSaMP (Compressive Sampling Matching Pursuit) (NEEDELL et al. 2009), SP (Subspace Pursuit) (DAI et al. 2009) et leurs variantes. Ces algorithmes vont construire $\tilde{x}$ en travaillant sur des supports avec des cardinalités plus grandes que k. Chacune de ces méthodes va permettre d'explorer différents supports possibles et de converger rapidement vers un minimum local.

Dans les cas avec peu de variables, il est possible d'utiliser des approches combinatoires de séparation et évaluation (de l'anglais Branch-and-Bound) (Andreas M TILLMANN et al. 2024), qui garantissent la convergence globale. Les algorithmes de type Branch-and-Bound (BEN MHENNI et al. 2022) permettent de résoudre des problèmes d'optimisation combinatoire en séparant l'espace des solutions en sous-ensembles plus petits, évitant ainsi l'évaluation exhaustive de l'ensemble des solutions.

Pour résoudre plus spécifiquement le problème de la factorisation de matrices en produit de matrices creuses, on utilise une méthode de minimisation alternée (BOLTE et al. 2014) couplée avec les méthodes que l'on vient de décrire. Cela permet de prendre en considération à la fois l'aspect séparément convexe (convexe en chacune des matrices de la factorisation) et la parcimonie du problème d'optimisation. Une seconde méthode consiste à résoudre le problème de factorisation de matrices dans un sous-espace de $\mathbb{R}^{m\times n}$ tel que l'espace des matrices Papillon (Y. LI et al. 2015) ou des matrices Monarque (DAO, B. CHEN et al. 2022). Plus précisément, au lieu de chercher un produit de matrices creuses ayant des contraintes de parcimonies quelconques, il s'agit ici de fixer la structure (voire le support) de chacun des facteurs. Cette méthode permet l'utilisation d'algorithmes avec des propriétés de convergence globale, voire de définir des projecteurs sur ces sous-espaces. Il existe de nombreux autres algorithmes pour résoudre ces deux problèmes, cependant j'ai préféré me concentrer sur les plus connus de chaque famille ainsi que ceux que je vais utiliser dans les prochains

chapitre.

1.4 Limites

Jusqu'ici, nous avons vu que la non-convexité et la combinatoire sous-jacente de nos problèmes était le principal obstacle à leur résolution. De ce fait, les fonctions que nous cherchons à minimiser possèdent de nombreux minima locaux. Atteindre un de ces minima locaux est relativement simple. Cependant, atteindre l'un des minima globaux ne l'est pas.

Dans le cas linéaire, on peut mesurer la difficulté théorique du problème à partir du dictionnaire A. C'est réalisable simplement en mesurant sa cohérence mutuelle.

Définition 1.4.1. (*Cohérence mutuelle (*MALLAT *et al. 1993 ; D.* DONOHO *et* HUO *2001 ;* TROPP *2004)*) Soit $A \in \mathbb{R}^{m\times n}$ un dictionnaire normalisé. La cohérence mutuelle de A est définie comme

$$\mu(A) := \|A^T A - I\|_\infty = \max_{\substack{i,j\in[\![1,n]\!]\\ i\neq j}} |\langle A_{|i}, A_{|j}\rangle|$$

Il s'agit du niveau de similitude le plus élevé entre deux colonnes du dictionnaire.

Remarque. (*Dictionnaire normalisé (par les colonnes, en norme ℓ_2)*) L'hypothèse de normalisation du dictionnaire est classique dans le cas linéaire. Si l'on pose $D_A = \operatorname{diag}\left(\frac{1}{\|A_{|1}\|_2}, \dots, \frac{1}{\|A_{|n}\|_2}\right)$, il est possible de remplacer A par $\hat{A} = AD_A$ sans perte de généralité. En effet, si $\hat{x} \in \underset{x\in\mathbb{R}^n,\ \|x\|_0\leq k}{\operatorname{argmin}} \frac{1}{2}\|\hat{A}x - y\|_2^2$, alors $\tilde{x} = D_A\hat{x} \in \underset{x\in\mathbb{R}^n,\ \|x\|_0\leq k}{\operatorname{argmin}} \frac{1}{2}\|Ax - y\|_2^2$.

Théorème 1.4.2. (*Unicité de la solution de* (SPARSE)*, démontrée dans le théorème 5.2 de (*ELAD *2010, page 85)*) Soient A, k, x^* et e les données du problème (SPARSE). Si $e = 0$ et $k < \frac{1}{2}\left(1 + \frac{1}{\mu(A)}\right)$, alors x^* est l'unique solution de (SPARSE).

Plus la cohérence mutuelle est élevée, plus il est simple (dans le pire des cas) de faire une erreur de reconstruction en sélectionnant une colonne non désirée plutôt qu'une autre en raison de leur similitude. En fonction du contexte, il peut être préférable d'utiliser une mesure issue de la propriété d'isométrie restreinte (de l'anglais RIP pour Restricted Isometry Property). Nous allons la préférer à la cohérence mutuelle, car elle rend plus précisément compte des spécificités de nos problèmes.

Définition 1.4.3. (*Propriété et constante d'isométrie restreinte (*CANDES *et al. 2005)*) Soit $A \in \mathbb{R}^{m\times n}$ un dictionnaire normalisé. Pour tout $k \in [\![1, n]\!]$, on définit la constante d'isométrie restreinte de A d'ordre k comme étant le plus petit

nombre positif δ_k, tel que, pour tout $x \in \mathbb{R}^n$, tel que $\|x\|_0 \leq k$, on a

$$(1-\delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1+\delta_k)\|x\|_2^2.$$

De plus, si $\delta_k < 1$, on dit que A satisfait la $k-$RIP (ou propriété d'isométrie restreinte d'ordre k).
Il s'agit de mesurer à quel point A est proche d'une isométrie lorsqu'on l'applique à des vecteurs parcimonieux.

Remarque. (*Croissance de δ_k*) La suite $(\delta_k)_{k\in[\![1,n]\!]}$ est croissante.

Remarque. (*Unicité de la solution de* (SPARSE) *avec le* $(2k)-$*RIP*) Si A satisfait la $(2k)-$RIP, alors x^* est l'unique solution de (SPARSE). En effet, pour tout vecteur $x \neq x^*$ de $\mathbb{R}^n$ tel que $\|x\|_0 \leq k$, on a $\|A(x^*-x)\|_2^2 > (1-\delta_{2k})\|x^*-x\|_2^2 > 0$.

Propriété 1.4.4. (*Définition équivalente*) Soient $k \in [\![1,n]\!]$ et $A \in \mathbb{R}^{m\times n}$ un dictionnaire normalisé. Sont équivalents :

1. A satisfait la $k-$RIP.
2. Pour tout $S \subseteq [\![1,n]\!]$, tel que $\mathrm{Card}(S) = k$, toutes les valeurs singulières de $A_{|S}$ sont dans l'intervalle $[\sqrt{1-\delta_k}, \sqrt{1+\delta_k}]$.

Démonstration. Supposons que A satisfasse la $k-$RIP, soit $\delta_k \in [0,1[$ la constante d'isométrie associée. Alors,

$$(1-\delta_k)\|x'\|_2^2 \leq \|Ax'\|_2^2 \leq (1+\delta_k)\|x'\|_2^2\,,\ \forall x' \in \mathbb{R}^n,\ \|x'\|_0 = k.$$

Cela équivaut à étudier A uniquement sur les supports de taille k :

$$(1-\delta_k)\|x\|_2^2 \leq \|A_{|S}x\|_2^2 \leq (1+\delta_k)\|x\|_2^2\,,\ \forall x \in \mathbb{R}^{|S|},\ S \subseteq [\![1,n]\!],\ |S| = k,$$

pour obtenir

$$1-\delta_k \leq \frac{\|A_{|S}x\|_2^2}{\|x\|_2^2} \leq 1+\delta_k.$$

Or, avec en notant $\sigma_{\max}(\cdot)$ (resp. $\sigma_{\min}(\cdot)$) l'opérateur renvoyant la plus grande (resp. la plus petite) valeur singulière d'une matrice, on a

$$\min_{x\neq 0} \frac{\|A_{|S}x\|_2^2}{\|x\|_2^2} = \sigma_{\min}(A_{|S}) \quad \text{et} \quad \max_{x\neq 0} \frac{\|A_{|S}x\|_2^2}{\|x\|_2^2} = \sigma_{\max}(A_{|S})$$

Ainsi, pour tout $x \neq 0$, on trouve par équivalence

$$1 - \delta_k \leq \sigma_{\min}(A_{|S})^2 \leq \frac{\|A_{|S}x\|_2^2}{\|x\|_2^2} \leq \sigma_{\max}(A_{|S})^2 \leq 1 + \delta_k,$$

et donc que

$$\sqrt{1-\delta_k} \leq \sigma_{\min}(A_{|S}) \leq \sigma_{\max}(A_{|S}) \leq \sqrt{1+\delta_k}.$$

Cette propriété permet de quantifier à quel point on peut se tromper de support lorsque l'on cherche un support de taille k. Ce qui est plus précis que la cohérence mutuelle. Grâce à cela, il est possible d'établir des limites théoriques de reconstruction du type $\delta_{lk} < c$ ou $\delta_{lk} < \frac{c}{\sqrt{k}}$ avec $c \in \mathbb{R}_+^*$ et $l \in \mathbb{N}^*$, pour les différents algorithmes approchant une solution de (SPARSE). De plus, dans la mesure où $(\delta_k)_{k\in[\![1,n]\!]}$ est croissante, on peut déduire une limite théorique de reconstruction sur k dépendant de A.

En pratique, chaque famille d'algorithmes renonce à quelque chose afin d'approcher une solution de notre problème. Les algorithmes gloutons vont explorer un nombre très restreint de minima locaux, à l'aide d'une heuristique simple à calculer, pour s'arrêter dans l'un d'entre eux. Les approches combinatoires ne sont utilisables qu'en petite dimension, en raison de la croissance exponentielle de la combinatoire de notre problème. Enfin, on sait que le problème relaxé en norme ℓ_1 n'est pas toujours équivalent au problème en norme ℓ_0 (voir figure 1.5). On a donc actuellement le choix entre une approximation très rapide, une résolution exacte en temps très long et une optimisation simple fonctionnant seulement pour les pus petites valeurs de k. Ainsi, le choix se fait très souvent en fonction de la taille du problème, des spécificités du dictionnaire A, de ce que l'on sait sur x^* et sur k.

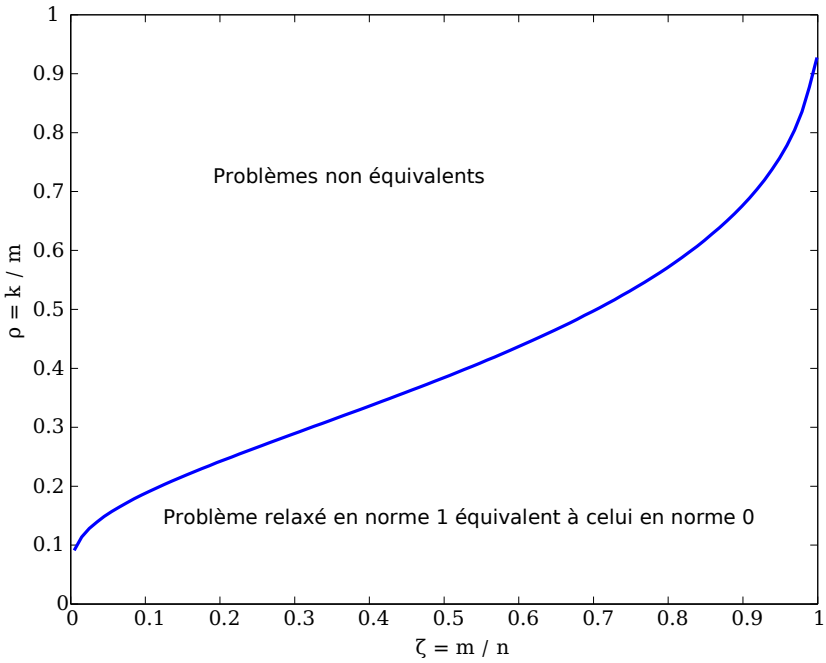

FIGURE 1.5 – Diagramme de phase théorique (adapté de (D. DONOHO et STODDEN 2006)). La ligne bleue (transition de phase) représente le seuil théorique à partir duquel la relaxation en norme ℓ_1 ne permet plus de résoudre le problème en norme ℓ_0. L'axe des abscisses représente le niveau de sous-détermination de A. L'axe des ordonnées représente le niveau de parcimonie du problème. Par exemple, s'il y a deux fois plus de colonnes que de lignes dans A ($\zeta = 0.5$), alors la relaxation n'est assurée de fonctionner que lorsque k est inférieur à 30% du nombre de lignes ($\rho \approx 0.3$).

Dans le cas matriciel, on retrouve les mêmes problématiques. Cependant, en raison de la non-convexité engendrée par le produit de matrices, les approches alternées n'ont pas de garantie de convergence globale. De plus, même lorsque l'on fixe les supports des matrices que l'on veut retrouver, la fonction à optimiser peut ne pas admettre de minimum global (LE, RICCIETTI et al. 2023). C'est pour cela que l'on privilégie aujourd'hui la factorisation dans des sous-espaces de $\mathbb{R}^{m\times n}$ où l'on peut concevoir des algorithmes de factorisation exacte. Cependant, ce type d'approche sacrifie l'expressivité de la factorisation sur $\mathbb{R}^{m\times n}$ tout entier au profit d'algorithmes capables de projeter les matrices $A \in \mathbb{R}^{m\times n}$ dans l'un de ses sous-espaces de matrices factorisables.

1.5 Optimisation de structures non convexes et non différentiables en apprentissage automatique

L'optimisation de fonctions non différentiables et non continues est une problématique également présente en optimisation de réseaux de neurones. En effet, par construction, optimiser les paramètres d'un réseau de neurones constitué des couches les plus communes (dense, convolution, attention, activations) revient à optimiser une fonction non convexe, parfois non différentiable en un nombre restreint de points. Cela n'a pas empêché ces structures d'obtenir leur succès actuel. Dans cette situation, l'objectif n'est pas de trouver le minimum global de la fonction, mais de la minimiser dans une certaine mesure. Bien que cette optimisation repose sur des techniques de descente de gradient, il est tout de même possible d'utiliser des composants non différentiables et non continus dans ces réseaux. C'est ce qu'il se passe dans les réseaux de neurones à poids binaires comme BinaryConnect (COURBARIAUX et al. 2015 ; SAYED et al. 2023).

La construction d'une couche dense d'un réseau de neurones à poids binaires est illustrée dans la figure 1.6 et la figure 1.7. Ce réseau utilise des poids binaires qui sont utilisés pendant l'évaluation et le calcul du gradient. Ces poids binaires ne sont pas directement appris. En effet, chacun de ces poids est le résultat d'une fonction de «binarisation» appliquée à une valeur flottante à apprendre. Par définition, une fonction de binarisation qui va de $\mathbb{R}$ dans $\{-1,1\}$ ne peut pas être continue, ni différentiable. Ainsi, afin de permettre la rétropropagation du gradient jusqu'aux valeurs flottantes à apprendre, il suffit de faire appel à l'estimateur direct.

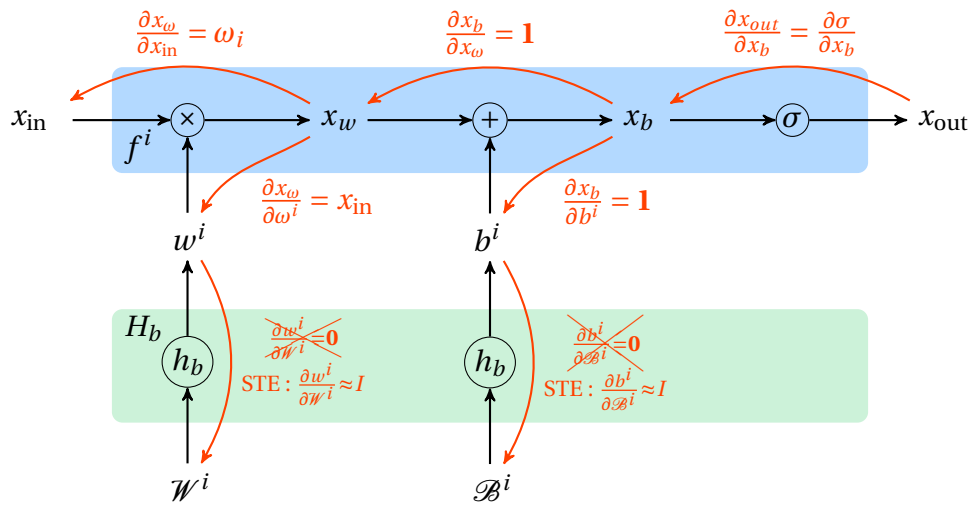

FIGURE 1.6 – Couche d'un réseau de neurones à poids binaires. Prenons $x_{\text{in}} \in \mathbb{R}^n$ et $x_{\text{out}} \in \mathbb{R}^m$ et σ une fonction d'activation. Les poids flottants $\mathcal{W}^i \in \mathbb{R}^{m \times n}$ et $\mathcal{B}^i \in \mathbb{R}^m$ sont appris par descente de gradient. Les poids binaires $w^i \in \{-1,1\}^{m \times n}$ et $b^i \in \{-1,1\}^m$ sont obtenus en binarisant les poids flottants à l'aide de l'opérateur h_b. L'estimateur direct permet de propager le gradient de la fonction de perte jusqu'aux poids flottants. Les flèches rouges représentent les gradients de chaque opération par rapport à leurs entrées. Pour obtenir le gradient de la fonction de perte par rapport aux poids flottants, il faut appliquer la règle de la chaine (multiplier les gradients de chaque opération).

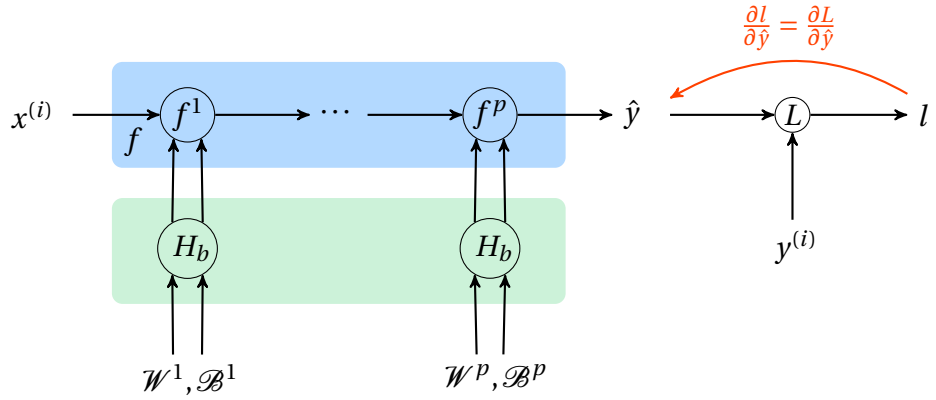

FIGURE 1.7 – Réseau de neurones à poids binaires. Prenons $x^{(i)} \in \mathbb{R}^n$ et $y^{(i)} \in \mathbb{R}^m$ et L une fonction de perte de $\mathbb{R}^m \times \mathbb{R}^m$ dans $\mathbb{R}$. Le réseau de neurones est composé de p couches $f^1, \ldots, f^p$. Le détail d'une couche est donné dans la figure 1.6.

Définition 1.5.1. (*Estimateur direct (STE) (G. E.* HINTON *2012 ;* BENGIO *et al. 2013)*) Soit F une fonction différentiable d'un espace vectoriel $\mathscr{D}$ de dimension n, à valeurs réelles. Soit H une fonction non différentiable de $\mathscr{D}$ dans $\mathscr{S} \subset \mathscr{D}$. Soit $\mathscr{X} \in \mathscr{D}$. Appliquer la technique de l'estimateur direct (de l'anglais STE, Straight-Through Estimator) revient à réaliser l'approximation :

$$\begin{aligned} \left.\frac{\partial F \circ H}{\partial \mathscr{X}}\right|_{\mathscr{X}} &= \left.\frac{\partial F}{\partial x}\right|_{x} \left.\frac{\partial H}{\partial \mathscr{X}}\right|_{\mathscr{X}} \\ &\approx \left.\frac{\partial F}{\partial x}\right|_{x} \cdot I_n \end{aligned}$$

Cela consiste à remplacer, dans la règle de la chaine, le gradient de la fonction non différentiable H, $\left.\frac{\partial H}{\partial \mathscr{X}}\right|_{\mathscr{X}}$; par celui de la fonction identité, I_n.

L'utilisation de cette technique permet d'utiliser des algorithmes de gradient sur des fonctions non différentiables ayant un gradient nul sur l'ensemble des points où elles sont différentiables. Afin d'illustrer son fonctionnement, la figure 1.8 présente l'application de la technique de l'estimateur direct à la quantification d'une droite identité et d'une parabole. On voit que cela revient à remplacer le gradient de la fonction que l'on cherche à différencier par le gradient de son approximation en fonction affine par morceaux. Grâce à cette technique, il est possible d'inclure des fonctions non continues et non différentiables, telles que la quantification et la binarisation dans les réseaux de neurones. De nombreux travaux ultérieurs améliorent ces méthodes sous divers aspects (C. YUAN et al. 2023 ; SAYED et al. 2023).

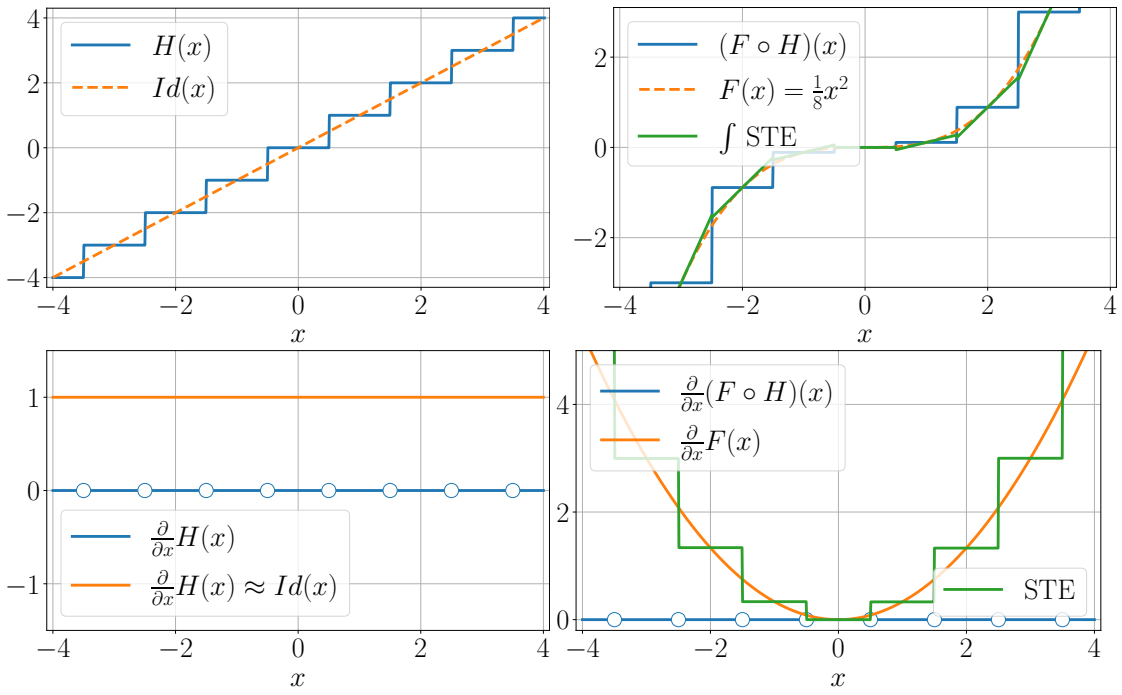

FIGURE 1.8 – Exemple d'application de l'estimateur direct (STE) pour différencier la fonction d'arrondi H qui renvoie l'entier le plus proche de son entrée. Les deux figures du haut montrent la fonction d'arrondi appliquée à la droite identité (en haut à gauche) et à une parabole (en haut à droite). Les deux figures du bas montrent les dérivées de ces fonctions ainsi que l'approximation de la dérivée de la fonction d'arrondi par celle de l'identité. Il n'y a pas de courbe verte dans les figures de gauches, car elle serait confondue avec la courbe orange.

Reprenons le réseau de neurones binaires de la figure 1.7 et écrivons le problème d'optimisation que l'on doit résoudre pour apprendre ses poids.

$$\underset{\mathscr{W}^j,\mathscr{B}^j \text{ flottants}, j\in[\![1,p]\!]}{\text{Minimiser}} \quad \sum_{i=1}^{M} L(y^{(i)}, f(x^{(i)}, H_b(\mathscr{W}^1,\mathscr{B}^1),\cdots,H_b(\mathscr{W}^p,\mathscr{B}^p))),$$

avec f le réseau de neurones, $H_b : \mathbb{R} \to \{-1,1\}$ une fonction de binarisation, $L : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ la fonction de perte à minimiser, et $\{(x^{(i)}, y^{(i)})\}_{i\in[\![1,M]\!]}$ le jeu de données d'entrainement où $x^{(i)} \in \mathbb{R}^n$ est une entrée et $y^{(i)} \in \mathbb{R}^m$ une étiquette. Ce problème d'optimisation est équivalent au problème suivant :

$$\underset{w^j,b^j \text{ binaires}, j\in[\![1,p]\!]}{\text{Minimiser}} \quad \sum_{i=1}^{M} L(y^{(i)}, f(x^{(i)}, w^1, b^1, \cdots, w^p, b^p)).$$

On constate qu'il s'agit en réalité d'un problème d'optimisation discret, où le but est de connaître la position des 1 (ou de manière équivalente celle des -1). Et, grâce à l'estimateur direct, on peut résoudre ce problème sous forme continue (sur $\mathscr{W}^j$ et $\mathscr{B}^j$)

à l'aide d'un algorithme de descente de gradient. Lors de la descente de gradient, les poids flottants $(\mathscr{W}^j, \mathscr{B}^j)$ vont évoluer dans un espace continu afin d'explorer l'espace discret dans lequel w^j et b^j évoluent. Cette exploration, qui est un problème de combinatoire, va dépendre de la fonction de binarisation et de la méthode de descente employée.

Ces considérations d'exploration d'un espace discret et les difficultés sous-jacentes du problème d'optimisation associé sont très proches de celles du problème de reconstruction de support parcimonieux. Ainsi, nous nous proposons de dériver, dans le chapitre 3, l'estimateur direct utilisé en binarisation afin de développer un estimateur direct utilisable dans les problèmes de reconstruction de support. Cela permettra d'avoir accès à toute la littérature sur la descente de gradient pour résoudre les problèmes de reconstruction de support parcimonieux en théorie, dans le chapitre 4, et en pratique, dans le chapitre 5.

2 Une application de la parcimonie à l'acoustique musicale

Sommaire

2.1 Introduction 40
2.2 Du modèle physique de la trompette et du musicien aux descripteurs de performance 41
2.3 Analyse des descripteurs provenant de trompettes haut de gamme . . . 45
2.3.1 Classification des trompettes 46
2.3.2 Interprétabilité des résultats de la classification 47
2.4 Prédiction des descripteurs sur des trompettes virtuelles 49
2.4.1 Construction du jeu de données 49
2.4.2 Étude comparative d'algorithmes d'apprentissage automatique 52
2.5 Zoom sur le LASSO-LARS 54
2.5.1 Formulation du problème d'optimisation 54
2.5.2 Interprétabilité du modèle 55
2.6 Outil pour les concepteurs de trompettes 57
2.7 Conclusions 58

Lorsque l'on réalise une thèse en France, il est possible de réaliser trois types de mission en tant que doctorant : une mission d'enseignement, de vulgarisation scientifique ou d'expertise. Dans mon cas, j'ai consacré un an et demi pour une mission d'expertise puis un an et demi à une mission d'enseignement. La mission consistait à mener un projet en collaboration avec le LMA (Laboratoire de Mécanique et d'Acoustique de Marseille) afin de développer un outil d'aide à la conception de trompettes basé sur l'apprentissage automatique. Ce projet a donné lieu à deux publications : la première (FRÉOUR, MOHAMED et al. 2023) dans la conférence européenne d'acoustique *Forum Acusticum 2023*, et la seconde (MOHAMED, FRÉOUR et al. 2024) (qui est une extension de la première) dans l'édition spéciale *Acoustique musicale - Derniers progrès dans les méthodes analytiques, numériques et expérimentales* du journal européen d'acoustique *Acta Acustica*. Il a également permis le développement d'un outil numérique, sous la forme d'une plateforme web, pour les ingénieurs de Yamaha qui permet d'entraîner et d'utiliser des modèles d'apprentissage automatique afin de prédire les performances d'une trompette en fonction de ses paramètres de conception. Ce chapitre porte sur le contenu de ces deux articles. Les notions d'acoustiques clés, nécessaires à la compréhension de ce chapitre, seront présentées. Pour une étude

plus approfondie, je vous invite à consulter les livres de référence (FLETCHER et al. 2012) ou (CHAIGNE et al. 2013 ; CHAIGNE et al. 2016) qui sont un peu plus techniques. Ici, le but est de montrer comment une méthode d'apprentissage automatique favorisant la parcimonie peut être utilisée pour prédire des résultats de simulations numériques complexes, et comment cette méthodologie peut être utilisée pour aider les concepteurs d'instruments de musique.

2.1 Introduction

Dans divers domaines de l'ingénierie, la modélisation physique est devenue une stratégie largement utilisée pour simuler et prédire le comportement des systèmes physiques. En acoustique musicale, des modèles physiques d'instruments de musique ont été développés au cours des dernières décennies pour la synthèse sonore et pour mieux comprendre les mécanismes sous-jacents au fonctionnement des instruments de musique. Ces modèles permettent également d'élucider les relations entre la conception et les réponses acoustiques telles que l'intonation, la facilité de souffler et le timbre. Rendre possible le calcul numérique de descripteurs quantitatifs associés au fonctionnement des instruments de musique est important pour les fabricants d'instruments, car cela leur permet d'utiliser le prototypage virtuel comme outil dans le processus de développement de nouveaux instruments. Ce prototypage virtuel est très apprécié, car il permet d'économiser le temps et les ressources associées à la production de prototypes matériels par les procédures traditionnelles d'essais et d'erreurs. Ainsi, il est intéressant de fournir aux concepteurs d'instruments de musique des outils numériques basés sur des modèles physiques pour faciliter le développement de nouveaux instruments.

En acoustique musicale, des modèles physiques de cuivres ont été proposés en utilisant une description «modale» de l'excitateur (les lèvres) et du résonateur (la colonne d'air de l'instrument), ainsi qu'une équation de flux (passage de l'air de la bouche à l'instrument) non linéaire qui couple les deux systèmes linéaires (CAMPBELL et al. 2021). Ces modèles ont montré une grande capacité à reproduire le comportement des cuivres (CULLEN et al. 2000 ; VELUT et al. 2017 ; KABURAGI et al. 2023) et offrent une bonne description de l'instrument en utilisant un nombre limité d'équations. Ils offrent également une précision suffisante pour comparer des instruments dont la conception a été légèrement modifiée si les paramètres modaux de ces instruments sont calculés ou extraits de mesures suffisamment précises (FRÉOUR, GUILLOT, MASUDA, TOMINAGA et al. 2020 ; FRÉOUR, MASUDA et al. 2023 ; PETIOT et al. 2023). Différents résultats peuvent être extraits de ces modèles, en fonction de la méthode numérique appliquée. Parmi ces résultats, le diagramme de bifurcation est un outil permettant d'analyser le comportement d'un instrument de musique en fonction de ses paramètres de conception et des paramètres du musicien. Un fabricant peut utiliser ces résultats pour évaluer et comparer les performances des instruments à vent.

Cependant, le maniement de ces méthodes nécessite des connaissances théoriques

et techniques poussées, ce qui les rend difficiles à utiliser pour un novice. En outre, le choix des paramètres du modèle, en particulier ceux des lèvres pour les cuivres, est crucial, car les paramètres mécaniques des lèvres sont difficiles à estimer expérimentalement. Ils peuvent également avoir un impact significatif sur le comportement du modèle. Pour tenir compte de la variabilité et de l'incertitude des valeurs des paramètres des lèvres, il est possible de résoudre des équations pour un grand nombre de valeurs de paramètres des lèvres (joueurs virtuels). Néanmoins, elle induit un coût de calcul élevé, ce qui constitue une limitation importante, en particulier lorsqu'il est nécessaire d'étudier de nombreuses configurations d'instruments et de joueurs virtuels. Compte tenu de ces contraintes, le transfert de cette technologie dans une application « facile à utiliser » dans l'atelier d'un fabricant n'est pas simple, et des solutions doivent être trouvées pour surmonter ce problème.

Une stratégie explorée dans des travaux antérieurs consiste à entraîner un modèle d'apprentissage automatique peu coûteux en tant que modèle de substitution pour calculer ces résultats de modélisation. Cela a l'avantage d'avoir un coût de calcul bien inférieur à celui des méthodes numériques usuelles. Cette stratégie a été appliquée à l'optimisation de la perce de la trompette (TOURNEMENNE et al. 2017 ; PETIOT et al. 2023). Cette stratégie, qui associe la modélisation physique à l'apprentissage automatique et aux méthodes basées sur les données, a également été appliquée à d'autres instruments, tels que le violon (GONZALEZ et al. 2021) et le piano (MOKDAD et al. 2013).

Dans cette étude, nous avons suivi cette approche pour développer un outil numérique pour les concepteurs de trompettes. Cet outil permet de calculer plusieurs descripteurs associés au diagramme de bifurcation d'une trompette à l'aide d'une méthode d'apprentissage automatique favorisant la parcimonie, ce qui n'a jamais été fait auparavant. Le chapitre est organisé comme suit : La section 2.2 décrit comment obtenir les descripteurs de performance de jeu à partir du modèle physique de la trompette. La section 2.3 étudie la pertinence des descripteurs de jeu sélectionnés. La section 2.4 compare expérimentalement les méthodes permettant de générer les descripteurs de performance par apprentissage automatique. La section 2.5 décrit de manière approfondie la méthode la plus performante pour prédire ces descripteurs. Enfin, la section 2.6 présente l'outil développé pour les ingénieurs de Yamaha ainsi que les informations qu'ils peuvent en extraire. La conclusion est dans la section 2.7.

2.2 Du modèle physique de la trompette et du musicien aux descripteurs de performance

L'anatomie de la trompette est décrite dans la figure 2.1. En ce qui concerne la trompette et le musicien, le modèle physique classique est basé sur trois équations qui supposent une propagation linéaire dans le résonateur (la colonne d'air de la

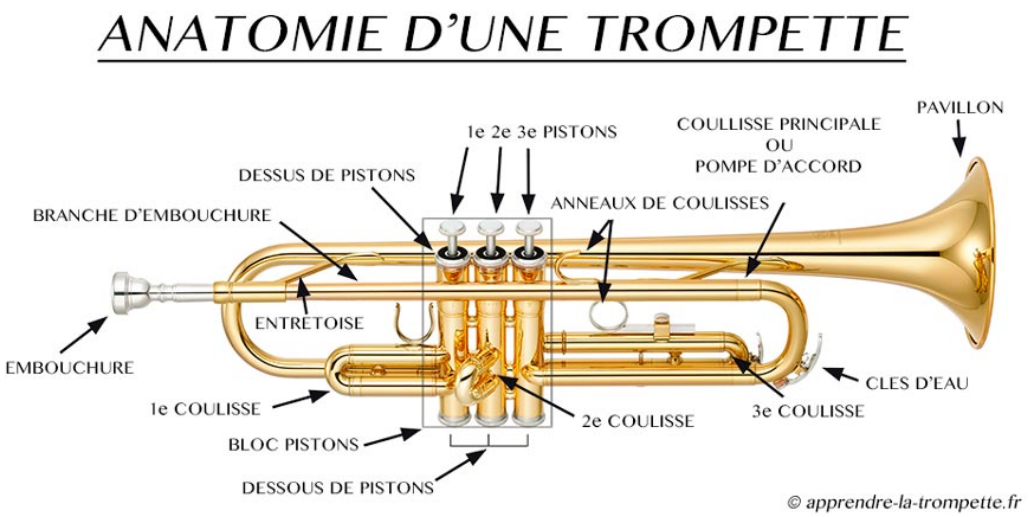

FIGURE 2.1 – Schéma d'une trompette. En jouant, le musicien souffle dans l'embouchure en faisant vibrer ses lèvres. Les lèvres vibrent à une fréquence proche de la fréquence de résonance de la trompette, ce qui génère une onde acoustique dans la colonne d'air de la trompette. La colonne d'air commence au niveau de l'embouchure et se termine au niveau du pavillon. Appuyer sur les pistons et tirer sur la coulisse permet de changer la longueur de la colonne d'air, ce qui modifie les fréquences de résonance de l'instrument (et donc la note jouée).
Source de l'image : Apprendre la trompette [1].

trompette). Le modèle s'écrit comme suit en fonction du temps $t \in \mathbb{R}^+$:

$$\begin{cases} \frac{d^2y}{dt^2}(t) + \frac{\omega_l}{Q_l}\frac{dy}{dt}(t) + {\omega_l}^2(y(t) - y_0) = \frac{1}{\mu_l}(p_0 - p(t)) \\ \frac{dp_i}{dt}(t) = Z_c C_i u(t) + s_i p_i(t), \forall i \in [\![1, N]\!] \\ u(t) = \frac{|y(t)|+y(t)}{2}\sqrt{\frac{2|p_0-p(t)|}{\rho}}b \cdot sign(p_0 - p(t)), \end{cases} \tag{2.1}$$

avec $p(t) = 2\sum_{i=1}^{N} \Re(p_i(t))$.

Les fonctions inconnues sont la position verticale de la lèvre du musicien $y(t)$, la pression de l'air dans l'embouchure (l'entrée) de la trompette $p(t)$ ainsi que le débit volumique de l'air entrant dans la trompette $u(t)$. La première équation caractérise mécaniquement l'excitateur (les lèvres du musicien). Ce modèle ne prend en compte qu'une seule lèvre, car on considère que les deux lèvres vibrent de manière symétrique. La deuxième equation caractérise le résonateur (la colonne d'air de la trompette) pour différents modes de résonance indexés par i. La troisième équation décrit le couplage

1. https://apprendre-la-trompette.fr/la-trompette/les-differentes-parties-de-lanatomie-dune-trompette-a-pistons/

entre le musicien et l'instrument.

Les différents paramètres sont les suivants. Pour caractériser les lèvres, on utilise la position verticale de la lèvre au repos y_0, la fréquence angulaire de résonance ω_l, le facteur de qualité Q_l, la masse par surface μ_l et la largeur d'ouverture des lèvres b. Les valeurs de ces paramètres, tirées de (FRÉOUR, GUILLOT, MASUDA, TOMINAGA et al. 2020), sont déterminées sur la base des valeurs fournies dans la littérature, en tenant compte des contraintes pour générer un son dans une trompette. Pour caractériser la trompette, on utilise l'impédance caractéristique Z_c ainsi que les pôles s_i et résidus C_i, avec $i \in [\![1, N]\!]$, associés aux $N = 11$ fréquences de résonance de la trompette. Enfin, on utilise la pression de l'air dans la bouche du musicien p_0 et la densité de l'air ρ.

Ce système d'équations peut être résolu en utilisant une méthode numérique appelée «continuation numérique pour solutions périodiques» décrite dans (FRÉOUR, GUILLOT, MASUDA, TOMINAGA et al. 2020, section 3). Cette méthode permet de résoudre les équations différentielles en fonction d'un paramètre, ici la pression de l'air dans la bouche p_0. Plus précisément, cela veut dire que l'on résout les équations pour une valeur de p_0 donnée, puis, que l'on augmente ou diminue p_0 pour obtenir une nouvelle fonction solution $p(t)$. En répétant cette opération, on peut obtenir un diagramme de bifurcation, c'est-à-dire un graphique montrant l'évolution de l'amplitude de $p(t)$: $|p|$ (car on cherche $p(t)$ périodique) en fonction de p_0. De la même manière, on peut obtenir l'évolution de la fréquence fondamentale (hauteur) f_0 du son joué en fonction de p_0. Ceci peut être fait pour chaque note de musique jouable par la trompette à l'aide du logiciel MANLAB (ARQUIER et al. 2005).

La figure 2.2 montre un exemple de diagramme de bifurcation pour une trompette réglée pour jouer un Si♭4. Il s'agit du son que l'on a montrée dans la figure 1.2 du chapitre 1. Sur ce diagramme de bifurcation, on peut voir la branche stable (les solutions accessibles du système) et la branche instable. Cela donne lieu à ce que l'on appelle un phénomène d'hystérésis (FRÉOUR, GUILLOT, MASUDA, VERGEZ et al. 2022), c'est-à-dire que l'amplitude de la pression de l'air dans la trompette ne se comporte pas de la même manière suivant que l'on augmente (crescendo) ou diminue (decrescendo) la pression de l'air dans la bouche. En effet, les flèches rouges indiquent les variations de pression dans l'embouchure de la trompette $|p|$ en fonction de la pression de l'air dans la bouche p_0 lors d'un crescendo. Les flèches vertes indiquent les variations de pression dans l'embouchure de la trompette $|p|$ en fonction de la pression de l'air dans la bouche p_0 lors d'un decrescendo.

Un certain nombre de valeurs spécifiques peuvent être extraites du diagramme de bifurcation, en particulier des tracés de $|p|$ et f_0, comme reporté sur la figure. Ces quantités décrivent les principales caractéristiques des solutions du système d'équations (2.1) qui constituent le diagramme de bifurcation. Elles sont étroitement liées à la performance de l'instrument, à savoir la facilité de jeu, la qualité et stabilité du son, l'intonation, etc. Nous avons donc choisi d'appeler ces valeurs « descripteurs de performance ». Les définitions des 10 descripteurs de performance qui nous intéressent sont fournies dans le tableau 2.1. On peut considérer que pour chaque paire {joueur — instrument}, ces 10 descripteurs constituent une évaluation quantitative des performances de l'instrument. En utilisant ces descripteurs, un fabricant de trompette

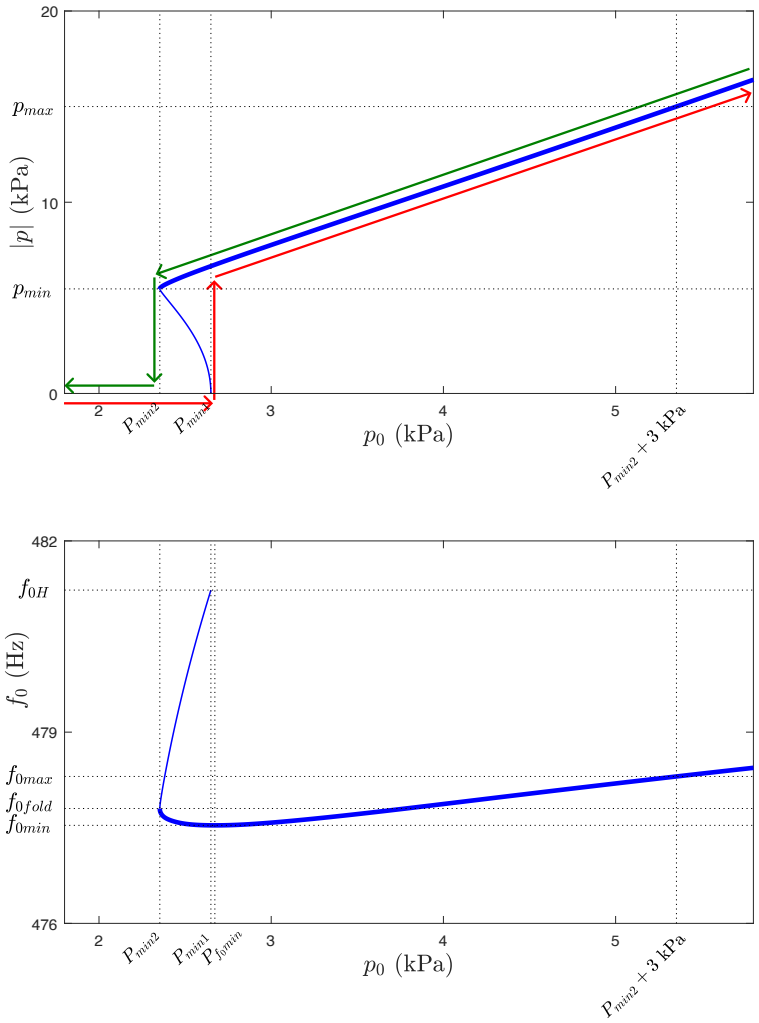

FIGURE 2.2 – Diagramme de bifurcation de la trompette pour un Si♭4. Le tracé de la pression de l'air dans l'embouchure de la trompette $|p|$ en fonction de la pression de l'air dans la bouche p_0 est représenté en haut. Le tracé de la fréquence fondamentale f_0 en fonction de la pression de l'air dans la bouche p_0 est représenté en bas. La ligne fine représente la branche instable, tandis que la ligne épaisse représente la branche stable (les solutions accessibles du système). La description des descripteurs de performance (abscisses et ordonnées notées sur les graphiques) est donnée dans le tableau 2.1.

peut aisément comparer quantitativement les trompettes sur plusieurs aspects. De plus, l'utilisation de ces descripteurs permet de reconstruire une approximation affine par morceaux du diagramme de bifurcation associé.

Comme décrit dans la figure 2.3, l'objectif de cette étude est de prédire ces descrip-

Descripteur	Définition
$P_{\min 1}$	Valeur minimale de p_0 lors d'un crescendo
$P_{\min 2}$	Valeur minimale de p_0 lors d'un decrescendo
$p_{\min}$	Amplitude de p lorsque $p_0 = P_{\min 2}$
$p_{\max}$	Amplitude de p lorsque $p_0 = P_{\min 2} + 3$ kPa (kilopascal)
$slope$	Pente de la branche stable du diagramme sur $\lvert p \rvert$, entre les abscisses $P_{\min 2}$ et $P_{\min 2} + 3$ kPa
$f_{0\min}$	Valeur minimale de f_0 sur la branche stable entre $P_{\min 2}$ et $P_{\min 2} + 3$ kPa
$f_{0\max}$	Valeur maximale de f_0 sur la branche stable entre $P_{\min 2}$ et $P_{\min 2} + 3$ kPa
$P_{f_{0\min}}$	Valeur de p_0 lorsque $f_0 = f_{0\min}$
$f_{0\text{fold}}$	Valeur de f_0 lorsque $p_0 = P_{\min 2}$
$f_{0\text{H}}$	Valeur de f_0 lorsque $p_0 = P_{\min 1}$

TABLEAU 2.1 – Définition des descripteurs de performance de jeu extraits du diagramme de bifurcation de la figure 2.2. La pression $P_{\min 2} + 3$ kPa est considérée comme une pression maximale raisonnable dans la bouche du musicien pour servir de point de repère.

teurs à l'aide d'un modèle d'apprentissage automatique, pour s'affranchir de l'utilisation des méthodes de continuation. Notons tout d'abord que d'après la figure 2.2 (bas), il apparaît que $P_{\min 1}$ et P_{f0min} sont presque identiques (P_{f0min} est légèrement plus grand que $P_{\min 1}$), bien que nous ne connaissions aucune raison théorique justifiant que la pression minimale nécessaire pour commencer un crescendo soit proche de celle permettant d'atteindre la valeur minimale de f_0. En outre, on peut déduire des 10 descripteurs du tableau 2.1, d'autres descripteurs de performances potentiellement significatifs pour les joueurs : l'« amplitude de l'hystérésis » $H = P_{\min 1} - P_{\min 2}$, la « plage dynamique » (variation d'amplitude du son sur la plage de pressions étudiée) $D = p_{\max} - p_{\min}$, ou la « stabilité de la hauteur » du son joué $\Delta f_0 = f_{0\max} - f_{0\min}$.

2.3 Analyse des descripteurs provenant de trompettes haut de gamme

Dans cette section, nous évaluons la capacité des descripteurs à classifier les trompettes haut de gamme, sans prendre en compte leurs paramètres modaux. Il est en effet intéressant de confirmer la pertinence des descripteurs extraits du diagramme de bifurcation pour distinguer les instruments existants au sein d'une même gamme de produits (en particulier les instruments utilisés par des professionnels) avant d'essayer de les générer. Ce sera l'objet de la section suivante.

Pour réaliser cette évaluation, nous considérons 4 trompettes réelles issues du commerce étiquetées B, S, V et W, ainsi que 800 joueurs virtuels pour construire un jeu de données composé de 3 200 diagrammes de bifurcation, un pour chaque couple joueur/trompette. L'ensemble des 800 joueurs virtuels est généré en utilisant

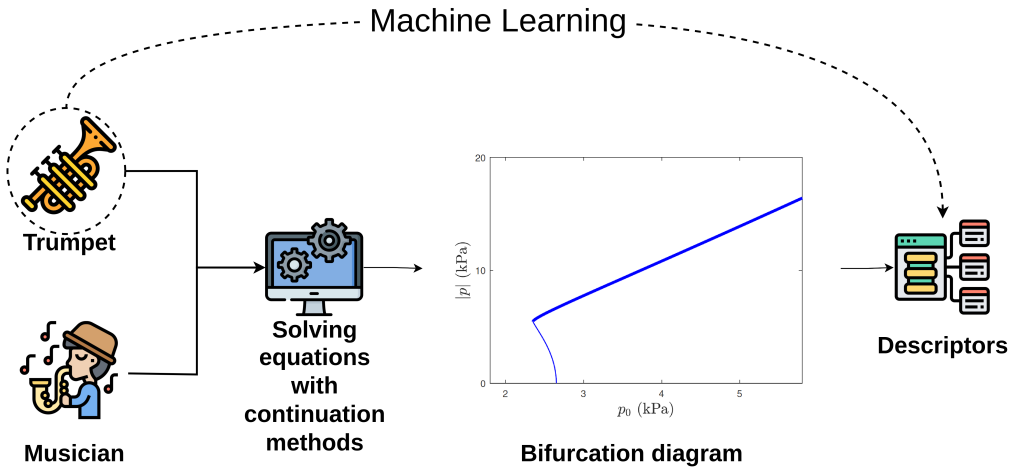

FIGURE 2.3 – Objectif du projet : prédire les descripteurs de performance à partir d'un modèle d'apprentissage automatique. Ce modèle ne doit ni requérir de connaissance des méthodes numériques, ni demander beaucoup de temps de calcul (a contrario de l'approche physique traditionnelle). Les avantages de cette approche sont simples :
1) le modèle peut être intégré dans une application logicielle simple d'utilisation, utilisable par un concepteur en autonomie,
2) les calculs peuvent être effectués pour plusieurs joueurs virtuels afin de tenir compte des incertitudes liées aux paramètres des lèvres, ce qui permet une analyse plus riche de la trompette.

un échantillonnage uniforme des paramètres des lèvres (Q_l, μ_l, et y_0), dans des limites correspondant à ±10% des valeurs des paramètres utilisés dans (FRÉOUR, GUILLOT, MASUDA, TOMINAGA et al. 2020). Ces limites dans l'espace des joueurs virtuels sont quelque peu arbitraires ; nous supposons que les variations de ±10% représentent des attentes raisonnables pour les différences entre des joueurs humains ou des variations au sein d'un groupe de joueurs d'un niveau de jeu similaire. Dans un premier temps, dans la section 2.3.1, nous allons tenter de classifier les trompettes à partir des 10 descripteurs de performance extraits de ces diagrammes de bifurcation à l'aide d'un modèle d'apprentissage automatique. Ensuite, dans la section 2.3.2, nous interprèterons le modèle d'apprentissage afin de déterminer les descripteurs les plus importants pour réaliser cette classification.

2.3.1 Classification des trompettes

Nous souhaitons maintenant démontrer que les descripteurs extraits des diagrammes de bifurcation permettent de classer efficacement différentes trompettes. Pour ce faire, nous utilisons le classifieur XGBoost (T. CHEN et al. 2016), un modèle d'apprentissage automatique reposant sur les arbres de décision ainsi que le boosting. Nous suivons une procédure standard pour entraîner ce classifieur. Le jeu de données est

divisé en un jeu d'entraînement (75 % du jeu de données) et un jeu de test (25 % du jeu de données) tout en conservant la même proportion de chaque trompette dans chacun des deux nouveaux jeux de données. Le réglage des hyperparamètres de XGBoost est réalisé par validation croisée stratifiée (BERRAR 2019) à cinq plis sur le jeu d'entraînement.

Nous utilisons la précision moyenne (moyenne des pourcentages de trompettes bien classées), désignée par $PRECISION_{moyenne}$, pour évaluer la performance de la classification. En outre, nous reproduisons l'expérience en utilisant des descripteurs représentés dans l'espace des composantes principales. Pour ce faire, nous appliquons l'analyse en composantes principales (ACP) (HOTELLING 1933) aux descripteurs normalisés [2] du jeu d'entraînement avant d'entraîner XGBoost. C'est une méthode courante de prétraitement qui permet souvent d'améliorer les performances des modèles basés sur les arbres de décision. Ici, nous ne nous en servons pas pour réduire la dimensionnalité de l'espace des descripteurs, mais pour améliorer la séparabilité des classes.

Les matrices de confusion résultant de la classification avec les descripteurs bruts et les descripteurs projetés dans l'espace des composantes principales sont présentées dans la figure 2.4. L'examen des matrices de confusion révèle que la classification des trompettes basée sur les descripteurs bruts atteint une précision moyenne de 81,12%. Cependant, le prétraitement des descripteurs à l'aide de l'ACP améliore considérablement les performances de la classification, avec une précision moyenne de 99,50%. De plus, nous savons qu'en réalité, les trompettes S et W sont très similaires. Et ce sont également les deux trompettes qui sont le plus souvent confondues par le modèle. Ainsi, ces résultats montrent que les descripteurs de performance suffisent à distinguer efficacement ces quatre instruments, et que les trompettes les plus proches sont celles qui sont le plus souvent confondues.

2.3.2 Interprétabilité des résultats de la classification

Nous allons maintenant interpréter le classifieur. Comme ce dernier est basé sur des arbres de décision, il est possible d'évaluer l'importance relative de chaque caractéristique dans la classification en calculant le gain qui lui est associé. Il s'agit de l'amélioration de la précision apportée par une caractéristique aux branches sur lesquelles elle se trouve. Autrement dit, plus le gain est élevé, plus la caractéristique est importante pour la classification. La figure 2.5 montre le gain de chaque caractéristique dans les deux configurations de l'expérience : en utilisant les descripteurs représentés dans l'espace d'origine et dans l'espace des composantes principales.

Dans l'espace des descripteurs, aucun descripteur ne semble inutile pour la classification. Les descripteurs les plus importants sont $f_{0\text{fold}}$, $slope$ et $f_{0\text{max}}$. Cela montre que les bornes du tracé de f_0 sont des descripteurs de performance cruciaux pour distinguer les trompettes dans cet espace. Toutefois, ils ne sont pas suffisants, car

2. Il est inutile de normaliser des données utilisées pour entrainer un arbre de décision. C'est pour cela que la normalisation n'est utilisée que lorsque l'on utilise l'ACP.

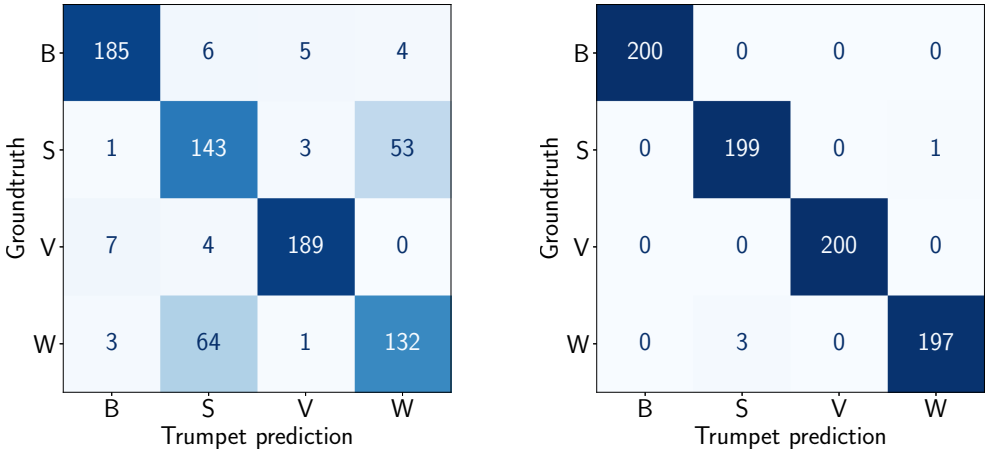

FIGURE 2.4 – Matrices de confusion de la classification des instruments avec XGBoost.
À gauche : les entrées du modèle sont les descripteurs bruts, $PRECISION_{moyenne} = 81,12\%$.
À droite : les entrées du modèle sont les descripteurs représentés dans l'espace des composantes principales, $PRECISION_{moyenne} = 99,50\%$.

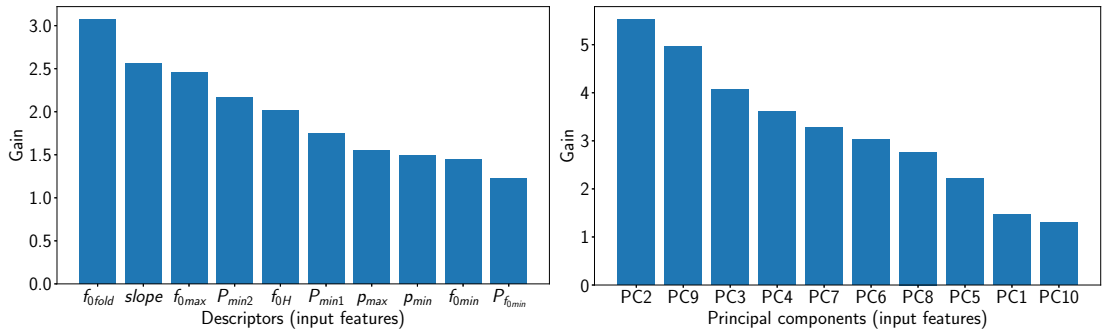

FIGURE 2.5 – Gain de chaque caractéristique après entraînement du classifieur.
À gauche : Descripteurs bruts.
À droite : Descripteurs représentés dans l'espace des composantes principales.

on sait que deux trompettes maniées par deux joueurs virtuels différents peuvent produire les mêmes bornes sur leur tracé de f_0.

En examinant les descripteurs dans l'espace des composantes principales, on remarque que les axes les plus importants pour la classification ne correspondent pas aux premières composantes principales. Au lieu de cela, le deuxième, puis le neuvième axe semblent les plus importants. C'est pour cette raison que nous avons choisi de ne pas réduire la dimensionnalité de l'espace des descripteurs en utilisant l'ACP, mais de conserver l'ensemble des axes principaux. En examinant les biplots de ces axes dans la figure 2.6, on observe la présence de clusters par instrument proches d'être séparables. Même si le pourcentage de variance du neuvième axe semble négligeable, nous avons vérifié que ces résultats sont stables sur tous les splits de validation du

schéma de validation croisée.

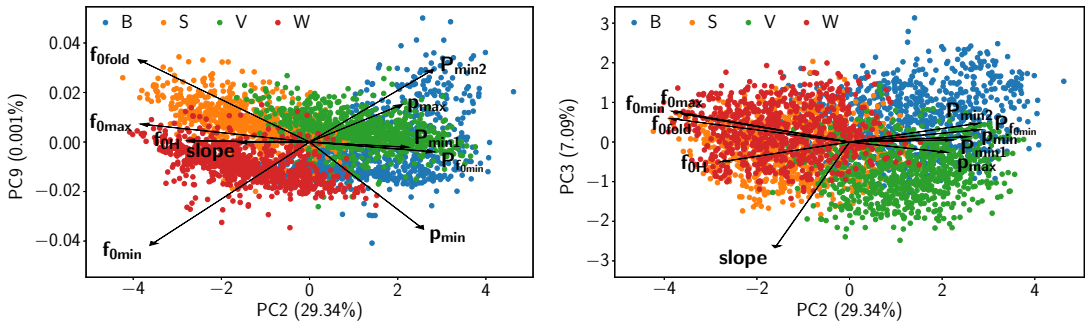

FIGURE 2.6 – Projections sur les axes de l'analyse en composantes principales possédant les gains les plus élevés lors de la classification avec XGBoost. Le pourcentage de variance expliquée par les axes est indiqué entre parenthèses. Les flèches dans le biplot représentent les descripteurs. La direction de chaque flèche indique les valeurs croissantes de ce descripteur, tandis que la longueur de la flèche représente la contribution du descripteur à la variation globale dans le jeu de données.

Cette étude démontre que l'ensemble des descripteurs de performance (issus de l'espace de joueurs virtuels considéré) permet d'identifier les trompettes avec une erreur très faible. Par conséquent, nous pouvons maintenant essayer de générer ces descripteurs sans utiliser le modèle physique classique, comme expliqué dans la figure 2.3. Cette approche permettra de créer un outil rapide et facile à utiliser pour les concepteurs de trompettes.

2.4 Prédiction des descripteurs sur des trompettes virtuelles

Maintenant que nous avons montré que les descripteurs de performance extraits du diagramme de bifurcation permettent de distinguer efficacement les trompettes dans la section 2.3, nous allons les générer à l'aide d'un modèle d'apprentissage automatique. Dans cette section, nous décrivons les modèles d'apprentissage utilisés pour prédire les descripteurs de performance extraits du diagramme de bifurcation de la trompette. Tout d'abord, nous allons expliquer dans la section 2.4.1, comment nous avons construit le jeu de données. Ensuite, nous comparerons dans la section 2.4.2, différents modèles d'apprentissage automatique pour prédire les descripteurs.

2.4.1 Construction du jeu de données

Comme expliqué dans la section 2.2, l'objectif principal de ce travail est de remplacer les calculs numériques de continuation par un modèle d'apprentissage automatique capable de prédire les 10 descripteurs décrits dans le tableau 2.1 pour une trompette

spécifique identifiée par son impédance d'entrée. Cependant, la modélisation du musicien est encore mal comprise et les paramètres des lèvres sont difficiles à mesurer. Nous avons donc décidé de prédire les valeurs des descripteurs de performance sur la base d'un ensemble de joueurs virtuels de référence. Notre objectif est de remplacer chacun de ces joueurs virtuels de référence, caractérisé par les paramètres de ses lèvres, par un modèle d'apprentissage capable de prédire l'un des 10 descripteurs du diagramme de bifurcation en fonction des paramètres modaux d'une trompette (les C_i et s_i.). Pour y parvenir, nous devons générer, pour chaque joueur virtuel que l'on souhaite remplacer, un jeu de données comprenant des descripteurs de performance pour un nombre suffisamment grand d'instruments virtuels. En outre, le modèle doit être suffisamment précis et fiable pour capturer l'impact de légères différences de conception sur la valeur des descripteurs. Par conséquent, les paramètres des instruments du jeu de données généré doivent, pour la plupart, appartenir à l'espace des paramètres des instruments réels, afin de garantir que le modèle fonctionne avec précision dans cet espace.

Pour s'en assurer, les paramètres modaux de neuf trompettes commerciales de différents fabricants sont extraits avec précision à l'aide d'une méthode nommée ESPRIT (Roy et al. 1989). Ces paramètres modaux fixent des valeurs limites pour chacun des $N = 11$ modes définissant ce qu'on appelle l'impédance d'entrée des instruments. Cette dernière, qui se décompose donc en 11 modes (d'où le nom paramètres modaux) complexes caractérisés par leurs pôles complexes s_i et leurs résidus C_i, définit un espace de conception de dimension $44 = 11$ (modes) $\times 2$ (pôle et résidu) $\times$ 2 (parties réelle et imaginaire). Ensuite, un total de 200 trompettes virtuelles est généré en échantillonnant au hasard ces 44 paramètres modaux à partir d'une distribution uniforme dans les limites fournies par les mesures des neuf trompettes commerciales.

La figure 2.7 représente les impédances correspondantes des 200 trompettes virtuelles. Il s'agit de ce qui caractérise comment chaque trompette réagit à la vibration des lèvres du musicien. On y voit clairement que ces impédances sont constituées de 11 pics. C'est de ces 11 pics que sont extraits les paramètres modaux des instruments. Dans cette étude, nous avons décidé d'uniquement étudier la note Si♭4, dont la fréquence fondamentale (470 Hz) se situe entre les quatrième et cinquième pics d'impédance. Pour fournir un examen détaillé des impédances des instruments virtuels autour de cette fréquence fondamentale, un zoom autour des quatrième et cinquième pics est présenté dans la figure 2.8.

Comme indiqué au début du chapitre, l'un des principaux avantages de l'apprentissage automatique est son faible coût de calcul (en ce qui concerne les modèles que l'on considère), ce qui permet de prendre en compte plusieurs joueurs virtuels pour chaque instrument. Comme dans la section précédente, chaque joueur virtuel est défini par quatre paramètres (Q_l, μ_l, y_0 et b). Cette fois-ci, nous générons un ensemble de 60 joueurs virtuels de référence. Les paramètres Q_l, μ_l, y_0 et b sont tirés, de l'espace limité par les valeurs correspondant à ±10% des valeurs des paramètres décrits dans (Fréour, Guillot, Masuda, Tominaga et al. 2020), e utilisant la méthode d'échantillonnage par hypercube latin. Comme le montre la figure 2.9, l'objectif est

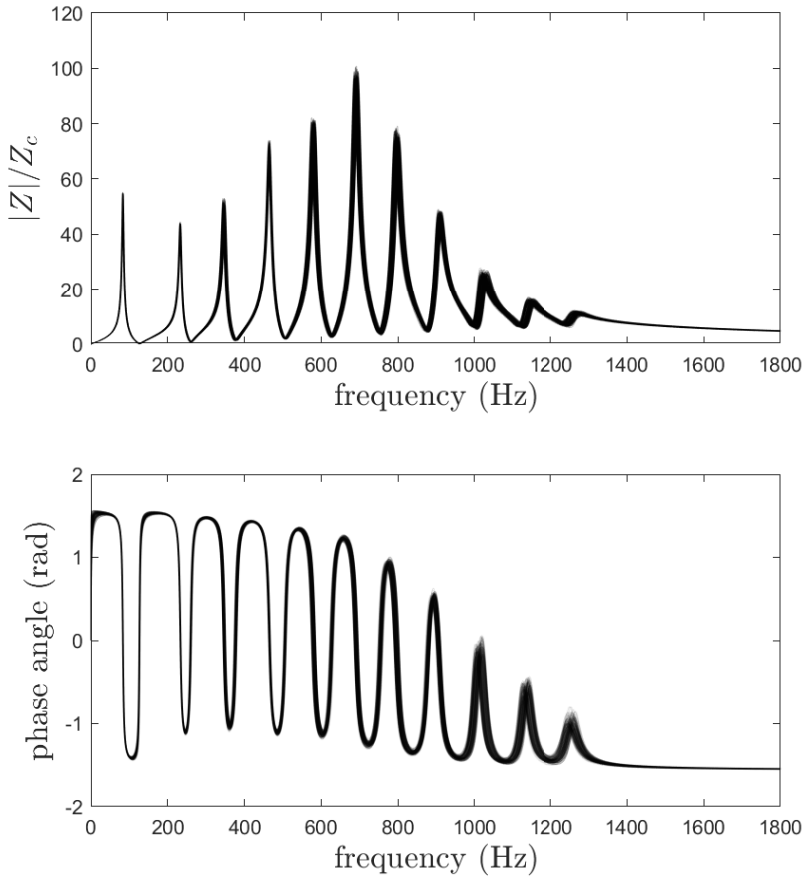

FIGURE 2.7 – Amplitude normalisée et angle de phase des impédances des 200 instruments virtuels générés pour entraîner les modèles d'apprentissage. Même si les courbes semblent très similaires, cela est suffisant pour obtenir des instruments sensiblement différents.

ici d'apprendre une grille de joueurs virtuels de référence bien répartis dans l'espace. Néanmoins, comme mentionné dans la section précédente, les limites de l'espace des joueurs virtuels sont quelque peu arbitraires ; nous supposons que les variations de ±10% représentent des attentes raisonnables pour les différences entre des joueurs humains ou des variations au sein d'un groupe de joueurs d'un niveau de jeu similaire. De plus, il faut également garder à l'esprit que, puisque nous allons entraîner un modèle d'apprentissage automatique par joueur virtuel, le choix de l'intervalle de dispersion par rapport aux valeurs de base des paramètres des lèvres devrait avoir un effet limité sur la performance de la prédiction.

Au total, l'ensemble d'apprentissage comprend 12 000 diagrammes de bifurcation pour la note Si♭4, chacun produisant un ensemble de 10 descripteurs. Ces diagrammes sont calculés à l'aide d'une routine Matlab automatisée et du code source de MANLAB.

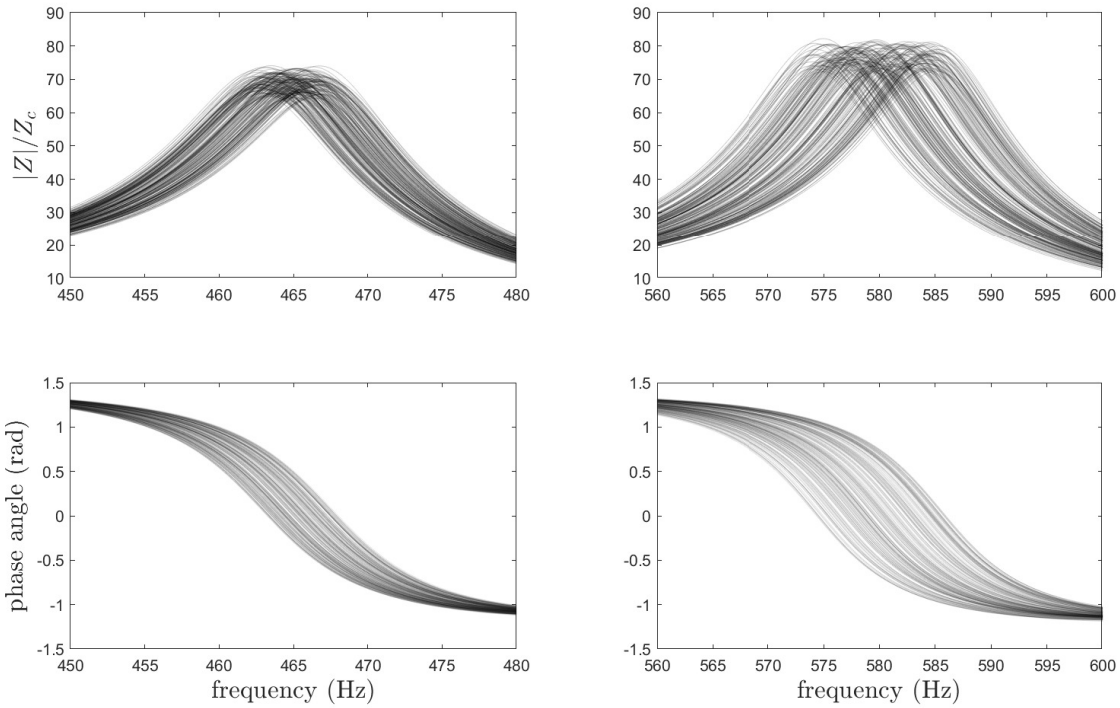

FIGURE 2.8 – Zoom autour du quatrième (à gauche) et du cinquième (à droite) pic des impédances des 200 instruments virtuels. Ici, on voit qu'il y a beaucoup moins de variation sur le quatrième pic que sur le cinquième, car sa fréquence est plus proche de la fréquence fondamentale de la note jouée.

Étant donné qu'il faut environ 3 minutes pour calculer un diagramme de bifurcation sur l'ordinateur utilisé pour les calculs, la génération de l'ensemble du jeu de données a nécessité environ 600 heures (équivalant à 25 jours) de temps de calcul.

2.4.2 Étude comparative d'algorithmes d'apprentissage automatique

Nous avons comparé plusieurs algorithmes d'apprentissage pour prédire les descripteurs des diagrammes de bifurcation en utilisant le jeu de données généré dans la section 2.4.1. En raison de la taille relativement petite du jeu de données, nous nous sommes concentrés sur les algorithmes d'apprentissage automatique connus pour leurs performances sur les petits jeux de données. Le problème posé est une tâche de régression, dont l'objectif est de prédire chaque descripteur à partir des paramètres modaux de la trompette pour chaque paire {musicien-descripteur} donnée. Nous avons comparé plusieurs approches possibles en régression, notamment des approches linéaires telles que la régression aux moindres carrés, le LASSO résolu par la méthode du LARS (EFRON et al. 2004) et la régression Ridge (HOERL et al. 1970),

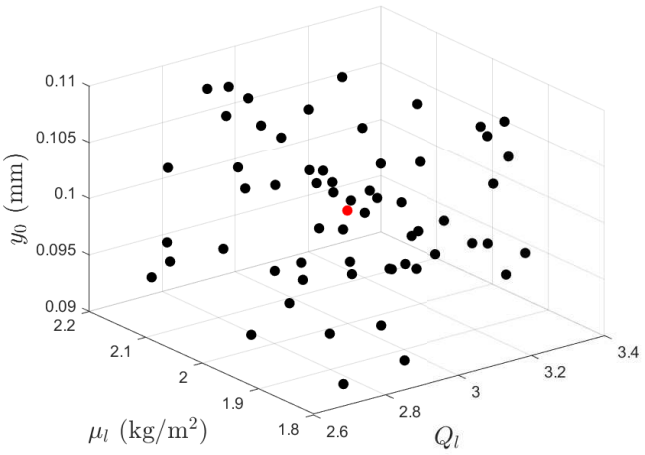

FIGURE 2.9 – Paramètres des lèvres des 60 musiciens virtuels de référence tirés via échantillonnage par hypercube latin. Le point rouge représente les paramètres des lèvres du musicien de (FRÉOUR, GUILLOT, MASUDA, TOMINAGA et al. 2020) ($Q_l = 3$, $\mu_l = 2$ kg/m^2, $y_0 = 0.1$ mm).

ainsi que les machine à vecteurs de support pour la régression (SVR) (VAPNIK 2013) avec différents noyaux, et une approche de type plus proches voisins (ALTMAN 1992). En outre, nous avons inclus un algorithme naïf qui prédit une valeur fixe pour chaque descripteur, fixée à la moyenne du descripteur dans le jeu de données d'entraînement. Nous suivons la procédure classique d'entraînement d'un modèle d'apprentissage, pour chaque paire musicien-descripteur. Tout d'abord, nous normalisons chacune des caractéristiques : les paramètres modaux des trompettes. Ensuite, nous divisons le jeu de données en un jeu d'entraînement (75% du jeu de données) et un jeu de test (25% du jeu de données). Pour chaque algorithme, nous utilisons une procédure de validation croisée à cinq plis pour affiner ses hyperparamètres et évaluer ses performances. La mesure de performance choisie est l'erreur absolue moyenne en pourcentage (MAPE), calculée comme la moyenne des différences en valeur absolue entre les valeurs prédites et les valeurs réelles des descripteurs, divisée par la valeur réelle des descripteurs. Le tableau 2.2 résume ces résultats. Malgré sa relative simplicité, le LASSO résolu par le LARS présente des performances supérieures à celles des autres méthodes pour prédire les valeurs des descripteurs des diagrammes de bifurcation, affichant l'erreur la plus faible pour chaque descripteur. En particulier, pour les descripteurs de fréquence ($f_{0\min}$, $f_{0\max}$, $f_{0\text{fold}}$ et $f_{0\text{H}}$), les erreurs de prédiction se situent entre 0.5 et 0.7 cent. Sachant que 100 cents représente un demi-ton (l'écart de hauteur entre deux touches consécutives sur un piano) et qu'une oreille, même entrainée, ne peut pas distinguer des différences de hauteur de moins de cinq cents (LOEFFLER 2006), ces résultats sont très satisfaisants. Nous allons présenter dans la section suivante, le problème d'optimisation résolu par le LASSO-LARS.

	$P_{\text{min}1}$ (%)	$P_{\text{min}2}$ (%)	p_{min} (%)	p_{max} (%)	$slope$ (%)
Naïf	5.75 ± 0.01	4.75 ± 0.01	4.24 ± 0.01	1.56 ± 0.01	1.38 ± 0.01
LASSO-LARS	**0.57 ± 0.02**	**0.41 ± 0.03**	**0.76 ± 0.03**	**0.48 ± 0.02**	**0.8 ± 0.05**
Régression linéaire	0.72 ± 0.06	0.67 ± 0.07	1.09 ± 0.07	0.66 ± 0.05	1.04 ± 0.07
SVR (linéaire)	0.71 ± 0.07	0.65 ± 0.05	1.1 ± 0.08	0.67 ± 0.05	1.06 ± 0.07
Moindres carrés	0.72 ± 0.06	0.67 ± 0.07	1.08 ± 0.07	0.66 ± 0.05	1.04 ± 0.07
SVR (non linéaire)	1.21 ± 0.08	1.17 ± 0.06	1.43 ± 0.07	0.7 ± 0.04	0.98 ± 0.04
Plus proches voisins	4.62 ± 0.15	3.71 ± 0.12	3.49 ± 0.12	1.29 ± 0.04	1.13 ± 0.04
	$f_{0\text{min}}$ (%)	$f_{0\text{max}}$ (%)	$P_{f_{0\text{min}}}$ (%)	$f_{0\text{fold}}$ (%)	$f_{0\text{H}}$ (%)
Naïf	0.26 ± 0.001	0.27 ± 0.001	3.81 ± 0.01	0.26 ± 0.001	0.25 ± 0.001
LASSO-LARS	**0.04 ± 0.002**	**0.04 ± 0.003**	**0.59 ± 0.03**	**0.03 ± 0.002**	**0.04 ± 0.002**
Moindres carrés	0.05 ± 0.003	0.05 ± 0.004	0.85 ± 0.06	0.05 ± 0.003	0.06 ± 0.01
SVR (linéaire)	0.05 ± 0.005	0.06 ± 0.005	0.86 ± 0.06	0.05 ± 0.004	0.06 ± 0.01
Régression Ridge	0.05 ± 0.003	0.05 ± 0.004	0.84 ± 0.06	0.05 ± 0.003	0.06 ± 0.01
SVR (non linéaire)	0.05 ± 0.003	0.06 ± 0.004	1.1 ± 0.06	0.05 ± 0.003	0.07 ± 0.01
Plus proches voisins	0.2 ± 0.01	0.2 ± 0.01	3.12 ± 0.1	0.2 ± 0.01	0.19 ± 0.01

TABLEAU 2.2 – Erreur moyenne absolue en pourcentage (MAPE) et écart-type de l'erreur absolue en pourcentage (StdAPE) dans la prédiction des valeurs des descripteurs à l'aide de divers algorithmes d'apprentissage automatique sur le jeu de test.

2.5 Zoom sur le LASSO-LARS

Dans cette section, nous décrivons l'approche LASSO-LARS, qui surpasse les autres algorithmes dans la prédiction des valeurs des descripteurs des diagrammes de bifurcation. La section 2.5.1 présente la formulation du problème d'optimisation et les résultats obtenus par le LASSO-LARS. Ensuite, la section 2.5.2 explique comment les résultats obtenus par le LASSO-LARS peuvent être interprétés pour identifier les paramètres modaux les plus importants pour la prédiction des descripteurs.

2.5.1 Formulation du problème d'optimisation

Considérons un descripteur d, parmi les 10 qui nous intéressent. La question est de savoir si, pour un musicien m donné, il est possible de prédire la valeur de ce descripteur à l'aide d'une combinaison linéaire $w^{m,d}$ des 44 paramètres modaux de la trompette et d'un biais $w_0^{m,d}$. Pour cela, il faut résoudre le problème inverse correspondant au modèle direct :

$$y^{m,d} \approx X w^{m,d} + w_0^{m,d},$$

où $y^{m,d} \in \mathbb{R}^{200}$ est le vecteur des 200 valeurs du descripteur d pour le musicien m, $X \in \mathbb{R}^{200\times 44}$ est la matrice contenant les 44 paramètres modaux de chacune des 200 trompettes. Afin de favoriser la parcimonie, nous cherchons une solution du problème

d'optimisation suivant pour chaque descripteur d et chaque musicien m :

$$\{\tilde{w}^{m,d},\ \tilde{w}_0^{m,d}\} = \underset{w^{m,d}\in\mathbb{R}^{44},\ w_0^{m,d}\in\mathbb{R}}{\operatorname{argmin}} \|y^{m,d} - X w^{m,d} - w_0^{m,d}\|_2^2 + \lambda \|w^{m,d}\|_1. \quad \text{(L2-L1)}$$

Ici, $\lambda \in \mathbb{R}^+$ est le paramètre de régularisation qui contrôle le niveau de parcimonie de la solution. Plus il est élevé, plus la solution sera parcimonieuse. Ce problème d'optimisation est connu sous le nom de problème du LASSO (TIBSHIRANI 1996).

Pour résoudre ce problème, nous utilisons l'algorithme du LARS (EFRON et al. 2004), un algorithme qui résout (L2-L1) efficacement pour un ensemble de valeurs λ bien choisies (appelé chemin de régularisation). Cet algorithme, couplé à la méthode de validation croisée, nous permet de trouver la valeur optimale (minimisant le critère d'erreur) de λ pour chaque couple descripteur/musicien.

Regardons de plus près les prédictions obtenues par cette approche. La figure 2.10 représente les squelettes des diagrammes de bifurcation générés à l'aide des valeurs des descripteurs prédits par le LARS en lignes pleines, et à l'aide du modèle physique (méthode de continuation) en lignes pointillées, pour trois joueurs et deux trompettes du jeu de test. Dans le tracé de $|p|$, on constate que le LARS prédit avec précision les diagrammes de bifurcation des trompettes qu'il n'a pas rencontrées lors de son apprentissage, avec une erreur bien inférieure à la variabilité inter-instrument et inter-joueur. Dans le tracé f_0, bien que le chevauchement entre les résultats de la prédiction et de la continuation ne soit pas aussi frappant que dans le tracé $|p|$, la différence relative entre les différentes courbes est relativement bien respectée, et la hauteur légèrement plus élevée pour une trompette (lignes bleues) est bien prédite par le modèle d'apprentissage. Notons que la différence maximale de valeur f_0 (le long des branches stables) entre les deux instruments est assez faible (environ sept cents), ce qui contribue à expliquer pourquoi la prédiction du modèle LARS peut sembler moins précise dans cette figure malgré les très bonnes performances de la prédiction rapportée dans le tableau 2.2.

Nous pouvons conclure que l'approche LASSO-LARS peut prédire les valeurs des descripteurs des diagrammes de bifurcation avec une précision suffisante. En outre, comme nous le verrons dans la section suivante, la régularisation en norme ℓ_1, qui favorise la parcimonie, va permettre d'obtenir une interprétation simple sur la capacité des paramètres modaux à prédire les descripteurs des diagrammes de bifurcation.

2.5.2 Interprétabilité du modèle

Nous analysons l'influence des paramètres modaux sur les descripteurs des diagrammes de bifurcation à l'aide du modèle LARS en examinant l'amplitude des coefficients $\tilde{w}^{m,d}$. Puisque ce modèle est un modèle linéaire, chaque paramètre modal est associé à un coefficient de régression. Les entrées étant normalisées, on peut regarder l'amplitude relative de ces coefficients, moyennée sur les 60 musiciens virtuels, pour déterminer l'importance de chaque paramètre modal dans la prédiction des descrip-

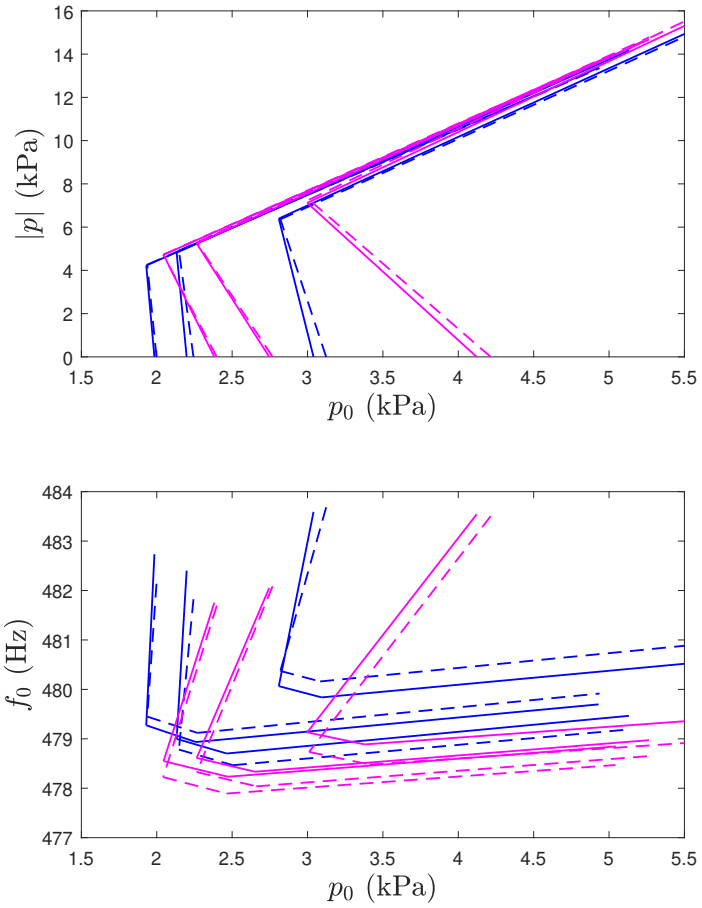

FIGURE 2.10 – Approximation affine par morceaux des diagrammes de bifurcation générés à l'aide des valeurs des descripteurs prédits par l'approche LASSO-LARS (ligne pleine) et à l'aide de la méthode de continuation (ligne pointillée) pour deux trompettes (bleu et violet) et trois joueurs virtuels.

teurs. Ces amplitudes moyennes sont calculées de la façon suivante pour chaque descripteur d :

$$\tilde{w}_i^{Moyenne,d} = \frac{1}{60}\sum_{m=1}^{60}\frac{|\tilde{w}_i^{m,d}|}{\|\tilde{w}^{m,d}\|_1}, \quad i \in [\![1,44]\!].$$

La figure 2.11 illustre l'importance normalisée des paramètres modaux dans la prédiction du descripteur $P_{\min 1}$, moyennée sur tous les joueurs virtuels. On remarque que les paramètres modaux associés aux modes acoustiques 4 et 5 contribuent à plus de 60 % de la prédiction de $P_{\min 1}$. Cela est tout à fait cohérent avec la physique sous-jacente de la production du son. En effet, comme expliqué précédemment, la fréquence fondamentale de la note jouée (Si♭4) se situe entre les quatrième et

cinquième pics d'impédance. Il est donc attendu que ces deux modes aient un impact significatif sur la prédiction de $P_{\min 1}$.

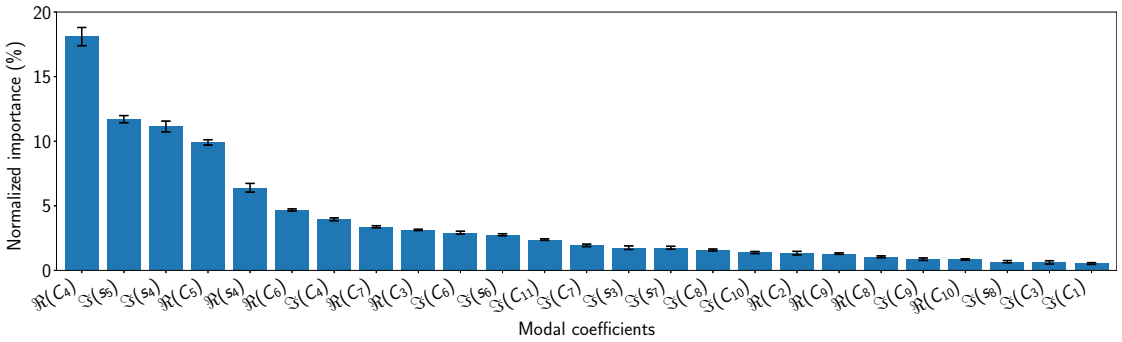

FIGURE 2.11 – Moyenne et écart-type de l'importance normalisée des 25 paramètres modaux les plus importants pour la prédiction du descripteur $P_{\min 1}$. $\Re(s_i)$ et $\Im(s_i)$ sont les parties réelle et imaginaire du pôle de la $i^{\text{ième}}$ fréquence de résonance de l'impédance acoustique de l'instrument. $\Re(C_i)$ et $\Im(C_i)$ sont les parties réelle et imaginaire du résidu de la $i^{\text{ième}}$ fréquence de résonance de l'impédance acoustique de l'instrument.

2.6 Outil pour les concepteurs de trompettes

Pour faciliter l'utilisation de ce procédé par les concepteurs de trompettes, j'ai développé une plateforme web permettant de prédire les valeurs des 10 descripteurs pour nos 60 musiciens virtuels de référence à partir de paramètres modaux donnés en entrée et de les afficher d'une manière pratique et lisible.

En pratique, les utilisateurs peuvent saisir des paramètres modaux pour une ou plusieurs trompettes, et l'outil renvoie les valeurs prédites des descripteurs pour tous les musiciens virtuels sous la forme de la figure 2.12. Cette figure donne un aperçu des caractéristiques des trois instruments du point de vue des descripteurs de performance. La représentation choisie permet d'avoir une vision globale de la réponse de chaque instrument à notre groupe de musiciens virtuels de référence. Par exemple, il existe des différences au niveau de l'intonation (fréquences de jeu $f_{0\min}$ et $f_{0\max}$) entre les trois trompettes. Nous observons également une valeur plus faible du descripteur $slope$ (la sensibilité à la variation de pression dans la bouche du musicien) pour la trompette 3 par rapport aux autres. De plus, cela met en lumière la sensibilité de l'instrument aux différents musiciens virtuels. Par exemple, la trompette 1 semble moins sensible aux paramètres du musicien en termes de pression seuil $P_{\min 1}$ que les trompettes 2 et 3, et la variation de la fréquence de jeu à ce niveau de pression ($f_{0\mathrm{H}}$) en fonction du musicien est plus importante sur la trompette 2 que sur les autres. Ce dernier aspect est particulièrement intéressant pour la comparaison des instruments et pour prédire la qualité perçue. De plus, notre modèle a l'avantage d'effectuer des calculs très rapides des descripteurs pour plusieurs musiciens virtuels.



FIGURE 2.12 – Diagrammes en boîte à moustaches des prédictions des valeurs des descripteurs pour les 60 musiciens virtuels de référence sur trois instruments du jeu de test.

Enfin, les utilisateurs peuvent également entraîner un nouveau modèle en fournissant un nouveau jeu de données qui associe des paramètres modaux à des descripteurs pour un groupe de musiciens de référence (virtuel ou réel). En ce qui concerne les temps de calcul, le code est écrit en Python et fait appel aux librairies Scikit-learn (PEDREGOSA et al. 2011) (pour le LASSO-LARS) et Streamlit (pour la plateforme web), et permet un calcul presque en temps réel des descripteurs. L'entraînement effectué dans ce projet n'a duré que deux minutes et la prédiction moins d'une seconde sur un ordinateur portable classique, alors que le calcul de la continuation numérique prend environ 3 minutes par diagramme de bifurcation. Une vidéo de démonstration de l'outil est disponible à cette adresse : https ://youtu.be/0Lhyj1tpmI0.

2.7 Conclusions

Dans ce chapitre, nous avons proposé une approche permettant aux concepteurs de trompettes d'accéder aux résultats des calculs de dynamique non linéaire. Plus précisément, en utilisant le modèle classique qu'est le LASSO-LARS, nous avons montré qu'il était possible de substituer l'approche physique traditionnelle qui nécessite du temps et un savoir-faire par un modèle classique, rapide et interprétable pour obtenir les descripteurs de performance de jeu d'une trompette. Ainsi, un concepteur de trompette peut directement obtenir ces valeurs sans avoir besoin d'une expertise

en dynamique non linéaire.

Nous avons comparé, de manière non exhaustive, plusieurs approches classiques d'apprentissage automatique. Ce chapitre montre que le modèle le plus précis pour réaliser cette tâche est obtenu par la méthode LASSO-LARS. Pour obtenir le jeu de données nécessaire à l'entraînement des modèles (12 000 fois 10 descripteurs), nous avons tiré parti de l'expérience acquise ces dernières années dans l'analyse de la dynamique non linéaire des modèles physiques de trompette à l'aide de la continuation numérique. Cette approche permet de construire des diagrammes de bifurcation détaillés qui tiennent compte des caractéristiques acoustiques de l'instrument (via ses paramètres modaux) et de la manière dont il est joué (via les valeurs des paramètres des lèvres du musicien). La construction du jeu de données est l'étape la plus longue de notre approche puisqu'elle implique à la fois le modèle physique et des analyses complexes de son fonctionnement. Cela correspond à environ 600 heures de calcul.

D'un point de vue pratique, la solution proposée dans l'article d'une plateforme web permettant la création et l'interrogation du modèle de substitution dans un contexte de conception de trompettes répond à deux difficultés : d'une part, les codes de calcul en dynamique non linéaire nécessitent un savoir-faire et des connaissances scientifiques que tous les concepteurs d'instruments de musique ne possèdent pas. D'autre part, une base de données de diagrammes de bifurcation, même importante, est par nature incomplète, puisque les concepteurs d'instruments veulent a priori concevoir des instruments qui n'existent pas encore. Ainsi, la solution présentée dans l'article permet aux concepteurs d'instruments de musique d'être autonomes dans leur exploration des performances de différentes trompettes. Ils sont donc maintenant capables d'instantanément comparer une géométrie de trompette à celle d'autres trompettes, pour différents types de musiciens.

Cette étude s'est concentrée sur une note de musique. L'analyse peut être étendue à d'autres notes de musique, il suffirait de générer les diagrammes de bifurcation correspondants. Cette nouvelle analyse pourrait permettre de mieux comprendre la dynamique de la trompette dans son ensemble. En outre, l'approfondissement de la modélisation du joueur virtuel pourrait fournir des indications précieuses sur les exigences nuancées de la conception d'une trompette. En effet, nous ne savons pas si un modèle linéaire est suffisant pour prédire les descripteurs des diagrammes de bifurcation à partir des paramètres modaux de la trompette et des paramètres des lèvres du joueur virtuel en même temps. Dans le cas contraire, il serait intéressant d'explorer comment des modèles non linéaires, tels que les réseaux de neurones informés par la physique (PINNs) (Raissi et al. 2019), DeepONet (Lu et al. 2019) ou les Fourrier Neural Operator (FNO) (Z. Li et al. 2023), pourraient être utilisé pour prédire les descripteurs. Toutefois, cette étape vers un modèle de substitution plus complet dépassait le cadre du projet. En ce qui concerne les trompettes elles-mêmes, l'approche proposée dans ce travail pourrait être étendue à des instruments dont les paramètres modaux dépassent les limites des instruments commerciaux que nous avons considérés. Une telle extension permettrait aux concepteurs de trompettes d'explorer le comportement de l'instrument au-delà de la gamme traditionnelle des paramètres modaux et de les comparer aux instruments existants.

3 SEA : Algorithme exploratoire pour le modèle parcimonieux linéaire

Sommaire

3.1 Introduction 60
3.1.1 Problème de reconstruction de supports parcimonieux 61
3.1.2 Estimateur direct (STE) 61
3.1.3 Contribution 62
3.2 Reformulation du problème de reconstruction de supports parcimonieux 63
3.3 Application de l'estimateur direct : l'algorithme SEA 65
3.3.1 Principe de l'algorithme SEA 65
3.3.2 Comparaison à l'état de l'art 67
3.4 Complexité de l'exploration 70
3.4.1 Implémentation efficace de SEA 70
3.4.2 Mesure de la complexité de l'exploration 71
3.5 Mise en évidence de l'efficacité de l'exploration 73
3.6 Conclusions 74

Ce chapitre est consacré à la présentation de l'algorithme SEA (Support Exploration Algorithm), un algorithme exploratoire utilisant un estimateur direct (STE) pour la reconstruction de supports parcimonieux de modèles linéaires. Il est question ici d'expliquer le raisonnement qui a conduit à la construction de l'algorithme SEA, de décrire son fonctionnement et de discuter de sa complexité tout en le comparant à l'état de l'art. La preuve théorique de son fonctionnement est présentée dans le chapitre 4. Enfin, une validation expérimentale de son efficacité est présentée dans le chapitre 5. Ce chapitre reprend une partie du contenu de (MOHAMED, MALGOUYRES et al. 2024).

3.1 Introduction

Comme nous l'avons présenté dans les généralités, au chapitre 1, les problèmes d'optimisation qui nous intéressent principalement sont les problèmes de reconstruction avec des contraintes de parcimonie. Nous voulons étudier une nouvelle approche

de résolution de ces problèmes utilisant un estimateur direct (STE), analogue à celui utilisé en quantification ou en binarisation.

3.1.1 Problème de reconstruction de supports parcimonieux

Nous nous intéressons à la reconstruction de signaux parcimonieux dans un cadre linéaire. Il s'agit de résoudre le problème décrit dans la section 1.2.1 du chapitre 1, que l'on rappelle ici.

Soit $x^* \in \mathbb{R}^n$ un signal inconnu, de support $S^* = \text{supp}(x^*)$ inconnu, k-parcimonieux (c'est-à-dire que $\|x^*\|_0 = |S^*| = k \in [\![1, n]\!]$). Soit $A \in \mathbb{R}^{m\times n}$ une matrice connue et

$$y = Ax^* + e \in \mathbb{R}^m$$

une observation linéaire de x^*, contaminée par un bruit additif $e \in \mathbb{R}^m$. Le problème est défini comme suit.

Définition 3.1.1. (*Problème de reconstruction de supports parcimonieux*) Le problème de reconstruction de supports, aussi appelé sélection de modèle ou sélection de variables, consiste à retrouver un support S de cardinalité inférieure ou égale à k tel que $S \subseteq S^*$ à partir de y et A.
On dit qu'un algorithme reconstruit S^* lorsqu'il parvient à retrouver un tel support.

En utilisant le critère de moindres carrés

$$F(x) = \frac{1}{2}\|Ax - y\|_2^2, \quad \forall x \in \mathbb{R}^n,$$

le problème de reconstruction de supports parcimonieux se formule comme suit :

$$\underset{x\in\mathbb{R}^n,\ \|x\|_0\leq k}{\text{Minimiser}}\ F(x). \tag{SPARSE}$$

Comme nous l'avons vu dans le chapitre 1, ce problème est NP-difficile (Davis et al. 1997) en raison de la non-différentiabilité de la norme ℓ^0. Il existe trois grandes familles d'algorithmes pour rechercher une solution à ce problème d'optimisation : les algorithmes de relaxation, les algorithmes gloutons et les algorithmes combinatoires (voir la section 1.3 du chapitre 1 pour plus de détails).

3.1.2 Estimateur direct (STE)

Comme nous l'avons vu dans la section 1.5 du chapitre 1, l'estimateur direct (STE de l'anglais Straight-Through Estimator) est une technique d'optimisation qui permet de résoudre des problèmes d'optimisation non différentiables. Il est utilisé en quantification et binarisation pour l'entraînement des réseaux de neurones profonds. Nous rappelons qu'il s'agit d'une méthode qui consiste à remplacer le gradient d'une fonction de coût non différentiable par celui d'une fonction de coût différentiable,

lors de l'étape de rétropropagation du gradient. Par exemple, minimiser $\bar{F}(H_q(\mathscr{W}))$ à la place de $\bar{F}(w)$ où H_q est un opérateur de quantification et $\bar{F}$ la fonction de coût que l'on souhaite minimiser, peut s'illustrer à partir du graphe de calcul de la figure 3.1, où $\mathscr{W}$ est un poids flottant, w est un poids quantifié et $l = \bar{F}(w)$.

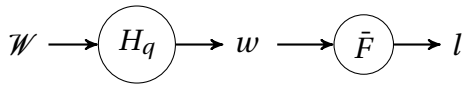

FIGURE 3.1 – Graphe de calcul de $\bar{F}(H_q(\mathscr{W}))$.

En prenant un pas de gradient $\eta \in \mathbb{R}_+^*$, on peut alors mettre à jour $\mathscr{W}$:

$$\mathscr{W} \leftarrow \mathscr{W} - \eta \left.\frac{\partial \bar{F}}{\partial \mathscr{W}}\right|_{\mathscr{W}} \tag{3.1}$$

La motivation, comme expliqué dans les travaux de (G. E. HINTON 2012 ; BENGIO et al. 2013), est que, puisque $\left.\frac{\partial H_q}{\partial \mathscr{W}}\right|_{\mathscr{W}}$ est soit non défini, soit égal à 0, il est impossible d'utiliser la règle de chaîne $\left.\frac{\partial \bar{F}\circ H_q}{\partial \mathscr{W}}\right|_{\mathscr{W}} = \left.\frac{\partial \bar{F}}{\partial w}\right|_{w} \left.\frac{\partial H_q}{\partial \mathscr{W}}\right|_{\mathscr{W}}$ pour effectuer la rétropropagation du gradient. L'estimateur direct effectue une approximation grossière $\left.\frac{\partial \bar{F}\circ H_q}{\partial \mathscr{W}}\right|_{\mathscr{W}} \approx \left.\frac{\partial \bar{F}}{\partial w}\right|_{w}$ pour rétropropager le gradient à travers l'opérateur H_q.

3.1.3 Contribution

L'utilisation de l'estimateur direct permet d'atteindre des performances au niveau de l'état de l'art dans le cadre des réseaux de neurones quantifiés. Cependant, son fonctionnement est mal compris et il a été peu étudié en dehors du contexte de la quantification. Nous allons introduire dans ce chapitre un principe d'estimateur direct adapté à la parcimonie. Nous allons l'appliquer au problème bien connu de la reconstruction de supports parcimonieux afin de comparer son comportement, ses performances empiriques et ses garanties théoriques à celles de l'état de l'art. Tout cela nous a conduit à la création d'un nouvel algorithme nommé *Support Exploration Algorithm* (SEA). Cet algorithme utilise l'historique complet des gradients comme une heuristique pour sélectionner le prochain support à explorer.

Si on le compare à d'autres algorithmes de reconstruction de supports, notre algorithme est proche de la famille des algorithmes gloutons. La principale différence entre SEA et les algorithmes gloutons est l'introduction d'un vecteur dense (non parcimonieux) $\mathscr{X}^t \in \mathbb{R}^n$ qui évolue au cours des itérations et indique quel support doit être testé à l'itération t. On appelle ce vecteur $\mathscr{X}^t$ la *variable d'exploration de supports*. Il est l'analogue des poids flottants $\mathscr{W}$ mentionnés précédemment et utilisés par BinaryConnect qui implémente également un estimateur direct pour optimiser les poids binaires des réseaux de neurones (COURBARIAUX et al. 2015 ; HUBARA et al. 2016). Nous montrons que l'adaptation de l'estimateur direct pour la parcimonie permet un compromis entre exploration et exploitation différent de celui proposé par l'état

de l'art, car il favorise davantage l'exploration. Cela permet d'obtenir de meilleures performances que l'état de l'art sur des problèmes très difficiles, c'est-à-dire ceux pour lesquels la matrice A possède une cohérence élevée.

Le chapitre est organisé comme suit : Dans la section 3.2, nous reformulons le problème de reconstruction de supports parcimonieux en un problème d'optimisation pour lequel l'estimateur direct peut être utilisé. Dans la section 3.3, nous montrons comment l'estimateur direct a permis de construire l'algorithme SEA et comparons son fonctionnement à celui de l'état de l'art. Dans la section 3.4, nous discutons la complexité de SEA et analysons son coût computationnel. Enfin, dans la section 3.5, nous présentons une preuve expérimentale de l'efficacité de SEA à l'aide de diagrammes de phase, avant de conclure dans la section 3.6.

3.2 Reformulation du problème de reconstruction de supports parcimonieux

Nous commençons par reformuler le problème de reconstruction de supports parcimonieux en un problème d'optimisation pour lequel l'estimateur direct peut être utilisé. Considérons l'opérateur suivant :

$$H(\mathscr{X}) \in \underset{\substack{x \in \mathbb{R}^n \\ \mathrm{supp}(x) \subseteq \mathrm{largest}_k(\mathscr{X})}}{\mathrm{argmin}} F(x), \quad \forall \mathscr{X} \in \mathbb{R}^n, \tag{3.2}$$

où $\mathrm{largest}_k(\mathscr{X})$ est l'ensemble des indices des k plus grands coefficients de $\mathscr{X}$ en valeur absolue. Il s'agit de l'opérateur qui, à un vecteur dense $\mathscr{X}$, associe un vecteur k-parcimonieux x de support $\mathrm{supp}(x) = S \subseteq \mathrm{largest}_k(\mathscr{X})$ minimisant l'erreur $F(x) = \frac{1}{2}\|Ax - y\|_2^2$. En d'autres termes, $H(\mathscr{X})_{|S}$ est une estimation par moindres carrés issue du modèle direct $y = A_{|S}H(\mathscr{X})_{|S}$. Dans le cas où $A_{|S}$ est de rang plein, cette estimation est unique [1] et s'écrit de la façon suivante :

$$\begin{cases} H(\mathscr{X})_{|S} = A_{|S}^{\dagger} y, \\ H(\mathscr{X})_{|\overline{S}} = 0, \end{cases} \tag{3.3}$$

avec $A_{|S}^{\dagger} = (A_{|S}^T A_{|S})^{-1} A_{|S}^T$, la pseudo-inverse de Moore-Penrose de $A_{|S}$.

On reformule le problème de reconstruction de supports parcimonieux (SPARSE) comme suit :

$$\underset{\mathscr{X} \in \mathbb{R}^n}{\text{Minimiser}}\ F(H(\mathscr{X})). \tag{SPARSE-STE}$$

Si l'on se reporte au graphe de calcul de $F(H(\mathscr{X}))$ de la figure 3.2, on remarque deux choses. La première est que la reformulation déplace l'espace d'optimisation

1. Comme généralement $k \ll m$, c'est a priori le cas. Dans le cas contraire, on peut choisir arbitrairement des minima de F de support S par descente de gradient.

vers l'ensemble de tous les vecteurs de $\mathbb{R}^n$, au lieu de le restreindre aux vecteurs k-parcimonieux de $\mathbb{R}^n$. On peut se demander si les deux problèmes sont équivalents. C'est ce que nous allons démontrer dans la proposition 3.2.1. La seconde chose que l'on remarque est que ce nouveau graphe possède de nombreuses similarités avec le graphe de calcul des réseaux de neurones binaires de la figure 3.1. En effet, H est un opérateur non différentiable constant par morceaux, F est une fonction différentiable et $F(H(\mathscr{X}))$ est une fonction de coût que l'on souhaite minimiser. On pourra donc utiliser un estimateur direct pour minimiser $F(H(\mathscr{X}))$. C'est ce que nous allons faire dans la section suivante.

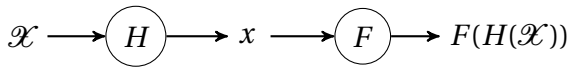

FIGURE 3.2 – Graphe de calcul de $F(H(\mathscr{X}))$.

Proposition 3.2.1. (*Équivalence entre* (SPARSE) *et* (SPARSE-STE)) Pour tout $m, n \in \mathbb{N}$, $k \in [\![1, n]\!]$, $A \in \mathbb{R}^{m\times n}$ et $y \in \mathbb{R}^m$. Le problème (SPARSE) est équivalent au problème (SPARSE-STE), c'est-à-dire :

1. pour toute solution $\mathscr{X}^* \in \operatorname{argmin}_{\mathscr{X}\in\mathbb{R}^n} F(H(\mathscr{X}))$ de (SPARSE-STE), $H(\mathscr{X}^*)$ est une solution de (SPARSE).
2. toute solution x' de (SPARSE) est aussi une solution de (SPARSE-STE).

Démonstration. **Preuve du premier point**
Prenons une solution $\mathscr{X}^* \in \operatorname{argmin}_{\mathscr{X}\in\mathbb{R}^n} F(H(\mathscr{X}))$ de (3.2). Par définition de H en (3.2), $H(\mathscr{X}^*)$ est k-parcimonieux. Maintenant, considérons $x \in \mathbb{R}^n$ k-parcimonieux, nous avons alors :

$$\begin{aligned} F(H(\mathscr{X}^*)) &\leq F(H(x)) && \text{(par hypothèse sur } \mathscr{X}^*) \\ &\leq F(x) && \text{(par définition de } H \text{ en (3.2))} \end{aligned}$$

Par conséquent, $H(\mathscr{X}^*)$ est solution de (SPARSE).
Preuve du second point
Prenons un minimiseur x' de (SPARSE) et $\mathscr{X} \in \mathbb{R}^n$. Par définition de H (3.2), $H(\mathscr{X})$ est k-parcimonieux. Comme x' est solution de (SPARSE), nous avons :

$$F(x') \leq F(H(\mathscr{X})). \tag{3.4}$$

De plus, puisque x' est k-parcimonieux, nous avons $\operatorname{supp}(x') \subseteq \operatorname{largest}_k(x')$. Ainsi, en utilisant de nouveau la définition de H, nous avons

$$F(H(x')) \leq F(x').$$

En combinant cette équation avec (3.4), nous obtenons $F(H(x')) \leq F(H(\mathscr{X}))$, pour tout $\mathscr{X} \in \mathbb{R}^n$, et concluons que x' est solution de (SPARSE-STE).

3.3 Application de l'estimateur direct : l'algorithme SEA

Maintenant que nous avons reformulé le problème de reconstruction de supports parcimonieux en un problème d'optimisation pour lequel l'estimateur direct peut être utilisé, nous montrons comment l'estimateur direct permet de construire l'algorithme SEA dans la section 3.3.1. Nous comparons ensuite son fonctionnement à celui de l'état de l'art dans la section 3.3.2.

3.3.1 Principe de l'algorithme SEA

Nous voulons minimiser $F(H(\mathscr{X}))$ sur $\mathbb{R}^n$ en utilisant la méthode de l'estimateur direct. Nous rappelons que, pour tout x et $\mathscr{X}$ appartenant à $\mathbb{R}^n$:

$$F(x) = \frac{1}{2}\|Ax - y\|_2^2 \quad \text{et} \quad H(\mathscr{X}) \in \underset{\substack{x \in \mathbb{R}^n \\ \text{supp}(x) \subseteq \text{largest}_k(\mathscr{X})}}{\text{argmin}} F(x).$$

Appliquer la méthode de l'estimateur direct comme défini dans la définition 1.5.1 du chapitre 1 revient à réaliser l'approximation :

$$\left.\frac{\partial F \circ H}{\partial \mathscr{X}}\right|_{\mathscr{X}} = \left.\frac{\partial F}{\partial x}\right|_{H(\mathscr{X})} \left.\frac{\partial H}{\partial \mathscr{X}}\right|_{\mathscr{X}} \approx \left.\frac{\partial F}{\partial x}\right|_{H(\mathscr{X})}. \tag{3.5}$$

Avec un pas de gradient $\eta \in \mathbb{R}_+^*$, cela conduit à une mise à jour de $\mathscr{X}$ analogue à celle de l'équation (3.1) :

$$\mathscr{X} \leftarrow \mathscr{X} - \eta \left.\frac{\partial F}{\partial x}\right|_{H(\mathscr{X})} = \mathscr{X} - \eta A^T(AH(\mathscr{X}) - y). \tag{3.6}$$

En répétant cette mise à jour, on obtient l'algorithme 1. En termes de pseudocode, l'algorithme semble proche d'autres algorithmes de l'état de l'art et partage des similitudes avec HTP (FOUCART 2011) et d'IHT (BLUMENSATH et Mike E. DAVIES 2009). Cependant, il se distingue par son comportement exploratoire lié à l'utilisation de l'estimateur direct. Une analyse plus approfondie de ces similitudes est donnée dans la section 3.3.2.

Chaque itération commence par le calcul de (3.2), où l'estimée parcimonieuse courante x^t est calculée en appliquant l'opérateur de parcimonie H à un vecteur dense $\mathscr{X}^t$. Ce calcul est effectué en deux étapes. D'abord, S^t est déterminé à la ligne 6 en extrayant les indices des k plus grands coefficients de $\mathscr{X}^t$ en valeur absolue. Ensuite, x^t est calculé à la ligne 7 en minimisant la fonction de coût F sur le support S^t. La rétropropagation du gradient utilise le principe du STE présenté dans l'équation (3.6) pour mettre à jour $\mathscr{X}$ à la ligne 8. Pour continuer à illustrer l'analogie avec BinaryConnect (COURBARIAUX et al. 2015), le vecteur dense $\mathscr{X}$ est l'analogue des poids de flottants, et $H(\mathscr{X})$ est l'analogue des poids quantifiés. À notre connaissance, il

Algorithme 1 Support Exploration Algorithm (SEA)

1 : **Entrées :** observation bruitée y, matrice A, parcimonie k, pas η
2 : **Sortie :** vecteur k-parcimonieux x
3 : Initialiser $\mathscr{X}^0$
4 : $t \leftarrow 0$
5 : **Répéter**
6 : $\quad S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
7 : $\quad \begin{cases} x^t{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^t} \\ x^t{}_{|S^t} & \leftarrow A^\dagger_{|S^t} y \end{cases}$
8 : $\quad \mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - \eta A^T(Ax^t - y)$
9 : $\quad t \leftarrow t+1$
10 : **Jusqu'à** ce que le critère d'arrêt soit `vrai`
11 : $t_{BEST} \leftarrow \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$
12 : **renvoyer** $x^{t_{BEST}}$

s'agit de la première utilisation de l'estimateur direct pour résoudre un problème de reconstruction de supports parcimonieux.

L'idée clé est que le support courant S^t est désigné à la ligne 6 par une variable dense $\mathscr{X}^t$ appelée *variable d'exploration du support*. Elle offre un mécanisme original pour explorer les supports d'une manière plus diversifiée que les algorithmes existants. La variable $\mathscr{X}^{t+1} = \mathscr{X}^0 - \eta \sum_{t'=0}^{t} A^T(Ax^{t'} - y)$ est une accumulation de gradients calculés à chaque estimée parcimonieuse. Cette accumulation est utilisée pour désigner le support de l'estimée parcimonieuse suivante x^{t+1}. Par conséquent, contrairement à d'autres algorithmes basés sur une descente, $\mathscr{X}^t$ n'est pas confiné au voisinage des vecteurs k-parcimonieux. Son évolution n'a pas pour but de faire diminuer la fonction objectif à chaque itération. Puisque l'algorithme explore les supports d'une manière qui permet à la fonctionnelle de parfois augmenter, la solution retenue est la meilleure rencontrée au fil des itérations (ligne 11). Des illustrations de ce phénomène sont données dans la figure 3.3 de la section suivante. On peut observer sur cette figure que le comportement de la fonction à optimiser au fil des itérations montre des variations importantes lorsqu'un nouveau support est exploré. Il s'agit d'une différence importante par rapport aux algorithmes de l'état de l'art, qui se traduit par une exploration accrue.

Une caractéristique importante de SEA est qu'il peut être utilisé comme post-traitement de la solution $\tilde{x}$ d'un autre algorithme. Pour ce faire, il suffit d'initialiser $\mathscr{X}^0$ à $\tilde{x}$. Dans ce cas, $S^0 = \text{supp}(\tilde{x})$ (ligne 6) et x^0 est au moins aussi bien que $\tilde{x}$ (ligne 7). Dans les expériences présentées dans la section 3.5 et au chapitre 5, nous étudions l'initialisation avec le résultat d'OMP (PATI et al. 1993), ELS (AXIOTIS et al. 2020 ; SHALEV-SHWARTZ et al. 2010) et une initialisation avec $\mathscr{X}^0 = 0$. Dans la suite du manuscrit, nous désignons naturellement ces trois versions par SEA_{OMP}, SEA_{ELS}

et SEA_0. Nous observons que l'initialisation avec ELS est généralement préférable, sauf pour les problèmes difficiles, lorsque les colonnes de A sont très cohérentes (voir section 5.2 du chapitre 5 pour plus de détails).

Enfin, comme souvent, il existe de nombreuses stratégies possibles pour concevoir le critère d'arrêt de la boucle « répéter » de l'algorithme 1. Il est clair qu'un critère plus permissif permet une plus grande exploration et de meilleurs résultats, au détriment du temps de calcul. Nous avons axé nos expériences sur l'illustration des avantages potentiels de SEA et, par conséquent, nous utilisons toujours un grand nombre fixe de passages dans la boucle « Répéter » de l'algorithme 1.

Le rôle du pas η est différent des cas usuels d'algorithme de descente. On remarque cela en considérant l'expression $\mathscr{X}^t = \mathscr{X}^0 - \eta \sum_{t'=0}^{t-1} A^T(Ax^{t'} - y)$ pour tout $t \geq 1$. Lorsque $\mathscr{X}^0 = 0$, η n'a aucun impact sur S^t, x^t, et donc sur la sortie $x^{t_{BEST}}$ de l'algorithme 1 car $\text{largest}_k(\mathscr{X}^t)$ ne dépend pas de η. Dans ce cas, en effet, l'ensemble de la trajectoire $(\mathscr{X}^t)_{t\in\mathbb{N}}$ est dilaté par $\eta > 0$ et la dilatation n'a aucun effet sur les supports sélectionnés S^t. Lorsque $\mathscr{X}^0 \neq 0$, la variable d'exploration de supports initiale est oubliée progressivement au fil des itérations. De plus, plus η augmente, plus elle est oubliée rapidement. Enfin, l'ensemble des propriétés de convergence de SEA seront étudiées dans le chapitre 4

3.3.2 Comparaison à l'état de l'art

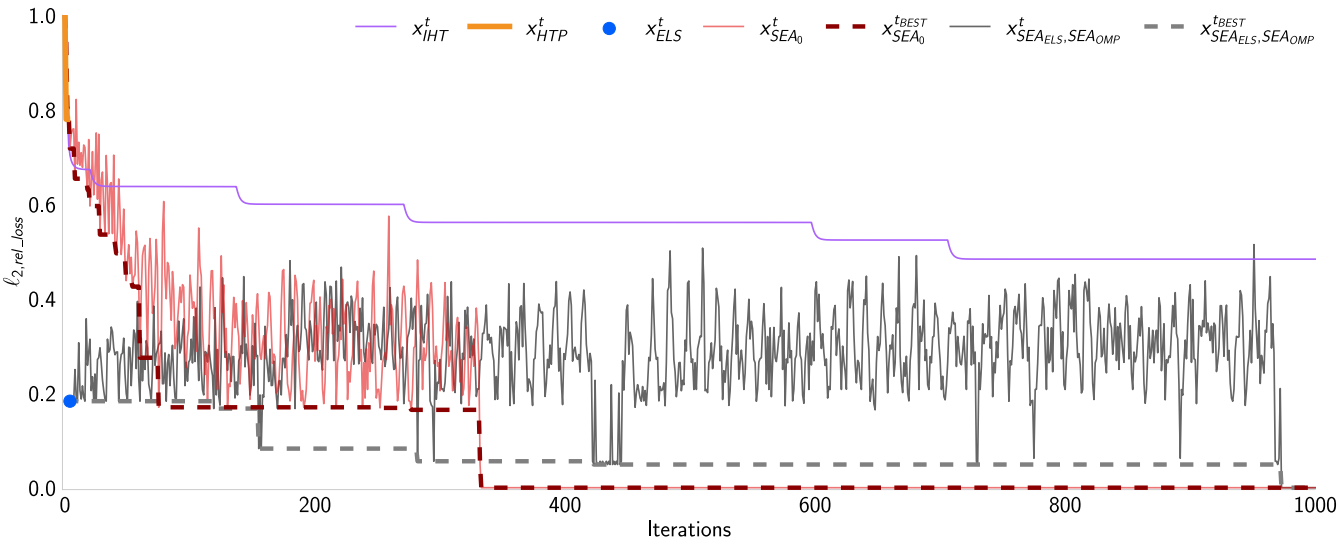

FIGURE 3.3 – Représentation de $\ell_{2,\text{rel_loss}}(x^t) = \|Ax^t - y\|_2 / \|y\|_2$ (lignes pleines) et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t)})$ (lignes pointillées) pour chaque itération de HTP, IHT, ELS et SEA initialisé à 0 (SEA_0) ou à la solution de OMP ou ELS (SEA_{OMP} ou SEA_{ELS}) sur un problème de déconvolution non bruité ($y = Ax^*$) avec $k = 20$, $n = m = 500$ et A est un opérateur de convolution. La description complète de l'expérience sera faite dans le chapitre 5.

Algorithme 2 SEA
(copie de l'algorithme 1)

Entrées :
observation bruitée y,
matrice A,
parcimonie k,
pas η
Sortie : vecteur k-parcimonieux x
Initialiser $\mathscr{X}^0$
$t \leftarrow 0$
Répéter
  $S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
  $\begin{cases} x^t{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^t} \\ x^t{}_{|S^t} & \leftarrow A^\dagger_{|S^t} y \end{cases}$
  $\mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t+1$
Jusqu'à ce que le critère d'arrêt soit `vrai`
$t_{BEST} \leftarrow \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$
renvoyer $x^{t_{BEST}}$

Algorithme 3 HTP
(FOUCART 2011)

Entrées :
observation bruitée y,
matrice A,
parcimonie k,
pas η
Sortie : vecteur k-parcimonieux x
Initialiser $\mathscr{X}^0$
$t \leftarrow 0$
Répéter
  $S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
  $\begin{cases} x^t{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^t} \\ x^t{}_{|S^t} & \leftarrow A^\dagger_{|S^t} y \end{cases}$
  $\mathscr{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t+1$
Jusqu'à ce que le critère d'arrêt soit `vrai`

renvoyer x^t

Nous allons maintenant comparer SEA à l'état de l'art, en nous concentrant sur les algorithmes gloutons, qui sont les plus similaires. En termes de pseudocode, SEA ressemble à Hard Thresholding Pursuit (HTP) (algorithme 3, (FOUCART 2011)) et, dans une moindre mesure, à Iterative Hard Thresholding (IHT) (algorithme 5, (BLUMENSATH et Mike E. DAVIES 2009)). Nous avons souligné les différences avec SEA (algorithmes 2 et 4) en rouge.

HTP et IHT sont des algorithmes de descente projetée qui alternent une étape de pas de gradient réalisé sur une estimée k-parcimonieuse x^t et une projection de la variable résultante $\mathscr{X}^t$ sur l'espace des vecteurs k-parcimonieux. Toute la différence avec SEA réside dans l'introduction de la variable d'exploration de supports $\mathscr{X}^t$ et dans son interaction avec le vecteur k-parcimonieux x^t. En effet, HTP et IHT effectuent un pas de gradient classique $\mathscr{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$ (où $\mathscr{X}$ désigne ici une variable intermédiaire, et non une variable d'exploration de supports) tandis que SEA utilise une mise à jour à l'aide de l'estimateur direct $\mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - \eta A^T(Ax^t - y)$ de la variable d'exploration de supports elle-même ($\mathscr{X}^t$) avec un gradient calculé en x^t. Par conséquent, le vecteur $\mathscr{X}^t$ dans HTP ou IHT est toujours exactement à un pas de gradient du vecteur k-parcimonieux x^t. Ils n'explorent donc pas beaucoup, contrairement à SEA. La variable d'exploration de supports $\mathscr{X}^t$ n'est pas censée minimiser l'objectif : elle accumule plutôt tous les gradients qui sont calculés en x^t.

Algorithme 4 SEA
(copie de l'algorithme 1)

Entrées :
observation bruitée y,
matrice A,
parcimonie k,
pas η
Sortie : vecteur k-parcimonieux x
Initialiser $\mathscr{X}^0$
$t \leftarrow 0$
Répéter
  $S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
  $\begin{cases} x^t{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^t} \\ x^t{}_{|S^t} & \leftarrow A^\dagger_{|S^t} y \end{cases}$
  $\mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t+1$
Jusqu'à ce que le critère d'arrêt soit `vrai`
$t_{BEST} \leftarrow \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$
renvoyer $x^{t_{BEST}}$

Algorithme 5 IHT
(Blumensath et Mike E. Davies 2009)

Entrées :
observation bruitée y,
matrice A,
parcimonie k,
pas η
Sortie : vecteur k-parcimonieux x
Initialiser $\mathscr{X}^0$
$t \leftarrow 0$
Répéter
  $S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
  $\begin{cases} x^t{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^t} \\ x^t{}_{|S^t} & \leftarrow \mathscr{X}^t{}_{|S^t} \end{cases}$
  $\mathscr{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t+1$
Jusqu'à ce que le critère d'arrêt soit `vrai`

renvoyer x^t

C'est là tout l'intérêt de l'estimateur direct. En particulier, $(\mathscr{X}^t)_{t\in\mathbb{N}}$ n'est pas restreint à l'espace des vecteurs de $\mathbb{R}^n$ à proximité des vecteurs k-parcimonieux. Elle peut ainsi explorer beaucoup plus que dans HTP et IHT. Cela explique pourquoi SEA possède un compromis exploration/exploitation différent et explore davantage.

Afin d'évaluer expérimentalement les différences entre SEA et les algorithmes de l'état de l'art en termes d'exploration, nous étudions un exemple simple de déconvolution sans bruit. Nous générons un vecteur aléatoire 20-parcimonieux x^* de taille $n = 500$. L'observation $y = Ax^*$, où A représente un opérateur de convolution, est ensuite calculée. Nous demandons enfin aux différents algorithmes de reconstruire x^* à partir de y et de A. Les variations de F au cours des itérations sont reportées pour toutes les versions de SEA, ainsi que pour ELS, HTP et IHT, dans la figure 3.3. Une description complète de l'expérience et une analyse détaillée des résultats seront fournies dans le chapitre 5. Ces premiers résultats montrent que les valeurs de F varient fortement y compris à la hausse, au cours du processus itératif de SEA, en raison de sa nature exploratoire. Néanmoins, l'algorithme conserve la meilleure solution $x^{t_{BEST}}$ rencontrée durant l'exploration. C'est à cela que l'on voit que SEA n'est pas basé sur un principe de descente comme IHT, HTP et les autres algorithmes couramment utilisés. De plus, si l'on s'attarde un peu plus sur la figure 3.3, on peut aussi remarquer que HTP s'arrête au bout de quelques itérations. Il s'agit du moment où le gradient est trop

petit pour que le support change entre deux itérations successives. En revanche, SEA continue d'accumuler des gradients de sorte que le support puisse rester inchangé pendant plusieurs itérations avant qu'un nouveau support ne soit exploré. Ainsi, SEA_0 trouve une meilleure approximation de S^* que ELS dans les 80 premières itérations et de réussir à reconstruire S^* malgré la cohérence élevée de A. Pour toutes les initialisations de SEA (dans cet exemple les initialisations SEA_{OMP} et SEA_{ELS} sont identiques), une fois que S^* est reconstruit, $x^t = x^*$. On trouve donc que $Ax^t - y = 0$, car nous sommes dans le cas non bruité. En utilisant la règle de mise à jour de $\mathscr{X}^t$ décrite dans la ligne 8 de algorithme 1, nous observons que $\mathscr{X}^t$ n'évolue plus, et qu'aucun nouveau support n'est exploré. IHT quant à lui surpasse HTP en explorant un peu plus, mais n'arrive pas à reconstruire S^*. Les performances médiocres de IHT et HTP sont dues à la cohérence élevée de A ($\mu = 0.97$), qui rend la reconstruction de S^* difficile.

3.4 Complexité de l'exploration

Avant de calculer la complexité de SEA, nous allons d'abord décrire une implémentation efficace.

3.4.1 Implémentation efficace de SEA

L'algorithme 1 de la section 3.3.1 est présenté d'une manière qui favorise la clarté et simplifie l'analyse théorique. En pratique, on peut remarquer que si le support S^t ne change pas (ligne 6), alors le vecteur k-parcimonieux x^t et le gradient $\eta A^T(Ax^t - y)$ ne changent pas non plus. Le pseudocode de l'algorithme 6 est une implémentation efficace, en termes de calcul, de l'algorithme 1 qui exploite cette observation. Les calculs les plus coûteux (l'estimation par moindres carrés à la ligne 10 et le gradient à la ligne 12) ne sont nécessaires que lorsque le support n'a jamais été exploré auparavant. De plus, le meilleur vecteur k-parcimonieux peut être mémorisé à la volée (ligne 16). Par conséquent, les opérations restantes, qui sont effectuées à chaque itération, ont un faible coût de calcul : extraction du support (ligne 8), recherche d'un support identique à un support précédent (ligne **??**) et mise à jour en utilisant l'estimateur direct (ligne 21). Cette version de SEA efficace en termes de calcul présente une complexité spatiale plus importante en raison de la mémorisation de tous les supports et gradients observés au cours des itérations. Cependant, ce nouveau besoin en mémoire est limité pour deux raisons. La première est que pour chaque support exploré, seuls deux vecteurs sont mémorisés, l'un d'entre eux étant k-parcimonieux. La seconde est que le nombre de supports explorés est généralement beaucoup plus faible que le nombre d'itérations. Par exemple, comme on peut le voir dans la figure 3.4, moins de 1000 vecteurs de taille 500 (dont 500 sont k-parcimonieux) sont stockés en moyenne au cours de 1000 itérations de SEA pour la résolution d'un problème de déconvolution. En outre, la résolution du système linéaire (sans contrainte) $A_{|S^t}^T A_{|S^t} x_{|S^t} = A_{|S^t}^T y$ peut également être effectuée efficacement. Bien que la pseudo-inverse soit une notation pratique à la ligne 10, la solution peut être obtenue plus efficacement, par exemple

en $\mathcal{O}(k^2 n)$ pour calculer $A_{|S^t}^T A_{|S^t}$ et $A_{|S^t}^T y$, et appliquer l'algorithme du gradient conjugué (SHEWCHUK et al. 1994). Cette complexité est dans le pire des cas. Dans la pratique, la solution est souvent obtenue plus rapidement.

Algorithme 6 Support Exploration Algorithm (SEA) : implémentation efficace

1 : **Entrées :** observation bruitée y, matrice A, parcimonie k, pas η
2 : **Sortie :** vecteur k-parcimonieux x
3 : Initialiser $\mathscr{X}^0$
4 : $F^{BEST} \leftarrow +\infty$
5 : $t \leftarrow 0$
6 : $\mathscr{S} \leftarrow \{\}$, $\boldsymbol{g} \leftarrow \{\}$
7 : **Répéter**
8 : $S \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
9 : **Si** $S \notin \mathscr{S}$ **alors**
▷ Calcule le vecteur k-parcimonieux et son gradient seulement pour les supports non explorés
10 : $\begin{cases} x^S{}_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S} \\ x^S{}_{|S} & \leftarrow A_{|S}^{\dagger} y \end{cases}$
11 : $loss^S \leftarrow \frac{1}{2}\|Ax^S - y\|_2^2$
12 : $g^S \leftarrow \eta A^T\left(Ax^S - y\right)$
13 : $\mathscr{S} \leftarrow \mathscr{S} \cup \{S\}$, $\boldsymbol{g} \leftarrow \boldsymbol{g} \cup \left\{g^S\right\}$ ▷ Mémorise le support et son gradient
14 : **Si** $loss^S < F^{BEST}$ **alors** ▷ Mémorise la meilleure solution
15 : $F^{BEST} \leftarrow loss^S$
16 : $x^{BEST} \leftarrow x^S$
17 : **Fin si**
18 : **Sinon**
19 : Récupérer g^S depuis $\boldsymbol{g}$
20 : **Fin si**
21 : $\mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - g^S$ ▷ Mise à jour de la variable d'exploration de supports
22 : $t \leftarrow t+1$
23 : **Jusqu'à** ce que le critère d'arrêt soit `vrai`
24 : **renvoyer** x^{BEST}

3.4.2 Mesure de la complexité de l'exploration

L'analyse de la complexité de SEA est basée sur deux faits. Premièrement, si le support S^t obtenu à la ligne 6 de l'algorithme 1 (respectivement S à la ligne 8 de l'algorithme 6) a déjà été exploré, alors le vecteur k-parcimonieux x^t (resp. x^S) et le gradient $\eta A^T(Ax^t - y)$ (resp. g^S) ont déjà été calculés. Ainsi, ces quantités étant mémorisées dans l'algorithme 6, le coût de l'itération est négligeable. Le coût global

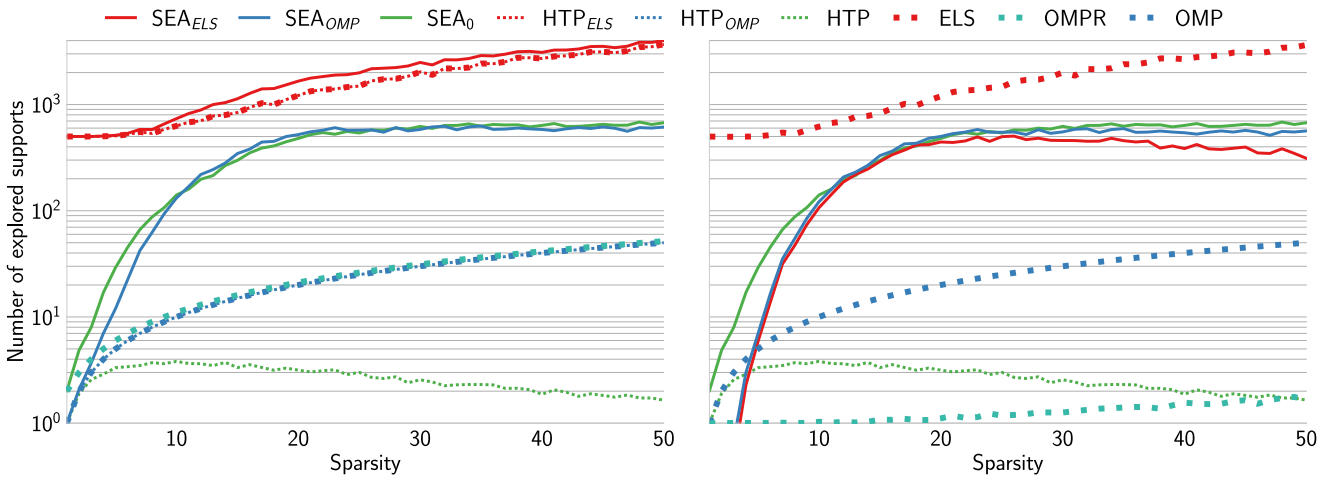

FIGURE 3.4 – Gauche : Moyenne du nombre de supports explorés (en considérant ceux vus pendant l'initialisation le cas échéant) par les algorithmes résolvant 200 problèmes de déconvolution non bruités en dimension $n = m = 500$ (comme ceux de l'expérience de la figure 3.3), pour des niveaux de parcimonie $k \in [\![1, 50]\!]$. Droite : Moyenne du nombre de supports explorés après l'initialisation des algorithmes dans la même configuration. HTP_{ELS} et HTP_{OMP} font référence à HTP initialisé avec ELS et OMP respectivement. Chaque algorithme effectue 1000 itérations. L'expérience complète est décrite dans le chapitre 5.

dépend donc du nombre de supports explorés plutôt que du nombre d'itérations. Deuxièmement, à chaque fois qu'un nouveau support est exploré, le coût de l'itération est dominé par la résolution du système linéaire (sans contrainte) $A_{|S^t}^T A_{|S^t} x_{|S^t} = A_{|S^t}^T y$ qui est en $\mathcal{O}(k^2 n)$ comme expliqué dans la section précédente. La complexité globale est donc de $\mathcal{O}(n_{\text{supp}} k^2 n)$ où n_{supp} est le nombre de supports effectivement explorés.

La complexité de HTP, OMP, OMPR et ELS est également dominée par le nombre de fois où le système linéaire $A_{|S}^T A_{|S} x_{|S} = A_{|S}^T y$ est résolu avec S respectant $|S| = k$. Comme pour SEA, des implémentations efficaces de HTP, OMPR et ELS permettent d'économiser des calculs en stockant tous les supports explorés et les itérés correspondants. Les complexités de HTP, OMP et OMPR dépendent donc du nombre de supports explorés de la même manière que SEA. OMP résout seulement k instances du système linéaire, ce qui réduit l'exploration et le coût de calcul. ELS est beaucoup plus exigeant, car il explore $(n - k)$ supports à chaque itération, dont beaucoup ne sont pas pertinents. IHT est moins complexe que SEA parce qu'il n'inverse jamais le système $A_{|S}^T A_{|S} x_{|S} = A_{|S}^T y$.

Comme nous l'avons vu dans la figure 3.3 à la section 3.3.2, SEA surpasse ELS tout en explorant deux fois moins de supports (voir la figure 3.4 de la section précédente). Aussi, on pourrait penser qu'effectuer une recherche aléatoire des supports pourrait être plus efficace que l'exploration de SEA. Cependant, lorsque l'on fait les calculs, on se rend compte que cela est en réalité peu probable. Par exemple, si nous tirons

$r = 10^5$ supports aléatoires indépendants d'une distribution uniforme, la probabilité de retrouver le support correct de taille $k = 20$ d'un signal de taille $n = 500$ est de $1 - \left(1 - \frac{1}{\binom{n}{k}}\right)^r \sim 3.75 \times 10^{-31}$. Nous avons aussi confirmé expérimentalement [2] qu'une recherche aléatoire est totalement inefficace dans ces conditions.

3.5 Mise en évidence de l'efficacité de l'exploration

Pour terminer ce chapitre, nous illustrons l'efficacité de l'exploration de SEA en comparant expérimentalement son comportement à celui d'autres algorithmes de l'état de l'art. Les algorithmes testés sont Iterative Hard Thresholding (IHT) (BLUMENSATH et Mike E. DAVIES 2009), Hard Thresholding Pursuit (HTP) (FOUCART 2011), Orthogonal Matching Pursuit (MALLAT et al. 1993 ; PATI et al. 1993), OMP with Replacement (OMPR) (JAIN et al. 2011) et Exhaustive Local Search (ELS) (AXIOTIS et al. 2020). OMPR et ELS sont initialisés avec la solution de OMP. Trois versions de SEA sont étudiées : la version avec démarrage à froid SEA_0, où SEA est initialisé avec le vecteur nul, et les versions avec démarrage à chaud SEA_{ELS} et SEA_{OMP}, où SEA est initialisé avec les solutions d'ELS et d'OMP, respectivement. Pour des raisons d'équité, nous avons également étudié HTP et IHT initialisés avec OMP et ELS. Ils sont appelés HTP_{OMP}, HTP_{ELS}, IHT_{OMP} et IHT_{ELS}.

Pour tous les algorithmes, les colonnes de A sont normalisées avant l'exécution, et, chaque estimation par moindres carrés pour un support fixe, comme à la ligne 7 de algorithme 1, est résolue en utilisant l'algorithme de descente de gradient conjugué de SciPy (VIRTANEN et al. 2020). Les résultats de HTP et, dans une moindre mesure, d'IHT et de SEA dépendent du choix du pas η. L'étude de l'influence de ce pas sera réalisée dans le chapitre 5. Nous avons choisi ici de fixer [3] $\eta = \frac{1,8}{L}$, où L est le carré du rayon spectral de A, pour les algorithmes HTP, IHT et SEA. Pour illustrer l'efficacité de l'exploration et les capacités de reconstruction de supports de chaque algorithme, nous allons construire un diagramme de transition de phase.

Introduits par Donoho et Tanner (D. DONOHO et TANNER 2009) et utilisés en acquisition comprimée (D. L. DONOHO et al. 2009 ; FOUCART et al. 2013), les diagrammes de transition de phase montrent les limites de reconstruction d'un algorithme en fonction du sous-échantillonnage/indétermination $\zeta = \frac{m}{n}$ de $A \in \mathbb{R}^{m \times n}$, et de la parcimonie/densité $\rho = \frac{k}{m}$ de x^*.

Nous fixons $n = 500$, m prend 18 valeurs dans $[\![1, n]\!]$ et k toutes les valeurs dans $[\![1, 0.5m]\!]$. Pour chaque triplet (m, n, k) et chaque algorithme, nous effectuons $r = 1000$ expériences (décrites ci-dessous) pour évaluer le taux de réussite $\frac{s_{\zeta,\rho}}{r}$ de l'algorithme, où $s_{\zeta,\rho}$ est le nombre de problèmes résolus avec succès. Un problème est considéré comme résolu avec succès si le support renvoyé par l'algorithme, après $256k$ itérations,

2. Dans la mesure où l'on observe exactement ce à quoi on s'attend, nous n'avons pas reporté ces résultats.

3. Nous avons réalisé l'expérience pour $\eta \in \{\frac{2^l}{L} \mid \text{for } l = [\![-3, +3]\!]\}$. Cela ne modifie pas de manière significative les résultats en termes de temps d'exécution, de stabilité, de performance.

est égal à S^*. Pour chaque exécution, les entrées de $A \in \mathbb{R}^{m\times n}$ sont tirées indépendamment selon la loi normale centrée réduite. Le vecteur k-parcimonieux $x^* \in \mathbb{R}^n$ est aléatoire. Les indices du support S^* sont choisis au hasard, uniformément et sans remplacement. Les entrées non nulles de x^* sont tirées uniformément dans $[-2,-1]\cup[1,2]$ comme dans (ELAD 2010). Le bruit e est tiré uniformément en utilisant la méthode décrite dans (BLANCHARD et al. 2015), c'est-à-dire qu'il est tiré uniformément depuis la sphère de rayon $0.01\|Ax^*\|_2$ dans $\mathbb{R}^m$. Le code est implémenté en Python 3 et est disponible publiquement [4].

La figure 3.5 montre les résultats de cette expérience. Chaque courbe indique le seuil en dessous duquel l'algorithme a un taux de réussite supérieur à 95%. Plus la courbe est élevée, mieux c'est. On sait que les constantes d'isométrie restreintes sont très mauvaises lorsque $\zeta = \frac{m}{n}$ est petit et s'améliorent lorsque m augmente (BAH et al. 2014). On s'attend donc à ce que les algorithmes obtiennent de meilleurs résultats pour des valeurs de ζ plus élevées. Nous constatons que les algorithmes OMP, HTP et IHT obtiennent de mauvais résultats en termes de reconstruction. La partie lisse et décroissante des courbes HTP et IHT sur la gauche est un artefact dû aux valeurs discrètes de (m,n,k) et correspond en fait à une transition de phase située à $k=1$.

Sur la figure, on voit que SEA_0 surpasse OMP, HTP et IHT lorsque $\frac{m}{n} < 0.6$. Tous les algorithmes initialisés à OMP (en bleu) améliorent les performances d'OMP sauf dans les cas les plus cohérents ($\frac{m}{n} < 0.2$) où HTP_{OMP} et IHT_{OMP} échouent alors que SEA_{OMP} présente la meilleure amélioration. Contrairement à HTP_{ELS} et IHT_{ELS}, SEA_{ELS} (en rouge) améliore encore les performances d'ELS et surpasse les autres algorithmes pour tous les $\frac{m}{n}$. Les principales améliorations apparaissent lorsque $\frac{m}{n}$ est petit ($\frac{m}{n} < 0.4$), c'est-à-dire pour les matrices A les plus cohérentes. Ainsi, SEA raffine un bon support candidat en un meilleur support en en explorant de nouveaux. Il réussit également à reconstruire S^* pour des valeurs de parcimonie k plus élevées que celles de ses concurrents. La supériorité réelle de SEA_{ELS} et SEA_{OMP} pour les matrices cohérentes ($\frac{m}{n} < 0.3$) est une conclusion majeure de cette expérience et illustre la capacité de SEA à explorer avec succès les supports dans des problèmes difficiles où les concurrents échouent.

L'analogue des courbes de la figure 3.5 se trouve dans la figure 3.6 dans le cas non bruité. À nouveau, un artefact provenant des valeurs discrètes de (m,n,k) est responsable de la partie lisse et décroissante observée sur le côté gauche des courbes de transition de phase, dans une région où $k=1$. Sans bruit, tous les algorithmes présentent une courbe de transition de phase similaire et conservent le même classement que dans le cas bruité. Ainsi, les conclusions tirées de la figure 3.6 dans le cas sans bruit sont analogues à celles que l'on vient de faire dans le cas bruité.

3.6 Conclusions

Dans ce chapitre, nous avons introduit SEA, un nouvel algorithme d'exploration de support pour la reconstruction parcimonieuse dans le cas linéaire. Nous avons

4. https://gitlab.lis-lab.fr/valentin.emiya/sea-icml-2024

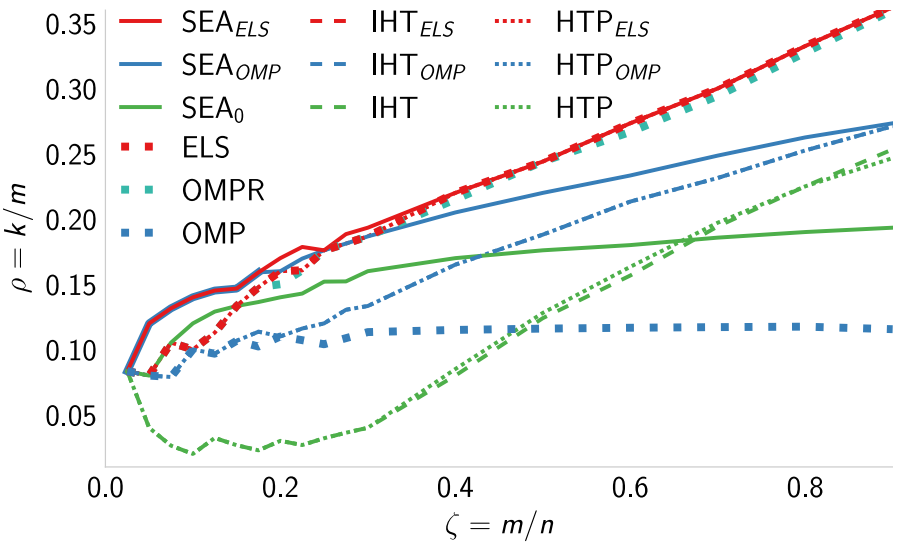

FIGURE 3.5 – Diagramme de transition de phase : chaque courbe (ou ligne de transition de phase) représente le seuil en dessous duquel l'algorithme correspondant reconstruit au moins 95% des supports S^*. La quantité ζ représente le rapport entre le nombre de lignes et le nombre de colonnes dans A tandis que ρ représente le rapport entre la parcimonie et le nombre de lignes de A. Les matrices A ont des entrées tirées depuis la loi normale centrée réduite et sont indépendantes et identiquement distribuées. Les entrées non nulles de x^* sont tirées uniformément dans $[-2,-1]\cup[1,2]$. Le nombre de lignes de A, $n = 500$, est fixé et les résultats sont obtenus à partir de 1000 problèmes résolus.

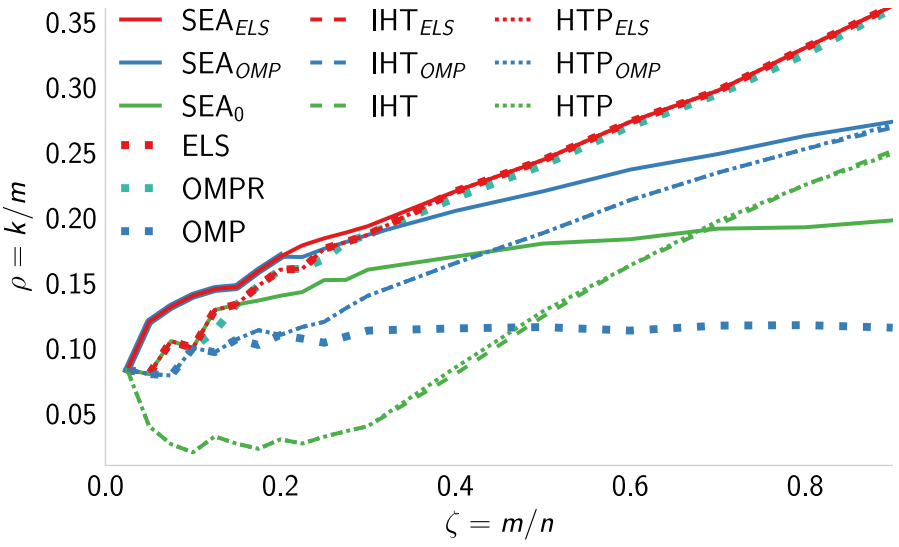

FIGURE 3.6 – Diagramme de transition de phase analogue à la figure 3.5 dans le cas non bruité ($e = 0$).

d'abord montré qu'il était possible de reformuler le problème de reconstruction de support parcimonieux en un problème pour lequel on peut utiliser un estimateur direct. L'utilisation de cet estimateur direct couplé à l'écriture d'un algorithme de type « descente de gradient » nous a permis de proposer SEA. Cependant, SEA n'est pas un algorithme de descente de gradient classique. Nous avons montré que l'utilisation de l'estimateur direct mène à l'apparition d'une variable d'exploration de supports $\mathscr{X}$ qui accumule les itérés de gradient. Cette variable d'exploration de supports permet à SEA d'avoir un compromis exploration/exploitation différent des autres algorithmes de reconstruction parcimonieuse. Nous avons par ailleurs montré que SEA explore beaucoup plus et beaucoup plus efficacement que les autres algorithmes, mais avec une complexité maîtrisée. Ce qui lui permet de surpasser les algorithmes de l'état de l'art dans les cas où les colonnes de la matrice A sont cohérentes. C'est ce que nous avons observé dans l'illustration de l'efficacité de l'exploration de SEA en déconvolution et dans l'expérience du diagramme de phase. Les deux chapitres suivants seront consacrés à l'étude théorique et expérimentale de SEA.

4 Une preuve théorique du succès de SEA

Sommaire

4.1 Introduction 78
4.2 Analyse d'un oracle pour l'exploration 82
4.2.1 Principe de l'oracle et énoncé du théorème de reconstruction . 82
4.2.2 Préliminaires 84
4.2.3 Preuve du théorème 4.2.1 : reconstruction par l'oracle 85
4.3 Analyse de SEA dans le cas général et garanties de reconstructions . . . 86
4.3.1 Énoncé du théorème de reconstruction dans le cas général . . . 87
4.3.2 Préliminaires 88
4.3.3 Preuve du théorème 4.3.1 : reconstruction dans le cas général . 92
4.4 Corollaire : Le cas des matrices orthogonales 94
4.5 Analyse de SEA dans le cas RIP et garanties de reconstructions 95
4.5.1 Énoncé du théorème de reconstruction et de son corollaire . . . 96
4.5.2 Propriétés issues du RIP 98
4.5.3 Préliminaires 99
4.5.4 Preuve du théorème 4.5.3 : reconstruction dans le cas RIP 102
4.5.5 Preuve du corollaire 4.5.4 : le cas RIP non bruité 104
4.5.6 Discussion 104
4.6 Conclusions 107

Ce chapitre est consacré à la preuve théorique du succès de SEA, l'algorithme que nous avons introduit au chapitre précédent. Ensuite, nous démontrons, à travers plusieurs théorèmes et corollaires, que SEA est capable de retrouver le bon support dans plusieurs situations. Nous concluons ce chapitre par une discussion sur les résultats théoriques obtenus. Le chapitre suivant est consacré à l'application de SEA à la déconvolution de signaux et à l'apprentissage supervisé. Comme le précédent et le suivant, ce chapitre reprend une partie du contenu de (MOHAMED, MALGOUYRES et al. 2024).

4.1 Introduction

Les algorithmes d'apprentissage faisant appel à l'estimateur direct (STE) ont contribué à améliorer l'état de l'art. Cependant, il existe très peu d'études théoriques garantissant leur succès. Dans (H. LI et al. 2017), les auteurs montrent dans le cadre des réseaux de neurones quantifiés que l'estimateur direct se comporte bien sur les problèmes convexes et non-convexes alors qu'une variante stochastique ne fonctionne pas sur les problèmes non convexes. Dans le cadre des réseaux de neurones linéaires à deux couches avec une fonction d'activation quantifiée, un estimateur direct bien choisi converge vers un point critique du risque empirique (YIN et al. 2019) ou agit en tant qu'oracle (LONG et al. 2021). Ce sont les seules garanties formelles connues qui incluent l'estimateur direct. Des travaux étudient la géométrie des vecteurs binaires en grande dimension (ANDERSON et al. 2018) pour expliquer le succès des réseaux de neurones binaires. Dans (HELWEGEN et al. 2019), les auteurs interprètent les poids flottants $\mathscr{W}$ (introduits dans la section 3.1.2) comme une variable d'inertie pour concevoir un nouvel algorithme. Enfin, dans (CHENG et al. 2018), les auteurs considèrent l'estimateur direct comme un flux de gradient de Wasserstein projeté.

Notre algorithme a été conçu dans un contexte de reconstruction du support. Dans le cas bruité $e \neq 0$, la reconstruction du support est une garantie plus forte que celle du cadre, plus standard, de l'acquisition comprimée, initiée dans (CANDÈS et al. 2006 ; D. L. DONOHO 2006), dont le but est de contrôler $\|Ax - Ax^*\|_2$, pour un x bien choisi. La première particularité de la reconstruction de supports est qu'elle suppose que x^* est vraiment k-parcimonieux (et pas seulement compressible). De plus, les garanties de reconstruction de support impliquent toujours une hypothèse sur $\min_{i \in S^*} |x^*_{|i}|$, en plus de l'hypothèse d'incohérence sur A (WAINWRIGHT 2009 ; MEINSHAUSEN et al. 2006 ; ZHAO et al. 2006 ; CAI et al. 2011 ; X. YUAN et al. 2016). Nous ne pouvons en effet pas espérer récupérer un élément $i \in S^*$ si $|x^*_{|i}|$ est négligeable par rapport au bruit (WAINWRIGHT 2009).

SEA s'appuie sur des garanties de reconstruction du support. Ces garanties sont fondées sur des conditions impliquant x^* et la propriété d'isométrie restreinte (RIP) de A. Ces conditions sont légèrement plus strictes, mais comparables à celles de l'état de l'art. Leur intérêt réside principalement dans leur application à une instance de l'estimateur direct, pour lequel il existe très peu de garanties de convergence. En effet, les succès de l'estimateur direct observés dans les expériences du chapitre 5 s'étendent à des problèmes cohérents pour lesquels l'hypothèse d'isométrie restreinte n'est plus satisfaite.

Les garanties reposent sur des preuves basées sur une interprétation de SEA comme une version bruitée d'un algorithme oracle (donné dans la section 4.2). Ces preuves reposent sur le principe suivant : lorsque les hypothèses du théorème sont satisfaites, la trajectoire $(\mathscr{X}^t)_{t\in\mathbb{N}}$ dans SEA est proche de la trajectoire de $(\mathscr{X}^t)_{t\in\mathbb{N}}$ dans cet algorithme oracle. Dans l'ensemble du chapitre, nous considérons sans perte de généralité [1] que les colonnes de A sont normalisées, c'est-à-dire que $\|A_{|i}\|_2^2 = 1$ pour

1. Nous l'avons montré dans une remarque de la section 1.4 à la page 30.

tout $i \in [\![1, n]\!]$. Nous rappelons au lecteur que nous souhaitons reconstruire le support S^* de x^*, à partir de $y = Ax^* + e$ et de A, en résolvant le problème d'optimisation suivant, avec $F(x) = \frac{1}{2}\|Ax - y\|_2^2, \forall x \in \mathbb{R}^n$:

$$\underset{x \in \mathbb{R}^n,\ \|x\|_0 \leq k}{\text{Minimiser}}\ F(x). \tag{SPARSE}$$

Avant d'aller plus loin, nous insistons sur le fait qu'aucune des preuves ne repose sur la décroissance d'une fonction. En particulier, comme nous l'avons vu dans la figure 3.3 du chapitre précédent, $(F(H(\mathscr{X}^t)))_{t\in\mathbb{N}}$ présente généralement un comportement non monotone. En effet, par construction dans l'algorithme 1, le support suivant, désigné par $\mathscr{X}^{t+1}$, n'est pas limité aux supports pour lesquels $F \circ H$ décroît. Ainsi, SEA explore plus de supports que les algorithmes possédant cette restriction et n'est pas non plus piégé dans des minima locaux.

Pour terminer cette introduction, nous énonçons les résultats principaux de ce chapitre. Il s'agit ici de rendre compte du fil conducteur des preuves qui sont détaillées dans les sections suivantes. Les définitions, théorèmes et corollaires sont donc tous énoncés de manière abrégée et simplifiée. Ils seront détaillés dans les sections suivantes.

Le raisonnement que nous suivons est le suivant : nous commençons par introduire un algorithme oracle. Cet algorithme s'écrit de la même manière que SEA, cependant il utilise une règle de mise à jour différente. Plus précisément, la mise à jour de $\mathscr{X}$ par l'estimateur direct (STE) dans SEA est remplacée par une mise à jour idéale, qui dépend de la vraie solution x^*. Ainsi, nous démontrons d'abord que cet algorithme oracle, défini en section 4.2, est capable de reconstruire le support de x^* dans toutes les situations.

Théorème 4.1.1. (*Reconstruction par la règle de l'oracle - Version abrégée du théorème 4.2.1*) Pour toute matrice A, bruit e, initialisation $\mathscr{X}^0$ et $\eta > 0$, l'algorithme oracle reconstruit le support S^* en un nombre fini d'itérations.

Ensuite, dans la section 4.3, nous partons du principe que SEA est une version bruitée de l'algorithme oracle. En effet, les preuves des théorèmes reposent sur la mesure de la déviation de la trajectoire $(\mathscr{X}^t)_{t\in\mathbb{N}}$ définie dans l'algorithme 1 par rapport à la trajectoire définie avec la règle de mise à jour de l'oracle. Plus précisément, nous fournissons une condition suffisante (*RC*) sur l'écart entre la mise à jour de l'oracle et la mise à jour de l'estimateur direct (et donc de SEA) garantissant que SEA visite le vrai support S^*.

Définition 4.1.2. (*Condition de reconstruction (RC) - Version abrégée*) Soient ε l'écart maximal entre la mise à jour de SEA et celle de l'oracle, $\eta > 0$ et x^* le vecteur k-parcimonieux que l'on souhaite reconstruire. On dit que la condition de reconstruction (*RC*) est vérifiée (ou que x^* vérifie (*RC*)) si l'inégalité suivante est satisfaite :

$$\varepsilon < \frac{\eta}{2k} \min_{i \in S^*} |x^*_{|i}|. \tag{RC}$$

Théorème 4.1.3. (*Reconstruction par SEA : cas général - Version abrégée du théorème 4.3.1*) Si (*RC*) est vérifiée, alors pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe une itération où l'algorithme SEA visite [a] S^*.

a. Nous employons volontairement le verbe visiter, ou explorer, car nous ne pouvons pas utiliser le verbe reconstruire ici. Cette différence sera explicitée dans la section 4.3.

La condition (*RC*) est une condition de contrôle de l'écart entre SEA et l'algorithme oracle. Cependant, en pratique, nous n'avons pas accès à l'algorithme oracle. Par conséquent, nous ne pouvons ni calculer ε, ni garantir que (*RC*) soit vérifiée. Nous cherchons donc à déterminer des conditions sur A et x^* qui garantissent que (*RC*) est satisfaite. Dans un premier temps, nous considérons le cas des matrices orthogonales, dans la section 4.4, car il est très simple de montrer que $\varepsilon = 0$. Ainsi, nous établissons dans le corollaire 4.4.1 que, lorsque A orthogonale et que $e = 0$, SEA reconstruit x^* en $k+1$ itérations. Bien qu'il s'agisse d'une vérification sans intérêt pour les applications, ce résultat fournit un cas élémentaire où la règle de mise à jour de l'oracle et la règle de mise à jour de l'estimateur direct coïncident.

Corollaire 4.1.4. (*Reconstruction par SEA : cas orthogonal non bruité - Version abrégée du corollaire 4.4.1*) Si A est une matrice orthogonale et $e = 0$, alors $\varepsilon = 0$. C'est-à-dire que SEA se comporte comme l'algorithme oracle. Par conséquent, pour tout x^* et $\eta > 0$, SEA reconstruit le support S^* en $k+1$ itérations au plus.

Cependant, les matrices orthogonales constituent une classe de matrices très restreinte. Dans un second temps, nous généralisons donc ce résultat à une classe plus variées de matrices. Ainsi, dans la section 4.5, nous considérons le cas des matrices qui satisfont la propriété d'isométrie restreinte (RIP).

Définition 4.1.5. (*Condition de reconstruction sous hypothèse RIP (RC_{RIP}) - Version abrégée*) Soient α_k^{RIP} et γ_k^{RIP}, deux quantités dépendant uniquement des constantes d'isométrie restreinte de la matrice A. Soient x^* le vecteur k-parcimonieux que l'on souhaite reconstruire, et e le bruit contaminant l'observation. On dit que la condition de reconstruction sous hypothèse RIP (RC_{RIP}) est vérifiée (ou que x^* vérifie (RC_{RIP})) si l'inégalité suivante est satisfaite :

$$\gamma_k^{RIP}\|e\|_2 < \frac{\min_{i\in S^*}|x^*_{|i}|}{2k} - \alpha_k^{RIP}\|x^*\|_2. \qquad (RC_{RIP})$$

Théorème 4.1.6. (*Reconstruction par SEA : cas RIP - Version abrégée du théorème 4.5.3*) Si (RC_{RIP}) est vérifiée et que A satisfait le $(2k+1)$-RIP, alors pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe une itération où l'algorithme SEA visite S^*. De plus, si le niveau de bruit est suffisamment faible, alors SEA reconstruit S^* en un nombre fini d'itérations.

Cependant, la condition (RC_{RIP}) est difficile à interpréter. Nous verrons qu'il en va de même pour le théorème que nous venons d'énoncer. Nous allons donc chercher à simplifier la formulation de cette condition et de ce théorème.

Définition 4.1.7. (*Condition de reconstruction simplifiée sous hypothèse RIP (RC_{SRIP}) - Version abrégée*) Soit α_k^{RIP}, une quantité dépendant uniquement des constantes d'isométrie restreinte de la matrice A. Soit x^* le vecteur k-parcimonieux que l'on souhaite reconstruire. On dit que la condition de reconstruction simplifiée sous hypothèse RIP (RC_{SRIP}) est vérifiée (ou que x^* vérifie (RC_{SRIP})) si

$$\Lambda := 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} < 1. \qquad (RC_{SRIP})$$

Corollaire 4.1.8. (*Reconstruction par SEA : cas RIP non bruité - Version abrégée du corollaire 4.5.4*) Si (RC_{SRIP}) est vérifiée et que A est $(2k+1)$-RIP, alors pour tout x^* et $\eta > 0$, SEA reconstruit le support S^* en $\frac{k+1}{1-\Lambda}$ itérations.

Devant la multitude de résultats obtenus et des conditions nécessaires à leur validité, nous résumons l'ensemble des résultats théoriques du chapitre dans la figure 4.1 afin de faciliter leur lecture. Nous détaillons maintenant ces théorèmes, corollaires et leurs preuves, en commençant par la description de l'algorithme oracle dans la section suivante.

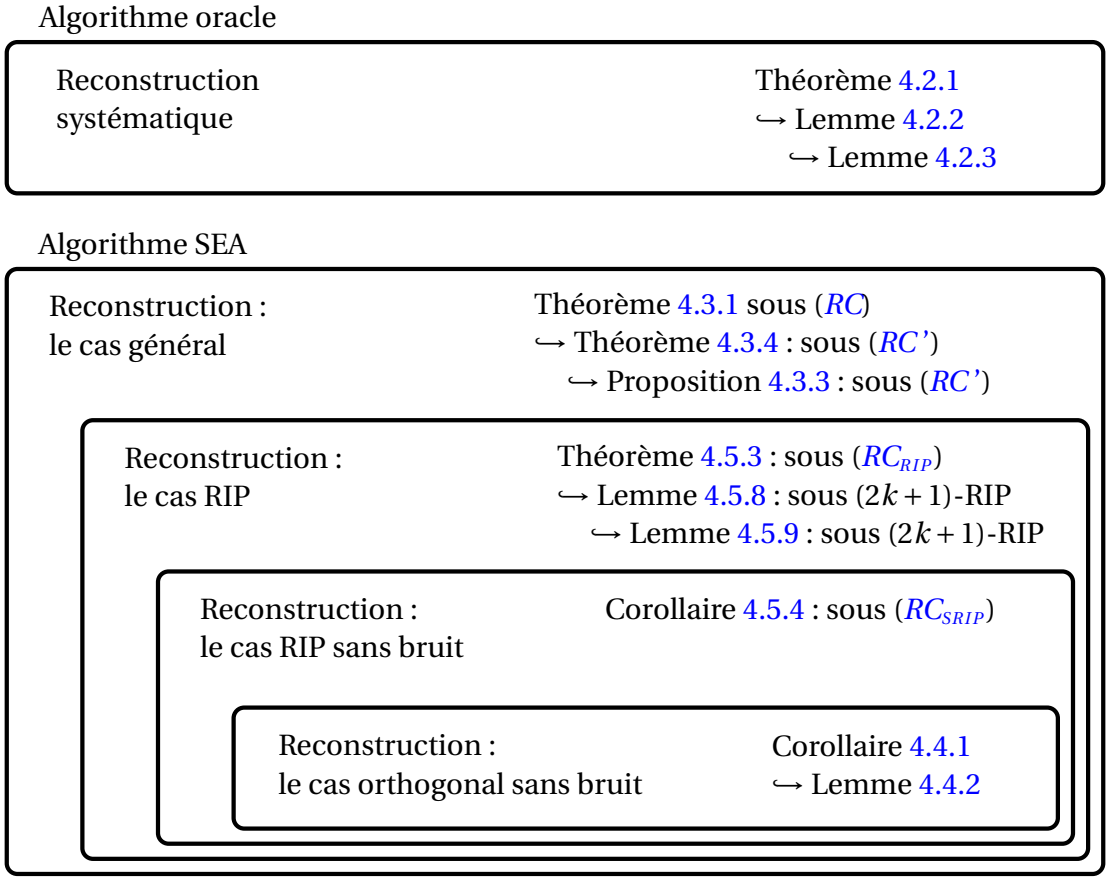

FIGURE 4.1 – Résumé des résultats théoriques du chapitre. Chaque flèche ↪ pointe vers un résultat qui est utilisé pour prouver le résultat à partir duquel elle part. Les boites sont organisées en fonction de leur niveau de généralité ; lorsqu'une boite est située à l'intérieur d'une autre, cela signifie qu'elle représente un cas particulier de cette dernière.

4.2 Analyse d'un oracle pour l'exploration

4.2.1 Principe de l'oracle et énoncé du théorème de reconstruction

L'analyse théorique de SEA et la compréhension du comportement sous-jacent de l'algorithme reposent sur l'introduction d'un cas oracle où la vraie solution x^* et son support S^* sont connus de l'algorithme. Dans ce cas, à l'itération t, nous pouvons utiliser la règle de mise à jour de l'oracle $\mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - u^t$, en utilisant la direction u^t définie pour tout indice $i \in [\![1, n]\!]$ par

$$u^t{}_{|i} = \begin{cases} -\eta x^*_{|i} & i \in S^* \cap \overline{S^t} \\ 0 & i \in \overline{S^*} \cup S^t, \end{cases} \tag{4.1}$$

où $S^t = \text{largest}_k\left(\mathscr{X}^t\right)$ contient les indices des k plus grandes entrées de $\mathscr{X}^t$ en valeur absolue et $\eta > 0$ est un pas arbitraire. Une représentation graphique des supports importants est fournie dans la figure 4.2. Le pseudo-code correspondant est donné dans l' algorithme 7. Cette règle de mise à jour dépend de x^* et n'a aucune utilité en pratique. Son intérêt réside dans le fait qu'elle fournit une mise à jour « idéale » permettant une reconstruction rapide du vrai support, comme nous le verrons dans le théorème 4.2.1

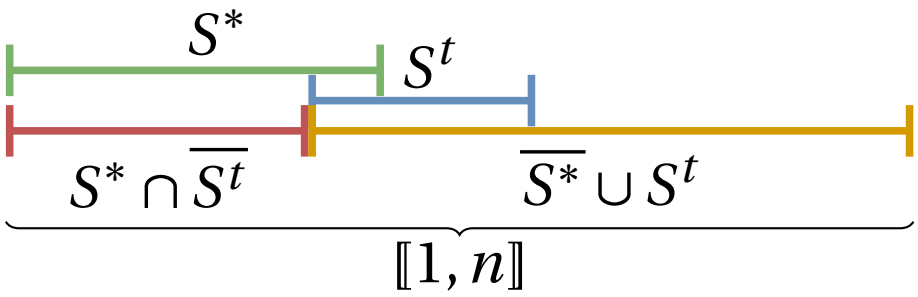

FIGURE 4.2 – Représentation visuelle des principaux ensembles d'indices rencontrés dans l'article.

Algorithme 7 SEA utilisant la règle de mise à jour de l'oracle

1 : **Entrées :** solution x^*, support S^*, parcimonie k, pas η, observation y, matrice A
2 : **Sortie :** vecteur k-parcimonieux x
3 : Initialiser $\mathscr{X}^0$
4 : $t \leftarrow 0$
5 : **Répéter**
6 : $\quad S^t \leftarrow \text{largest}_k\left(\mathscr{X}^t\right)$
7 : $\quad \mathscr{X}^{t+1} \leftarrow \mathscr{X}^t - u^t$
8 : $\quad t \leftarrow t+1$
9 : **Jusqu'à** $u^{t-1} = 0$
10 : $\begin{cases} x_{|i} & \leftarrow 0 \text{ pour } i \in \overline{S^{t-1}} \\ x_{|S^{t-1}} & \leftarrow A^{\dagger}_{|S^{t-1}} y \end{cases}$
11 : **renvoyer** x

On remarque que ${u^t}_{|i}$ est non nul pour les indices i du vrai support S^* pour lesquels $|{\mathscr{X}^t}_{|i}|$ est trop petit pour être sélectionné dans S^t à la ligne 6. Quel que soit le contenu initial de $\mathscr{X}^0$, la règle de mise à jour de l'oracle ajoute toujours le même incrément à ${\mathscr{X}^t}_{|i}$, pour $i \in S^* \cap \overline{S^t}$. Cela garantit que, lors d'une itération ultérieure $t' \geq t$, le vrai support S^* figure parmi les indices des k plus grandes entrées en valeur absolue de $\mathscr{X}^{t'}$. Autrement dit, $S^* \subseteq S^{t'}$ et, l'intersection $S^* \cap \overline{S^t}$ est vide. Ainsi, $u^{t'} = 0$, et l'algorithme 7 s'arrête.

Dans le théorème suivant, nous formalisons cette affirmation et donnons une borne supérieure au nombre d'itérations requis par l'algorithme d'exploration de supports en utilisant la règle de mise à jour de l'oracle pour reconstruire S^*.

Théorème 4.2.1. (*Reconstruction par la règle de l'oracle*) Pour toute matrice A, erreur e, initialisation $\mathscr{X}^0$ et pas $\eta > 0$, il existe

$$t_s \leq T_{max}^{oracle} = k\left(1 + \frac{2\|\mathscr{X}^0\|_\infty}{\eta \min_{i \in S^*} |x^*_{|i}|}\right)$$

tel que $S^* \subseteq S^{t_s}$, où S^t est défini à la ligne 6 de l'algorithme 7.
De plus, $u^{t_s} = 0$ et l'algorithme 7 retourne $\underset{x \in \mathbb{R}^n,\ \mathrm{supp}(x) \subseteq S^{t_s}}{\operatorname{argmin}} \|Ax - y\|_2^2$.

Avant de démonter le théorème 4.2.1, nous introduisons les définitions et lemmes qui sont utilisés dans la preuve.

4.2.2 Préliminaires

Nous définissons, pour tout $t \in \mathbb{N}^*$, tout $i \in [\![1, n]\!]$, et avec S^t défini à la ligne 6 de l'algorithme 7, la $i$ème composante du vecteur de comptage [2] c à l'itération t par :

$$c^t_{|i} = |\{t' \in [\![0, t-1]\!] : i \in S^* \cap \overline{S^{t'}}\}|. \tag{4.2}$$

Nous étendons la définition à $t = 0$ en fixant, pour tout $i \in [\![1, n]\!]$, $c^0_{|i} = 0$.

Nous pouvons démontrer par récurrence sur t en utilisant la définition de $\mathscr{X}^t$ dans l'algorithme 7 et celle de u^t dans (4.1), que, pour tout $t \in \mathbb{N}$,

$$\mathscr{X}^t = \mathscr{X}^0 + \eta\, c^t \odot x^*, \tag{4.3}$$

où $\odot$ désigne le produit de Hadamard.

Le lemme suivant indique que si $c^t_{|i}$ est grand, i est toujours sélectionné par l'algorithme 7.

Lemme 4.2.2. Pour tout $i \in S^*$ et tout $t \in \mathbb{N}^*$,

$$\text{si} \quad c^t_{|i} > \frac{2\|\mathscr{X}^0\|_\infty}{\eta |x^*_{|i}|} \quad \text{alors} \quad \forall t' \geq t, \quad i \in S^{t'}.$$

Démonstration. Soient $i \in S^*$ et $t \in \mathbb{N}^*$ tels que $c^t_{|i} > \frac{2\|\mathscr{X}^0\|_\infty}{\eta |x^*_{|i}|}$. Considérons

2. On l'appelle vecteur de comptage, car il comptabilise, pour chaque $i \in S^*$, le nombre de fois où i n'a pas été sélectionné dans le support courant S^t depuis le début de l'algorithme.

$t' \geq t$. Puisque $t \mapsto c_{|i}^t$ est croissant, nous avons

$$c_{|i}^{t'} \geq c_{|i}^t > \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|}.$$

Par conséquent, pour tout $j \in \overline{S^*}$,

$$|\mathscr{X}_{|i}^{t'}| = |\mathscr{X}_{|i}^0 + \eta c_{|i}^{t'} x_{|i}^*| \geq |\eta c_{|i}^{t'} x_{|i}^*| - |\mathscr{X}_{|i}^0| > 2\|\mathscr{X}^0\|_\infty - |\mathscr{X}_{|i}^0| \geq \|\mathscr{X}^0\|_\infty \geq |\mathscr{X}_{|j}^0| = |\mathscr{X}_{|j}^{t'}|.$$

Ainsi, $|\mathscr{X}_{|i}^{t'}|$ est plus grand qu'au moins $n-k$ éléments de $\{|\mathscr{X}_{|j}^{t'}| : j \in [\![1,n]\!]\}$.
Autrement dit, $i \in \text{largest}_k(\mathscr{X}^{t'}) = S^{t'}$.
Ceci conclut la preuve du lemme 4.2.2.

Cela conduit à la borne supérieure suivante.

Lemme 4.2.3. Pour tout $i \in S^*$ et tout $t \in \mathbb{N}^*$,

$$c_{|i}^t \leq \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|} + 1.$$

Démonstration. Si le lemme 4.2.3 est faux, il existe $i \in S^*$ et $t \in \mathbb{N}^*$ tels que $c_{|i}^t > \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|} + 1$.
Posons

$$t' = \min\left\{ t \in \mathbb{N}^* : c_{|i}^t > \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|} + 1 \right\}.$$

On a $c_{|i}^{t'} > \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|} + 1 \geq 1$ et donc $t' > 1$. Par conséquent, $t'-1 \in \mathbb{N}^*$ et $c_{|i}^{t'-1}$ existe et est défini par (4.2). En raison des définitions de t' et de $c_{|i}^{t'}$, nous devons avoir $c_{|i}^{t'} = c_{|i}^{t'-1} + 1$. Ainsi, $c_{|i}^{t'-1} > \frac{2\|\mathscr{X}^0\|_\infty}{\eta|x_{|i}^*|}$. Et, puisque $i \in S^*$ et $t'-1 \in \mathbb{N}^*$, en utilisant le lemme 4.2.2, nous concluons que $i \in S^{t'-1}$ et, en utilisant la définition de $c_{|i}^{t'}$ dans (4.2), nous concluons également que $c_{|i}^{t'} = c_{|i}^{t'-1}$.
Ceci est impossible. Nous pouvons donc conclure que le lemme 4.2.3 est vrai.

4.2.3 Preuve du théorème 4.2.1 : reconstruction par l'oracle

Nous commençons par rappeler l'énoncé du théorème.

Théorème 4.2.1. (*Reconstruction par la règle de l'oracle*) Pour toute matrice A, erreur e, initialisation $\mathscr{X}^0$ et pas $\eta > 0$, il existe

$$t_s \leq T_{max}^{oracle} = k\left(1 + \frac{2\|\mathscr{X}^0\|_\infty}{\eta \min_{i \in S^*}|x_{|i}^*|}\right)$$

tel que $S^* \subseteq S^{t_s}$, où S^t est défini dans l'algorithme 7 à la ligne 6.
De plus, $u^{t_s} = 0$ et l'algorithme 7 retourne $\underset{x \in \mathbb{R}^n,\ \mathrm{supp}(x) \subseteq S^{t_s}}{\operatorname{argmin}} \|Ax - y\|_2^2$.

Démonstration. Posons

$$t_s = \min\{t \in \mathbb{N} : S^* \subseteq S^t\}. \tag{4.4}$$

Par convention, si pour tout $t \in \mathbb{N}$, $S^* \not\subseteq S^t$ (l'oracle ne reconstruit jamais S^*), on considèrera que $t_s = +\infty$. La première affirmation du théorème 4.2.1 est évidente si $t_s = 0$. Nous supposons ci-dessous que $t_s \geq 1$.
Considérons $t \in [\![0, t_s - 1]\!]$. En utilisant la définition de t_s dans (4.4), il existe $i \in S^* \cap \overline{S^t}$. En utilisant la définition de $c_{|i}^{t+1}$ dans (4.2), nous obtenons $c_{|i}^{t+1} = c_{|i}^t + 1$. Puisque pour tout $j \in [\![1, n]\!]$, $t \mapsto c_{|j}^t$ est croissant, nous concluons donc que

$$\text{pour tout } t \in [\![0, t_s - 1]\!], \qquad \sum_{i \in S^*} c_{|i}^{t+1} \geq \sum_{i \in S^*} c_{|i}^t + 1.$$

Nous obtenons donc

$$\begin{aligned} \sum_{i \in S^*} c_{|i}^{t_s} &= \sum_{i \in S^*} \left(\sum_{i=0}^{t_s - 1} (c_{|i}^{t+1} - c_{|i}^t) \right) \\ &= \sum_{i=0}^{t_s - 1} \left((\sum_{i \in S^*} c_{|i}^{t+1}) - (\sum_{i \in S^*} c_{|i}^t) \right) \\ &\geq \sum_{i=0}^{t_s - 1} 1 = t_s. \end{aligned}$$

En utilisant le lemme 4.2.3, nous obtenons

$$t_s \leq \sum_{i \in S^*} c_{|i}^{t_s} \leq \sum_{i \in S^*} \left(\frac{2\|\mathscr{X}^0\|_\infty}{\eta |x_{|i}^*|} + 1 \right) \leq k \left(1 + \frac{2\|\mathscr{X}^0\|_\infty}{\eta \min_{i \in S^*} |x_{|i}^*|} \right).$$

Pour conclure la preuve, nous remarquons simplement que, puisque $S^* \subseteq S^{t_s}$, la définition de u^{t_s} dans (4.1) nous donne

$$u^{t_s} = 0.$$

4.3 Analyse de SEA dans le cas général et garanties de reconstructions

Comme x^* et S^* ne sont pas disponibles en pratique, nous remplaçons dans l'algorithme 7 la mise à jour de l'oracle u^t par le substitut $\eta A^T(Ax^t - y)$ (ligne 8 de l'algorithme 1). Le choix de ce substitut est une application de l'estimateur direct. Il

est de surcroît naturel. Par exemple, on peut montrer que $u^t = \eta A^T(Ax^t - y)$ dans le cas simple où A est orthogonale et où l'observation est sans bruit (voir le corollaire 4.4.1 et le lemme 4.4.2 dans la section 4.4). Dans le cadre général, SEA peut être interprété comme une version bruitée de l'algorithme d'exploration du support utilisant la mise à jour de l'oracle. Nous verrons plus tard que le théorème 4.5.3 et sa preuve dans la section 4.5.4 sont basés sur le fait que la norme de $u^t - \eta A^T(Ax^t - y)$ est petite sous des hypothèses appropriées sur x^* et les constantes RIP de A. Avant d'en arriver là, nous étudions d'abord ce qu'il se passe dans le cas général.

4.3.1 Énoncé du théorème de reconstruction dans le cas général

Nous montrons maintenant que si l'écart entre la mise à jour de l'oracle et la mise à jour de l'estimateur direct est suffisamment faible, SEA visite S^*.

Pour cela, nous définissons pour tout $t \in \mathbb{N}$, le bruit du gradient $b^t \in \mathbb{R}^n$ par

$$b^t = u^t - \eta A^T(Ax^t - y), \tag{4.5}$$

la norme maximale du bruit du gradient par,

$$\varepsilon = \sup_{t\in\mathbb{N}} \|b^t\|_\infty \in \mathbb{R}^+, \tag{4.6}$$

et la condition de reconstruction (*RC*)

$$\varepsilon < \frac{\eta}{2k} \min_{i\in S^*} |x^*_{|i}|. \tag{RC}$$

Dans cette section, nous démontrons le théorème suivant.

Théorème 4.3.1. (*Reconstruction par SEA : cas général*) Si (*RC*) est vérifiée, alors, pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe $t_s \leq T_{max}$ tel que $S^* \subseteq S^{t_s}$, où S^t est défini à la ligne 6 de l'algorithme 1 et

$$T_{max} = \frac{2k\|\mathscr{X}^0\|_\infty + (k+1)\eta \min_{i\in S^*} |x^*_{|i}|}{\eta \min_{i\in S^*} |x^*_{|i}| - 2k\varepsilon}. \tag{4.7}$$

L'intérêt principal du théorème 4.3.1 est de formaliser quantitativement que, lorsque $u^t - \eta A^T(Ax^t - y)$ est suffisamment petit, SEA visite le vrai support. Cependant, la condition (*RC*) est difficile à utiliser et à interpréter, car elle implique à la fois A, x^* et tous les itérés k-parcimonieux x^t. C'est pourquoi nous fournissons dans les sections suivantes des conditions suffisantes sur A, e et x^* garantissant que (*RC*) est vérifiée.

La conclusion du théorème 4.3.1 est que le processus itératif de SEA visite le vrai support S^* à une certaine itération t_s. En général, nous n'avons aucune garantie que ce pas de temps t_s est égal à t_{BEST}. Nous avons cependant la garantie que SEA renvoie une solution k-parcimonieuse telle que $\|Ax^{t_{BEST}} - y\|_2 \leq \|Ax^{t_s} - y\|_2 \leq \|Ax^* - y\|_2$. Cela ne donne pas de garantie sur la reconstruction du support, mais sur l'erreur de

reconstruction de y. En apprentissage automatique, cette limite supérieure peut être utilisée pour dériver une limite supérieure du risque.

En ce qui concerne la valeur de T_{max}, une analyse rapide montre que T_{max} augmente avec ε, lorsque (*RC*) est vérifiée. En d'autres termes, le nombre d'itérations nécessaires à l'algorithme pour fournir la solution augmente avec l'écart entre u^t et $\eta A^T(Ax^t - y)$.

Les initialisations $\mathscr{X}^0 \neq 0$ ont un impact négatif apparent sur le nombre d'itérations nécessaires dans le pire des cas. En effet, dans le pire des cas, $\mathscr{X}^0$ est mal choisi et SEA aurait besoin de plusieurs itérations pour corriger ce mauvais choix.

En ce qui concerne η, il convient de noter que puisque, u^t est proportionnel à $\eta > 0$ et que ε est proportionnel à $\eta > 0$, (*RC*) est indépendant de η. Lorsque cela est possible, tout η permet de reconstruire S^*. La seule influence de η est sur T_{max} défini dans (4.7). À cet égard, comme ε est proportionnel à $\eta > 0$, le dénominateur de (4.7) est proportionnel à η. Au numérateur, nous voyons que plus η est grand, plus vite SEA écrase l'initialisation $\mathscr{X}^0$. Le choix de η est donc étroitement lié à la question de la qualité de l'initialisation comme discutée ci-dessus.

4.3.2 Préliminaires

Pour démontrer le théorème 4.3.1, nous adaptons dans cette section la formule explicite de la variable exploratoire $\mathscr{X}^t$ déjà rencontrée dans la preuve du théorème 4.2.1 (celle de (4.3)). Ensuite, nous étudions les propriétés de cette formule explicite à travers le vecteur de comptage c^t pour trouver une condition suffisante de reconstruction du support. Nous démontrons enfin le théorème 4.3.1 dans la section suivante.

Nous rappelons au lecteur que la figure 4.2 contient une description graphique de l'ensemble des supports rencontrés dans les preuves. Nous rappelons également que pour chaque itération $t \in \mathbb{N}$ et $i \in [\![1, n]\!]$, la mise à jour de l'oracle (définie en (4.1)) est

$$u^t_{|i} = \begin{cases} -\eta x^*_{|i} & i \in S^* \cap \overline{S^t} \\ 0 & i \in \overline{S^*} \cup S^t. \end{cases}$$

Nous montrons tout d'abord, le lemme suivant.

Lemme 4.3.2. Pour tout $t \in \mathbb{N}$ et tout $i \in S^t$,

$$b^t_{|i} = 0.$$

Démonstration. Cette égalité est démontrée en remarquant que, pour tout $i \in S^t$, $u^t_{|i} = 0$, et $(A^T(Ax^t - y))_{|i} = 0$. En effet, cette dernière égalité est vraie, car $i \in S^t$ et $x^t_{|S^t} = A^\dagger_{|S^t} y$ (voir algorithme 1, ligne 7). Par définition de la pseudo-inverse de Moore-Penrose, $A_{|S^t} A^\dagger_{|S^t}$ est le projecteur orthogonal de $\mathbb{R}^m$ sur $\mathrm{Im}(A_{|S^t})$. Par conséquent, $A_{|S^t} A^\dagger_{|S^t} y - y$ est orthogonal à $\mathrm{Im}(A_{|S^t})$ et, pour tout $x' \in \mathbb{R}^k$, $0 = \langle A_{|S^t} A^\dagger_{|S^t} y - y, A_{|S^t} x' \rangle = \langle A^T_{|S^t}(A_{|S^t} A^\dagger_{|S^t} y - y), x' \rangle$. Ainsi, $A^T_{|S^t}(A_{|S^t} A^\dagger_{|S^t} y - y) = 0$. Puisque $x^t_{|i} = 0$ pour tout $i \in \overline{S^t}$, nous avons aussi

que $A_{|S^t} A^{\dagger}_{|S^t} y = A_{|S^t} x^t_{|S^t} = Ax^t$. Nous déduisons ainsi que pour tout $i \in S^t$, $(A^T(Ax^t - y))_{|i} = (A^T_{|S^t}(A_{|S^t} A^{\dagger}_{|S^t} y - y))_{|j} = 0$, où la ligne $j \in [\![1,k]\!]$ de $A^T_{|S^t}$ correspond à la ligne $i \in S^t$ de A^T. Ainsi, $b^t_{|i} = 0$. Ceci conclut la preuve du lemme 4.3.2.

En utilisant la définition de b^t et SEA (ligne 8), nous obtenons, pour tout $t \in \mathbb{N}$, une nouvelle écriture de la mise à jour de la variable d'exploration de supports :

$$\mathscr{X}^{t+1} = \mathscr{X}^t + b^t - u^t. \tag{4.8}$$

Le bruit du gradient b^t est l'erreur qui empêche le gradient (calculé dans SEA) d'être dans la direction de la mise à jour de l'oracle u^t. À chaque itération, cette erreur s'accumule dans $\mathscr{X}^t$. En posant $\beta^0 = 0$, nous définissons cette erreur accumulée, pour tout $t \in \mathbb{N}^*$, par

$$\beta^t = \sum_{t'=0}^{t-1} b^{t'} \in \mathbb{R}^n. \tag{4.9}$$

Comme nous l'avons déjà fait dans la preuve du théorème 4.2.1 pour la suite de supports $(S^t)_{[\![0,t_s]\!]}$ définie dans l'algorithme 7 (SEA avec oracle), nous définissons des vecteurs de comptage c^t. Cependant, cette fois-ci, ils sont définis pour la suite de supports définie dans l'algorithme 1(SEA). Pour des raisons de simplicité, nous conservons les mêmes notations que dans la section précédente. Nous fixons à nouveau $c^0 = 0$, et, pour tout $t \in \mathbb{N}^*$ et $i \in [\![1,n]\!]$, nous définissons également le vecteur de comptage par

$$c^t_{|i} = |\{t' \in [\![0, t-1]\!] : i \in S^* \cap \overline{S^{t'}}\}|. \tag{4.10}$$

Nous allons utiliser la forme récursive de c^t. Pour tout $t \in \mathbb{N}$, $i \in [\![1,n]\!]$

$$c^{t+1}_{|i} = \begin{cases} c^t_{|i} + 1 & \text{if } i \in S^* \cap \overline{S^t} \\ c^t_{|i} & \text{if } i \in \overline{S^*} \cup S^t. \end{cases} \tag{4.11}$$

Pour tout $i \in [\![1,n]\!]$, on remarque que la suite $(c^t_{|i})_{t\in\mathbb{N}}$ est croissante.

En utilisant (4.8), (4.9) et (4.10), nous établissons par récurrence sur t que, pour tout $t \in \mathbb{N}$,

$$\mathscr{X}^t = \mathscr{X}^0 + \eta c^t \odot x^* + \beta^t, \tag{4.12}$$

où $\odot$ désigne le produit de Hadamard. Ceci généralise (4.3) dans notre nouveau cadre bruité.

Comme nous pouvons le voir dans (4.12), l'accumulation d'erreurs β^t est responsable de l'exploration dans les mauvaises directions. Alors que $c^t \odot x^*$ encourage l'exploration dans la direction des composants manqués de x^*. Nous présentons ci-dessous les propriétés importantes de $(c^t)_{t\in\mathbb{N}}$.

Pour chaque itération de SEA, en utilisant (4.11) lorsque $S^* \nsubseteq S^t$, nous voyons qu'il existe au moins un $i \in S^*$ tel que $c^{t+1}_{|i} = c^t_{|i} + 1$. En utilisant également que, pour tout

$i \in S^*$, $(c^t_{|i})_{t\in\mathbb{N}}$ est croissante, nous obtenons

$$\text{pour tout } t \in \mathbb{N} \text{ tel que } S^* \not\subseteq S^t, \quad \sum_{i\in S^*} c^{t+1}_{|i} \geq \Big(\sum_{i\in S^*} c^t_{|i}\Big) + 1 \tag{4.13}$$

Nous définissons la premiere itération de reconstruction[3] t_s comme la première itération t telle que $S^* \subseteq S^t$. Plus précisément,

$$t_s = \min\{t : S^* \subseteq S^t\} \in \mathbb{N}. \tag{4.14}$$

Comme dans la démonstration de l'oracle, si S^* n'est jamais reconstruit, $t_s = +\infty$ par convention. Par récurrence sur t, en utilisant (4.13), nous obtenons une borne inférieure de $\sum_{i\in S^*} c^t_{|i}$:

$$\text{Pour tout } t \leq t_s, \quad \sum_{i\in S^*} c^t_{|i} \geq t. \tag{4.15}$$

Établissons une borne supérieure de $\sum_{i\in S^*} c^t_{|i}$. Nous rappelons d'abord que la définition de ε est donnée dans (4.6). Nous définissons la condition de reconstruction forte

$$\varepsilon < \frac{1}{2\sum_{i\in S^*} \frac{1}{\eta|x^*_{|i}|}} \tag{RC'}$$

et

$$T'_{max} = \frac{\sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta|x^*_{|i}|} + k + 1}{1 - 2\varepsilon\sum_{i\in S^*} \frac{1}{\eta|x^*_{|i}|}}. \tag{4.16}$$

Si (*RC'*) est vérifiée, nous définissons, pour tout $i \in S^*$, le $i^{ème}$ seuil de comptage[4] par

$$C_i = \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}| + 2T'_{max}\varepsilon}{\eta|x^*_{|i}|}. \tag{4.17}$$

Proposition 4.3.3. (*Borne supérieure de* $c^t_{|i}$) Si (*RC'*) est vérifiée, alors pour tout $i \in S^*$ et tout $t \leq T'_{max}$, nous avons $c^t_{|i} \leq C_i + 1$.

Démonstration. Supposons que (*RC'*) est vérifiée. Alors, $T'_{max} > 0$. Prenons $i \in S^*$, nous distinguons deux cas :
$\underline{1^{er}\ cas}$: Si pour tout $t \leq T'_{max}$, $c^t_{|i} \leq C_i$: Alors évidemment, pour tout $t \leq T'_{max}$, $c^t_{|i} \leq C_i + 1$.
$\underline{2^{nd}\ cas}$: S'il existe $t \leq T'_{max}$, tel que $c^t_{|i} > C_i$:

3. À nouveau, une quantité similaire est définie dans la preuve de théorème 4.2.1 pour les supports S^t issus de algorithme 7. Nous utilisons la même notation bien que cette fois la quantité soit définie pour les ensembles S^t définis dans algorithme 1. Il ne devrait pas y avoir d'ambiguïté puisque les notations sont utilisées dans différentes preuves et sections.
4. Par analogie au vecteur de comptage c.

On définit $t_i = \min\{t \in \mathbb{N} : c^t_{|i} > C_i\}$. On a donc $t_i \leq T'_{max}$. La preuve suit deux étapes :

1. Nous montrons que pour tout $t \in [\![t_i, T'_{max}]\!]$, $c^t_{|i} = c^{t_i}_{|i}$. (4.18)
2. Nous montrons que pour tout $t \leq T'_{max}$, $c^t_{|i} \leq C_i + 1$. (4.19)

1. Soit $t \in [\![t_i, T'_{max}]\!]$, nous avons, en utilisant (4.12), l'inégalité triangulaire et le fait que $c^t_{|i} \geq c^{t_i}_{|i} > C_i$,

$$\begin{aligned}|\mathscr{X}^t_{|i}| &= |\mathscr{X}^0_{|i} + \eta c^t_{|i} x^*_{|i} + \beta^t_{|i}| \\ &\geq \eta c^t_{|i}|x^*_{|i}| - |\mathscr{X}^0_{|i}| - |\beta^t_{|i}| \\ &> \eta C_i |x^*_{|i}| - |\mathscr{X}^0_{|i}| - |\beta^t_{|i}|.\end{aligned}$$

   En utilisant la définition de C_i, donnée dans (4.17), nous obtenons

$$\begin{aligned}|\mathscr{X}^t_{|i}| &> \eta \frac{\max_{j \notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}| + 2T'_{max}\varepsilon}{\eta |x^*_{|i}|} |x^*_{|i}| - |\mathscr{X}^0_{|i}| - |\beta^t_{|i}| \\ &= \max_{j \notin S^*} |\mathscr{X}^0_{|j}| + 2T'_{max}\varepsilon - |\beta^t_{|i}|.\end{aligned}$$

   Puisque, pour tout $j \in [\![1, n]\!]$, $|\beta^t_{|j}| \leq \sum_{t'=0}^{t-1} |b^{t'}_{|j}| \leq t\varepsilon \leq T'_{max}\varepsilon$, nous avons

$$\begin{aligned}|\mathscr{X}^t_{|i}| &> \max_{j \notin S^*} |\mathscr{X}^0_{|j}| + \max_{j \notin S^*} |\beta^t_{|j}| + |\beta^t_{|i}| - |\beta^t_{|i}| \\ &\geq \max_{j \notin S^*} |\mathscr{X}^0_{|j} + \beta^t_{|j}| \\ &= \max_{j \notin S^*} |\mathscr{X}^t_{|j}|, \end{aligned} \tag{4.20}$$

   avec la dernière égalité justifiée par (4.12) et que pour tout $j \notin S^*$, $t \in \mathbb{N}$, $c^t{}_{|j} = 0$.
   L'équation (4.20) implique que $|\mathscr{X}^t_{|i}|$ est plus grand que $|\{j \notin S^*\}|$ éléments de $\{|\mathscr{X}^t_{|j}| : j \in [\![1, n]\!]\}$ et, puisque $|S^*| \leq k$, nous avons $|\{j \notin S^*\}| = n - |S^*| \geq n - k$. Finalement, $|\mathscr{X}^t_{|i}|$ est plus grand que $n - k$ éléments de $\{|\mathscr{X}^t_{|j}| \mid j \in [\![1, n]\!]\}$ avec $i \in \text{largest}_k(\mathscr{X}^t) = S^t$.
   On conclut que, pour tout $t \in [\![t_i, T'_{max}]\!]$, $i \in S^t$. En utilisant (4.11), on trouve que $c^{t+1}_{|i} = c^t_{|i}$. C'est pourquoi, pour tout $t \in [\![t_i, T'_{max} + 1]\!]$, $c^t_{|i} = c^{t_i}_{|i}$. Ceci conclut la preuve de la première étape.
2. Comme $t_i = \min\{t \in \mathbb{N} : c^t_{|i} > C_i\}$ et que $c^0_{|i} = 0$, $t_i \geq 1$. Avec la définition de t_i, $c^{t_i-1}_{|i} \leq C_i$ et $c^{t_i}_{|i} \neq c^{t_i-1}_{|i}$, on trouve que $c^{t_i}_{|i} = c^{t_i-1}_{|i} + 1 \leq C_i + 1$.
   En utilisant (4.18), pour tout $t \in [\![t_i, T'_{max}]\!]$, $c^t_{|i} = c^{t_i}_{|i} \leq C_i + 1$. Enfin, puisque $(c^t_{|i})_{t \in \mathbb{N}^*}$ est croissante, il s'ensuit que pour tout $t \leq t_i - 1$, $c^t_{|i} \leq c^{t_i-1}_{|i} \leq C_i$. Ceci conclut la preuve de (4.19).

4.3.3 Preuve du théorème 4.3.1 : reconstruction dans le cas général

Pour démontrer le théorème 4.3.1, nous démontrons d'abord un théorème plus fort, mais difficile à interpréter.

Théorème 4.3.4. (*Reconstruction forte par SEA : cas général*) Si (*RC'*) est vérifiée, alors pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe $t_s \leq T'_{max}$ tel que $S^* \subseteq S^{t_s}$, où S^t est défini à la ligne 6 de l'algorithme 1 et

$$T'_{max} = \frac{\sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta |x^*_{|i}|} + k + 1}{1 - 2\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x^*_{|i}|}}.$$

Démonstration. Supposons que (*RC'*) est vérifiée et démontrons le théorème 4.3.4 en utilisant les résultats des préliminaires de la section 4.3.2.
Pour faire cela, nous démontrons d'abord que $T'_{max} = \sum_{i\in S^*} C_i + k + 1$, puis démontrons que $t_s \leq T'_{max}$.
Puisque (*RC'*) est vérifiée, en utilisant la définition de T'_{max}, on calcule

$$T'_{max} = \frac{1}{1 - 2\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x^*_{|i}|}} \left(\sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta |x^*_{|i}|} + k + 1 \right)$$

$$\left(1 - 2\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x^*_{|i}|}\right) T'_{max} = \sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta |x^*_{|i}|} + k + 1$$

$$T'_{max} = \sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta |x^*_{|i}|} + k + 1 + 2T'_{max}\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x^*_{|i}|}$$

$$T'_{max} = \sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}| + 2T'_{max}\varepsilon}{\eta |x^*_{|i}|} + k + 1.$$

En utilisant (4.17), on obtient que $T'_{max} = \sum_{i\in S^*} C_i + k + 1$.
Nous démontrons finalement le théorème 4.3.1 par l'absurde. Supposons par l'absurde que $t_s > T'_{max}$, où t_s est défini dans (4.14). En utilisant (4.15) avec $t = \lfloor T'_{max} \rfloor < t_s$, on trouve

$$\sum_{i\in S^*} c_{|i}^{\lfloor T'_{max} \rfloor} \geq \lfloor T'_{max} \rfloor = \lfloor \sum_{i\in S^*} C_i + k + 1 \rfloor > \sum_{i\in S^*} C_i + k. \tag{4.21}$$

Cependant, en utilisant que $|S^*| \leq k$ et la proposition 4.3.3, on trouve, pour $t = \lfloor T'_{max} \rfloor$ que

$$\sum_{i\in S^*} C_i + k \geq \sum_{i\in S^*} (C_i+1) \geq \sum_{i\in S^*} c_{|i}^{\lfloor T'_{max}\rfloor}$$

Cela contredit (4.21). Nous pouvons donc conclure que $t_s \leq T'_{max}$. Le théorème 4.3.4 est démontré.

En utilisant le théorème 4.3.4, nous démontrons le théorème 4.3.1que nous rappelons ici.

Théorème 4.3.1. (*Reconstruction par SEA : cas général*) Si (*RC*) est vérifiée, alors, pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe $t_s \leq T_{max}$ tel que $S^* \subseteq S^{t_s}$, où S^t est défini à la ligne 6 de l'algorithme 1 et

$$T_{max} = \frac{2k\|\mathscr{X}^0\|_\infty + (k+1)\eta \min_{i\in S^*} |x^*_{|i}|}{\eta \min_{i\in S^*} |x^*_{|i}| - 2k\varepsilon}. \tag{4.7}$$

Démonstration. Si (*RC*) est vérifiée, c'est-à-dire que $\varepsilon < \frac{\eta}{2k}\min_{i\in S^*}|x^*_{|i}|$, alors, puisque $\sum_{i\in S^*}\frac{1}{|x^*_{|i}|} \leq \frac{k}{\min_{i\in S^*}|x^*_{|i}|}$, nous avons

$$\varepsilon < \frac{1}{2\sum_{i\in S^*}\frac{1}{\eta|x^*_{|i}|}}.$$

Ainsi, (*RC'*) est vérifiée, et nous pouvons appliquer le théorème 4.3.4. Il assure que pour toute initialisation $\mathscr{X}^0$ et tout $\eta > 0$, il existe $t_s \leq T'_{max}$ tel que $S^* \subseteq S^{t_s}$. Pour démontrer le théorème 4.3.1, il suffit de prouver que $T'_{max} \leq T_{max}$. En utilisant que $\sum_{i\in S^*}\frac{1}{|x^*_{|i}|} \leq \frac{k}{\min_{i\in S^*}|x^*_{|i}|}$, on obtient

$$1 - 2\varepsilon\sum_{i\in S^*}\frac{1}{\eta|x^*_{|i}|} \geq 1 - 2\varepsilon\frac{k}{\eta\min_{i\in S^*}|x^*_{|i}|}$$

et en utilisant $\min_{j\notin S^*}|\mathscr{X}^0_j| + |\mathscr{X}^0_i| \leq 2\|\mathscr{X}^0\|_\infty$ on trouve que

$$T'_{max} = \frac{\sum_{i\in S^*}\frac{\max_{j\notin S^*}|\mathscr{X}^0_{|j}| + |\mathscr{X}^0_{|i}|}{\eta|x^*_{|i}|} + k + 1}{1 - 2\varepsilon\sum_{i\in S^*}\frac{1}{\eta|x^*_{|i}|}} \leq \frac{\frac{2k\|\mathscr{X}^0\|_\infty}{\eta\min_{i\in S^*}|x^*_{|i}|} + k + 1}{1 - 2\varepsilon\frac{k}{\eta\min_{i\in S^*}|x^*_{|i}|}} = T_{max}.$$

Nous avons démontré le théorème 4.3.1, qui garantit que SEA peut reconstruire le support de x^* sous la condition (*RC*). Cependant, cette condition est impossible à vérifier en pratique en toute généralité. En effet, elle dépend de la norme maximale du bruit du gradient ε, laquelle dépend de la norme du bruit du gradient b, qui elle-même dépend de x^* et de toutes les estimations parcimonieuses x^t. Dans la suite de ce chapitre, nous étudions des cas particuliers dans lesquels il est possible de majorer ε et de vérifier (*RC*).

4.4 Corollaire : Le cas des matrices orthogonales

Le corollaire suivant particularise le théorème 4.3.1 au cas sans bruit ($e = 0$) et orthogonal. En pratique, un algorithme complexe tel que SEA est bien sûr inutile dans un tel cas, et les algorithmes de l'état de l'art mentionnés dans les généralités et au chapitre précédent ont des propriétés de reconstruction similaires. Nous donnons ce corollaire principalement pour illustrer la diversité des liens entre les propriétés du triplet (A, x^*, e) et ε ainsi que le comportement de SEA. Nous rappelons que les définitions de b^t et ε sont respectivement introduites dans (4.5) et (4.6). Le corollaire du cas orthogonal n'est pas seulement une vérification de la convergence de SEA sous des hypothèses simplistes. Il fournit également et surtout un cas d'étude pour comprendre la preuve sous hypothèse RIP que l'on présentera dans la section suivante, qui est plus technique. En effet, le corollaire donne un cas où les mises à jour de l'oracle coïncident avec la mise à jour faite avec l'estimateur direct dans SEA, comme mentionné dans le dernier paragraphe de la section 4.2.3.

Corollaire 4.4.1. (*Reconstruction par SEA : cas orthogonal*) Si A est une matrice orthogonale et que $\|e\|_2 = 0$, alors $\varepsilon = 0$. Donc, pour tout x^*, toute initialisation $\mathscr{X}^0 = 0$ et tout $\eta > 0$, si SEA réalise plus de $k+1$ itérations, nous avons $S^* \subseteq S^{t_{BEST}}$ et $x^{t_{BEST}} = x^*$.

Pour démontrer le corollaire 4.4.1, nous montrons d'abord dans le lemme 4.4.2 que le bruit du gradient b^t est nul pour tout $t \in \mathbb{N}$. Ensuite, nous appliquons le théorème 4.3.1 et démontrons que $S^* \subseteq S^{t_{BEST}}$ et $x^{t_{BEST}} = x^*$.

Lemme 4.4.2. Si A est une matrice orthogonale et que $\|e\|_2 = 0$, alors pour tout $t \in \mathbb{N}$ et tout $\eta > 0$,

$$\eta A^T \left(Ax^t - y\right) = u^t,$$

Cela veut dire que $b^t = 0$.

Démonstration. Soit $t \in \mathbb{N}$. Remarquons d'abord que, puisque $\|e\|_2 = 0$ et que A est une matrice orthogonale, nous avons

$$A^T \left(Ax^t - y\right) = A^T A\left(x^t - x^*\right) = x^t - x^*. \tag{4.22}$$

Pour démontrer le lemme 4.4.2, nous distinguons trois cas : $i \in S^t$, $i \in \overline{S^*} \cap \overline{S^t}$ et $i \in S^* \cap \overline{S^t}$.
$\underline{1^{er}\ cas}$: Si $i \in S^t$, alors $\eta\left(A^T \left(Ax^t - y\right)\right)_{|i} = 0 = {u^t}_{|i}$. La première égalité est une conséquence de la définition de x^t donnée à la ligne 7 de l'algorithme 1 . La seconde est due à la définition de u^t, donnée dans (4.1).
$\underline{2^{ème}\ cas}$: Si $i \in \overline{S^*} \cap \overline{S^t}$, en prenant la $i$ème composante de (4.22) et en utilisant les contraintes de support de x^t et x^*, on trouve que

$$\eta\left(A^T \left(Ax^t - y\right)\right)_{|i} = 0 = {u^t}_{|i},$$

où la seconde égalité est due à la définition de u^t, donnée dans (4.1).
$\underline{3^{ème}\ cas}$: Si $i \in S^* \cap \overline{S^t}$, la $i$ème composante de (4.22) devient

$$\eta\left(A^T\left(Ax^t - y\right)\right)_{|i} = -\eta x^*_{|i} = u^t_i,$$

où encore une fois, la seconde égalité est due à la définition de u^t, donnée dans (4.1).

Nous démontrons maintenant le corollaire 4.4.1.

Démonstration. Supposons que A est une matrice orthogonale, $\|e\|_2 = 0$ et $\mathscr{X}^0 = 0$. Nous rappelons que T_{max} est défini dans (4.7).
En utilisant le lemme 4.4.2, (4.6) et (4.5), on trouve que $\varepsilon = 0$. Ainsi (*RC*) est vérifiée pour tout x^* et le théorème 4.3.1 implique qu'il existe $t_s \leq T_{max}$ tel que $S^* \subseteq S^{t_s}$. Aussi, puisque $\mathscr{X}^0 = 0$ et $\varepsilon = 0$, on a $T_{max} = k+1$.
Ensuite, comme $\|e\|_2 = 0$, le théorème 4.3.1 et les définitions de t_{BEST} et x^t dans l'algorithme 1 nous donnent que

$$\|Ax^{t_{BEST}} - y\|_2 \leq \|Ax^{t_s} - y\|_2 \leq \|Ax^* - y\|_2 = 0.$$

Enfin, le fait que A soit une matrice orthogonale, et $\|e\|_2 = 0$, nous mène à

$$\begin{aligned} 0 &= Ax^{t_{BEST}} - y \\ &= A^T A(x^{t_{BEST}} - x^*) \\ &= x^{t_{BEST}} - x^*. \end{aligned}$$

Ainsi, $S^* = \text{supp}(x^*) = \text{supp}(x^{t_{BEST}}) \subseteq S^{t_{BEST}}$.
Cela conclut la preuve du corollaire 4.4.1.

Nous avons démontré le corollaire 4.4.1, qui garantit que SEA peut retrouver le support de x^* lorsque A est orthogonal et qu'il n'y a pas de bruit. En effet, dans ce cas, SEA se comporte exactement comme l'algorithme oracle. Nous allons à présent étudier un cas plus général, où A n'est pas nécessairement orthogonale[5], mais où A satisfait l'hypothèse RIP (Restricted Isometry Property).

4.5 Analyse de SEA dans le cas RIP et garanties de reconstructions

Avant de présenter le théorème de reconstruction sous hypothèse RIP, nous rappelons au lecteur la définition de l'hypothèse d'isométrie restreinte (RIP) ainsi que la définition équivalente démontrée dans le chapitre 1.

5. Les résultats établis dans cette section se généralisent aux matrices dont les colonnes forment une famille orthonormée. Cependant, étant donné qu'il s'agit d'un cas pédagogique, j'ai choisi de ne pas aborder cette nuance.

Définition 4.5.1. (*Propriété et constante d'isométrie restreinte (*CANDES *et al. 2005)*) Soit $A \in \mathbb{R}^{m\times n}$ un dictionnaire normalisé. Pour tout $k \in [\![1,n]\!]$, on définit la constante d'isométrie restreinte de A d'ordre k comme étant le plus petit nombre positif δ_k, tel que, pour tout $x \in \mathbb{R}^n$, tel que $\|x\|_0 \leq k$, on a

$$(1-\delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1+\delta_k)\|x\|_2^2.$$

De plus, si $\delta_k < 1$, on dit que A satisfait la $k-$RIP (ou propriété d'isométrie restreinte d'ordre k).
Il s'agit de mesurer à quel point A est proche d'une isométrie lorsqu'on l'applique à des vecteurs parcimonieux.

Propriété 4.5.2. (*Définition équivalente*) Soient $k \in [\![1,n]\!]$ et $A \in \mathbb{R}^{m\times n}$ un dictionnaire normalisé. Sont équivalents :

1. A satisfait la $k-$RIP.
2. Pour tout $S \subseteq [\![1,n]\!]$, tel que $\text{Card}(S) = k$, toutes les valeurs singulières de $A_{|S}$ sont dans l'intervalle $[\sqrt{1-\delta_k}, \sqrt{1+\delta_k}]$.

4.5.1 Énoncé du théorème de reconstruction et de son corollaire

Nous rappelons que pour tout $t \in \mathbb{N}$, le bruit du gradient $b^t \in \mathbb{R}^n$ est défini dans l'équation (4.5), et que la norme maximale du bruit du gradient est définie dans l'équation (4.6).

On définit la condition de reconstruction sous hypothèse RIP (RC_{RIP}) comme suit :

$$\gamma_k^{RIP}\|e\|_2 < \frac{\min_{i\in S^*}|x^*_{|i}|}{2k} - \alpha_k^{RIP}\|x^*\|_2. \qquad (RC_{RIP})$$

Théorème 4.5.3. (*Reconstruction par SEA : cas RIP*) Supposons que $A \in \mathbb{R}^{m\times n}$ satisfasse le $(2k+1)$-RIP et que pour tout $i \in [\![1,n]\!]$, $\|A_{|i}\|_2 = 1$.
Alors, pour tout $t \in \mathbb{N}$,

$$\frac{\|b^t\|_\infty}{\eta} = \|\frac{u^t}{\eta} - A^T(Ax^t - y)\|_\infty \leq \alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2, \qquad (4.23)$$

avec

$$\alpha_k^{RIP} = \delta_{2k+1}\left(\frac{\delta_{2k}}{1-\delta_k}+1\right) \in \mathbb{R}_+^* \qquad \text{et} \qquad \gamma_k^{RIP} = \delta_{2k+1}\frac{\sqrt{1+\delta_k}}{1-\delta_k} + 1 \in \mathbb{R}_+^*. \quad (4.24)$$

Si en outre x^* vérifie (RC_{RIP}), alors pour toutes les initialisations $\mathscr{X}^0$ et tout $\eta > 0$,

il existe $t_s \leq T_{RIP}$ tel que $S^* \subseteq S^{t_s}$, où

$$T_{RIP} = \frac{2k\frac{\|\mathscr{X}^0\|_\infty}{\eta} + (k+1)\min_{i\in S^*}|x^*_{|i}|}{\min_{i\in S^*}|x^*_{|i}| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)}. \tag{4.25}$$

Si de plus, x^* est tel que

$$\min_{i\in S^*}|x^*_{|i}| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2 \tag{4.26}$$

et que SEA réalise plus de T_{RIP} itérations, alors $S^* \subseteq S^{t_{BEST}}$ et

$$\|x^{t_{BEST}} - x^*\|_2 \leq \frac{2}{\sqrt{1-\delta_k}}\|e\|_2.$$

Remarque. (*Ordre de grandeur de α_k^{RIP} et γ_k^{RIP}*) Dès que δ_k est loin de 1 [a] (par exemple $\delta_k \leq \frac{1}{2}$), α_k^{RIP} est de l'ordre de grandeur de δ_{2k+1} (ou dans notre exemple $\delta_{2k+1} \leq \alpha_k^{RIP} \leq 3\delta_{2k+1}$) et γ_k^{RIP} est de l'ordre de grandeur de $1+\delta_{2k+1}$ (ou dans notre exemple $(1+\delta_{2k+1}) \leq \gamma_k^{RIP} \leq \sqrt{6}(1+\delta_{2k+1})$).

a. C'est à priori le cas, car A satisfait le $(2k+1)$-RIP.

Pour démontrer le théorème, nous allons d'abord rappeler dans la section 4.5.2 des propriétés connues des matrices satisfaisant le RIP. Nous bornons ensuite, dans le lemme 4.5.8 de la section 4.5.3, l'erreur commise lors de l'approximation de x^* sur un support spécifique S. Cette première borne conduit à une seconde borne, sur $\|b^t\|_\infty$ dans le lemme théorème 4.5.9 et à l'application du théorème 4.3.1 pour démontrer le théorème 4.5.3 dans la section 4.5.4. Nous appliquons enfin le théorème 4.5.3 dans la section 4.5.5 pour démontrer le corollaire 4.5.4 énoncé ci-après.

Nous rappelons que l'on définit la condition de reconstruction simplifiée sous hypothèse RIP (RC_{SRIP}) comme suit :

$$\Lambda := 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} < 1. \tag{RC_{SRIP}}$$

Corollaire 4.5.4. (*Reconstruction par SEA : cas RIP non bruité*) Supposons que $\|e\|_2 = 0$, que A satisfasse le $(2k+1)$-RIP et que pour tout $i \in [\![1,n]\!]$, $\|A_{|i}\|_2 = 1$. Si de plus x^* vérifie (RC_{SRIP}), alors x^* vérifie (RC_{RIP}). Par conséquent, pour $\eta > 0$, si $\mathscr{X}^0 = 0$ et que SEA réalise plus de $T_{SRIP} = \frac{k+1}{1-\Lambda}$ itérations, on a $S^* \subseteq S^{t_{BEST}}$ et $x^{t_{BEST}} = x^*$.

4.5.2 Propriétés issues du RIP

Nous partageons ici deux remarques et quatre propriétés des matrices satisfaisant le RIP.

Remarque. Si $A \in \mathbb{R}^{m\times n}$ satisfait la k-RIP, alors pour tout $S \subset [\![1,n]\!]$ tel que $|S| = k$, $A_{|S}$ est de rang plein. Ce sera toujours vrai, car $k < n < m$ (ou à défaut $k < m$) dans nos problèmes.

Remarque. La normalisation vise à simplifier les formules en garantissant que $\delta_1 = 0$. Elle est effectuée sans perte de généralité. Si A n'est pas normalisée, mais satisfait la k-RIP avec $k > 1$, sa normalisation n'a qu'un faible impact sur δ_k. En effet, si l'on considère $\Delta \in \mathbb{R}^{n\times n}$, une matrice diagonale définie par $\Delta_{i,i} = \|A_i\|_2$, alors $A\Delta^{-1}$ est normalisée, et, pour tout vecteur $x \in \mathbb{R}^n$ k-parcimonieux, on a

$$(1-\delta_k)\|\Delta^{-1}x\|_2^2 \leq \|A\Delta^{-1}x\|_2^2 \leq (1+\delta_k)\|\Delta^{-1}x\|_2^2.$$

En utilisant $1-\delta_1 \leq \|A_i\|_2^2 \leq 1+\delta_1$, il est possible de calculer la constante d'isométrie restreinte d'ordre k de la matrice normalisée $A\Delta^{-1}$.

Passons maintenant aux propriétés qui nous sont utiles par la suite.

Propriété 4.5.5. (*Croissance de la constante d'isométrie restreinte*) Pour tout $k, k' \in [\![1,n]\!]$, tels que $k \leq k'$, nous avons

$$\delta_k \leq \delta_{k'}.$$

Propriété 4.5.6. (*Contrôle de la norme de la composition avec la transposée*) Pour tout $R, S \subseteq [\![1,n]\!]$, tels que $R \cap S = \emptyset$ et A satisfaisant le $(|R|+|S|)$-RIP. Nous rappelons le lemme 1 de (Dai et al. 2009) (également décrit dans (Candes et al. 2005)). Pour tout $x \in \mathbb{R}^{|S|}$

$$\|A_{|R}^T A_{|S} x\|_2 \leq \delta_{|R|+|S|}\, \|x\|_2.$$

Démonstration. Par souci d'exhaustivité, nous allons prouver cette inégalité. Soient A, R, S et x tels que définis dans la propriété 4.5.6.

$$\|A_{|R}^T A_{|S} \frac{x}{\|x\|_2}\|_2 = \max_{x' \in \mathbb{R}^{|R|} : \|x'\|_2 = 1} \langle x', A_{|R}^T A_{|S} \frac{x}{\|x\|_2} \rangle.$$

En utilisant $\langle x', A_{|R}^T A_{|S} \frac{x}{\|x\|_2}\rangle = \langle A_{|R}x', A_{|S}\frac{x}{\|x\|_2}\rangle \leq \frac{1}{2}\|A_{|R}x' + A_{|S}\frac{x}{\|x\|_2}\|_2^2$, le fait que $R \cap S = \emptyset$, et le fait que A satisfasse le $(|R|+|S|)$-RIP défini dans la définition 4.5.1 ; on obtient que, pour tout $x' \in \mathbb{R}^{|R|}$ tel que $\|x'\|_2 = 1$,

$$\langle x', A_{|R}^T A_{|S} \frac{x}{\|x\|_2}\rangle \leq \frac{1}{2}(1+\delta_{|R|+|S|})\left(\|x'\|_2^2 + \|\frac{x}{\|x\|_2}\|_2^2\right) = (1+\delta_{|R|+|S|}).$$

Cela conclut la preuve de la propriété 4.5.6.

Nous terminons par un corollaire qui découle de la propriété 4.5.2.

Corollaire 4.5.7. (*Contrôle de la norme de $A_{|S}^T A_{|S}$*) Pour tout $S \subseteq [\![1, n]\!]$, tel que A satisfasse le $|S|$-RIP. Pour tout $x \in \mathbb{R}^{|S|}$ et tout $v \in \mathbb{R}^m$, on a

$$\|A_{|S}^T v\|_2 \leq \sqrt{1+\delta_{|S|}}\,\|v\|_2,$$
$$\|(A_{|S}^T A_{|S})^{-1} x\|_2 \leq \frac{1}{1-\delta_{|S|}}\|x\|_2,$$
$$\text{et } \|(A_{|S}^T A_{|S} - I_{|S|})x\|_2 \leq \delta_{|S|}\|x\|_2.$$

4.5.3 Préliminaires

Dans cette section, nous utilisons les propriétés énoncées dans la section 4.5.2 pour majorer l'erreur $\|(x^t - x^*)_{|S^t}\|_2$, où S^t est défini à la ligne 6 et x^t est défini à la ligne 7 de l'algorithme 1. Cette limite conduit à une majoration de $\|b^t\|_2$. Tout au long de la section, nous supposons que A satisfait le $(2k+1)$-RIP. Nous rappelons que la figure 4.2 à la page 83 peut aider à visualiser les différents ensembles d'indices considérés dans les preuves.

Lemme 4.5.8. (*Majoration de l'erreur d'approximation*) Si A satisfait le $(2k+1)$-RIP, alors pour tout $t \in \mathbb{N}$,

$$\|(x^t - x^*)_{|S^t}\|_2 \leq \frac{\delta_{2k}}{1-\delta_k}\frac{\|u^t\|_2}{\eta} + \frac{\sqrt{1+\delta_k}}{1-\delta_k}\|e\|_2.$$

Démonstration. Soit $t \in \mathbb{N}$, en utilisant la définition de x^t donnée dans l'algorithme 1 et celle de u^t dans (4.1), on trouve que

$$\begin{aligned} x^t{}_{|S^t} &= A_{|S^t}^\dagger y \\ &= A_{|S^t}^\dagger (A_{|S^*} x^*{}_{|S^*} + e) \\ &= A_{|S^t}^\dagger A_{|S^* \cap S^t} x^*{}_{|S^* \cap S^t} - \frac{1}{\eta} A_{|S^t}^\dagger A_{|S^* \cap \overline{S^t}} u^t{}_{|S^* \cap \overline{S^t}} + A_{|S^t}^\dagger e. \end{aligned} \tag{4.27}$$

On a également que

$$\begin{aligned} A_{|S^t}^\dagger A_{|S^* \cap S^t} x^*{}_{|S^* \cap S^t} &= A_{|S^t}^\dagger \begin{bmatrix} A_{|S^* \cap S^t} & A_{|S^t \setminus S^*} \end{bmatrix} \begin{bmatrix} x^*{}_{|S^* \cap S^t} \\ 0 \end{bmatrix} \\ &= A_{|S^t}^\dagger A_{|S^t} x^*{}_{|S^t}. \end{aligned} \tag{4.28}$$

Comme $\delta_{2k+1} < 1$, la propriété 4.5.5 implique que $\delta_k \leq \delta_{2k+1} < 1$ et la propriété 4.5.1 implique que la plus petite valeur singulière de $A_{|S^t}$ est plus grande

que $\sqrt{1-\delta_k} \geq \sqrt{1-\delta_{2k+1}} > 0$. Ainsi, $A_{|S^t}$ est de rang plein et

$$A^{\dagger}_{|S^t} = (A^T_{|S^t} A_{|S^t})^{-1} A^T_{|S^t}. \tag{4.29}$$

En combinant (4.27), (4.28) et (4.29), on obtient

$$x^t{}_{|S^t} = x^*{}_{|S^t} - \frac{1}{\eta} A^{\dagger}_{|S^t} A_{|S^* \cap \overline{S^t}} u^t{}_{|S^* \cap \overline{S^t}} + A^{\dagger}_{|S^t} e.$$

Puis, avec (4.29), on trouve

$$\begin{aligned}
\|(x^t - x^*)_{|S^t}\|_2 &= \|\frac{1}{\eta} A^{\dagger}_{|S^t} A_{|S^* \cap \overline{S^t}} u^t{}_{|S^* \cap \overline{S^t}} - A^{\dagger}_{|S^t} e\|_2 \\
&\leq \frac{1}{\eta} \|(A^T_{|S^t} A_{|S^t})^{-1} A^T_{|S^t} A_{|S^* \cap \overline{S^t}} u^t{}_{|S^* \cap \overline{S^t}}\|_2 + \|(A^T_{|S^t} A_{|S^t})^{-1} A^T_{|S^t} e\|_2.
\end{aligned}$$

Enfin, en utilisant le corollaire 4.5.7, puis la propriété 4.5.6, la propriété 4.5.5, le corollaire 4.5.7 et (4.1), on peut terminer la preuve par

$$\begin{aligned}
\|(x^t - x^*)_{|S^t}\|_2 &\leq \frac{1}{1-\delta_k} \left(\frac{1}{\eta} \|A^T_{|S^t} A_{|S^* \cap \overline{S^t}} u^t{}_{|S^* \cap \overline{S^t}}\|_2 + \|A^T_{|S^t} e\|_2 \right) \\
&\leq \frac{1}{1-\delta_k} \left(\frac{\delta_{2k}}{\eta} \|u^t{}_{|S^* \cap \overline{S^t}}\|_2 + \sqrt{1+\delta_k} \|e\|_2 \right) \\
&= \frac{\delta_{2k}}{1-\delta_k} \frac{\|u^t\|_2}{\eta} + \frac{\sqrt{1+\delta_k}}{1-\delta_k} \|e\|_2.
\end{aligned}$$

En utilisant le lemme que l'on vient de démontrer, nous pouvons établir la majoration de $\|b^t\|_\infty$ donnée dans le théorème 4.5.3.

Lemme 4.5.9. (*Majoration de* $\|b^t\|_\infty$) Si A satisfait le $(2k+1)$-RIP, alors pour tout $t \in \mathbb{N}$,

$$\|\frac{u^t}{\eta} - A^T(Ax^t - y)\|_\infty = \frac{1}{\eta} \|b^t\|_\infty \leq \alpha_k^{RIP} \|x^*\|_2 + \gamma_k^{RIP} \|e\|_2,$$

où α_k^{RIP} et γ_k^{RIP} sont définies dans (4.24).

Démonstration. Soient $t \in \mathbb{N}$ et $i \in [\![1, n]\!]$, nous rappelons que b^t est défini dans (4.5), on a

$$\begin{aligned}
|b^t_{|i}| &= |u^t{}_{|i} - \eta (A_{|i})^T (Ax^t - y)| \\
&= |u^t{}_{|i} - \eta (A_{|i})^T A(x^t - x^*) + \eta (A_{|i})^T e| \\
&= |u^t{}_{|i} - \eta (A_{|i})^T A_{|S^* \cup S^t} (x^t - x^*)_{|S^* \cup S^t} + \eta (A_{|i})^T e|.
\end{aligned} \tag{4.30}$$

On distingue trois cas : $i \in S^t$, $i \in \overline{S^*} \cap \overline{S^t}$ et $i \in S^* \cap \overline{S^t}$. Nous prouvons que, dans

ces trois cas,

$$|b^t_{|i}| \leq \eta\left(\delta_{2k+1}\|x^t - x^*\|_2 + \|e\|_2\right). \tag{4.31}$$

$\underline{1^{er}\ cas}$: Si $i \in S^t$, en utilisant le lemme 4.3.2 on voit que $b^t_{|i} = 0$ et que (4.31) est vraie.
$\underline{2^{ème}\ cas}$: Si $i \in \overline{S^*} \cap \overline{S^t}$, en utilisant la définition de u^t donnée dans (4.1), la propriété 4.5.6, la propriété 4.5.5 et le fait que $\|A_{|i}\|_2 = 1$ on obtient que

$$\begin{aligned}
|b^t_{|i}| &= |-\eta(A_{|i})^T A_{|S^*\cup S^t}(x^t - x^*)_{|S^*\cup S^t} + \eta(A_{|i})^T e| \\
&\leq \eta\left(\|(A_{|i})^T A_{|S^*\cup S^t}(x^t - x^*)_{|S^*\cup S^t}\|_2 + \|(A_{|i})^T e\|_2\right) \\
&\leq \eta\left(\delta_{2k+1}\|(x^t - x^*)_{|S^*\cup S^t}\|_2 + \|e\|_2\right) \\
&= \eta\left(\delta_{2k+1}\|x^t - x^*\|_2 + \|e\|_2\right)
\end{aligned}$$

$\underline{3^{ème}\ cas}$: Si $i \in S^* \cap \overline{S^t}$, comme $\overline{\{i\}}$ est le complémentaire de $\{i\} \subseteq [\![1,n]\!]$, et comme $(A_{|i})^T A_{|\{i\}} = \|A_{|i}\|_2 = 1$ et que $x^t{}_{|i} = 0$, (4.30) devient

$$\begin{aligned}
|b^t_{|i}| &= |-\eta x^*_{|i} - \eta(A_{|i})^T A(x^t - x^*) + \eta(A_{|i})^T e| \\
&= \eta|-x^*_{|i} - (A_{|i})^T A_{|\{i\}}(x^t - x^*)_{|\{i\}} - (A_{|i})^T A_{|\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_{|i})^T e| \\
&= \eta|-x^*_{|i} - (x^t{}_{|i} - x^*_{|i}) - (A_{|i})^T A_{|\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_{|i})^T e| \\
&= \eta|-(A_{|i})^T A_{|\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_{|i})^T e| \\
&\leq \eta\left(|(A_{|i})^T A_{|\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}}| + |(A_{|i})^T e|\right).
\end{aligned}$$

En utilisant la propriété 4.5.6, la propriété 4.5.5 et $\|A_{|i}\|_2 = 1$, on obtient que

$$\begin{aligned}
|b^t_{|i}| &\leq \eta\left(\delta_{2k}\|x^t - x^*\|_2 + \|e\|_2\right) \\
&\leq \eta\left(\delta_{2k+1}\|x^t - x^*\|_2 + \|e\|_2\right).
\end{aligned}$$

En regroupant les trois cas, on conclut que pour tout $i \in [\![1,n]\!]$, (4.31) est vraie.
Nous allons maintenant terminer la preuve.
En utilisant (4.1) suivi du lemme 4.5.8, on trouve que

$$\begin{aligned}
|b^t_{|i}| &\leq \eta\left(\delta_{2k+1}\left(\|(x^t - x^*)_{|S^t}\|_2 + \frac{\|u^t\|_2}{\eta}\right) + \|e\|_2\right) \\
&\leq \eta\left(\delta_{2k+1}\left(\frac{\delta_{2k}}{1-\delta_k} + 1\right)\frac{\|u^t\|_2}{\eta} + \left(\delta_{2k+1}\frac{\sqrt{1+\delta_k}}{1-\delta_k} + 1\right)\|e\|_2\right) \\
&\leq \eta\left(\alpha^{RIP}_k\|x^*\|_2 + \gamma^{RIP}_k\|e\|_2\right),
\end{aligned}$$

où la dernière inégalité est vraie car $\frac{\|u^t\|_2}{\eta} \leq \|x^*\|_2$.

4.5.4 Preuve du théorème 4.5.3 : reconstruction dans le cas RIP

Nous rappelons le théorème 4.5.3 avant de le démontrer.

Théorème 4.5.3. (*Reconstruction par SEA : cas RIP*) Supposons que $A \in \mathbb{R}^{m\times n}$ satisfasse le $(2k+1)$-RIP et que pour tout $i \in [\![1,n]\!]$, $\|A_{|i}\|_2 = 1$.
Alors, pour tout $t \in \mathbb{N}$,

$$\frac{\|b^t\|_\infty}{\eta} = \|\frac{u^t}{\eta} - A^T(Ax^t - y)\|_\infty \leq \alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2, \tag{4.23}$$

avec

$$\alpha_k^{RIP} = \delta_{2k+1}\left(\frac{\delta_{2k}}{1-\delta_k}+1\right) \in \mathbb{R}_+^* \qquad \text{et} \qquad \gamma_k^{RIP} = \delta_{2k+1}\frac{\sqrt{1+\delta_k}}{1-\delta_k}+1 \in \mathbb{R}_+^*. \tag{4.24}$$

Si en outre x^* vérifie (RC_{RIP}), alors pour toutes les initialisations $\mathscr{X}^0$ et tout $\eta > 0$, il existe $t_s \leq T_{RIP}$ tel que $S^* \subseteq S^{t_s}$, où

$$T_{RIP} = \frac{2k\frac{\|\mathscr{X}^0\|_\infty}{\eta} + (k+1)\min_{i\in S^*}|x_{|i}^*|}{\min_{i\in S^*}|x_{|i}^*| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)}. \tag{4.25}$$

Si, de plus, x^* est tel que

$$\min_{i\in S^*}|x_{|i}^*| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2 \tag{4.26}$$

et que SEA réalise plus de T_{RIP} itérations, alors $S^* \subseteq S^{t_{BEST}}$ et

$$\|x^{t_{BEST}} - x^*\|_2 \leq \frac{2}{\sqrt{1-\delta_k}}\|e\|_2.$$

Démonstration. Supposons que A satisfasse le $(2k+1)$-RIP et que x^* satisfasse (RC_{RIP}). Nous rappelons que T_{max} est défini dans (4.7) et T_{RIP} dans (4.25). En utilisant (4.6) et le lemme 4.5.9, on a

$$\varepsilon = \sup_{t\in\mathbb{N}}\|b^t\|_\infty \leq \eta\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right). \tag{4.32}$$

Combiné avec (RC_{RIP}), donné par $\gamma_k^{RIP}\|e\|_2 < \frac{\min_{i\in S^*}|x_{|i}^*|}{2k} - \alpha_k^{RIP}\|x^*\|_2$, on trouve que

$$\varepsilon < \frac{\eta}{2k}\min_{i\in S^*}|x_{|i}^*|.$$

Ainsi (RC) est vérifiée et le théorème 4.3.1 implique qu'il existe $t_s \leq T_{max}$ tel que

$S^* \subseteq S^{t_s}$, avec

$$T_{max} = \frac{2k\|\mathscr{X}^0\|_\infty + (k+1)\eta \min_{i\in S^*}|x^*_{|i}|}{\eta \min_{i\in S^*}|x^*_{|i}| - 2k\varepsilon}.$$

En utilisant (4.32), on obtient

$$T_{max} \leq \frac{2k\|\mathscr{X}^0\|_\infty + (k+1)\eta \min_{i\in S^*}|x^*_{|i}|}{\eta \min_{i\in S^*}|x^*_{|i}| - 2k\eta\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)} = T_{RIP}.$$

On doit encore prouver que lorsque $\min_{i\in S^*}|x^*_{|i}| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2$, t_{BEST} satisfait $S^* \subseteq S^{t_{BEST}}$ ainsi que la dernière majoration du théorème 4.5.3 .
Supposons par l'absurde que l'inégalité

$$\min_{i\in S^*}|x^*_{|i}| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2 \tag{4.33}$$

est vraie, mais que $S^* \not\subset S^{t_{BEST}}$. La construction de t_{BEST}, à la ligne 11 de l'algorithme 1, et l'existence de t_s tel que $S^* \subseteq S^{t_s}$ garantissent que

$$\|Ax^{t_{BEST}} - y\| \leq \|Ax^{t_s} - y\| \leq \|Ax^* - y\| = \|e\|_2.$$

Ainsi, en utilisant la partie gauche de l'inégalité donnée dans la définition 4.5.1, on obtient

$$\begin{aligned}\sqrt{1-\delta_{2k}}\|x^{t_{BEST}} - x^*\|_2 &\leq \|A(x^{t_{BEST}} - x^*)\|_2 \\ &\leq \|Ax^{t_{BEST}} - y\|_2 + \|Ax^* - y\|_2 \\ &\leq 2\|e\|_2.\end{aligned}$$

D'autre part, comme on a supposé que $S^* \not\subset S^{t_{BEST}}$, on a

$$\|x^{t_{BEST}} - x^*\|_2 \geq \min_{i\in S^*}|x^*_{|i}|.$$

On conclut que $\min_{i\in S^*}|x^*_{|i}| \leq \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2$. Ce qui contredit (4.33).
Pour conclure, lorsque $\min_{i\in S^*}|x^*_{|i}| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2$, on a $S^* \subseteq S^{t_{BEST}}$.
Dans ce cas, puisque le support de $x^{t_{BEST}} - x^*$ est de taille inférieure ou égale à k, on peut refaire les calculs ci-dessus et obtenir

$$\sqrt{1-\delta_k}\|x^{t_{BEST}} - x^*\|_2 \leq \|A(x^{t_{BEST}} - x^*)\|_2 \leq 2\|e\|_2.$$

Cela mène à la dernière inégalité du théorème 4.5.3 et conclut la preuve.

4.5.5 Preuve du corollaire 4.5.4 : le cas RIP non bruité

Nous rappelons le corollaire 4.5.4 avant de le démontrer.

Corrolaire 4.5.4. (*Reconstruction par SEA : cas RIP non bruité*) Supposons que $\|e\|_2 = 0$, que A satisfasse le $(2k+1)$-RIP et que pour tout $i \in [\![1, n]\!]$, $\|A_{|i}\|_2 = 1$. Si de plus x^* vérifie (RC_{SRIP}), alors x^* vérifie (RC_{RIP}). Par conséquent, pour $\eta > 0$, si $\mathscr{X}^0 = 0$ et que SEA réalise plus de $T_{SRIP} = \frac{k+1}{1-\Lambda}$ itérations, on a $S^* \subseteq S^{t_{BEST}}$ and $x^{t_{BEST}} = x^*$.

Démonstration. Supposons que x^* satisfasse (RC_{SRIP}) et que $\|e\|_2 = 0$. Prouvons d'abord que x^* vérifie (RC_{RIP}). En utilisant (RC_{SRIP}), on a

$$\begin{aligned} 0 < 1 - \Lambda \leq 1 - 2k\alpha_k^{RIP} \frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} \\ = \frac{2k}{\min_{i\in S^*}|x^*_{|i}|}\left(\frac{\min_{i\in S^*}|x^*_{|i}|}{2k} - \alpha_k^{RIP}\|x^*\|_2\right). \end{aligned}$$

Ainsi, puisque $2k > 0$ et que $\min_{i\in S^*}|x^*_{|i}| > 0$,

$$0 < \frac{\min_{i\in S^*}|x^*_{|i}|}{2k} - \alpha_k^{RIP}\|x^*\|_2. \tag{4.34}$$

On en conclut que x^* vérifie (RC_{RIP}) pour A.
En appliquant le théorème 4.5.3 avec $\|e\|_2 = 0$ et $\mathscr{X}^0 = 0$, on trouve qu'il existe $t \leq T_{RIP} = \frac{(k+1)}{1-2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|}}$ tel que $S^* \subseteq S^t$. Il n'est pas difficile de vérifier que la fonction $f : \mathbb{R} \longrightarrow \mathbb{R}$ définie pour tout $v \in \mathbb{R}$ par $f(v) = \frac{k+1}{1-v}$ est croissante $[0, 1[$.
En appliquant f à

$$0 \leq 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} \leq \Lambda < 1,$$

on obtient

$$T_{RIP} = f\left(2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|}\right) \leq f(\Lambda) = T_{SRIP},$$

où T_{SRIP} est définie dans le corollaire 4.5.4.
Ainsi, $t \leq T_{SRIP}$. On peut conclure qu'il existe $t \leq T_{SRIP}$ tel que $S^* \subseteq S^t$.
La dernière affirmation du corollaire 4.5.4 est une conséquence directe du théorème 4.5.3 car x^* satisfait (4.26) avec $\|e\| = 0$.
Cela conclut la preuve du corollaire 4.5.4.

4.5.6 Discussion

Dans cette section, nous discutons de la signification des résultats obtenus dans le théorème 4.5.3 et le corollaire 4.5.4.

Commençons par examiner T_{RIP}. Lorsque (RC_{RIP}) s'applique, T_{RIP} augmente, quand $\min_{i\in S^*}|x^*_{|i}| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)$ diminue. En particulier, le nombre d'itérations requises par l'algorithme pour fournir la solution correcte augmente lorsque l'information sur certaines colonnes indicées par S^* diminue, c'est-à-dire quand $\min_{i\in S^*}|x^*_{|i}|$ diminue. De même, les initialisations $\mathscr{X}^0 \neq 0$ ont un impact négatif apparent sur le nombre d'itérations nécessaires dans le pire des cas. En effet, dans le pire des cas, $\mathscr{X}^0$ est mal choisi et SEA a besoin d'un certain nombre d'itérations pour corriger ce mauvais choix (comme dans le cas général).

Quand les conditions du théorème 4.5.3 sont remplies, n'importe quel $\mathscr{X}^0$ et η permettent la reconstruction de S^*. $\mathscr{X}^0$ et η n'influencent que T_{RIP}. À cet égard, étant donné que plus η est grand, plus vite SEA écrase l'initialisation $\mathscr{X}^0$, le choix de η est très lié à la question de la qualité de l'initialisation. Cette dernière est souvent bénéfique dans la pratique (sauf dans les cas où la cohérence μ de A est élevée).

Par rapport aux garanties de reconstruction de supports pour le LASSO (Wainwright 2009 ; Meinshausen et al. 2006 ; Zhao et al. 2006), OMP (Cai et al. 2011), HTP (Foucart 2011 ; X. Yuan et al. 2016) et ARHT (Axiotis et al. 2020), les conditions de reconstruction fournies dans le théorème 4.5.3 et le corollaire 4.5.4 pour SEA sont plus strictes. Toutes les conditions des algorithmes de l'état de l'art impliquent une condition sur l'incohérence de A et une condition similaire à (4.26). Dans le cas de l'algorithme LASSO, cette dernière n'est pas très explicite. Cependant, aucune des conditions de reconstruction de supports n'implique une condition telle que (RC_{RIP}) et (RC_{SRIP}).

Développons ces deux conditions.

Une limitation notable de (RC_{RIP}) et (RC_{SRIP}) est que si $\alpha_k^{RIP} \neq 0$, il n'y a aucune garantie de reconstruire le support d'un x^* tel que $\max_{i\in S^*}|x^*_{|i}| \gg \min_{i\in S^*}|x^*_{|i}|$. En effet, en augmentant $x^*_{|j}$, pour un $j \in S^*$ tel que $|x^*_{|j}| \neq \min_{i\in S^*}|x^*_{|i}|$, nous pouvons maintenir $\min_{i\in S^*}|x^*_{|i}|$ tout en augmentant suffisamment $\|x^*\|_2$, ce qui fait échouer (RC_{RIP}) et (RC_{SRIP}). Nous n'avons pas ce problème avec la condition de l'équation (4.26). Nous avons expérimenté l'influence de $\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|}$ sur les performances de reconstruction dans les expériences de la section 5.2 du prochain chapitre.

Une autre question cruciale se pose : quelle condition (nécessaire et suffisante) sur α_k^{RIP}, γ_k^{RIP} et $\|e\|_2$ garantit l'existence de x^* satisfaisant (RC_{RIP}) ou (RC_{SRIP}) ?

Pour répondre à cette question pour la condition (RC_{RIP}), nous remarquons tout d'abord qu'il existe x^* satisfaisant (RC_{RIP}) si et seulement s'il existe une constante $c \in \mathbb{R}$ et x^*, telle que pour tout $i \in S^*$, $x^*_{|i} = c$. En effet, imaginons que x^* satisfasse (RC_{RIP}), et prenons $c = \min_{i\in S^*}|x^*_{|i}|$. Posons ensuite $\tilde{x}^* \in \mathbb{R}^n$, défini pour tout indice $i \in [\![1, n]\!]$ par,

$$\tilde{x}^*_{|i} = \begin{cases} c & \text{si } i \in S^*, \\ 0 & \text{sinon.} \end{cases}$$

Alors $\tilde{x}^*$ vérifie également (RC_{RIP}), puisque nous avons $\min_{i\in S^*}|\tilde{x}^*_{|i}| = \min_{i\in S^*}|x^*_{|i}|$ et $\|\tilde{x}^*\|_2 \leq \|x^*\|_2$. La réciproque est également vraie.

Pour trouver une condition garantissant l'existence d'un x^* satisfaisant (RC_{RIP}), nous pouvons réécrire (RC_{RIP}) lorsque pour tout $i \in S^*$, $x^*_{|i} = c$ pour une constante $c \in \mathbb{R}$ et obtenir

$$\gamma_k^{RIP}\|e\|_2 < |c|\left(\frac{1-2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}}{2k}\right).$$

L'existence de $c \in \mathbb{R}$ pour que cette condition soit remplie dépend uniquement du signe de $1-2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}$. Si $1-2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}} \leq 0$, il n'existe pas de c satisfaisant la condition ; si $1-2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}} > 0$, tout $c \in \mathbb{R}$ satisfaisant la condition

$$|c| \geq \left(\frac{2k\gamma_k^{RIP}}{1-2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}}\right)\|e\|_2$$

conduit à un x^* qui vérifie (RC_{RIP}). Par conséquent, lorsque $|S^*| = k$, la condition $\alpha_k^{RIP} < \frac{1}{2}k^{-\frac{3}{2}}$ est nécessaire et suffisante pour garantir l'existence d'un x^* satisfaisant (RC_{RIP}).

Nous rappelons que α_k^{RIP} est du même ordre de grandeur que δ_{2k+1}. Par conséquent, la condition $\alpha_k^{RIP} < \frac{1}{2}k^{-\frac{3}{2}}$ est plus stricte que les conditions équivalentes pour d'autres algorithmes de l'état de l'art. Par exemple, les conditions décrites dans (AXIOTIS et al. 2020) sont de la forme $\delta_{2k} < C$, pour une constante universelle $C < 1$.

Initialiser SEA avec la solution d'un algorithme bénéficiant de meilleures conditions de reconstruction est un moyen simple et efficace pour que SEA hérite de sa garantie de reconstruction de supports dès que (4.26) est vérifiée. Cela peut être prouvé formellement en utilisant la même preuve que dans la section 4.5.4.

Attardons nous maintenant sur la condition (RC_{SRIP}). Comme énoncé précédemment pour (RC_{RIP}), si α_k^{RIP} est trop grand, il n'existe pas de x^* satisfaisant (RC_{SRIP}). C'est par exemple le cas si $\alpha_k^{RIP} \geq 0.5$. Au contraire, une condition suffisante d'existence de vecteurs x^* satisfaisant (RC_{SRIP}) est que la constante α_k^{RIP} satisfasse $\Lambda = 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} < 1$. Dans ce cas, lorsque toutes les entrées non nulles de x^* sont égales, nous avons $\|x^*\|_2 = \sqrt{|S^*|}\min_{i\in S^*}|x^*_{|i}|$ et

$$\Lambda = 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x^*_{|i}|} = 2k\alpha_k^{RIP}\sqrt{|S^*|} \leq 2k^{\frac{3}{2}}\alpha_k^{RIP} < 1.$$

En résumé, comme pour (RC_{RIP}), lorsque $\alpha_k^{RIP} < \frac{1}{2}k^{-\frac{3}{2}}$, il existe des vecteurs x^* satisfaisant (RC_{SRIP}) et la condition du corollaire 4.5.4 n'est pas vide.

Comme nous le verrons plus loin dans les expériences du chapitre 5, SEA fonctionne bien même lorsque A est cohérent. Ceci n'est expliqué ni par le théorème 4.5.3, ni le corollaire 4.5.4 qui utilisent l'hypothèse RIP. Le principal intérêt des résultats théoriques de ce chapitre réside dans le fait qu'ils s'appliquent à une instance de l'estimateur direct. Une autre théorie doit être développée pour expliquer le bon comportement de SEA pour la reconstruction de support parcimonieux lorsque A est cohérent.

4.6 Conclusions

Dans ce chapitre, nous avons étudié les garanties de reconstruction de SEA. Les garanties de reconstruction de supports de SEA, contrairement aux garanties de reconstructions de supports des algorithmes de l'état de l'art, ne sont pas basées sur un argument de « décroissance » de F ou $\|x^{t+1} - x^t\|_2$. Elles sont basées sur la majoration de l'écart entre les différentes mises à jour de la variable d'exploration de supports $\mathscr{X}$. En effet, si la mise à jour utilisant l'estimateur direct (donc celle de SEA) et celle suivant la direction d'un oracle sont proches, SEA est capable de reconstruire le support de la solution. Ainsi, nous avons borné cet écart et établi des garanties lorsque la matrice A satisfait la propriété d'isométrie restreinte (RIP), ce qui, je l'espère, permet d'améliorer la compréhension de l'estimateur direct. Les garanties théoriques impliquent des conditions sur x^* qui ne sont pas présentes dans les garanties des autres algorithmes. Cela pourrait restreindre leur applicabilité.

5 Applications de SEA

Sommaire

5.1 Introduction . . . 108
5.2 Déconvolution de signaux . . . 109
5.2.1 Contexte . . . 109
5.2.2 Étude d'un exemple . . . 111
5.2.2.1 Résultats de la déconvolution . . . 112
5.2.2.2 Suivi de F pendant le processus itératif . . . 113
5.2.3 Généralisation des résultats . . . 116
5.2.3.1 Analyse des performances . . . 116
5.2.3.2 Analyse de l'exploration . . . 118
5.2.3.3 Impact du pas sur IHT et HTP . . . 120
5.2.4 Étude d'autres configurations . . . 120
5.2.4.1 Impact d'une mauvaise estimation de la parcimonie k . . . 120
5.2.4.2 Robustesse au bruit . . . 122
5.2.4.3 Impact d'une parcimonie approximative . . . 123
5.2.4.4 Impact de la variation de l'amplitude de x^* . . . 124
5.3 Apprentissage supervisé . . . 124
5.3.1 Contexte . . . 124
5.3.2 Tâches de régression . . . 125
5.3.3 Tâches de classification binaire par régression logistique . . . 127
5.4 Conclusions . . . 128

Nous avons présenté SEA dans le chapitre 3. Nous l'avons ensuite analysé théoriquement dans le chapitre 4. Ce chapitre est consacré à l'application de SEA à des problèmes artificiels. Nous montrons ici que SEA résout des problèmes qui sont hors des hypothèses théoriques du chapitre 4. Nous expliquons également comment utiliser SEA pour réaliser des tâches d'apprentissage automatique. Ce chapitre reprend une partie du contenu de (MOHAMED, MALGOUYRES et al. 2024).

5.1 Introduction

Comme nous l'avons vu dans le chapitre 1, la reconstruction d'objets parcimonieux est un problème qui peut se décliner sous de nombreuses formes et dans de nombreux domaines. Nous nous intéressons à deux problèmes, l'un provenant du traitement

du signal et l'autre de l'apprentissage automatique. Dans la section 5.2, nous nous intéressons à la déconvolution de signaux parcimonieux. Dans la section 5.3, nous nous intéressons à des tâches de régression linéaire et logistique pour lesquelles on souhaite effectuer une séléction de variable pour obtenir de bonnes performances.

Dans chacun des cas, nous comparons SEA à des algorithmes de l'état de l'art [1]. Comme dans la section 3.5 du chapitre 3, nous comparons SEA à Iterative Hard Thresholding (IHT) (BLUMENSATH et Mike E. DAVIES 2009), Hard Thresholding Pursuit (HTP) (FOUCART 2011), Orthogonal Matching Pursuit (MALLAT et al. 1993 ; PATI et al. 1993), OMP with Replacement (OMPR) (JAIN et al. 2011) et Exhaustive Local Search (ELS) (AXIOTIS et al. 2020). OMPR et ELS sont initialisés avec la solution de OMP. Trois versions de SEA sont étudiées : la version avec démarrage à froid SEA_0, où SEA est initialisé avec le vecteur nul, et les versions avec démarrage à chaud SEA_{ELS} et SEA_{OMP}, où SEA est respectivement initialisé avec les solutions d'ELS et d'OMP. Pour des raisons d'équité, nous étudions également HTP et IHT initialisés avec OMP et ELS. Ils sont respectivement appelés HTP_{OMP}, HTP_{ELS}, IHT_{OMP} et IHT_{ELS}.

Comme lors de l'étude menée dans la section 3.5, les colonnes de A sont normalisées avant l'exécution de chaque algorithme, et, chaque estimation par moindres carrés pour un support fixe, comme à la ligne 7 de algorithme 1, est résolue en utilisant l'algorithme de descente de gradient conjugué de SciPy (VIRTANEN et al. 2020). Les résultats de HTP et, dans une moindre mesure, d'IHT et de SEA dépendent du choix du pas. Dans un premier temps, par souci d'équité de la comparaison avec OMP, OMPR et ELS, nous n'optimisons pas le choix de la taille du pas. Le pas de SEA, HTP et IHT est arbitrairement fixé à $\eta = \frac{1,8}{L}$, où L est la constante de Lipschitz du gradient de F (carré du rayon spectral de A ou carré de sa plus grande valeur singulière). Dans un second temps, l'impact de la variation du pas de HTP et IHT est étudié spécifiquement dans la section 5.2.3.3. Contrairement à la section 3.5, le nombre maximal d'itérations est cette fois-ci fixé à 1000 pour tous les algorithmes.

5.2 Déconvolution de signaux

5.2.1 Contexte

Les problèmes de déconvolution sont présents dans de nombreux domaines du traitement du signal tels que la microscopie ou la télédétection. Un intérêt de la déconvolution de signaux parcimonieux, également connue sous le nom de déconvolution de sources ponctuelles (BERNSTEIN et al. 2019) ou de déconvolution de pics (DUVAL et al. 2015 ; DUVAL et al. 2017), réside dans l'hypothèse que l'opérateur de convolution est connu (contrairement aux approches aveugles (KUO et al. 2019)). Le problème est le suivant. Nous fixons $n = m = 500$ et $k = 20$. Prenons un signal $x^* \in \mathbb{R}^n$, k-parcimonieux

1. Nous ne nous sommes pas comparés aux algorithmes de type Branch-and-Bound dans les expériences, car, au moment de leur réalisation, nous n'avons pas pu installer la bibliothèque contenant leur implémentation optimisée. Étant donné la taille des problèmes traités, il ne nous a pas semblé pertinent de les implémenter nous-mêmes.

aléatoire. Les indices du support S^* sont choisis au hasard, uniformément et sans remplacement. Les entrées non nulles de x^* sont tirées uniformément dans $[-2,-1]\cup[1,2]$ comme dans (ELAD 2010). Une illustration d'un tel signal est présentée sur la figure 5.1.

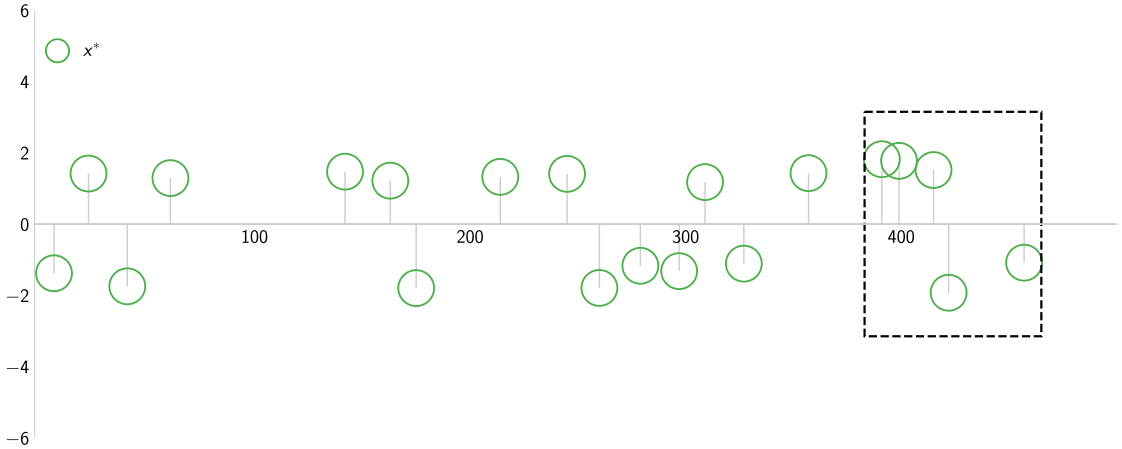

FIGURE 5.1 – Exemple d'instance d'un signal x^*, k-parcimonieux, de taille $n = 500$ avec $k = 20$.

Prenons une matrice de convolution $A \in \mathbb{R}^{m\times n}$ correspondant à un filtre gaussien d'écart-type égal à 3. Nous pouvons observer sur la figure 5.2 un signal non bruité $y \in \mathbb{R}^m$ qui est le résultat de la convolution de x^* par le filtre gaussien A.

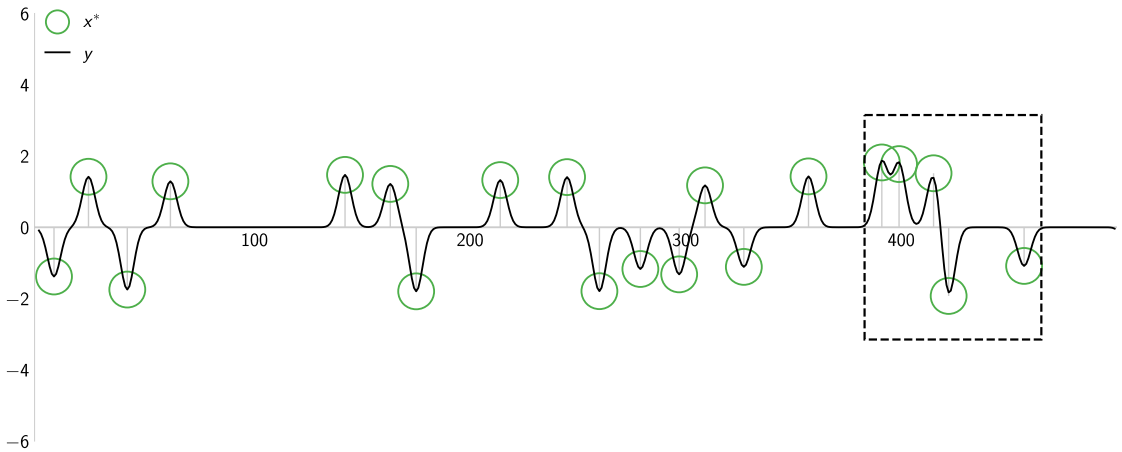

FIGURE 5.2 – Exemple d'instance d'observation y de taille $n = 500$ résultant de la convolution de x^* par un filtre gaussien A.

Résoudre le problème de déconvolution consiste à retrouver x^* (représenté par les cercles verts) à partir de y (la courbe noire) et A. Sur la figure 5.2, on peut remarquer deux types de situations. Une premiere dans laquelle les pics sont bien séparés (comme sur la partie gauche du signal), et un second où les pics sont proches (comme dans la zone autour de 400, délimitée par les tirets). Dans le premier cas,

les algorithmes de déconvolution ont tendance à bien fonctionner. Dans le second cas, qui est dû à la cohérence mutuelle élevée de A ($\mu = \max_{i \neq j} |\langle A_{|i}, A_{|j} \rangle| = 0.97$), les algorithmes rencontrent des difficultés pour retrouver le support. Le problème est plus difficile que cela, car le signal observé est souvent bruité. La figure 5.3 montre une version bruitée du signal de la figure 5.2, avec un bruit tiré uniformément sur la sphère de rayon $0.1\|Ax^*\|_2$ de $\mathbb{R}^m$, visant un rapport signal sur bruit de $SNR = 10\log_{10}(\|Ax^*\|_2 / \|e\|_2) = 20$dB, de manière similaire à (BLANCHARD et al. 2015).

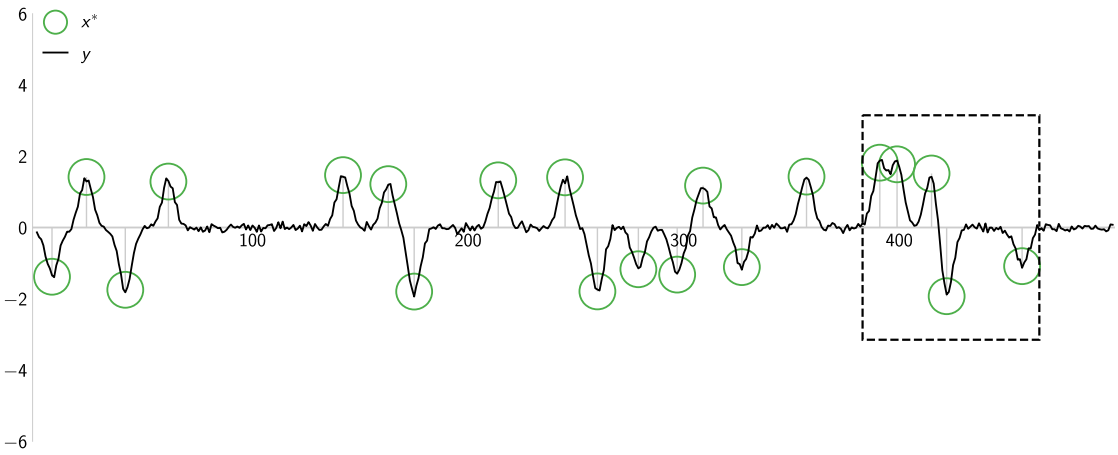

FIGURE 5.3 – Exemple d'instance d'observation bruitée y de taille $n = 500$ résultant de la convolution de x^* par un filtre gaussien A.

Comme on peut le voir, ce niveau de bruit est suffisant pour rendre le sommet des pics difficilement localisables.

Nous étudions l'efficacité de SEA pour résoudre ce problème de déconvolution dans les cas bruité et non bruité. Pour cela, nous étudions le comportement de SEA et des algorithmes de l'état de l'art sur les signaux des figures 5.2 et 5.3 dans la section 5.2.2. Nous analysons ensuite, dans la section 5.2.3, dans quelle mesure les résultats obtenus se généralisent à d'autres instances du problèmes. Enfin, dans la section 5.2.4, nous étudions l'impact de la modification de certains paramètres du problème tels que le niveau de bruit, la sur- ou sous-estimation de la parcimonie de x^* et ce qu'il se passe quand x^* est approximativement parcimonieux ou quand on change l'amplitude des pics de x^*.

5.2.2 Étude d'un exemple

Nous étudions le comportement de SEA et des algorithmes de l'état de l'art sur les signaux des figures 5.2 et 5.3. Dans un premier temps, nous analysons leur capacité à retrouver le support de x^* dans la section 5.2.2.1. Nous analysons ensuite l'évolution de la fonction de coût pendant le processus itératif dans la section 5.2.2.2.

5.2.2.1 Résultats de la déconvolution

Sur la figure 5.4, nous présentons le signal de la figure 5.3 accompagné des solutions fournies par les algorithmes étudiés. La figure 5.5 est une version agrandie de la région la plus dense du signal.

De manière générale, presque tous les algorithmes peuvent identifier des pics isolés. Cependant, les algorithmes ne parviennent pas toujours à identifier avec précision les pics lorsqu'ils sont proches les uns des autres. Par exemple, ELS, OMP et OMPR font des erreurs de détection dans la partie la plus énergétique du signal (autour de la position 400) et sont piégés dans un minimum local. On voit également que IHT et HTP font des erreurs de détection dans les parties les plus énergétiques du signal (autour des positions 140, 180, 260 et 400), se retrouvant piégés dans des minima locaux. Dans cette expérience (ce n'est pas le cas en général), l'initialisation de SEA avec ELS ou OMP permet à SEA_{ELS} et SEA_{OMP} de trouver une meilleure approximation de S^* que celle de SEA_0. De plus, seul SEA est capable de raffiner la solution fournie par OMP et ELS pour obtenir une solution plus précise. SEA_{ELS} et SEA_{OMP} reconstruisent avec succès le signal original, à l'exception de deux pics entre les positions 410 et 425. En revanche, les autres algorithmes échouent en raison de la cohérence mutuelle élevée de A et de la présence du bruit additif.

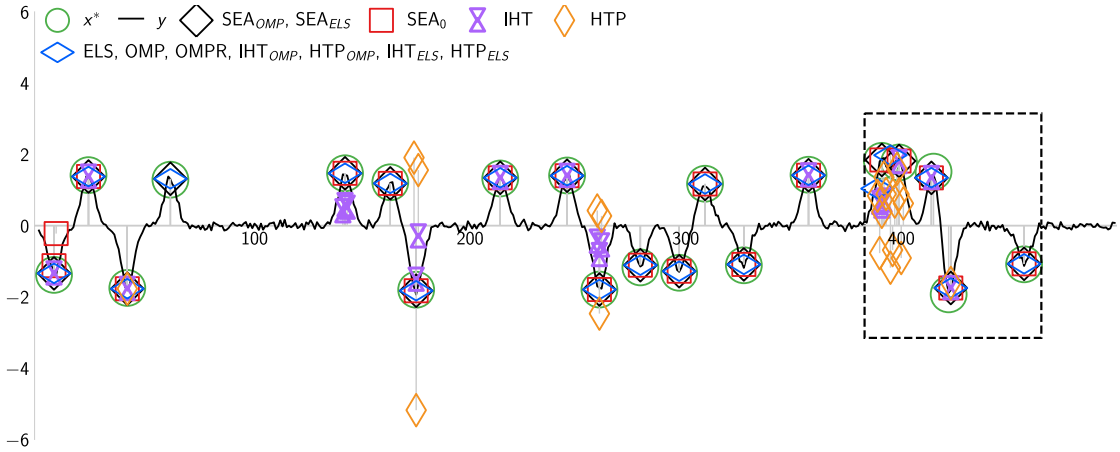

FIGURE 5.4 – Représentation de l'instance de x^* et y de la figure 5.3 avec les solutions fournies par les algorithmes. La solution d'OMP (identique à celle de HTP_{OMP}, IHT_{OMP} et OMPR) diffère de celle d'ELS (identique à HTP_{ELS} et IHT_{ELS}) par 3 pics. Afin de ne pas surcharger la figure, ces différences sont regroupées sous un même symbole. Cela ne compromet pas l'interprétation des résultats.

Les homologues des courbes des figures 5.4 et 5.5 dans le cas sans bruit sont présentés dans les figures 5.6 et 5.7. Ainsi, nous présentons les résultats des algorithmes sur le signal de la figure 5.2. Ici, les algorithmes se comportent de la même manière que dans le cas bruité. Cependant, en l'absence de perturbation du signal, toutes les versions de SEA parviennent à retrouver les positions exactes des pics, alors qu'aucun autre algorithme n'atteint une telle performance.

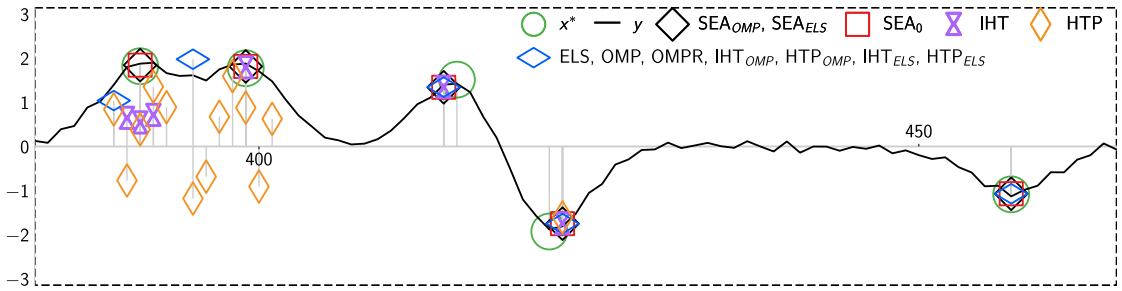

FIGURE 5.5 – Agrandissement de la région en pointillés de la figure 5.4 avec les résultats de tous les algorithmes analysés. Cette région correspond à la zone la plus dense du signal.

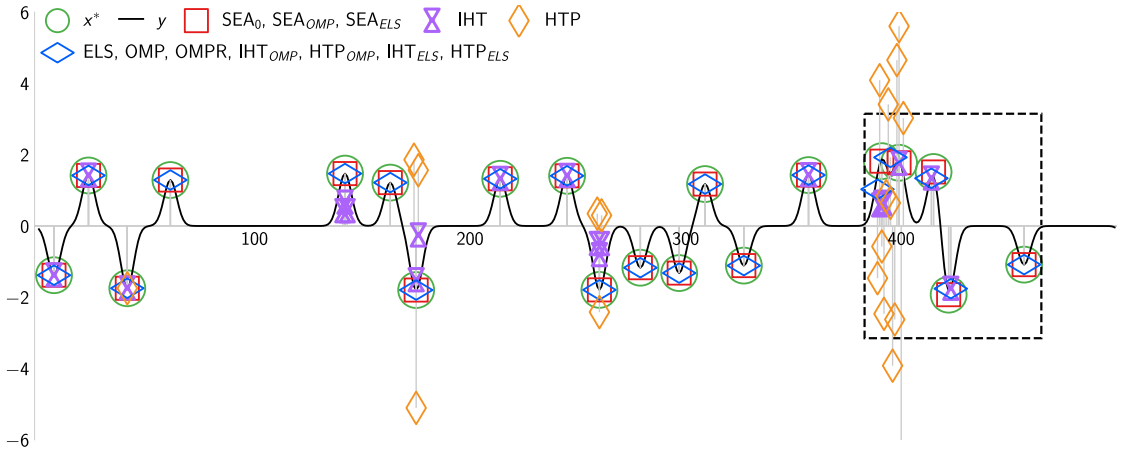

FIGURE 5.6 – Représentation d'une instance de x^* et y avec les solutions fournies par les algorithmes lorsque $k = 20$ et $n = 500$ dans le cas sans bruit.

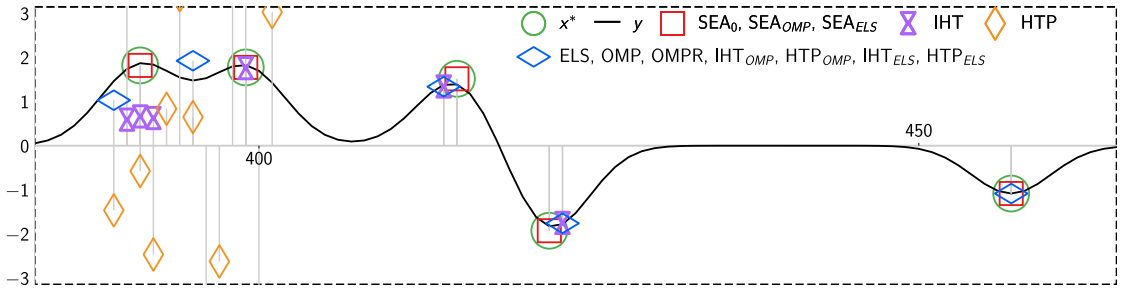

FIGURE 5.7 – Découpe de la région en pointillés de la figure 5.6. Cette région correspond à la zone la plus dense du signal.

5.2.2.2 Suivi de F pendant le processus itératif

Nous approfondissons l'analyse entamée dans la figure 3.3 de la section 3.3.2. Les figures 5.8 et 5.9 illustrent le comportement de HTP, IHT, ELS, SEA_0, SEA_{OMP} et SEA_{ELS}, pour retrouver le signal dans le cas bruité (figure 5.3) tout au long du processus itératif.

Plus précisément, dans la figure 5.8, les courbes pleines représentent $\ell_{2,\text{rel_loss}}(x^t)$

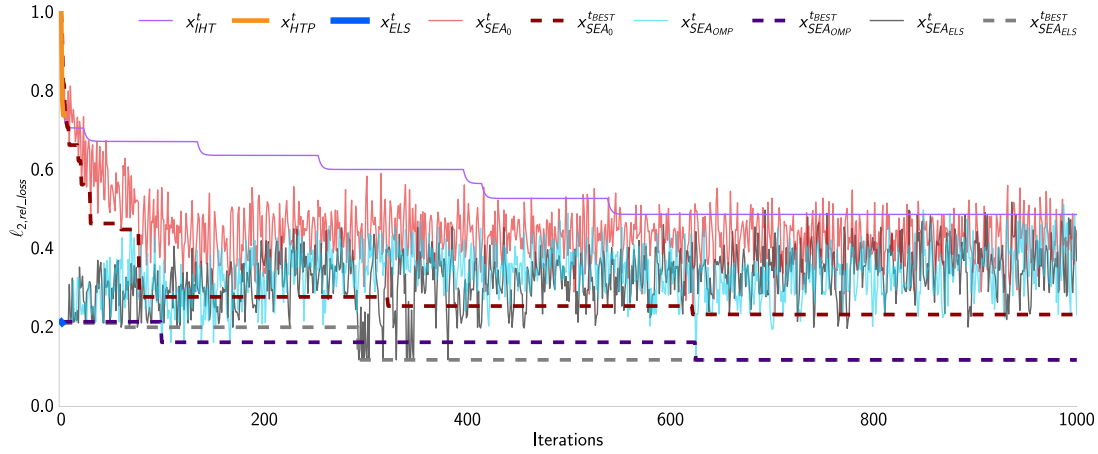

FIGURE 5.8 – Représentation de $\ell_{2,\text{rel_loss}}(x^t)$ (lignes pleines) et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t)})$ (lignes pointillées) pour chaque itération de plusieurs algorithmes, pour l'expérience de la figure 5.3.

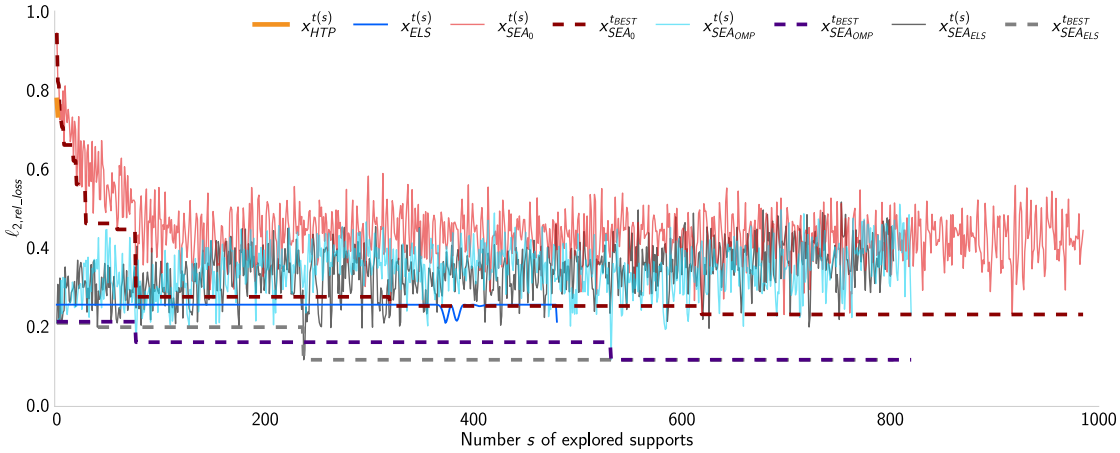

FIGURE 5.9 – Représentation de $\ell_{2,\text{rel_loss}}(x^{t(s)})$ (lignes pleines) et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t(s))})$ (lignes pointillées) pour chaque nouveau support exploré de plusieurs algorithmes, pour l'expérience de la figure 5.3.

lorsque t varie dans $[\![0, 1000]\!]$, où $\ell_{2,\text{rel_loss}}$ est définie par

$$\ell_{2,\text{rel_loss}}(x) = \frac{\|Ax - y\|_2}{\|y\|_2} = \frac{\sqrt{2F(x)}}{\|y\|_2}. \tag{5.1}$$

Les lignes en pointillés représentent $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t)})$ où $t_{BEST}(t) = \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$ et t varie dans $[\![0, 1000]\!]$.

Dans l'ensemble, aucun algorithme ne parvient à atteindre une erreur égale à 0. ELS n'effectue qu'une seule itération avant de s'arrêter dans un minimum local ($\ell_{2,\text{rel_loss}}(x^1) \approx 0.2$). HTP effectue quelques itérations avant de s'arrêter. IHT surpasse HTP en explorant un peu plus. On peut observer que, en raison de la nature explo-

ratoire de SEA, $\ell_{2,\text{rel_loss}}(x^t)$ varie fortement pour toutes les versions de SEA. Cette exploration permet à SEA_{ELS} d'affiner l'estimation d'ELS en 300 itérations. Enfin, malgré une décroissance plus rapide autour de la 100ème itération, SEA_{OMP} rejoint finalement SEA_{ELS} après 620 itérations.

Nous observons aussi que HTP et IHT ont des performances médiocres en raison de la cohérence mutuelle élevée de A. En effet, comme nous l'avons vu dans les figures de la section 5.2.2.1, ces algorithmes commettent l'erreur de sélectionner plusieurs pics voisins pour représenter le même pic large. Ils ne parviennent pas à corriger cette erreur au cours du processus itératif.

La figure 5.9 illustre le même processus itératif que figure 5.8, en se concentrant sur le nombre de supports explorés plutôt que sur le nombre d'itérations de chaque algorithme. Ici, les courbes pleines représentent $\ell_{2,\text{rel_loss}}(x^{t(s)})$ lorsque s varie de 0 au nombre total de supports explorés, où $t(s)$ est l'itération associée au $s^{ème}$ support exploré (sans redondance).

Comme dans la figure précédente, les courbes en pointillés représentent $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t(s))})$ où $t_{BEST}(t(s)) = \underset{t' \in [\![0, t(s)]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$. Il s'agit de l'erreur associée à la meilleure estimation trouvée lors de l'exploration du $s^{ème}$ support.

Sur cette deuxième figure, nous observons que HTP explore très peu avant de s'arrêter dans un minimum local. Malgré une seule itération, ELS explore 500 supports dans le voisinage de son initialisation par OMP. Cela ne lui permet d'améliorer que très légèrement la solution fournie par OMP. Ici, SEA_0 explore un nouveau support à chaque itération, tandis que SEA_{ELS} en explore moins, améliorant ELS en explorant moins de 250 supports. Encore une fois, malgré une décroissance plus rapide au début, SEA_{OMP} rejoins finalement SEA_{ELS} après avoir exploré environ 520 supports. Ceci révèle l'efficacité de chaque algorithme à trouver des supports pertinents.

Nous allons maintenant compléter l'analyse que nous avions entamée dans le cas non bruité au chapitre 3 avec la figure 3.3 de la section 3.3.2.

Comme dans le cas bruité, on peut observer dans la figure 5.10 (comme déjà observé dans la figure 3.3) qu'en raison de la nature exploratoire de SEA, $\ell_{2,\text{rel_loss}}(x^t)$ oscille pour toutes les versions de SEA. Pour chacune de ces versions, une fois que S^* est récupéré pour t suffisamment grand, $x^t = x^*$, et donc $Ax^t - y = 0$ (car $e = 0$). En effet, comme nous l'avions mentionné dans la section 3.3.2, la règle de mise à jour de $\mathscr{X}^t$ à la ligne 8 de l'algorithme 1, fait que $\mathscr{X}^t$ n'évolue plus, et qu'aucun nouveau support n'est exploré. Ce comportement, déjà très marqué dans la première analyse, l'est toujours dans la figure 5.10. En outre, comme on peut le voir dans figure 5.6, ELS n'améliore pas OMP, ce qui conduit à une initialisation identique pour SEA_{OMP} et SEA_{ELS}. Par conséquent, ces deux derniers algorithmes suivent la même trajectoire. D'après la figure 5.10, nous observons cependant que SEA_{ELS} et SEA_{OMP} explorent deux fois plus de supports que SEA_0.

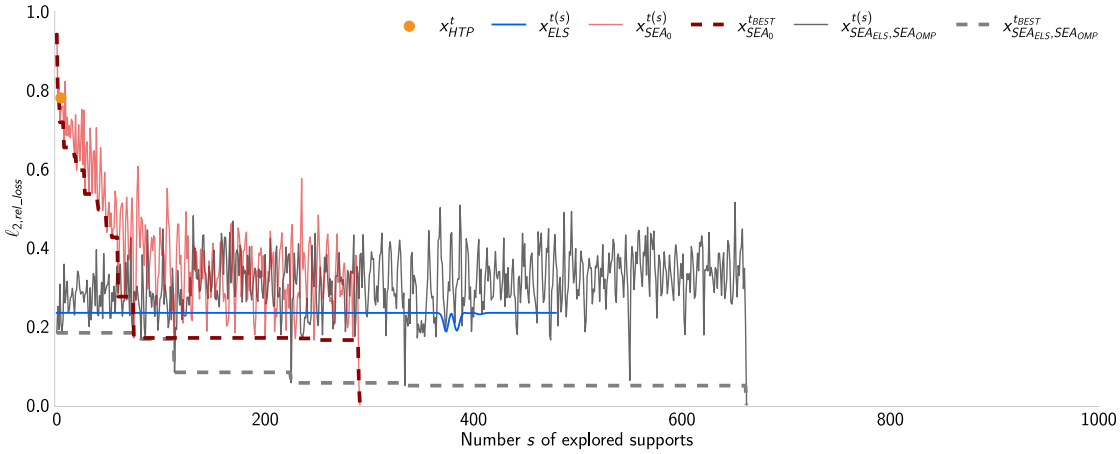

FIGURE 5.10 – Représentation de $\ell_{2,\text{rel_loss}}(x^{t(s)})$ (lignes pleines) et $\ell_{2,\text{rel_loss}}(x^{t_{BEST}(t(s))})$ (lignes pointillées) pour chaque nouveau support exploré de plusieurs algorithmes, pour l'expérience de la figure 5.6.

5.2.3 Généralisation des résultats

Nous nous intéressons maintenant à ce qu'il se passe lorsque l'on répète cette expérience de déconvolution sur différentes instances. Nous étudions les performances des algorithmes sur $r = 200$ réalisations de signaux x^* et y en faisant varier k de 1 à 50. Comme précédemment, nous étudions conjointement les cas bruité et non bruité, car les résultats sont très similaires. Pour cela, nous nous intéressons encore une fois à la capacité des algorithmes à retrouver le bon support (section 5.2.3.1) ainsi que l'efficacité de leur exploration (section 5.2.3.2). Nous allons également étudier l'impact de la variation du pas de HTP et IHT dans la section 5.2.3.3.

5.2.3.1 Analyse des performances

Nous commençons par regarder, à l'aide de la figure 5.11, une mesure de distance entre des supports, provenant de (ELAD 2010), moyennée sur les $r = 200$ problèmes résolus. Cette mesure est définie par

$$\text{dist}_{\text{supp}}(x) = \frac{k - |S^* \cap \text{supp}(x)|}{k}. \tag{5.2}$$

Cette distance, comprise entre 0 et 1, mesure la proportion d'erreur dans le support estimé par un algorithme. Plus la distance est faible, plus l'algorithme est performant. Pour toutes les valeurs de parcimonie considérées, SEA_0, SEA_{OMP} et SEA_{ELS} surpassent les autres algorithmes. SEA améliore les résultats de OMP et ELS alors qu'ils ne sont jamais améliorés par HTP ou IHT (les courbes se superposent). On peut noter que pour les petites valeurs de k, IHT a de mauvaises performances. En effet, comme on l'a déjà vu dans la sous-section précédente, il réalise plusieurs erreurs de détection autour du plus grand pic de y. Ces erreurs ne sont pas corrigées par la suite. Lorsque k augmente,

les algorithmes ont de plus en plus de difficultés à retrouver S^*. En utilisant une stratégie de démarrage à froid, SEA_0 est ici l'algorithme le plus performant. Comme on le confirmera plus tard, ces analyses suggèrent que l'exploration de SEA est plus efficace que les échanges d'éléments de support effectués par ELS. Ces expériences suggèrent également qu'une stratégie de démarrage à chaud, telle que SEA_{ELS} ou SEA_{OMP}, peut conduire l'algorithme à se retrouver piégé dans un moins bon minimum local. Le choix de la meilleure stratégie semble dépendre de la qualité de l'initialisation et du problème. En effet, lors de l'expérience de diagramme de phase de la section 3.5 du chapitre 3, nous avions vu que SEA_{ELS} était le plus performant et SEA_0 le moins performant. Ainsi, le choix de la version de SEA à utiliser ne peut se faire que sur la base de performances empiriques.

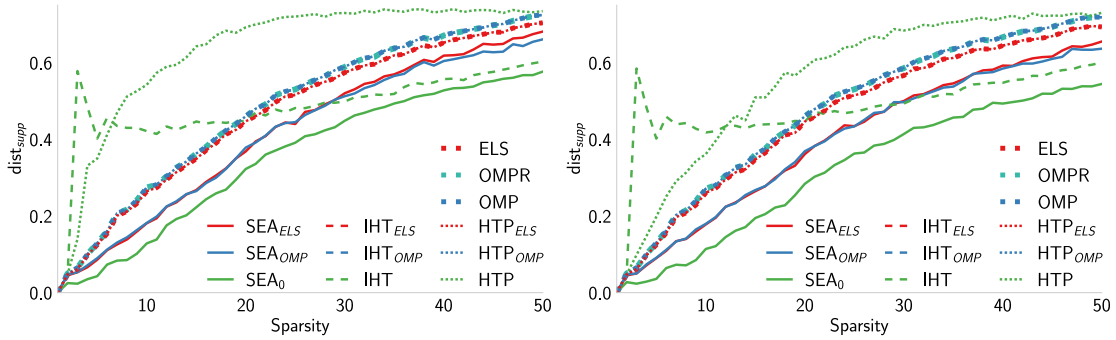

FIGURE 5.11 – Distance moyenne entre S^* et les supports renvoyés par plusieurs algorithmes sur les 200 problèmes pour chaque niveau de parcimonie $k \in [\![1, 50]\!]$ dans le cas bruité (à gauche) et non bruité (à droite).

Sur la figure 5.12, nous présentons la moyenne, sur les $r = 200$ problèmes, de l'erreur relative $\ell_{2,\text{rel_loss}}$, définie dans (5.1), pour les sorties de tous les algorithmes et pour chaque niveau de parcimonie $k \in [\![1, 50]\!]$. Nous observons que toutes les versions de SEA atteignent les erreurs les plus faibles dans les cas les plus parcimonieux ($k < 20$). L'écart le plus important entre SEA et ses concurrents est observé lorsque k est compris entre 9 et 13. Pour plus de clarté, les courbes pour HTP et IHT ne sont visibles que pour les petites valeurs de k. Encore une fois, la cohérence mutuelle élevée de A et la méthode de sélection simultanée de plusieurs éléments lors de l'estimation du support, conduisent IHT et HTP à des erreurs bien plus importantes que les autres algorithmes. Bien que tous les algorithmes soient impactés par l'ajout de bruit, SEA_0 l'est plus que les autres. Ainsi, on voit que même si l'erreur de reconstruction de l'observation de SEA_0 fait partie des plus élevées dans le cas bruité, SEA_0 reste l'algorithme le plus précis en termes de reconstruction de supports.

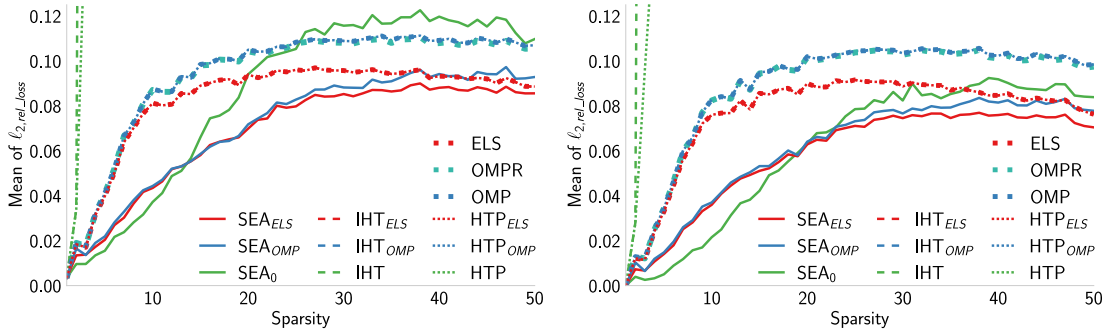

FIGURE 5.12 – Moyenne de $\ell_{2,\text{rel_loss}}(x)$ pour les sorties des algorithmes sur les 200 problèmes pour chaque niveau de parcimonie $k \in [\![1,50]\!]$ dans le cas bruité (à gauche) et non bruité (à droite).

Dans la figure 5.13, nous montrons la moyenne de la distance de Wasserstein-1 (également appelée distance du cantonnier, ou distance du terrassier) pour les mêmes problèmes. Elle illustre à quel point les pics choisis sont « éloignés » des vrais pics. Là encore, toutes les versions de SEA atteignent les distances les plus faibles pour $k < 18$. Lorsque k augmente, SEA_0 et, dans une moindre mesure, SEA_{OMP} et SEA_{ELS}, bien qu'ils soient les meilleurs pour trouver la position exacte des pics (voir figure 5.11), choisissent des pics « loins » des vrais lorsqu'ils se trompent, tandis que IHT s'améliore pour les k les plus élevés.

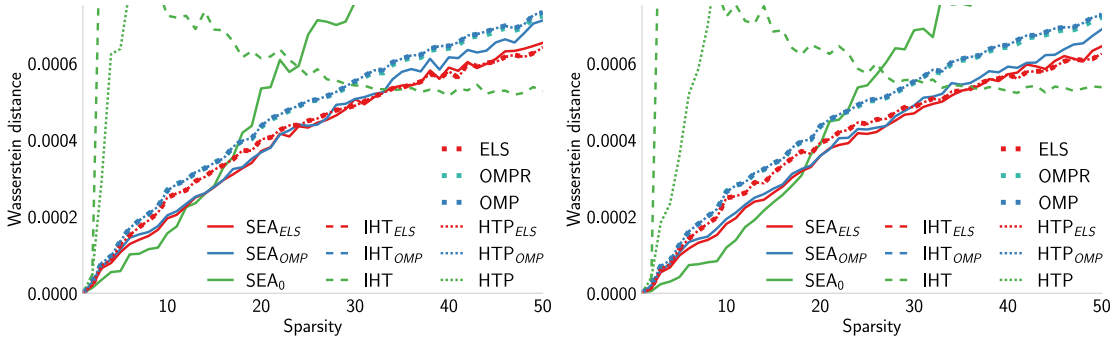

FIGURE 5.13 – Moyenne de la distance de Wasserstein-1 pour les sorties des algorithmes sur les 200 problèmes pour chaque niveau de parcimonie $k \in [\![1,50]\!]$ dans le cas bruité (à gauche) et non bruité (à droite).

5.2.3.2 Analyse de l'exploration

Comme nous l'avons expliqué dans la section 3.4.1 du chapitre 3, le coût global des algorithmes dépend du nombre de supports explorés. Dans la figure 5.14, nous illustrons le nombre de supports explorés de deux manières différentes, comme nous l'avions déjà fait dans la figure 3.4. Premièrement, sur la figure 5.14 à gauche, nous présentons le nombre moyen de supports explorés durant l'ensemble de la résolution du problème, ce qui représente le coût global. Cela inclut les supports explorés avant

l'initialisation. Par exemple, les valeurs atteintes par la courbe SEA_{OMP} incluent à la fois les supports explorés par OMP pour son initialisation et ceux explorés ultérieurement durant le processus itératif de SEA. Ensuite, sur la figure 5.14 à droite, nous présentons le nombre moyen de supports explorés qui ont effectivement nécessité un calcul après l'initialisation. Ces courbes révèlent le coût des algorithmes après l'initialisation, où les supports vus avant l'initialisation (par exemple, ceux de OMP pour SEA_{OMP}) ne sont pas inclus, car ils n'entraînent pas de temps de calcul supplémentaire.

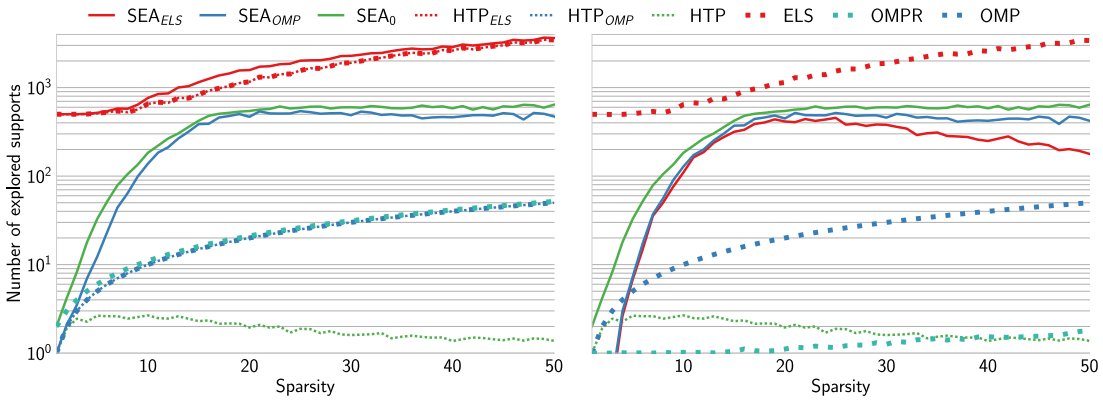

FIGURE 5.14 – Gauche : Moyenne du nombre de supports explorés (en considérant ceux vus pendant l'initialisation le cas échéant) par les algorithmes résolvant 200 problèmes de déconvolution bruités en dimension $n = m = 500$, à des niveaux de parcimonie $k \in [\![1, 50]\!]$.
Droite : Moyenne du nombre de supports explorés après l'initialisation des algorithmes dans la même configuration.

En examinant le coût global des algorithmes sur la gauche, nous observons trois types de profils d'exploration. Certains algorithmes, comme OMP, OMPR et HTP, ont une exploration minimale. De plus, lorsque k augmente, HTP explore moins de supports. D'autre part, des algorithmes comme ELS explorent beaucoup. Enfin, SEA se situe entre les deux, explorant quelques supports pour les petits k et plus au fur et à mesure que k augmente jusqu'à atteindre un seuil. Bien qu'il explore au moins deux fois moins de supports qu'ELS, la figure 5.11 montre que l'exploration plus efficace de SEA lui permet d'obtenir de meilleurs résultats.

La partie droite de la figure montre que l'ajout de SEA à ELS (SEA_{ELS}) ne modifie pas de manière significative l'ordre de grandeur du coût en calculs. De plus, HTP_{ELS} et HTP_{OMP} ne sont pas présents, car ils n'explorent pas après leur initialisation. On pouvait déjà le voir sur la courbe de HTP_{ELS} de la figure 5.9. Les courbes d'OMP et de SEA_0 restent inchangées, car elles n'ont pas d'algorithme d'initialisation.

Toutes les versions de SEA ont le même ordre de grandeur de supports explorés. Cependant, nous concluons que plus l'initialisation est forte (avec 0 < OMP < ELS), plus l'exploration devient difficile en raison de la cohérence mutuelle élevée de A et des minima locaux dans lesquels OMP et ELS aboutissent.

5.2.3.3 Impact du pas sur IHT et HTP

Jusqu'à présent, nous avons fixé arbitrairement le pas à $\eta = \frac{1.8}{L}$ où L est le carré du rayon spectral de A. Dans cette section, nous étudions l'influence du pas sur HTP et IHT. Nous allons également considérer l'algorithme NIHT (Normalized Iterative Thresholding) (BLUMENSATH et Mike E DAVIES 2010) qui est basé sur IHT mais utilise un pas adaptatif η^t qui dépend de l'itération t. Le pas est choisi comme suit :

$$\eta^t = \frac{\|(A^T(y - Ax^t))_{|S^t}\|_2^2}{\|A((A^T(y - Ax^t))_{|S^t})\|_2^2}.$$

De plus, comme expliqué dans (FOUCART 2011), lorsque $\eta^t > 0.99\|x^t - x^{t+1}\|_2^2 / \|A(x^t - x^{t+1})\|_2^2$, η^t est divisé par deux jusqu'à ce que cette inégalité ne soit plus satisfaite.

Ceci étant posé, nous considérons la même expérience que dans la section 5.2.3.1 pour NIHT, IHT, et HTP avec différents pas. Chaque algorithme est testé sur $r = 200$ problèmes bruités, pour différents x^* k-parcimonieux, avec $k[\![1, 50]\!]$. IHT et HTP ont été testés avec des pas $\eta \in \{\frac{2^l}{L} \mid \text{pour } l \in [\![-3, +3]\!]\}$.

L'analogue de la figure 5.11 est représenté dans la figure 5.15. Pour tous les pas considérés, IHT et HTP ne peuvent pas améliorer la solution fournie par ELS et OMP au regard de la distance $\text{dist}_{\text{supp}}$. Nous arrivons à la même conclusion pour NIHT. Le classement de HTP ne dépend pas du pas sélectionné. La plus faible distance $\text{dist}_{\text{supp}}$ que HTP atteint est avec un pas $\eta = \frac{8}{L}$. L'augmentation de la taille du pas n'améliore pas les résultats et fait diverger HTP. IHT, quant à lui, est plus sensible aux variations de la taille du pas que HTP. La plus faible distance $\text{dist}_{\text{supp}}$ qu'IHT atteint est avec un pas $\eta = \frac{2}{L}$. L'augmentation de la taille du pas fait également diverger IHT. Le pas dynamique de NIHT le rend meilleur que SEA_{OMP} et SEA_{ELS} pour $k > 25$. Cependant, IHT avec $\eta = \frac{2}{L}$ devient à son tour meilleur que NIHT pour $k > 36$. Ainsi, même en tenant compte de ces variabilités, SEA_0 reste le meilleur algorithme dans ce contexte.

5.2.4 Étude d'autres configurations

Nous étudions maintenant l'impact de la modification de certains paramètres du problème sur les performances des algorithmes. À chaque fois, nous nous intéressons uniquement au cas bruité. Dans la section 5.2.4.1, nous étudions l'impact d'une mauvaise estimation de la parcimonie, c'est-à-dire, lorsque k et inconnue et approchée. Puis, dans la section 5.2.4.2, nous étudions la robustesse des algorithmes au bruit. Ensuite, dans la section 5.2.4.3, nous étudions l'impact de l'ajout de bruit sur le signal x^* pour simuler une parcimonie non exacte. Enfin, dans la section 5.2.4.4, nous étudions l'impact de la modification de l'amplitude relative des éléments non nuls de x^*.

5.2.4.1 Impact d'une mauvaise estimation de la parcimonie k

Nous considérons la même expérience que dans la section 5.2.3. Cependant, nous nous plaçons dans le cas où la parcimonie k est mal estimée. Ainsi, nous fixons

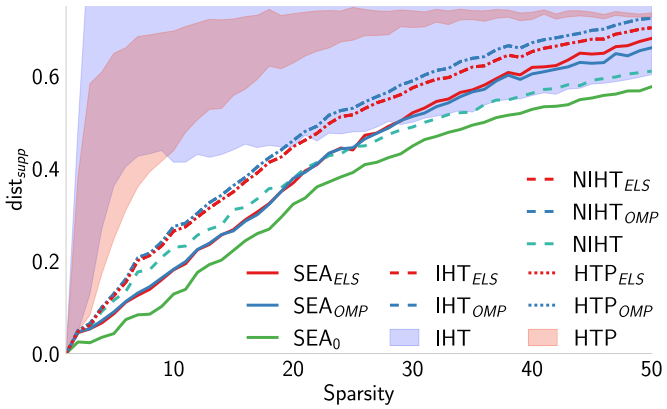

FIGURE 5.15 – Distance moyenne $\text{dist}_{\text{supp}}$ (définie dans (5.2)) entre S^* et le support des solutions fournies par plusieurs algorithmes en fonction du niveau de parcimonie k avec différents pas pour IHT et HTP. La zone entre la courbe la plus haute et la courbe la plus basse pour différentes tailles de pas est affichée en bleu et en rouge pour IHT et HTP. Les courbes NIHT_{OMP} et IHT_{OMP} sont superposées. Les courbes NIHT_{ELS} et IHT_{ELS} le sont également.

toujours $n = 500$ et utilisons une matrice de convolution A correspondant à un filtre gaussien d'écart-type 3. Chaque algorithme est testé sur $r = 200$ problèmes bruités, pour différents x^* k-parcimonieux, avec $k \in \{4p,\ p \in [\![1, 12]\!]\}$. La différence ici est que les algorithmes doivent retrouver un signal de parcimonie $k' \neq k$.

Les résultats analogues à ceux de la figure 5.11 lorsque $k' = 0.75k$ et $k' = 1.25k$ sont représentés dans la figure 5.16. Nous n'avons pas testé toutes les valeurs de parcimonie $k \in [\![1, 50]\!]$, car nous voulions maintenir un rapport $\frac{k'}{k}$ constant, égal à 0.75 ou 1.25. Pour garantir l'équité dans le cas d'une sous-estimation, nous introduisons une distance légèrement différente de $\text{dist}_{\text{supp}}$, qui permet aux algorithmes d'atteindre une distance égale à 0 :

$$\text{dist}_{\text{supp},k'}(x) = \frac{k' - |S^* \cap \text{supp}(x)|}{k'}.$$

Cette métrique montre la capacité de chaque algorithme à reconstruire uniquement les éléments de S^*.

Dans la partie gauche de la figure 5.16, on voit que SEA_{OMP} et SEA_{ELS} n'améliorent pas beaucoup OMP et ELS et que SEA_0 conserve toujours la distance la plus basse comme dans la figure 5.11. Dans la partie droite de la figure 5.16, nous constatons également que tous les algorithmes atteignent une distance plus petite que dans la figure 5.11. Ainsi, dans ces configurations, SEA_0 est toujours l'algorithme qui atteint la plus petite distance. En raison de la définition de $\text{dist}_{\text{supp},k'}$, les bonnes performances du cas $\frac{k'}{k} > 1$ (partie droite de la figure) étaient attendues. En outre, lorsque $\frac{k'}{k} < 1$, les algorithmes peuvent se concentrer sur les plus grandes entrées de x^* qui sont plus faciles à reconstruire.

La figure 5.17 complète les résultats présentés dans la figure 5.16 avec une autre mesure de distance. Nous illustrons ici la manière dont les performances se dégradent lorsque l'on évalue la distance moyenne entre les supports des plus grandes entrées de x^* et ceux des solutions fournies par les algorithmes, et inversement. Nous introduisons donc $K = \min\{k, k'\}$, et pour tout $x \in \mathbb{R}^n$ et $i \in [\![1, n]\!]$

$$\text{dist}_{\text{supp,largest}}(x) = \frac{K - |\text{supp}(x^*_{\text{largest}_K}) \cap \text{supp}(x_{\text{largest}_K})|}{K}$$

$$\text{avec } x_{\text{largest}_K|i} = \begin{cases} x_{|i} & \text{si } i \in \text{largest}_K(x) \\ 0 & \text{si } i \notin \text{largest}_K(x). \end{cases}$$

Lorsque k est sous-estimée (partie gauche de la figure 5.17), $\text{dist}_{\text{supp,largest}}$ indique la capacité de chaque algorithme à reconstruire $\text{supp}(x^*_{\text{largest}_K})$, le support des K plus grandes entrées de x^*. Alors que lorsque k est surestimée (partie droite de la figure 5.17), $\text{dist}_{\text{supp,largest}}$ représente la capacité de chaque algorithme à reconstruire S^* pour les plus grandes entrées de la solution fournie. Avec cette nouvelle métrique, nous observons dans la partie gauche de la figure 5.17 que les performances de tous les algorithmes se dégradent. Nous observons également le même phénomène dans la partie droite de la figure 5.17, SEA_0 étant moins affecté que les autres algorithmes et OMPR affichant la meilleure performance pour $k < 11$.

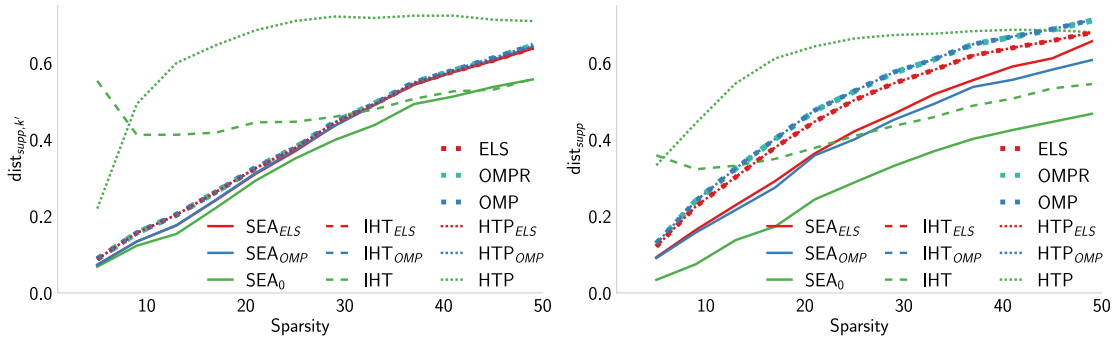

FIGURE 5.16 – Distance moyenne $\text{dist}_{\text{supp},k'}$ entre S^* et les supports renvoyés par plusieurs algorithmes pour chaque niveau de parcimonie $k \in \{4p,\ p \in [\![1, 12]\!]\}$ lorsque la parcimonie est sous-estimée (gauche, $k' = 0.75k$) et surestimée (droite, $k' = 1.25k$).

5.2.4.2 Robustesse au bruit

Nous considérons la même expérience que dans la section 5.2.3. Cependant, nous nous plaçons dans le cas où le bruit e est uniformément tiré sur la sphère de rayon $\alpha\|Ax^*\|_2$ avec $\alpha \in \{0.2, 0.3\}$ au lieu de $\alpha = 0.1$. De nouveau, nous fixons $n = 500$ et utilisons une matrice de convolution A correspondant à un filtre gaussien d'écart-type 3. Chaque algorithme est testé sur $r = 200$ problèmes bruités, pour différents x^*

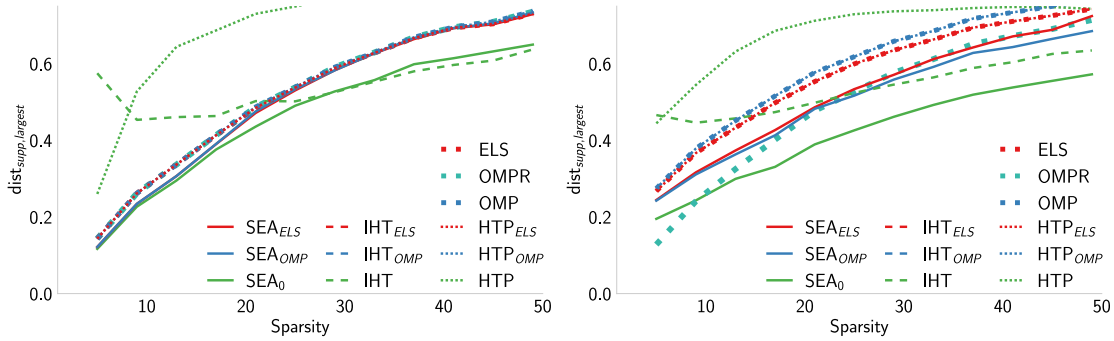

FIGURE 5.17 – Distance moyenne $\text{dist}_{\text{supp,largest}}$ entre S^* et les supports renvoyés par plusieurs algorithmes pour chaque niveau de parcimonie $k \in \{4p,\ p \in [\![1,12]\!]\}$ lorsque la parcimonie est sous-estimée (gauche, $k' = 0.75k$) et surestimée (droite, $k' = 1.25k$).

k-parcimonieux, avec $k \in [\![1,50]\!]$. La différence ici est que nous changeons l'amplitude du bruit.

Les résultats analogues à ceux de la figure 5.11 lorsque $\alpha = 0.2$ et $\alpha = 0.3$ sont représentés dans la figure 5.18. Au fur et à mesure que le bruit augmente, les performances de tous les algorithmes se dégradent, sans modifier leur classement ni les conclusions de l'expérience.

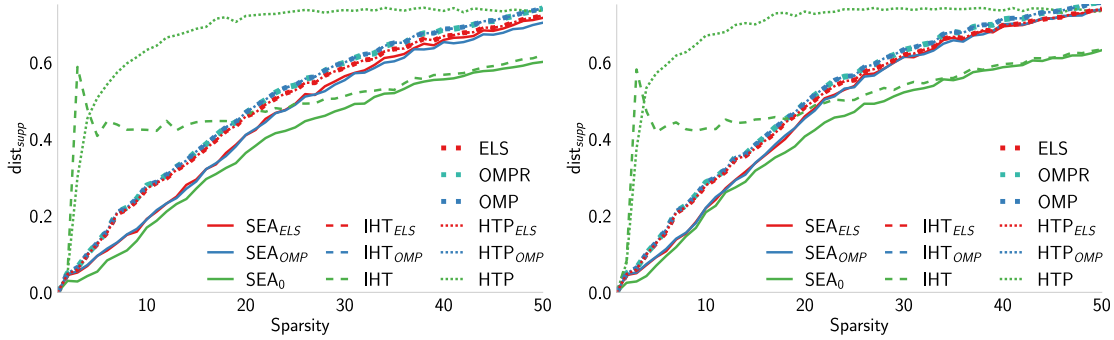

FIGURE 5.18 – Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes pour chaque niveau de parcimonie $k \in [\![1,50]\!]$ lorsque le bruit e est uniformément tiré sur la sphère de rayon $0.2\|Ax^*\|_2$ (gauche) et $0.3\|Ax^*\|_2$ (droite).

5.2.4.3 Impact d'une parcimonie approximative

Nous considérons la même expérience que dans la section 5.2.3. Cependant, nous nous intéressons au cas où le bruit e impacte différemment l'observation. Au lieu de générer $y = Ax^* + e$ avec e uniformément tiré sur la sphère de rayon $0.1\|Ax^*\|_2$, nous générons $y = A(x^* + e)$, avec e uniformément tiré sur la sphère de rayon $0.1\|x^*\|_2$. Cela revient à observer un signal qui n'est pas parcimonieux, mais qui peut être compressé

en signal k-parcimonieux. Une fois encore, nous fixons $n = 500$ et utilisons une matrice de convolution A correspondant à un filtre gaussien d'écart-type 3. Chaque algorithme est testé sur $r = 200$ problèmes bruités, pour différents x^* k-parcimonieux bruités, avec $k \in [\![1, 50]\!]$. La différence ici est que nous avons changé la manière d'ajouter le bruit dans les observations.

Les résultats analogues à ceux de la figure 5.11 sont représentés dans la figure 5.19. Les différences entre ces deux figures ne sont pas significatives. Ainsi, dans cette configuration, SEA_0 est l'algorithme qui atteint la plus basse distance $\text{dist}_{\text{supp}}$ et nous tirons les mêmes conclusions que dans la section 5.2.3.

5.2.4.4 Impact de la variation de l'amplitude de x^*

Le rapport $\frac{\|x^*\|_2}{\min_{i \in S^*} |x^*|_i}$ est important dans les conditions (RC_{RIP}) et (RC_{SRIP}). Nous voulons regarder expérimentalement si cela a un impact réel sur les performances, ou si c'est seulement un artefact de la preuve. Nous considérons la même expérience que dans la section 5.2.3. Cependant nous nous intéressons au cas où les entrées non nulles de x^* sont tirées uniformément dans $[\![-10, -1]\!] \cup [\![1, 10]\!]$ au lieu de $[\![-2, -1]\!] \cup [\![1, 2]\!]$. Chaque algorithme est testé sur $r = 200$ problèmes bruités, pour différents x^* k-parcimonieux, avec $k \in [\![1, 50]\!]$.

Les résultats analogues à ceux de la figure 5.11 sont représentés sur la figure 5.20. Ici, OMP, OMPR et ELS obtiennent des résultats légèrement meilleurs, tandis que toutes les versions de SEA obtiennent des résultats légèrement moins bons. IHT et HTP sont également moins performants. Cependant, le classement des algorithmes reste le même. Nous pouvons donc constater expérimentalement que l'augmentation du rapport $\frac{\|x^*\|_2}{\min_{i \in S^*} |x^*|_i}$ réduit les performances de SEA.

5.3 Apprentissage supervisé

5.3.1 Contexte

Dans un contexte d'apprentissage supervisé, les lignes de la matrice $A \in \mathbb{R}^{m \times n}$ (souvent désignée par X) sont les vecteurs de caractéristiques de dimension n associés à chacun des m exemples d'apprentissage, tandis que les étiquettes correspondantes se trouvent dans le vecteur $y \in \mathbb{R}^m$. Lors de la phase d'apprentissage, un vecteur x k-parcimonieux (souvent appelé β ou w) est optimisé à l'aide d'une fonction de coût appropriée pour traduire le fait que $y \approx Ax$. Dans ce contexte, la reconstruction de supports est appelée sélection de variables.

Sur la base de la configuration expérimentale de (Axiotis et al. 2020), nous comparons les valeurs prises par une fonction de coût à la fin d'un apprentissage. Cette comparaison est effectuée pour différents niveaux de parcimonie k, pour chacun des algorithmes, sur des tâches de régression linéaire et de régression logistique. Nous utilisons les jeux de données publiques préparés et fournis par (Axiotis et al. 2020), en suivant le même processus de prétraitement : nous ajoutons à A une colonne sup-

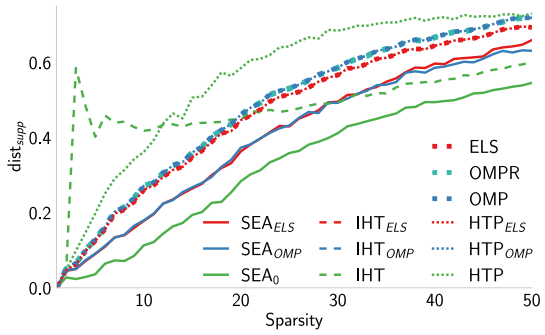

FIGURE 5.19 – Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes pour chaque niveau de parcimonie $k \in [\![1,50]\!]$ lorsque $y = A(x^* + e)$ avec un bruit e uniformément tiré depuis la sphère de rayon $0.1\|x^*\|_2$.

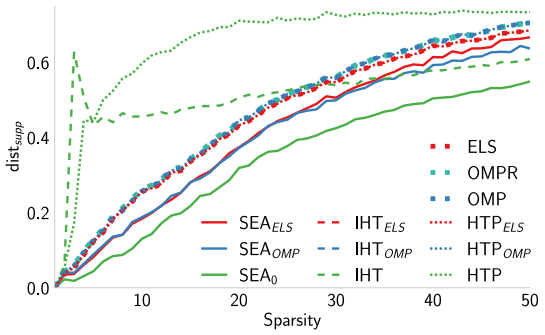

FIGURE 5.20 – Distance moyenne $\text{dist}_{\text{supp}}$ entre S^* et les supports renvoyés par plusieurs algorithmes pour chaque niveau de parcimonie $k \in [\![1,50]\!]$ lorsque les entrées non nulles de x^* sont tirées uniformément dans $[\![-10,-1]\!] \cup [\![1,10]\!]$.

plémentaire égale à 1 pour simuler l'ajout d'un biais et nous normalisons les colonnes de A.

Nous présentons les résultats pour les problèmes de régression dans la section 5.3.2 et pour les problèmes de classification dans la section 5.3.3.

5.3.2 Tâches de régression

Comme nous travaillons avec des jeux de données réelles sans vérité terrain, nous utilisons une fonction de coût adaptée aux problèmes de régression linéaire $\ell_2_\text{loss}(x) = \frac{1}{2}\|Ax - y\|_2^2$ pour tout $x \in \mathbb{R}^n$.

Comme le montre la figure 5.21, SEA_0, SEA_{OMP} et SEA_{ELS} sont au même niveau que ELS sur le jeu de données `comp-activ-harder`, qui est un jeu de données avec n petit. Dans le cadre d'un jeu de données de régression en plus grande dimension comme `year` (voir la figure 5.22), SEA_0 a des performances médiocres lorsque k augmente, mais SEA_{ELS} peut améliorer les performances de ELS et surpasser les autres algorithmes.

Dans un problème de petite dimension (n est petit) comme celui du jeu de données `cal_housing` ($m = 20639$ exemples, $n = 8$ caractéristiques) de la figure 5.23, nous voyons que SEA_{OMP} et SEA_{ELS} sont plus performants que ELS, et que SEA_0 les surpasse pour les parcimonies $k \in [\![5,8]\!]$. Il convient de mentionner que HTP obtient de bonnes performances pour ce jeu de données.

La même expérience est rapportée sur la figure 5.24, mais pour le jeu de données `slice` ($m = 53500$ exemples, $n = 384$ caractéristiques). Il s'agit d'un problème de

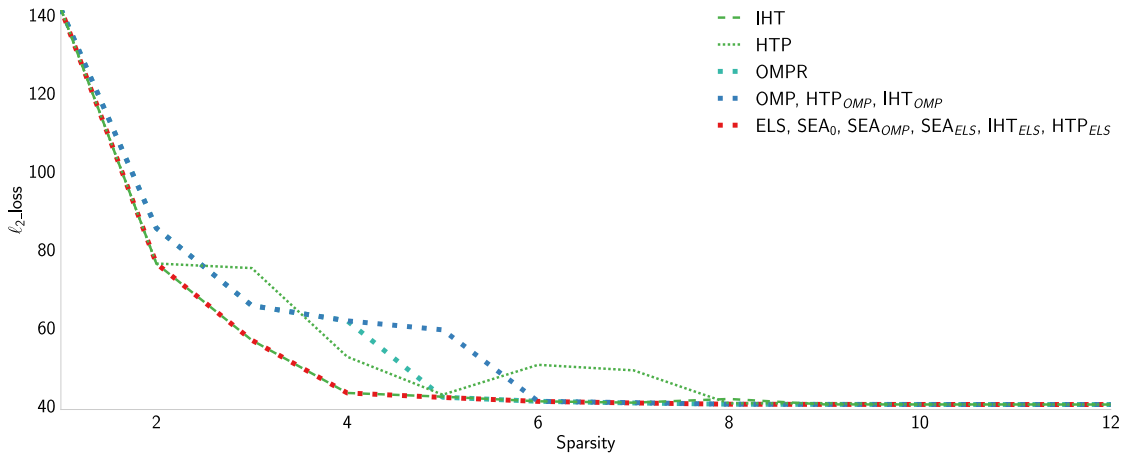

FIGURE 5.21 – Performances sur le jeu de données de régression `comp-activ-harder` ($m = 8191$ exemples, $n = 12$ caractéristiques).

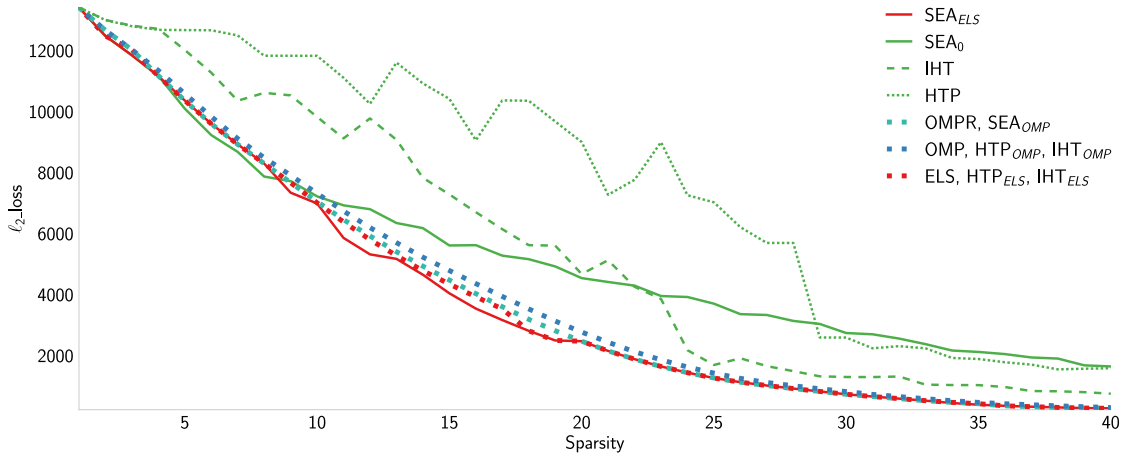

FIGURE 5.22 – Performances sur le jeu de données de régression `year` ($m = 463715$ exemples, $n = 90$ caractéristiques).

dimension intermédiaire. La figure 5.24 montre que SEA$_0$ obtient des résultats légèrement moins bons que SEA$_{ELS}$, SEA$_{OMP}$ et ELS. La courbe non décroissante de HTP provient de sa technique d'estimation du support. La cohérence mutuelle de l'ensemble de données `slice` étant de 1, HTP sélectionne des caractéristiques fortement corrélées et ne parvient pas à corriger cette erreur.

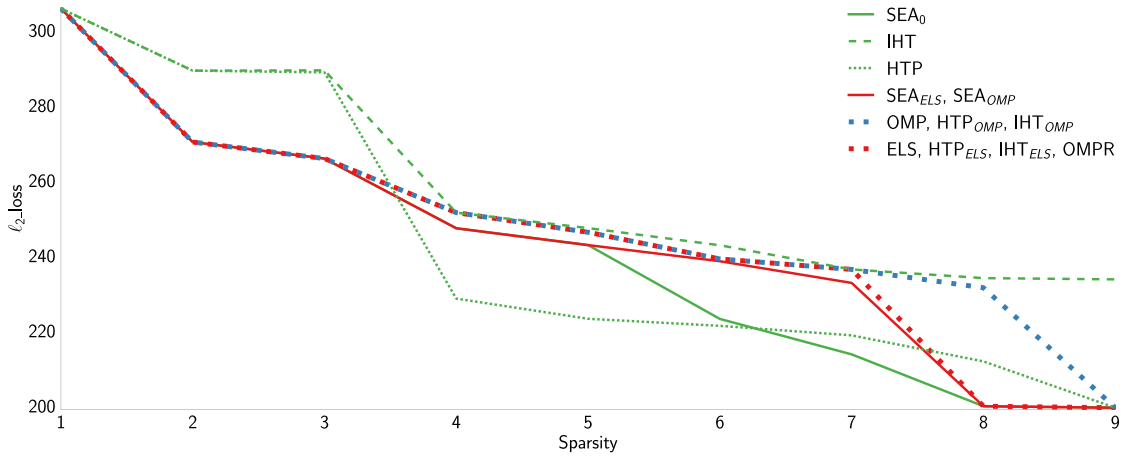

FIGURE 5.23 – Performances sur le jeu de données de régression `cal_housing` ($m = 20639$ exemples, $n = 8$ caractéristiques).

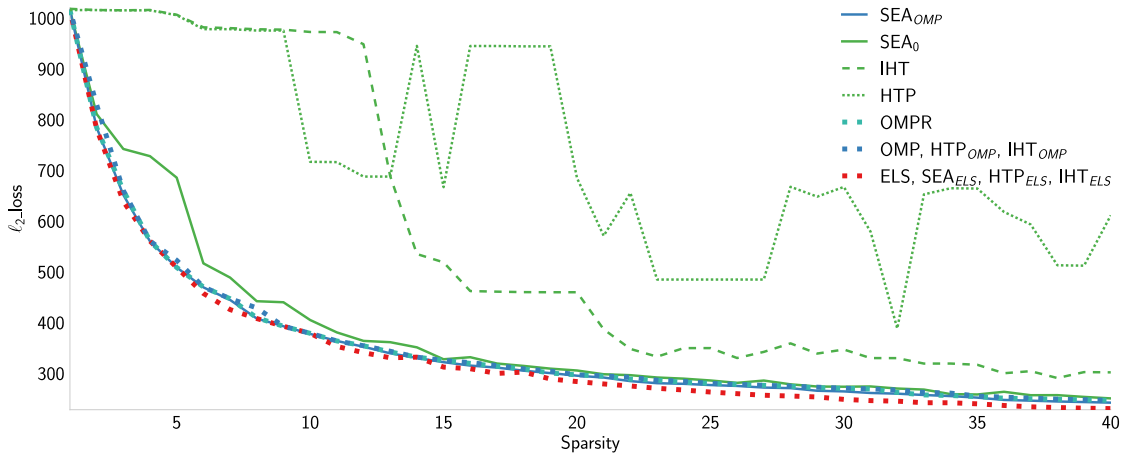

FIGURE 5.24 – Performances sur le jeu de données de régression `slice` ($m = 53500$ exemples, $n = 384$ caractéristiques).

5.3.3 Tâches de classification binaire par régression logistique

Dans ces expériences, nous considérons une fonction de coût adaptée au problème de régression logistique définie par

$$\text{log_loss}(x) = \sum_{i=1}^{m} \left(-y_i \log(\sigma((Ax)_{|i})) - (1-y_i)\log(1-\sigma((Ax)_{|i})\right),$$

où $\sigma(t) = \frac{1}{1+e^{-t}}$ est la fonction sigmoïde.

Nous devons adapter SEA à cette nouvelle fonction de coût. Dans l'algorithme 1, la ligne 7 est remplacée par $x^t = \underset{x\in\mathbb{R}^n,\ \text{supp}(x)\subseteq S^t}{\text{argmin}} \text{log_loss}(x)$ et la ligne 8 est remplacée par $\mathscr{X}^{t+1} = \mathscr{X}^t - \eta\nabla\,\text{log_loss}(x^t)$. Des adaptations similaires sont effectuées pour les

autres algorithmes.

Les valeurs prises par la fonction de coût log_loss(x), pour le jeu de données `letter` ($m = 20000$ exemples, $n = 16$ caractéristiques), pour tout $k \in [\![1, 12]\!]$ et pour chaque algorithme, sont représentées dans la figure 5.25. Les mêmes courbes obtenues sur le jeu de données `ijcnn1` ($m = 24995$ exemples, $n = 22$ caractéristiques) sont représentées dans la figure 5.26. Ces deux dernières figures montrent que SEA_0, SEA_{OMP} et SEA_{ELS} atteignent des performances similaires à ELS.

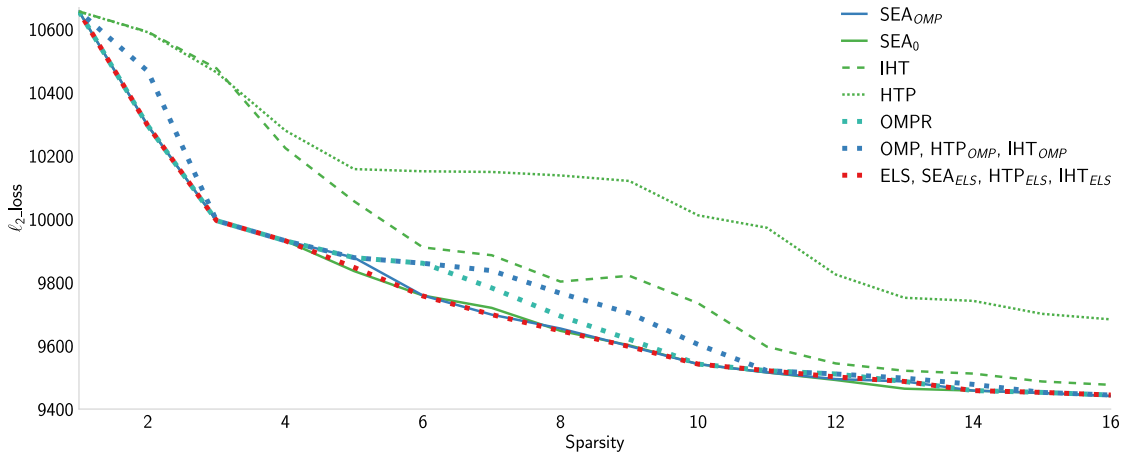

FIGURE 5.25 – Performances sur le jeu de données de classification `letter` ($m =$ 20000 exemples, $n = 16$ caractéristiques).

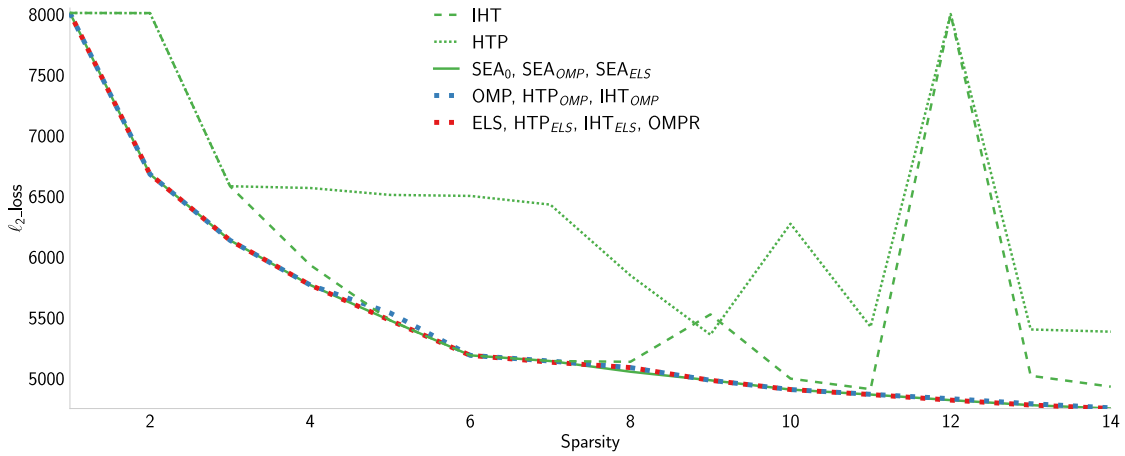

FIGURE 5.26 – Performances sur le jeu de données de classification `ijcnn1` ($m =$ 24995 exemples, $n = 22$ caractéristiques).

5.4 Conclusions

Dans ce chapitre, nous avons étudié les performances de SEA_0, SEA_{OMP} et SEA_{ELS} ainsi que celles d'algorithmes de l'état de l'art dans deux contextes différents : la

déconvolution et l'apprentissage supervisé. Dans le cas de la déconvolution, nous avons accès à une solution optimale du problème que l'on souhaite résoudre. Nous avons donc pu comparer les performances des algorithmes en fonction de la distance entre les supports des solutions fournies et ceux de la solution optimale dans de nombreuses configurations. Nous avons également pu étudier l'efficacité de l'exploration des différents algorithmes. En mettant ces résultats en perspective de ceux obtenus dans le cadre de l'expérience de diagramme de phase du chapitre 3, qui est aussi une expérience totalement contrôlée, nous avons vu que SEA était l'algorithme le plus performant. SEA est également le plus efficace en termes d'exploration. Il se distingue particulièrement de l'état de l'art dans les cas où la matrice A possède une cohérence mutuelle élevée. Ce qui correspond aux problèmes les plus difficiles.

Dans le cas de l'apprentissage supervisé, nous avons comparé les performances des algorithmes sur des jeux de données réels. Nous avons montré que SEA était compétitif avec les algorithmes de l'état de l'art. Il est particulièrement performant sur les jeux de données de petite dimension. Cependant, dans les cas de dimensions plus élevées, SEA se révèle moins performant et nécessite une bonne initialisation pour atteindre un résultat satisfaisant. Il se distingue très peu dans les tâches de classification que nous avons considérées.

Dans tous les cas, la question de l'initialisation est cruciale. On a vu que SEA était capable d'améliorer les performances des algorithmes de l'état de l'art. Cependant, initialiser SEA avec un algorithme de l'état de l'art ne garantit pas les meilleures performances. En effet, dans le cas du diagramme de phase, SEA_{ELS} est l'algorithme le plus performant. Dans le cas de la déconvolution, il s'agit de SEA_0. Et en apprentissage supervisé, cela dépend de plusieurs facteurs tels que la parcimonie et les dimensions du jeu de données. À l'heure actuelle, nous n'avons pas de méthode pour choisir la version de SEA la plus adaptée à une situation donnée.

6 STEAM : Algorithme exploratoire pour l'estimation de permutation

Sommaire

6.1 Introduction 130
6.1.1 Factorisation et matrices de Monarque 131
6.1.2 Permutations et algorithmes d'affectation 132
6.2 Principe de l'algorithme STEAM 133
6.3 Étude expérimentale de STEAM 137
6.3.1 Factorisation de matrices 137
6.3.2 Apprentissage supervisé 138
6.4 Conclusion 141

Dans le chapitre 3, nous avons introduit l'algorithme SEA. Cet algorithme explore l'espace discret des supports parcimonieux dans le cas linéaire à l'aide d'un estimateur direct (STE). Nous avons ensuite étudié ses propriétés théoriques et ses performances expérimentales dans le cadre de la reconstruction de signaux. Dans ce chapitre, nous utilisons à nouveau l'estimateur direct dans le cas des matrices creuses cette fois, plutôt que celui des vecteurs parcimonieux. Comme nous l'avons décrit dans la section 1.2.2 du chapitre 1, nous sommes confrontés à un problème d'optimisation sur l'espace discret des supports parcimonieux. Cependant, notre objectif ici est différent : il s'agit d'effectuer une recherche de matrices de permutations afin d'augmenter l'expressivité d'un modèle de factorisation de matrices en produit de matrices creuses. Ce chapitre reprend le contenu de (MOHAMED, EMIYA et al. 2025).

6.1 Introduction

Comme nous l'avons expliqué dans les généralités au chapitre 1, les problèmes d'optimisation qui nous intéressent concernent principalement la reconstruction sous des contraintes de parcimonie sur le support. Nous cherchons à résoudre ces problèmes à l'aide d'un estimateur direct (STE), similaire à ceux employés en quantification ou en binarisation. Dans le chapitre 3, nous avons introduit l'algorithme SEA, qui applique l'estimateur direct (STE) à la reconstruction de support de vecteurs parcimonieux. Nous allons à présent adapter cet algorithme à la recherche de matrices

de permutation dans le cadre de la factorisation de matrices en produit de matrices creuses.

6.1.1 Factorisation et matrices de Monarque

Les matrices jouent un rôle central en traitement du signal et en apprentissage automatique, que ce soit pour le stockage de données ou la représentation et manipulation d'opérateurs linéaires. À ce titre, elles constituent souvent un goulot d'étranglement en termes de calcul et de stockage, ce qui a motivé de nombreuses recherches pour surmonter ces limitations.

La factorisation de matrices en produit de matrices creuses a été récemment étudiée dans ce contexte, donnant naissance à divers modèles tels que les factorisations multi-couches (Le Magoarou et al. 2016 ; Le et Gribonval 2021 ; Giffon et al. 2021) et les factorisations Papillon (Y. Li et al. 2015 ; Dao, Gu et al. 2019 ; Dao, Sohoni et al. 2020). Ces modèles définissent des classes de matrices structurées capables de représenter de nombreux opérateurs linéaires pertinents, tout en réduisant la complexité spatiale et temporelle. De plus, ces classes disposent d'algorithmes d'approximation et de projection efficaces. Plus récemment, les auteurs de (Dao, B. Chen et al. 2022) ont introduit le modèle de Monarque, qui consiste à approcher des matrices par un produit de matrices bloc-diagonales et de matrices de permutation. Ce modèle a été utilisé pour concevoir des couches de réseau de neurones moins coûteuses (Fu et al. 2024), tout en étant comparables aux couches « Mixer » (Tolstikhin et al. 2021 ; Trockman et al. 2023) qui gagnent maintenant de l'attention (Qiu et al. 2024 ; Hayase et al. 2024 ; Tan et al. 2024 ; Lee et al. 2024). Les factorisations de Monarque et Papillon incluent des permutations, qui sont soit fixes (Dao, B. Chen et al. 2022), soit choisies dans un petit ensemble de candidats (Dao, Gu et al. 2019), ce qui limite l'expressivité du modèle.

En particulier, les matrices de Monarque permettent d'approximer les matrices denses par des factorisations qui réduisent significativement la complexité spatiale et temporelle lors des produits matriciels. Ces propriétés sont particulièrement intéressantes pour les couches denses et d'attention (Vaswani 2017) des réseaux de neurones.

Dans sa forme la plus générale, une matrice de Monarque d'ordre p est définie comme un produit $M = \prod_{i=1}^{p} (P_i B_i) P_0$ qui alterne des permutations P_i et des matrices bloc-diagonales B_i. En pratique, les travaux existants (Dao, B. Chen et al. 2022 ; Fu et al. 2024) se sont concentrés sur le modèle

$$M = P_2 \, L \, P_1 \, R \, P_0 \tag{6.1}$$

où $M \in \mathbb{R}^{N \times N}$, $P_0, P_1, P_2 \in \mathscr{P}_N$ sont des permutations fixes, et $L, R \in \mathscr{BD}^{(N)}$ sont des matrices bloc-diagonales composées de n blocs de taille $n \times n$, avec $n^2 = N$. Le modèle avec $P_0 = I_N$ (matrice d'identité) et $P_1 = P_2 = \bar{P}$ une variante en « base n » de la permutation par renversement de bits (« bit-reversal permutation » en anglais) a été étudié dans (Dao, B. Chen et al. 2022), tandis que le modèle avec $P_0 = P_1 = P_2 = \bar{P}$ a

été utilisé dans (FU et al. 2024). Dans les deux cas, la complexité en temps et en espace de la factorisation qui en résulte est de l'ordre de $\mathcal{O}\left(N^{\frac{3}{2}}\right)$ au lieu de $\mathcal{O}\left(N^2\right)$.

Pour toute matrice $A \in \mathbb{R}^{N\times N}$, (DAO, B. CHEN et al. 2022) ont proposé un algorithme de complexité $\mathcal{O}\left(N^{\frac{5}{2}}\right)$ en temps résolvant le problème de projection sur l'ensemble $\mathscr{M}^{(N)}$ des matrices de Monarque de taille N dans le cas où $P_0 = I_N$ et $P_1 = P_2 = \bar{P}$:

$$(\tilde{L}, \tilde{R}) = \pi_{\mathscr{M}}(A) \in \underset{L,R\in\mathscr{BD}^{(N)}}{\operatorname{argmin}} \frac{1}{2}\|\bar{P}L\bar{P}R - A\|_F^2 \tag{6.2}$$

Nous appelons $\mathrm{p}_{\mathscr{M}}(A) = \bar{P}\tilde{L}\bar{P}\tilde{R}$ la matrice de Monarque résultante (à différencier de $\pi_{\mathscr{M}}(A) = (\tilde{L}, \tilde{R})$ qui est seulement les matrices bloc-diagonales). Dans le cas $P_0 = P_1 = P_2 = \bar{P}$, une projection a été proposée dans (FU et al. 2024). Dans ce chapitre, nous nous intéressons à l'extension de ce modèle à des matrices de permutation autre que $\bar{P}$ afin d'augmenter l'expressivité du modèle de Monarque, c'est-à-dire mieux approximer les matrices en général.

6.1.2 Permutations et algorithmes d'affectation

Comme nous l'avons vu dans les généralités, une matrice de permutation est une matrice carrée binaire, où chaque ligne et chaque colonne contient exactement un unique 1. Cependant, dans les problèmes de recherche de permutation, on retrouve les mêmes difficultés que pour les problèmes de parcimonie en norme ℓ_0, à savoir la non-convexité et la non-différentiabilité du problème. Ainsi, l'apprentissage d'une permutation est souvent une tâche difficile. En général, deux types de méthodes sont utilisés pour traiter ce type de problème. Les premières reposent sur des algorithmes combinatoires génériques, tels que ceux basés sur l'opérateur de Sinkhorn (MENA et al. 2018 ; EMAMI et al. 2018). Ces méthodes nécessitent de grands jeux de données pour l'entraînement et sont souvent spécifiques à une tâche particulière. Les secondes reposent sur des relaxations, qui nécessitent également l'opérateur de Sinkhorn (LYU et al. 2020), et souffrent de limitations similaires. L'estimateur direct (STE) a déjà été utilisé pour apprendre des permutations afin d'aligner plusieurs réseaux de neurones (AINSWORTH et al. 2022). Cependant, il entraîne généralement une augmentation conséquente du coût de calcul.

Une autre façon de considérer une matrice de permutation consiste à la considérer comme une matrice d'affectation. Il s'agit d'une matrice binaire qui associe des lignes à des colonnes. Ainsi, elle peut être interprétée comme la solution d'un problème d'affectation, qui consiste à associer des éléments de deux ensembles de manière optimale. Le problème d'affectation est un problème classique en optimisation combinatoire, où l'objectif est de trouver une correspondance optimale entre les éléments de deux ensembles (ici de taille identique). Il existe plusieurs formulations équivalentes pour définir le problème d'affectation. Dans ce chapitre, nous adopterons la définition suivante.

Définition 6.1.1. (*Problème d'affectation*) Soient A et B deux ensembles de même cardinalité N. Soit C une matrice de coût de taille $N \times N$. Le problème d'affectation consiste à trouver une matrice d'affectation (ou de permutation) P de taille $N \times N$ telle que

$$P = \underset{P \in \mathscr{P}_N}{\operatorname{argmin}} \operatorname{Tr}(CP) = \underset{P \in \mathscr{P}_N}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{j=1}^{N} C_{i,j} P_{i,j},$$

où $\mathscr{P}_N$ est l'ensemble des matrices d'affectation (ou de permutation) de taille N.

Ce problème est aussi équivalent à un problème de couplage (ou appariement) parfait dans un graphe biparti complet (BURKARD et al. 2012).

Il existe plusieurs algorithmes permettant de résoudre le problème d'affectation en temps polynomial. Le plus connu est l'algorithme hongrois (KUHN 1955) et ses variantes. On peut aussi citer l'algorithme d'enchère (D. P. BERTSEKAS 1988) ainsi que ses améliorations récentes (D. BERTSEKAS 2024). Il fonctionne comme une vente aux enchères, où un ensemble de personnes A se disputent un ensemble d'objets B. Dans ce scénario, chaque objet se voit attribuer un prix qui, dans un certain sens, représente l'importance de l'objet pour les personnes. Le processus d'optimisation s'effectue comme dans le cadre d'un appel d'offres concurrentiel, où les prix des objets sont réduits de manière appropriée afin de rendre l'objet désiré par une personne moins désirable pour les autres personnes. Enfin, le dernier algorithme couramment utilisé est celui de Goldberg et Kennedy (GOLDBERG et al. 1995). Ce dernier applique des techniques de réseaux de flots au problème d'affectation, car ce dernier peut être vu comme une instance du problème de réduction de flots dans un réseau (ou graphe). Cet algorithme est basé sur la technique dite de « push-relabel », utilisée pour résoudre des problèmes de flots maximaux dans un réseau.

Ces trois algorithmes fournissent une solution exacte du problème d'affectation. Avec nos notations, leur complexité dans le pire des cas est respectivement en $\mathscr{O}\left(N^3\right)$, $\mathscr{O}\left(N^3 \log(N\|C\|_\infty)\right)$ et $\mathscr{O}\left(N^{5/2} \log(N\|C\|_\infty)\right)$ pour l'algorithme hongrois, l'algorithme d'enchère échelonné et l'algorithme de Goldberg et Kennedy. Dans ce travail, nous avons choisi d'utiliser l'algorithme d'enchère agressif pour résoudre les problèmes d'affectation. Ce choix est motivé par des résultats montrant qu'il est souvent le plus rapide en moyenne (ALFARO et al. 2022). De plus, l'algorithme d'enchère présente l'avantage de pouvoir ajuster efficacement ses coûts internes en cas de modification légère de la matrice de coût C, ce qui permet de trouver une nouvelle solution optimale plus rapidement que si le calcul était effectué depuis le début.

6.2 Principe de l'algorithme STEAM

Comme mentionné dans l'introduction, l'approximation d'une matrice par une factorisation de Monarque $P_2\ L\ P_1\ R\ P_0$ n'a été réalisée qu'avec des permutations fixes P_0, P_1 et P_2, en optimisant uniquement les matrices bloc-diagonales L et R. Notre

objectif est d'étudier la possibilité d'optimiser également certaines permutations, afin d'augmenter l'expressivité du modèle. Dans ce chapitre, nous allons donc apprendre les permutations P_0 et P_2 tout en gardant P_1 fixe, afin de pouvoir continuer à utiliser les projecteurs $\pi_{\mathscr{M}}$ et $\mathrm{p}_{\mathscr{M}}$ proposé dans (Dao, B. Chen et al. 2022). Le problème d'optimisation qui nous intéresse s'écrit :

$$\underset{\substack{L,R\in\mathscr{BD}^{(N)}\\ P_0,P_2\in\mathscr{P}_N}}{\text{Minimiser}} \frac{1}{2}\|P_2L\bar{P}RP_0 - A\|_F^2, \tag{6.3}$$

où $A \in \mathbb{R}^{N\times N}$ est la matrice que nous cherchons à approcher.

Ce problème peut se reformuler comme suit (en gardant $P_1 = \bar{P}$ fixe pour bénéficier de la projection $\mathrm{p}_{\mathscr{M}}$ sur l'ensemble des matrices de Monarque $\mathscr{M}^{(N)}$) :

Proposition 6.2.1. (*Problème de recherche de permutations*) Soit $A \in \mathbb{R}^{N\times N}$, on définit la fonction

$$F : \begin{cases} \mathscr{P}_N \times \mathscr{P}_N & \to \mathbb{R} \\ (P_0, P_2) & \mapsto \frac{1}{2}\|\mathrm{p}_{\mathscr{M}}(\bar{P}P_2^T AP_0^T)\|_F^2 \end{cases}$$

Alors, le problème (6.3) est équivalent au problème

$$\underset{P_0,P_2\in\mathscr{P}_N}{\text{Minimiser}} F(P_0, P_2). \tag{6.4}$$

Démonstration. Puisque $P_0, P_2, \bar{P}$ sont des matrices orthogonales, nous avons

$$\begin{aligned} &\min_{\substack{L,R\in\mathscr{BD}^{(N)}\\ P_0,P_2\in\mathscr{P}_N}} \frac{1}{2}\|P_2L\bar{P}RP_0 - A\|_F^2 \\ &\qquad = \min_{\substack{L,R\in\mathscr{BD}^{(N)}\\ P_0,P_2\in\mathscr{P}_N}} \frac{1}{2}\|\bar{P}L\bar{P}R\bar{P} - \bar{P}P_2^T AP_0^T\|_F^2 \\ &\qquad = \min_{P_0,P_2\in\mathscr{P}_N} F(P_0, P_2). \end{aligned}$$

Nous voulons écrire un algorithme utilisant un schéma alterné pour minimiser $F(P_0, P_2)$. Cependant, comme pour la reconstruction de supports parcimonieux, nous ne pouvons pas efficacement minimiser directement F à l'aide de techniques basées sur le gradient, car l'espace $\mathscr{P}_N \times \mathscr{P}_N$ est discret. Pour surmonter cette difficulté, nous allons utiliser le même raisonnement que dans le chapitre 3. Plus précisément, nous introduisons des variables denses $\mathscr{X}_0$ et $\mathscr{X}_2$, analogues à la variable d'exploration $\mathscr{X}$ utilisée dans SEA, ainsi qu'une application H, qui joue un rôle similaire à l'application du même nom dans SEA. Nous nous en servirons afin de résoudre le problème (6.4) en minimisant $F \circ H$ sur $\mathbb{R}^{N\times N} \times \mathbb{R}^{N\times N}$, plutôt que de minimiser F sur $\mathscr{P}_N \times \mathscr{P}_N$. Ici, nous avons volontairement utilisé les mêmes notations que dans le chapitre 3 pour souligner la similitude entre les deux approches. Toute ambiguïté est levée par le

contexte, puisque les applications F et H n'ont pas les mêmes définitions dans ces deux chapitres, bien qu'elles aient le même rôle.

Proposition 6.2.2. Soient $\mathscr{X}_0, \mathscr{X}_2 \in \mathbb{R}^{N\times N}$, posons $H(\mathscr{X}_0, \mathscr{X}_2) = (h(\mathscr{X}_0), h(\mathscr{X}_2))$ avec $h : \mathbb{R}^{N\times N} \to \mathscr{P}_N$ une application surjective. Alors le problème (6.4) est équivalent à

$$\underset{\mathscr{X}_0,\mathscr{X}_2\in\mathbb{R}^{N\times N}}{\text{Minimiser}} \; F(H(\mathscr{X}_0, \mathscr{X}_2)) \tag{6.5}$$

Démonstration. Puisque h est surjective, l'image de H est $\mathscr{P}_N \times \mathscr{P}_N$ et le minimum de F sur $\mathscr{P}_N \times \mathscr{P}_N$ et celui de $F \circ H$ coïncident.

Maintenant, si l'on considère $\mathscr{X}_0$ et $\mathscr{X}_2$ comme des matrices de coût, h peut être n'importe quel algorithme résolvant le problème d'affectation décrit dans la définition 6.1.1. Nous avons choisi d'utiliser l'algorithme d'enchères agressif (D. P. BERTSEKAS 1988) en raison de ses performances reconnues. Cet algorithme agit effectivement comme une application surjective sur $\mathscr{P}_N$, car, par définition du problème d'affectation, $h(P) = P$ pour tout $P \in \mathscr{P}_N$.

Ainsi, nous construisons l'algorithme 8, appelé STE Alternate minimization for Monarch (STEAM), pour trouver une solution approchée du problème (6.5). Tout comme SEA, il repose sur une procédure de rétropropagation du gradient basée sur l'estimateur direct (STE) pour minimiser $F(H(\mathscr{X}_0, \mathscr{X}_2))$. La figure 6.1 illustre le graphe de calcul correspondant à $F(H(\mathscr{X}_0, \mathscr{X}_2))$.

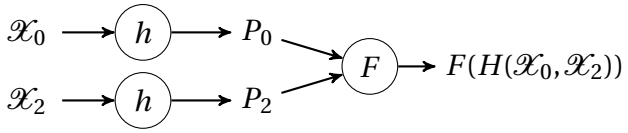

FIGURE 6.1 – Graphe de calcul de $F(H(\mathscr{X}_0, \mathscr{X}_2))$

STEAM alterne des mises à jour de $\mathscr{X}_i$, pour $i \in \{0, 2\}$, basées sur l'estimateur direct (STE) (ligne 9), et un pas contrôlé par la constante de Lipschitz du gradient (ligne 8). Cette mise à jour est similaire à une étape de gradient où le gradient

$$\left.\frac{\partial F \circ H}{\partial \mathscr{X}_i}\right|_{(\mathscr{X}_0,\mathscr{X}_2)} = \left.\frac{\partial F}{\partial P_i}\right|_{(P_0,P_2)} \left.\frac{\partial h}{\partial \mathscr{X}_i}\right|_{\mathscr{X}_i}$$

est approché par :

$$\begin{aligned}\left.\frac{\partial F}{\partial P_i}\right|_{(P_0,P_2)} &= \frac{\partial}{\partial P_i}\frac{1}{2}\|P_2 L\bar{P}RP_0 - A\|_F^2 \\ &= \begin{cases} (P_2 L\bar{P}R)^T(P_2 L\bar{P}RP_0 - A) & \text{si } i = 0, \\ (P_2 L\bar{P}RP_0 - A)(L\bar{P}RP_0)^T & \text{si } i = 2. \end{cases}\end{aligned}$$

Algorithme 8 STEAM

1 : **Entrées :** Matrice A, paramètre $\alpha \in \mathbb{R}^+$, nombre d'itérations T
2 : **Sorties :** Facteurs $\tilde{P}_2, \tilde{L}, \tilde{R}, \tilde{P}_0$ de $\hat{A} = \tilde{P}_2 \tilde{L} \bar{P} \tilde{R} \tilde{P}_0$
3 : $t \leftarrow 1$
4 : $P_0, P_2, \mathscr{X}_0, \mathscr{X}_2 \leftarrow \bar{P}$
5 : $(L, R) \leftarrow \pi_{\mathscr{M}}(AP_0^T)$
6 : **Pour** $t \in [\![1, T]\!]$, **faire** :
7 : **Pour** $i \in \{0, 2\}$, **faire** :
8 : $\eta \leftarrow \frac{1}{\alpha \|L\bar{P}R\|_2^2}$
9 : $\mathscr{X}_i \leftarrow \mathscr{X}_i - \eta \frac{\partial}{\partial P_i} \frac{1}{2} \|P_2 L \bar{P} R P_0 - A\|_F^2$
10 : $P_i \leftarrow h(\mathscr{X}_i)$
11 : $(L, R) \leftarrow \pi_{\mathscr{M}}(\bar{P} P_2^T A P_0^T)$
12 : **Fin pour**
13 : $L^t, R^t, P_0^t, P_2^t \leftarrow L, R, P_0, P_2$
14 : **Fin pour**
15 : $t_{BEST} = \underset{t' \in [\![1,T]\!]}{\operatorname{argmin}} \|P_2^{t'} L^{t'} \bar{P} R^{t'} P_0^{t'} - A\|_F$
16 : **renvoyer** $P_2^{t_{BEST}}, L^{t_{BEST}}, R^{t_{BEST}}, P_0^{t_{BEST}}$

La permutation P_i est calculée à partir de $\mathscr{X}_i$ (ligne 10), tandis que les facteurs bloc-diagonaux L et R sont obtenus à partir de la projection de Monarque (ligne 11) comme décrit dans la proposition 6.2.2 et le problème (6.4).

Le paramètre α (ligne 8) contrôle le pas de gradient. Il doit être choisi afin que $\frac{1}{\eta}$ soit strictement supérieur à la constante de Lipschitz du gradient de F (Bolte et al. 2014). En pratique, nous avons choisi $\alpha = 1.001$ comme dans (Le Magoarou et al. 2016). Une initialisation naturelle de l'algorithme est de fixer $P_0, P_2, \mathscr{X}_0, \mathscr{X}_2$ à $\bar{P}$ et d'utiliser la projection de Monarque pour obtenir L et R (lignes 4-5). À la fin, les meilleures permutations et matrices bloc-diagonales rencontrées au cours de la procédure sont renvoyées (ligne 15), car la fonction de coût peut ne pas diminuer à chaque itération, comme dans SEA. En effet, comme détaillé dans le chapitre 3 pour la reconstruction de supports parcimonieux, nous avons des variables exploratoires $\mathscr{X}_i$ qui accumulent des gradients au cours des itérations, ce qui nous permet d'explorer l'espace des permutations et échapper aux minima locaux. Un exemple typique de variation de F est représenté dans la figure 6.2.

Chaque itération de notre algorithme fait uniquement appel à des produits matriciels, dont la majorité inclut au moins une permutation, menant à une complexité temporelle en $\mathscr{O}(N^2)$. Les produits restants peuvent être réarrangés pour inclure un facteur bloc-diagonal, ce qui conduit à une complexité temporelle en $\mathscr{O}(N^{5/2})$. La complexité en temps de la projection $\pi_{\mathscr{M}}$ est en $\mathscr{O}(N^{5/2})$. La meilleure complexité théorique dans le pire des cas d'un algorithme d'affectation (Orlin et al. 1992) est en $\mathscr{O}(N^{5/2} \log(N\|\mathscr{X}\|_\infty))$. Cependant, (Alfaro et al. 2022) montre que la complexité

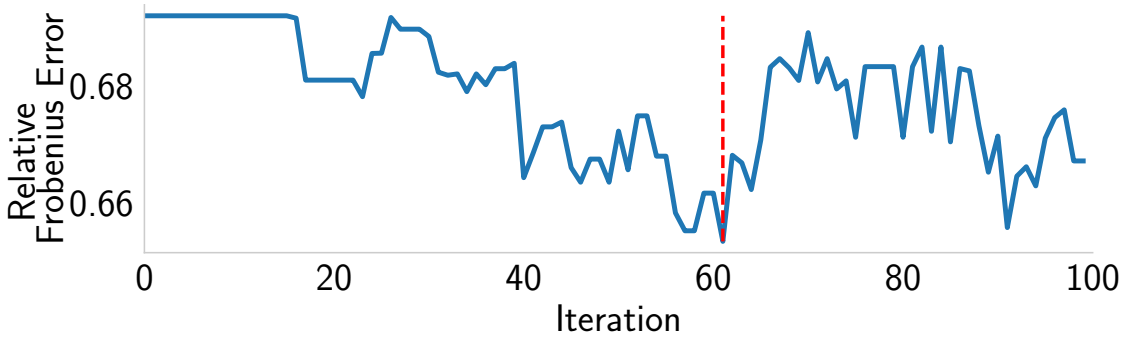

FIGURE 6.2 – Erreur de Frobenius relative entre A et la factorisation courante en fonction de l'itération de STEAM quand $N = 49$. La ligne rouge en pointillé indique t_{BEST}.

expérimentale de l'algorithme d'enchères est généralement inférieure, sa complexité dans le meilleur des cas étant en $\mathcal{O}(N^2)$. De plus, après la première itération, nous initialisons l'algorithme d'enchères avec des éléments de son instance précédente. On sait (D. BERTSEKAS 2024) qu'une telle stratégie réduit considérablement le temps de calcul. Ainsi, le temps d'exécution global de notre algorithme dépend fortement des performances empiriques de l'algorithme d'enchères. Dans le pire des cas, sa complexité reste sous-cubique, comme celle de la projection de Monarque.

6.3 Étude expérimentale de STEAM

Nous évaluons les performances de notre approche par rapport à la factorisation de Monarque originale qui garde les permutations fixes. Tout d'abord, nous examinons la capacité des modèles et des algorithmes à approcher une matrice générée à partir du modèle de l'équation (6.1). Ensuite, nous illustrons les capacités d'apprentissage des différents modèles dans une tâche d'apprentissage supervisé. Les expériences ont été réalisées sur un seul CPU. Le code Python est disponible à l'adresse https://gitlab.lis-lab.fr/valentin.emiya/STEAM.

6.3.1 Factorisation de matrices

Dans cette expérience, nous générons aléatoirement des matrices A à partir du modèle de l'équation (6.1). Les entrées bloc-diagonales de L et R sont tirées indépendamment selon la loi normale centrée réduite ; P_2 est tirée uniformément depuis $\mathscr{P}_N$; et P_0 est soit fixée à I_N, soit tirée uniformément depuis $\mathscr{P}_N$. Nous faisons varier la taille de la matrice $N \in \{n^2 \mid n \in [\![2,10]\!]\}$. Pour chaque valeur de N, nous générons 100 matrices A à factoriser. Les solutions sont estimées à partir de la projection de Monarque originale, de STEAM (en ignorant la boucle intérieure pour $i = 0$ quand $P_0 = I_N$), avec $\alpha = 1.001$ et $T \in \{100, 1000\}$ itérations, ainsi que d'une stratégie d'exploration purement aléatoire. Dans ce dernier cas, des permutations aléatoires sont

tirées uniformément depuis $\mathscr{P}_N$, T fois. Chaque tirage est suivi d'une projection de Monarque afin d'estimer L et R. Enfin, la meilleure factorisation rencontrée est renvoyée.

Les performances sont illustrées sur la figure 6.3 à l'aide de l'erreur de Frobenius relative $\frac{\|\hat{A}-A\|_F}{\|A\|_F}$ entre A et l'approximation $\hat{A}$ retournée par chaque algorithme, en fonction de N. Notre algorithme est plus performant que la projection de Monarque pour toutes les tailles de matrice considérées N. La différence la plus marquée est observée en petite dimension. Nous remarquons que l'augmentation du nombre d'itérations dans notre algorithme améliore ses performances. La stratégie purement aléatoire n'est pas représentée, car elle obtient de très mauvaises performances, avec des erreurs supérieures à 1 dans presque tous les cas. Nous avons également mené une expérience de factorisation similaire où les entrées de la matrice sont tirées selon la loi normale centrée réduite. Les résultats sont similaires à ceux de la figure 6.3 et notre approche conserve les meilleures performances. Cette expérience suggère que notre méthode est capable de fournir une meilleure factorisation que la projection de Monarque, bien que l'exploration de l'espace des permutations soit une tâche difficile. En effet, notre algorithme ne parvient généralement pas à retrouver la meilleure permutation (il n'atteint pas une erreur de nulle). De plus, il est plus performant dans le cas où seule une permutation est recherchée (cas $P_0 = I_N$) que dans le cas plus difficile où les deux permutations sont inconnues. Cependant, dans les deux cas, il parvient à trouver de bonnes permutations, contrairement à la stratégie d'exploration pure qui ne renvoie pas de factorisations intéressantes.

6.3.2 Apprentissage supervisé

Pour comparer la capacité d'apprentissage de notre modèle avec celle du modèle de Monarque, nous proposons une procédure d'entraînement complète inspirée de (Dao, B. Chen et al. 2022). Nous effectuons une tâche de classification sur le jeu de données MNIST (LeCun et al. 1998) en utilisant les jeux d'entraînement et de test usuels. La classification est réalisée par un Perceptron multicouche (MLP) avec une couche cachée de taille 784, une activation ReLU, et une couche de décision LogSoftmax avant de calculer une fonction de coût basée sur l'anti log-vraisemblance. Nous entraînons ce Perceptron à l'aide des optimiseurs de premier ordre Adadelta (Zeiler 2012) et AdamW (Loshchilov 2017) sur 30 itérations. Ensuite, nous recommençons cet entraînement à partir de zéro en remplaçant la matrice dense 784×784 de la couche cachée soit par une factorisation de Monarque, soit par notre modèle de factorisation, illustré par le graphe de calcul de la figure 6.4.

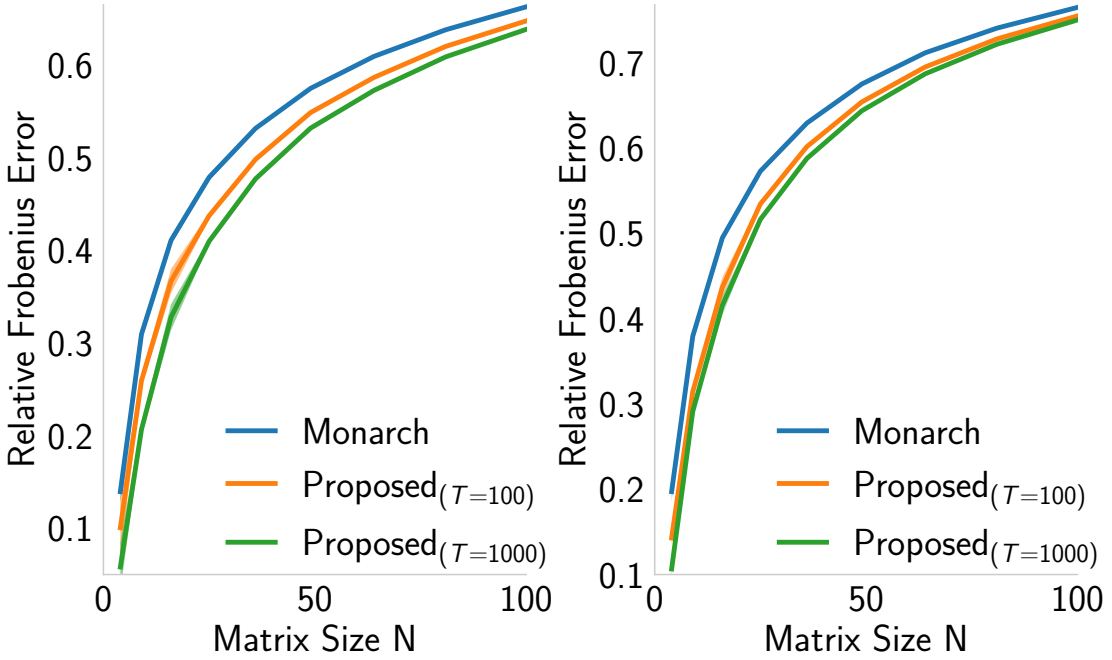

FIGURE 6.3 – Erreur de Frobenius relative pour diverses factorisations en fonction de la taille N de la matrice A. La matrice A est générée aléatoirement à partir de équation (6.1) avec $P_0 = I_N$ (gauche) ou $P_0 \in \mathscr{P}_N$ (droite). L'intervalle de confiance à 99% est très étroit (zones sombres à peine visibles).

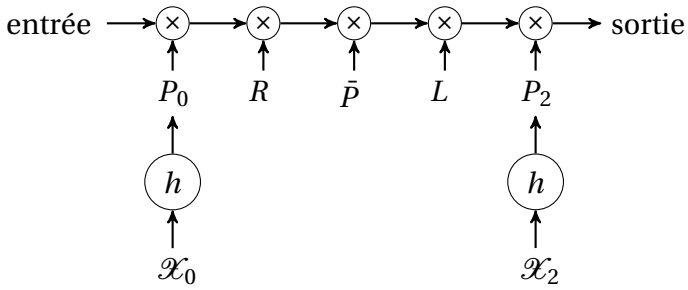

FIGURE 6.4 – Graphe de calcul de la factorisation remplaçant la matrice dense de la couche cachée du Perceptron.

Comme il n'y a pas de matrice apprise à factoriser, nous ne pouvons utiliser ni la projection de Monarque ni l'algorithme 8. Cependant, l'estimateur direct (STE) peut être utilisé pour rétropropager les gradients à travers h, et notre factorisation, tout comme celle de Monarque, peut être entraînée de la même manière qu'un Perceptron classique, c'est-à-dire en utilisant une méthode de descente de gradient. Dans cette expérience, nous nous concentrons sur les cas où nous apprenons P_0 et/ou P_2 en même temps que L et R. Lorsqu'elles sont apprises, P_0 et P_2 sont initialisées à I_N. Dans le cas contraire, elles sont égales à $\bar{P}$.

Le tableau 6.1 montre la précision obtenue sur le jeu de test de chaque modèle après

	Adadelta	**AdamW**
Matrice dense	98.41%	96.35%
Monarque (apprentissage de L et R)	96.26%	92.92%
STEAM (apprentissage de P_2, L et R)	96.12%	**97.49%**
STEAM (apprentissage de L, R et P_0)	**96.39%**	97.00%
STEAM (apprentissage de P_2, L, R et P_0)	96.38%	96.08%

TABLEAU 6.1 – Précision sur le jeu de données de MNIST en utilisant les optimiseurs AdamW et Adadelta.

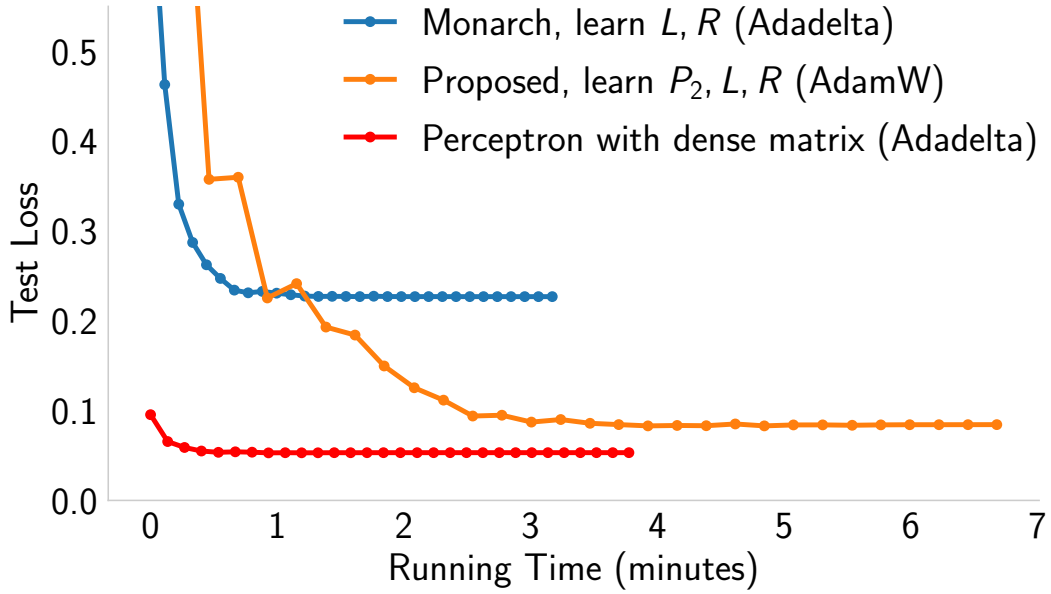

FIGURE 6.5 – Valeurs prises par la fonction de coût sur le jeu de données de test de MNIST en fonction du temps pendant l'entraînement d'un Perceptron avec une couche cachée composée d'une matrice dense (615 000 paramètres), d'une factorisation de Monarque classique (44 000 paramètres), ou de notre factorisation (entraînement : 659 000 paramètres, inférence : 45 000 paramètres). Chaque point représente une itération.

l'apprentissage. Alors que la précision du modèle de Monarque original (96.26%) est plus de 2% plus bas que celle du Perceptron (98.41%), l'apprentissage d'une permutation permet d'obtenir une meilleure précision (97.49%), seulement 1% en dessous de celle du Perceptron. Notre modèle est également moins sensible au choix de l'optimiseur. Comme dans l'expérience de factorisation, notre modèle est plus performant lorsqu'une seule permutation est apprise. La figure 6.5 présente la fonction de coût sur le jeu de données de test en fonction du temps d'exécution. Bien que notre code ne soit

pas entièrement optimisé, notre modèle obtient des performances similaires à celles du modèle de Monarque original à $t = 1$s, même si chaque itération prend plus de temps pour notre modèle. À partir de ce moment, le modèle de Monarque original plafonne en performance, tandis que notre modèle continue de s'améliorer. Ainsi, nous pouvons nous rapprocher significativement des performances du Perceptron grâce à une augmentation limitée du coût de calcul. Ceci est particulièrement intéressant lors de l'inférence, où les complexités spatiales et temporelles sont proportionnelles au nombre de paramètres appris dans cette couche, qui passe de $N^2 = 614\,656$ pour le Perceptron à $2N^{\frac{3}{2}} + N = 44\,688$ pour notre approche et $2N^{\frac{3}{2}} = 43\,904$ pour le modèle de Monarque original.

6.4 Conclusion

Dans ce chapitre, nous avons étendu le modèle de Monarque pour améliorer son expressivité, en apprenant certaines permutations à l'aide du principe de l'estimateur direct (STE). Nous avons décrit un algorithme (nommé STEAM) pour estimer une telle factorisation. La construction de ce dernier repose sur les mêmes principes que ceux ayant conduit à la conception de SEA dans le chapitre 3. Nous avons montré que notre approche est capable de fournir une meilleure factorisation que la projection de Monarque originale, en particulier pour les petites matrices. Nous avons également montré comment utiliser cette factorisation en tant que couche dense d'un réseau de neurones de type Perceptron dans une procédure d'entraînement classique complète. Les expériences de ce chapitre ont démontré l'efficacité de l'approche tout en maintenant un coût de calcul limité. Ainsi, cette approche peut être utilisée pour remplacer les matrices denses d'un réseau de neurones sans modifier la procédure d'entraînement.

Conclusions et perspectives

Dans cette thèse, nous avons développé une méthodologie pour la résolution de problèmes d'optimisation sous contrainte de parcimonie en norme ℓ_0, avec une attention particulière pour la reconstruction de support. À l'ère du « tout différentiable », nous avons proposé une approche basée sur la technique de l'estimateur direct (STE) utilisée en apprentissage profond, afin de tirer parti des nombreuses techniques de descente de gradient disponibles dans la littérature. Cette technique permet en effet de surmonter le problème de non-différentiabilité des opérateurs impliqués dans les problèmes de parcimonie.

Dans le chapitre 1, nous avons introduit les notions clés nécessaires à la compréhension de ce manuscrit. Nous avons présenté le problème de reconstruction de support ainsi que celui de la factorisation de matrices en produit de matrices creuses. Nous avons également décrit les méthodes de résolution existantes et leurs limites. Enfin, nous avons introduit le principe de l'estimateur direct (STE).

Dans le chapitre 2, nous avons proposé une approche permettant de substituer un modèle physique par un modèle d'apprentissage favorisant la parcimonie. Nous avons d'abord montré que les descripteurs de performance issus du diagramme de bifurcation de la trompette permettent de différencier des trompettes commerciales. Ensuite, suivant une méthodologie standard, nous avons développé un modèle générant ces descripteurs pour des musiciens de référence et une trompette donnée. Enfin, nous avons encapsulé ce modèle dans une application web, facilement utilisable par un concepteur de trompette, contrairement à l'approche physique traditionnelle.

La principale contribution de cette thèse s'étend sur les chapitres 3 à 5.

Dans le chapitre 3, nous avons proposé une approche basée sur l'estimateur direct (STE) pour la résolution du problème de reconstruction de support dans le cas linéaire. En effet, nous avons montré qu'en reformulant le problème de reconstruction de support, nous pouvions utiliser l'estimateur direct (STE) pour construire l'algorithme SEA (Support Estimation Algorithm). Cet algorithme se distingue par l'introduction d'une variable d'exploration de support $\mathscr{X}$ permettant une exploration de l'espace des supports avec un coût computationnel maîtrisé. Les capacités d'exploration de SEA ont été mises en évidence à travers des expériences de diagramme de phase, où nous l'avons comparé aux méthodes de l'état de l'art. Ces expériences ont montré que SEA, initialisé à ELS, offre des performances supérieures dans les zones du diagramme où les matrices sont les plus cohérentes. Cependant, l'influence des paramètres de SEA sur son fonctionnement demandent à être étudiés. En effet, le pas η n'a pas la même utilité que dans les algorithmes de descente classique. Ce paramètre contrôle la vitesse à laquelle l'initialisation de l'algorithme est remplacée par la somme des gradients calculés au cours des itérations. L'impact de la modification

de ce paramètre devrait être étudié à l'avenir. Enfin, chaque support est actuellement complètement exploré. Il serait intéressant d'examiner si une modification du compromis exploration/exploitation de SEA (modifier l'opérateur H) pourrait réduire le coût computationnel sans affecter significativement ses performances.

Dans le chapitre 4, nous avons démontré théoriquement que SEA est capable de retrouver le support de la solution du problème de reconstruction de support. Nous avons d'abord analysé un algorithme oracle, qui met à jour la variable $\mathscr{X}$ en se basant directement sur la solution du problème x^*, garantissant systématiquement la reconstruction correcte du support (théorème 4.2.1). En étudiant la différence entre les mises à jour de l'algorithme oracle et celles de SEA, nous avons établi que SEA retrouve le support sous certaines conditions. Dans un premier temps, nous avons formulé une condition suffisante générale (*RC*) (théorème 4.3.1). Cependant, cette condition, dépendant de la solution du problème et des gradients calculés par SEA, n'est pas utilisable en pratique. Pour pallier cela, nous avons dérivé le corollaire 4.4.1, garantissant la reconstruction du support lorsque la matrice A est orthogonale. En exploitant la propriété d'isométrie restreinte (RIP), nous avons reformulé (*RC*) en une condition (RC_{RIP}), dépendant des constantes d'isométrie restreinte de A, du bruit e, et de la solution x^*. Si A satisfait le $(2k+1)$-RIP, le théorème 4.5.3 garantit que SEA retrouve le support. Cependant, cette condition reste difficile à interpréter. Nous avons donc dérivé du théorème 4.5.3, le corollaire 4.5.4, basé sur une condition (RC_{SRIP}) plus simple à comprendre. Cette condition met en évidence des cas non triviaux où SEA réussit à reconstruire le support, montrant notamment que la reconstruction est facilitée lorsque les pics du vecteur à reconstruire sont proches en amplitude.

Par ailleurs, comme il s'agit d'une méthode novatrice, cette étude théorique de l'algorithme n'en est qu'à ses débuts. Bien que SEA soit performant pour des matrices A cohérentes, cette observation n'est pas expliquée par l'analyse théorique actuelle, qui se limite aux matrices satisfaisant la propriété d'isométrie restreinte (RIP), donc peu cohérentes. Étant donné que la vérification explicite des conditions RIP est NP-difficile (Andreas M. TILLMANN et al. 2014), il serait pertinent de développer des garanties théoriques basées sur la cohérence mutuelle de A (D. DONOHO et HUO 2001), pour rendre les résultats théoriques applicables en pratique. Il serait également intéressant d'adapter la stratégie développée pour établir des garanties théoriques dans d'autres contextes, tels que l'optimisation de réseaux neuronaux quantifiés. En outre, il est actuellement difficile de comparer nos résultats à ceux d'autres algorithmes. Un travail plus approfondi sur les conditions des théorèmes pourrait permettre de dériver des bornes de reconstruction facilitant une comparaison théorique.

Dans le chapitre 5, nous avons évalué SEA par rapport à l'état de l'art dans deux contextes : la déconvolution de signaux et l'apprentissage supervisé avec une contrainte de budget sur les caractéristiques. En déconvolution, nous avons à la fois étudié avec précision le comportement des algorithmes sur une instance spécifique, ainsi que de manière intensive sur de nombreuses configurations, en utilisant des mesures de performance adaptées. Ces expériences ont montré que SEA offre un meilleur compromis entre exploration et exploitation que ses concurrents. En effet, il explore davantage que les autres algorithmes (excepté ELS), ce qui fait de lui le plus performant pour

retrouver le support, quelle que soit l'initialisation. SEA initialisé à zéro, quant à lui, obtient les meilleurs résultats. Cette capacité d'exploration réduit les erreurs d'estimation dans les zones du signal où les pics sont proches. En apprentissage automatique, nous avons utilisé SEA pour des tâches de régression et de classification avec une contrainte sur le nombre de caractéristiques à utiliser. Pour la classification, une légère modification de SEA était nécessaire, montrant sa capacité à minimiser toute fonction de coût continue et différentiable, sous contrainte de parcimonie en norme ℓ_0. Les expériences ont révélé que SEA est particulièrement efficace dans des problèmes de petite dimension et lorsque le niveau de parcimonie est élevé. Ainsi, la question de l'initialisation reste ouverte : dans le cas de la déconvolution, SEA initialisé à 0 est le plus performant, tandis que dans le diagramme de phase, SEA initialisé à ELS donne de meilleurs résultats. Dans le contexte de l'apprentissage, les performances dépendent fortement du jeu de données. Nous n'avons pas encore identifié de critère général permettant de choisir l'initialisation la plus pertinente. Cette question mérite d'être approfondie dans de futures recherches.

À plus long terme, il serait intéressant de revisiter certains choix de conception de SEA. En particulier, l'opérateur H :

$$H(\mathscr{X}) \in \underset{\substack{x \in \mathbb{R}^n \\ \mathrm{supp}(x) \subseteq \mathrm{largest}_k(\mathscr{X})}}{\mathrm{argmin}} F(x), \quad \forall \mathscr{X} \in \mathbb{R}^n,$$

ainsi que l'estimateur direct (STE) utilisé pour le calcul de son gradient. En effet, l'opération de projection aux moindres carrés (sans contrainte) étant différentiable, il pourrait être judicieux de limiter l'utilisation de l'estimateur direct (STE) à la seule partie non différentiable de l'opérateur H, à savoir l'opérateur $\mathrm{largest}_k$. Une autre question ouverte concerne l'utilisation d'un opérateur H non déterministe, qui pourrait permettre une exploration plus large de l'espace des supports. Cela pourrait être envisagé en traitant $\mathscr{X}$ comme une distribution discrète à partir de laquelle on échantillonne k éléments sans remise, comme proposé dans les travaux de (AHMED et al. 2023). Dans ces travaux, H est une opération d'échantillonnage pour laquelle un estimateur du gradient est disponible. Enfin, un axe de recherche intéressant serait de comparer notre méthode à une approche reposant sur une relaxation de l'opérateur $\mathrm{largest}_k$. Cette comparaison pourrait mettre en lumière les différences en termes d'exploration, de coût computationnel, et de reconstruction de support.

Le chapitre 6 prolonge la contribution principale. En tirant parti de l'expérience acquise sur le cas linéaire, nous avons abordé le problème de recherche de permutation dans le cadre de la factorisation de matrices en produits de matrices creuses. En partant du modèle de Monarque (combinant matrices de permutation fixes et matrices bloc-diagonales à estimer), nous avons proposé un modèle plus expressif, où les matrices de permutation sont également apprises. En reformulant ce problème selon la stratégie utilisée au chapitre 3, nous avons obtenu un problème d'optimisation non contraint adapté à l'utilisation de l'estimateur (STE). En combinant cette approche avec une méthode d'optimisation alternée, nous avons développé l'algorithme STEAM (STE Alternate minimization for Monarch). Similairement à SEA, STEAM utilise des

variables d'exploration pour parcourir l'espace des permutations, tout en maintenant un coût computationnel comparable à celui de la projection dans le modèle de Monarque. Nous avons comparé notre modèle et notre algorithme au modèle et à la projection de Monarque dans deux contextes. Dans un premier temps, nous avons étudié les capacités de chaque modèle et algorithme à approcher des matrices denses ou suivant notre nouveau modèle. Cette expérience a montré que STEAM améliore non seulement l'approximation des matrices de notre modèle, mais aussi celle des matrices denses. Dans un second temps, nous avons évalué les capacités d'apprentissage de notre modèle lorsqu'il est utilisé à la place des matrices denses dans un Perceptron pour résoudre un problème de classification sur MNIST. Cette expérience a révélé que notre modèle atteint, pour un même temps de calcul, la même erreur que celui de Monarque, tout en continuant à s'améliorer si le temps de calcul est prolongé.

Cette étude met en lumière les défis liés à l'apprentissage des permutations et les limites de cette approche. En effet, en travaillant sur la procédure d'optimisation, de meilleures performances pourraient être obtenues. De plus, l'apprentissage simultané de plusieurs permutations reste un problème particulièrement complexe. L'utilisation de matrices d'exploration $\mathscr{X}_i$ dans notre procédure entraîne une complexité en espace quadratique, ce qui contraste avec la complexité sous-quadratique des algorithmes basés sur des permutations fixes. Par ailleurs, les modèles de Monarque d'ordre supérieur à 2 nécessitent une étude plus approfondie, notamment pour leur application aux couches d'attention dans les réseaux de neurones.

Dans STEAM, nous utilisons un estimateur direct (STE) pour rétropropager des gradients à travers un algorithme d'affectation (algorithme d'enchère). Similairement à ce qu'il est possible pour SEA, à plus long terme, on pourrait remplacer l'algorithme d'affectation par l'échantillonnage d'une permutation depuis $\mathscr{X}_i$. Cela impliquerait d'étendre les travaux de (AHMED et al. 2023) dans un premier temps, pour dériver un estimateur du gradient de cette opération, puis de les adapter à notre problème. Encore une fois, il serait intéressant d'étudier l'impact d'une relaxation de toutes les opérations non différentiables de l'algorithme d'enchère en utilisant cette fois-ci les techniques de (BLONDEL et al. 2024).

Cette thèse, intitulée « Algorithmes exploratoires pour la frugalité par la parcimonie », a permis de développer des méthodes novatrices pour contribuer à la frugalité des modèles à l'aide de la parcimonie. Cependant, de nombreux défis restent à relever. Cette thèse apporte une réponse technique au besoin croissant de frugalité provenant de la course aux performances ayant lieu dans le domaine de l'apprentissage automatique, et plus généralement en informatique depuis les débuts de la discipline. Maintenant, on sait également que la réponse à ce besoin ne peut être seulement technique. En effet, l'histoire a monté (et continue de montrer) qu'en raison de l'effet rebond, les gains de frugalité obtenus par des avancées technologiques sont souvent annulés par une augmentation encore plus importante de la consommation énergétique et des ressources. Ainsi, la frugalité ne peut être obtenue que par une approche systémique, remettant en question non seulement les usages, mais aussi les modèles économiques et sociaux actuels.

Bibliographie

[Ahm+23] Kareem AHMED, Zhe ZENG, Mathias NIEPERT et al. « SIMPLE : A Gradient Estimator for k-Subset Sampling ». In : *The Eleventh International Conference on Learning Representations*. 2023 (cf. p. 144, 145).

[AHS22] Samuel K AINSWORTH, Jonathan HAYASE et Siddhartha SRINIVASA. « Git re-basin : Merging models modulo permutation symmetries ». In : *arXiv preprint arXiv :2209.04836* (2022) (cf. p. 132).

[Alf+22] Carlos A ALFARO, Sergio L PEREZ, Carlos E VALENCIA et al. « The assignment problem revisited ». In : *Optimization Letters* 16.5 (2022), p. 1531-1548 (cf. p. 133, 136).

[Alt92] Naomi S ALTMAN. « An introduction to kernel and nearest-neighbor nonparametric regression ». In : *The American Statistician* 46.3 (1992), p. 175-185 (cf. p. 53).

[AB18] Alexander G. ANDERSON et Cory P. BERG. « The High-Dimensional Geometry of Binary Neural Networks ». In : *International Conference on Learning Representations*. 2018 (cf. p. 78).

[ACV05] Rémi ARQUIER, Bruno COCHELIN et Christophe VERGEZ. « Manlab : logiciel de continuation interactif ». In : *7e Colloque national en calcul des structures*. 2005 (cf. p. 43).

[Aud+23] Alexandre AUDIBERT, Massih R AMINI, Konstantin USEVICH et al. « Low-rank updates of pre-trained weights for multi-task learning ». In : *Findings of the Association for Computational Linguistics : ACL 2023*. 2023, p. 7544-7554 (cf. p. 17).

[AS20] Kyriakos AXIOTIS et Maxim SVIRIDENKO. « Sparse Convex Optimization via Adaptively Regularized Hard Thresholding ». In : *Proc. Int. Conf. Mach. Learn.* T. 119. Proceedings of Machine Learning Research. PMLR, juill. 2020, p. 452-462 (cf. p. 29, 66, 73, 105, 106, 109, 124).

[BT14] Bubacarr BAH et Jared TANNER. « Bounds of restricted isometry constants in extreme asymptotics : formulae for Gaussian matrices ». In : *Lin. Algebra Appl.* 441 (2014), p. 88-109 (cf. p. 74).

[BT09] Amir BECK et Marc TEBOULLE. « A fast iterative shrinkage-thresholding algorithm for linear inverse problems ». In : *SIAM journal on imaging sciences* 2.1 (2009), p. 183-202 (cf. p. 28).

[BBN22] Ramzi BEN MHENNI, Sébastien BOURGUIGNON et Jordan NININ. « Global optimization for sparse solution of least squares problems ». In : *Optim. Methods Softw.* 37.5 (2022), p. 1740-1769 (cf. p. 29).

[BLC13] Yoshua BENGIO, Nicholas LÉONARD et Aaron COURVILLE. « Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation ». In : *CoRR* arXiv :1308.3432 (2013). arXiv : 1308 . 3432 (cf. p. 19, 36, 62).

[BF19] Brett BERNSTEIN et Carlos FERNANDEZ-GRANDA. « Deconvolution of Point Sources : A Sampling Theorem and Robustness Guarantees ». In : *Comm. Pure Appl. Math.* 72.6 (2019), p. 1152-1230 (cf. p. 109).

[Ber19] Daniel BERRAR. « Cross-Validation ». In : *Encyclopedia of Bioinformatics and Computational Biology* 1.April (2019), p. 542-545 (cf. p. 47).

[Ber+24] Adrien BERTHELOT, Eddy CARON, Mathilde JAY et al. « Estimating the environmental impact of Generative-AI services using an LCA-based methodology ». In : *Procedia CIRP* 122 (2024), p. 707-712 (cf. p. 17).

[Ber24] Dimitri BERTSEKAS. « New auction algorithms for the assignment problem and extensions ». In : *Results in control and optimization* 14 (2024), p. 100383 (cf. p. 133, 137).

[Ber88] Dimitri P BERTSEKAS. « The auction algorithm : A distributed relaxation method for the assignment problem ». In : *Ann. Oper. Res.* 14.1 (1988), p. 105-123 (cf. p. 133, 135).

[BKM16] Dimitris BERTSIMAS, Angela KING et Rahul MAZUMDER. « Best Subset Selection via a Modern Optimization Lens ». In : *The Annals of Statistics* 44.2 (2016), p. 813-852 (cf. p. 26).

[BT15] Jeffrey D BLANCHARD et Jared TANNER. « Performance comparisons of greedy algorithms in compressed sensing ». In : *Numerical Linear Algebra with Applications* 22.2 (2015), p. 254-282 (cf. p. 74, 111).

[BR24] Mathieu BLONDEL et Vincent ROULET. « The elements of differentiable programming ». In : *arXiv preprint arXiv :2403.14606* (2024) (cf. p. 145).

[BD10] Thomas BLUMENSATH et Mike E DAVIES. « Normalized iterative hard thresholding : Guaranteed stability and performance ». In : *IEEE J. Sel. Topics Signal Process.* 4.2 (2010), p. 298-309 (cf. p. 29, 120).

[BD09] Thomas BLUMENSATH et Mike E. DAVIES. « Iterative hard thresholding for compressed sensing ». In : *Appl. Comput. Harmon. Analysis* 27.3 (2009), p. 265-274. ISSN : 1063-5203 (cf. p. 29, 65, 68, 69, 73, 109).

[BST14] Jérôme BOLTE, Shoham SABACH et Marc TEBOULLE. « Proximal alternating linearized minimization for nonconvex and nonsmooth problems ». In : *Mathematical Programming* 146.1 (2014), p. 459-494 (cf. p. 28, 29, 136).

[BDM12] Rainer BURKARD, Mauro DELL'AMICO et Silvano MARTELLO. *Assignment problems : revised reprint*. SIAM, 2012 (cf. p. 133).

[CW11] T Tony CAI et Lie WANG. « Orthogonal matching pursuit for sparse signal recovery with noise ». In : *IEEE Trans. Inform. Theory* 57.7 (2011), p. 4680-4688 (cf. p. 78, 105).

[CGM21] Murray CAMPBELL, Joël GILBERT et Arnold MYERS. *The science of brass instruments*. T. 436. Springer, 2021 (cf. p. 40).

[CT05] Emmanuel J CANDES et Terence TAO. « Decoding by linear programming ». In : *IEEE transactions on information theory* 51.12 (2005), p. 4203-4215 (cf. p. 30, 96, 98).

[CRT06] Emmanuel J CANDÈS, Justin ROMBERG et Terence TAO. « Robust uncertainty principles : Exact signal reconstruction from highly incomplete frequency information ». In : *IEEE Trans. Inform. Theory* 52.2 (2006), p. 489-509 (cf. p. 18, 26, 78).

[CK13] Antoine CHAIGNE et Jean KERGOMARD. *Acoustique des instruments de musique (2e édition revue et augmentée)*. 2013 (cf. p. 40).

[CK16] Antoine CHAIGNE et Jean KERGOMARD. *Acoustics of musical instruments*. Springer, 2016 (cf. p. 40).

[CDS01] Scott Shaobing CHEN, David L DONOHO et Michael A SAUNDERS. « Atomic decomposition by basis pursuit ». In : *SIAM Rev.* 43.1 (2001), p. 129-159 (cf. p. 28).

[CG16] Tianqi CHEN et Carlos GUESTRIN. « Xgboost : A scalable tree boosting system ». In : *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, p. 785-794 (cf. p. 46).

[Che+18] Pengyu CHENG, Chang LIU, Chunyuan LI et al. « Straight-Through Estimator as Projected Wasserstein Gradient Flow ». In : *Third workshop on Bayesian Deep Learning (NeurIPS)*. 2018 (cf. p. 78).

[CBD15] Matthieu COURBARIAUX, Yoshua BENGIO et Jean-Pierre DAVID. « Binaryconnect : Training deep neural networks with binary weights during propagations ». In : *Advances in neural information processing systems* 28 (2015) (cf. p. 17, 34, 62, 65).

[CGC00] JS CULLEN, Joël GILBERT et DM CAMPBELL. « Brass instruments : linear stability analysis and experiments with an artificial mouth ». In : *Acta Acustica united with Acustica* 86.4 (2000), p. 704-724 (cf. p. 40).

[DM09] Wei DAI et Olgica MILENKOVIC. « Subspace pursuit for compressive sensing signal reconstruction ». In : *IEEE Trans. Inform. Theory* 55.5 (2009), p. 2230-2249 (cf. p. 29, 98).

[Dao+22] Tri DAO, Beidi CHEN, Nimit S SOHONI et al. « Monarch : Expressive structured matrices for efficient and accurate training ». In : *International Conference on Machine Learning*. PMLR. 2022, p. 4690-4721 (cf. p. 18, 29, 131, 132, 134, 138).

[Dao+19] Tri DAO, Albert GU, Matthew EICHHORN et al. « Learning Fast Algorithms for Linear Transforms Using Butterfly Factorizations ». In : *Proc. Int. Conf. Mach. Learn.* Juin 2019 (cf. p. 18, 131).

[Dao+20] Tri DAO, Nimit SOHONI, Albert GU et al. « Kaleidoscope : An Efficient, Learnable Representation For All Structured Linear Maps ». In : *Eighth International Conference on Learning Representations*. 2020 (cf. p. 18, 131).

[DMA97] Geoff DAVIS, Stephane MALLAT et Marco AVELLANEDA. « Adaptive greedy approximations ». In : *Constr. Approx.* 13.1 (1997), p. 57-98 (cf. p. 18, 27, 61).

[DH01] David DONOHO et X. HUO. « Uncertainty principles and ideal atomic decomposition ». In : *IEEE Trans. Inform. Theory* 47.7 (2001), p. 2845-2862 (cf. p. 30, 143).

[DS06] David DONOHO et Victoria STODDEN. « Breakdown point of model selection when the number of variables exceeds the number of observations ». In : *The 2006 IEEE international joint conference on neural network proceedings*. IEEE. 2006, p. 1916-1921 (cf. p. 33).

[DT09] David DONOHO et Jared TANNER. « Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing ». In : *Phil. Trans. R. Soc. A* 367.1906 (2009), p. 4273-4293 (cf. p. 73).

[Don06] David L DONOHO. « Compressed sensing ». In : *IEEE Trans. Inform. Theory* 52.4 (2006), p. 1289-1306 (cf. p. 18, 26, 78).

[DMM09] David L DONOHO, Arian MALEKI et Andrea MONTANARI. « Message-passing algorithms for compressed sensing ». In : *Proceedings of the National Academy of Sciences* 106.45 (2009), p. 18914-18919 (cf. p. 73).

[DP15] Vincent DUVAL et Gabriel PEYRÉ. « Exact Support Recovery for Sparse Spikes Deconvolution ». In : *Found. Comput. Math.* 15.5 (2015), p. 1315-1355 (cf. p. 109).

[DP17] Vincent DUVAL et Gabriel PEYRÉ. « Sparse spikes super-resolution on thin grids I : the LASSO ». In : *Inverse Problems* 33.5 (2017), p. 055008 (cf. p. 109).

[Efr+04] Bradley EFRON, Trevor HASTIE, Iain JOHNSTONE et al. « Least angle regression ». In : *The Annals of Statistics* 32.2 (2004), p. 407-499 (cf. p. 28, 52, 55).

[Ela10] Michael ELAD. *Sparse and redundant representations : from theory to applications in signal and image processing*. Springer Science & Business Media, 2010 (cf. p. 30, 74, 110, 116).

[ER18] Patrick EMAMI et Sanjay RANKA. « Learning permutations with sinkhorn policy gradient ». In : *arXiv preprint arXiv :1805.07010* (2018) (cf. p. 132).

[FR12] Neville H FLETCHER et Thomas D ROSSING. *The physics of musical instruments*. Springer Science & Business Media, 2012 (cf. p. 40).

[Fou11] Simon FOUCART. « Hard thresholding pursuit : an algorithm for compressive sensing ». In : *SIAM J. Numer. Anal.* 49.6 (2011), p. 2543-2563 (cf. p. 29, 65, 68, 73, 105, 109, 120).

[Fou+13] Simon FOUCART, Holger RAUHUT, Simon FOUCART et al. *An invitation to compressive sensing*. Springer, 2013 (cf. p. 73).

[Fré+20] Vincent FRÉOUR, Louis GUILLOT, Hideyuki MASUDA, Eiji TOMINAGA et al. « Numerical continuation of a physical model of brass instruments : Application to trumpet comparisons ». In : *The Journal of the Acoustical Society of America* 148.2 (2020), p. 748-758 (cf. p. 40, 43, 46, 50, 53).

[Fré+22] Vincent FRÉOUR, Louis GUILLOT, Hideyuki MASUDA, Christophe VERGEZ et al. « Parameter identification of a physical model of brass instruments by constrained continuation ». In : *Acta Acustica* 6 (2022), p. 9 (cf. p. 43).

[Fré+23a] Vincent FRÉOUR, Hideyuki MASUDA, Bruno COCHELIN et al. « Identification of lip-parameters associated to different trumpets using constrained continuation ». In : *Forum Acusticum 2023*. 2023 (cf. p. 40).

[Fré+23b] Vincent FRÉOUR, Mimoun MOHAMED, Keita ARIMOTO et al. « Machine learning applied to the prediction of trumpet bifurcation diagrams : towards a tool for trumpet designers ». In : *Forum Acusticum 2023-10th Convention of the European Acoustics Association*. 2023 (cf. p. 5, 7, 19, 39).

[Fu+24] Dan FU, Simran ARORA, Jessica GROGAN et al. « Monarch mixer : A simple sub-quadratic gemm-based architecture ». In : *Adv. Neural Inf. Process. Syst.* 36 (2024) (cf. p. 131, 132).

[GRC09] Gilles GASSO, Alain RAKOTOMAMONJY et Stéphane CANU. « Recovering sparse signals with a certain family of nonconvex penalties and DC programming ». In : *IEEE Transactions on Signal Processing* 57.12 (2009), p. 4686-4698 (cf. p. 29).

[GSB17] Simon GAZAGNES, Emmanuel SOUBIES et Laure BLANC-FÉRAUD. « High density molecule localization for super-resolution microscopy using CEL0 based sparse approximation ». In : *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*. IEEE. 2017, p. 28-31 (cf. p. 23).

[Gif+21] Luc GIFFON, Stéphane AYACHE, Hachem KADRI et al. « PSM-nets : Compressing neural networks with product of sparse matrices ». In : *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2021, p. 1-8 (cf. p. 131).

[GK95] Andrew V GOLDBERG et Robert KENNEDY. « An efficient cost scaling algorithm for the assignment problem ». In : *Mathematical Programming* 71.2 (1995), p. 153-177 (cf. p. 133).

[Gon+21] Sebastian GONZALEZ, Davide SALVI, Daniel BAEZA et al. « A data-driven approach to violin making ». In : *Scientific reports* 11.1 (2021), p. 9455 (cf. p. 41).

[GTT18] Jun-ya GOTOH, Akiko TAKEDA et Katsuya TONO. « DC formulations and algorithms for sparse optimization problems ». In : *Mathematical Programming* 169 (2018), p. 141-176 (cf. p. 29).

[HMD15] Song HAN, Huizi MAO et William J DALLY. « Deep compression : Compressing deep neural networks with pruning, trained quantization and huffman coding ». In : *arXiv preprint arXiv :1510.00149* (2015) (cf. p. 18).

[HK24] Tomohiro HAYASE et Ryo KARAKIDA. « Understanding MLP-Mixer as a wide and sparse MLP ». In : *Proceedings of the 41st International Conference on Machine Learning*. 2024 (cf. p. 131).

[Hel+19] Koen HELWEGEN, James WIDDICOMBE, Lukas GEIGER et al. « Latent weights do not exist : Rethinking binarized neural network optimization ». In : *Advances in neural information processing systems* 32 (2019) (cf. p. 78).

[Hin15] Geoffrey HINTON. « Distilling the Knowledge in a Neural Network ». In : *arXiv preprint arXiv :1503.02531* (2015) (cf. p. 17).

[Hin12] Geoffrey E. HINTON. *Neural networks for machine learning*. Coursera, video lectures. Lecture 15b. 2012 (cf. p. 19, 36, 62).

[HK70] Arthur E HOERL et Robert W KENNARD. « Ridge regression : Biased estimation for nonorthogonal problems ». In : *Technometrics* 12.1 (1970), p. 55-67 (cf. p. 52).

[Hot33] Harold HOTELLING. « Analysis of a complex of statistical variables into principal components. » In : *Journal of educational psychology* 24.6 (1933), p. 417 (cf. p. 47).

[Hu+21] Edward J HU, Yelong SHEN, Phillip WALLIS et al. « Lora : Low-rank adaptation of large language models ». In : *arXiv preprint arXiv :2106.09685* (2021) (cf. p. 17).

[Hub+16] Itay HUBARA, Matthieu COURBARIAUX, Daniel SOUDRY et al. « Binarized neural networks ». In : *Adv. Neural Inf. Process. Syst.* 29 (déc. 2016) (cf. p. 62).

[JTD11] Prateek JAIN, Ambuj TEWARI et Inderjit DHILLON. « Orthogonal matching pursuit with replacement ». In : *Adv. Neural Inf. Process. Syst.* 24 (déc. 2011) (cf. p. 29, 73, 109).

[Kab+23] Tokihiko KABURAGI, Chiho KUROKI, Shunsuke HIDAKA et al. « Numerical method for analyzing steady-state oscillation in trumpets ». In : *Acoustical Science and Technology* 44.3 (2023), p. 269-280 (cf. p. 40).

[KT19] Konstantinos KOUNAKIS et Nektarios TAVERNARAKIS. « The cytoskeleton as a modulator of aging and neurodegeneration ». In : *Reviews on Biomarker Studies in Aging and Anti-Aging Research* (2019), p. 227-245 (cf. p. 23).

[Kuh55] Harold W KUHN. « The Hungarian method for the assignment problem ». In : *Naval research logistics quarterly* 2.1-2 (1955), p. 83-97 (cf. p. 133).

[Kuo+19] Han-Wen KUO, Yenson LAU, Yuqian ZHANG et al. « Geometry and Symmetry in Short-and-Sparse Deconvolution ». In : *Proc. Int. Conf. Mach. Learn.* T. 97. Proceedings of Machine Learning Research. PMLR, juin 2019, p. 3570-3580 (cf. p. 109).

[LG21] Quoc-Tung LE et Rémi GRIBONVAL. « Structured Support Exploration For Multilayer Sparse Matrix Factorization ». In : *Proc. Int. Conf. Acoust. Speech Signal Process.* Juin 2021 (cf. p. 131).

[LRG23] Quoc-Tung LE, Elisa RICCIETTI et Rémi GRIBONVAL. « Spurious valleys, np-hardness, and tractability of sparse matrix factorization with fixed support ». In : *SIAM Journal on Matrix Analysis and Applications* 44.2 (2023), p. 503-529 (cf. p. 28, 33).

[LG16] Luc LE MAGOAROU et Rémi GRIBONVAL. « Flexible Multilayer Sparse Approximations of Matrices and Applications ». In : *IEEE J. Sel. Topics Signal Process.* 10.4 (2016), p. 688-700 (cf. p. 28, 131, 136).

[LeC+98] Yann LECUN, Léon BOTTOU, Yoshua BENGIO et al. « Gradient-based learning applied to document recognition ». In : *Proc. IEEE* 86.11 (1998), p. 2278-2324 (cf. p. 138).

[LK24] Changwoo LEE et Hun-Seok KIM. « Differentiable Learning of Generalized Structured Matrices for Efficient Deep Neural Networks ». In : *The Twelfth International Conference on Learning Representations*. 2024 (cf. p. 131).

[Li+17] Hao LI, Soham DE, Zheng XU et al. « Training quantized nets : A deeper understanding ». In : *Advances in Neural Information Processing Systems*. 2017, p. 5811-5821 (cf. p. 78).

[Li+15] Yingzhou LI, Haizhao YANG, Eileen R MARTIN et al. « Butterfly factorization ». In : *Multiscale Modeling & Simulation* 13.2 (2015), p. 714-732 (cf. p. 18, 29, 131).

[Li+23] Zongyi LI, Daniel Zhengyu HUANG, Burigede LIU et al. « Fourier Neural Operator with Learned Deformations for PDEs on General Geometries ». In : *Journal of Machine Learning Research* 24.388 (2023), p. 1-26 (cf. p. 59).

[Loe06] Dominik B LOEFFLER. « Instrument timbres and pitch estimation in polyphonic music ». Mém. de mast. Citeseer, 2006 (cf. p. 53).

[LYX21] Ziang LONG, Penghang YIN et Jack XIN. « Learning quantized neural nets by coarse gradient method for nonlinear classification ». In : *Research in the Mathematical Sciences* 8 (2021), p. 1-19 (cf. p. 78).

[Los17] I LOSHCHILOV. « Decoupled weight decay regularization ». In : *arXiv preprint arXiv :1711.05101* (2017) (cf. p. 138).

[LJK19] Lu LU, Pengzhan JIN et George Em KARNIADAKIS. « DeepONet : Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators ». In : *CoRR* abs/1910.03193 (2019) (cf. p. 59).

[LJS24] Sasha LUCCIONI, Yacine JERNITE et Emma STRUBELL. « Power hungry processing : Watts driving the cost of AI deployment? » In : *The 2024 ACM Conference on Fairness, Accountability, and Transparency*. 2024, p. 85-99 (cf. p. 17).

[Lyu+20] Jiancheng LYU, Shuai ZHANG, Yingyong QI et al. « Autoshufflenet : Learning permutation matrices via an exact lipschitz continuous penalty in deep convolutional neural networks ». In : *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, p. 608-616 (cf. p. 132).

[Mal+20] Eran MALACH, Gilad YEHUDAI, Shai SHALEV-SCHWARTZ et al. « Proving the lottery ticket hypothesis : Pruning is all you need ». In : *International Conference on Machine Learning*. PMLR. 2020, p. 6682-6691 (cf. p. 17).

[MZ93] Stéphane G MALLAT et Zhifeng ZHANG. « Matching pursuits with time-frequency dictionaries ». In : *IEEE Trans. Signal Process.* 41.12 (1993), p. 3397-3415 (cf. p. 29, 30, 73, 109).

[Mar+18] Elaine Crespo MARQUES, Nilson MACIEL, Lirida NAVINER et al. « A review of sparse recovery algorithms ». In : *IEEE access* 7 (2018), p. 1300-1322 (cf. p. 29).

[MR17] Rahul MAZUMDER et Peter RADCHENKO. « The discrete Dantzig selector : Estimating sparse linear models via mixed integer linear optimization ». In : *IEEE Transactions on Information Theory* 63.5 (2017), p. 3053-3075 (cf. p. 26).

[MB06] Nicolai MEINSHAUSEN et Peter BÜHLMANN. « High-dimensional graphs and variable selection with the Lasso ». In : *Ann. Statist.* 34.3 (2006), p. 1436-1462 (cf. p. 78, 105).

[Men+18] Gonzalo MENA, David BELANGER, Scott LINDERMAN et al. « Learning latent permutations with gumbel-sinkhorn networks ». In : *arXiv preprint arXiv :1802.08665* (2018) (cf. p. 132).

[MEC25] Mimoun MOHAMED, Valentin EMIYA et Caroline CHAUX. « Learning Permutations in Monarch Factorization ». In : *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2025 (cf. p. 5, 7, 19, 130).

[Moh+24a] Mimoun MOHAMED, Vincent FRÉOUR, Christophe VERGEZ et al. « Prediction of trumpet performance descriptors using machine learning ». In : *Acta Acustica* 8 (2024), p. 65 (cf. p. 5, 7, 19, 39).

[Moh+24b] Mimoun MOHAMED, Francois MALGOUYRES, Valentin EMIYA et al. « Straight-Through Meets Sparse Recovery : the Support Exploration Algorithm ». In : *Proceedings of the 41st International Conference on Machine Learning*. Sous la dir. de Ruslan SALAKHUTDINOV, Zico KOLTER, Katherine HELLER et al. T. 235. Proceedings of Machine Learning Research. PMLR, juill. 2024, p. 35968-36011 (cf. p. 5, 7, 19, 60, 77, 108).

[MM13] Fatma MOKDAD et Samy MISSOUM. « A fully parameterized finite element model of a grand piano soundboard for sensitivity analysis of the dynamic behavior ». In : *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. T. 55997. American Society of Mechanical Engineers. 2013, V008T13A072 (cf. p. 41).

[NT09] Deanna NEEDELL et Joel A TROPP. « CoSaMP : Iterative signal recovery from incomplete and inaccurate samples ». In : *Appl. Comput. Harmon. Analysis* 26.3 (2009), p. 301-321 (cf. p. 29).

[Neh+18] Elias NEHME, Lucien E WEISS, Tomer MICHAELI et al. « Deep-STORM : super-resolution single-molecule microscopy by deep learning ». In : *Optica* 5.4 (2018), p. 458-464 (cf. p. 23).

[OA92] James B ORLIN et Ravindra K AHUJA. « New scaling algorithms for the assignment and minimum mean cycle problems ». In : *Math. Programm.* 54.1 (1992), p. 41-56 (cf. p. 136).

[PRK93] Yagyensh Chandra PATI, Ramin REZAIIFAR et Perinkulam Sambamurthy KRISHNAPRASAD. « Orthogonal matching pursuit : Recursive function approximation with applications to wavelet decomposition ». In : *Proc. Asilomar Conf. Signal Syst. Comput.* T. 1. IEEE. Pacific Grove, CA, USA, 1993, p. 40-44 (cf. p. 29, 66, 73, 109).

[Ped+11] Fabian PEDREGOSA, Gaël VAROQUAUX, Alexandre GRAMFORT et al. « Scikit-learn : Machine Learning in Python ». In : *J. Mach. Learn. Res.* 12 (2011), p. 2825-2830 (cf. p. 58).

[Pet+23] Jean-François PETIOT, Misael ROATTA, Vincent FRÉOUR et al. « Contribution of machine learning and physics-based sound simulations for the characterization of brass instruments ». In : *Forum Acusticum 2023*. 2023 (cf. p. 40, 41).

[Qiu+24] Shikai QIU, Andres POTAPCZYNSKI, Marc Anton FINZI et al. « Compute Better Spent : Replacing Dense Layers with Structured Matrices ». In : *Proceedings of the 41st International Conference on Machine Learning*. 2024 (cf. p. 18, 131).

[RPK19] M. RAISSI, P. PERDIKARIS et G.E. KARNIADAKIS. « Physics-informed neural networks : A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations ». In : *Journal of Computational Physics* 378 (2019) (cf. p. 59).

[RK89] Richard ROY et Thomas KAILATH. « ESPRIT-estimation of signal parameters via rotational invariance techniques ». In : *IEEE Transactions on acoustics, speech, and signal processing* 37.7 (1989), p. 984-995 (cf. p. 50).

[Say+23] Ratshih SAYED, Haytham AZMI, Heba SHAWKEY et al. « A systematic literature review on binary neural networks ». In : *IEEE Access* 11 (2023), p. 27546-27578 (cf. p. 34, 36).

[Sch+20] Klaske M SCHUKKEN, Yu-Chih LIN, Petra L BAKKER et al. « Altering microtubule dynamics is synergistically toxic with spindle assembly checkpoint inhibition ». In : *Life science alliance* 3.2 (2020) (cf. p. 23).

[SSZ10] Shai SHALEV-SHWARTZ, Nathan SREBRO et Tong ZHANG. « Trading accuracy for sparsity in optimization problems with sparsity constraints ». In : *SIAM J. Optim.* 20.6 (2010), p. 2807-2832 (cf. p. 66).

[She+94] Jonathan Richard SHEWCHUK et al. « An introduction to the conjugate gradient method without the agonizing pain ». In : (1994) (cf. p. 71).

[Tan+24] Wenxuan TAN, Nicholas ROBERTS, Tzu-Heng HUANG et al. « MoRe Fine-Tuning with 10x Fewer Parameters ». In : *ICML 2024 Workshop on Foundation Models in the Wild*. 2024 (cf. p. 131).

[Tib96] Robert TIBSHIRANI. « Regression shrinkage and selection via the lasso ». In : *Journal of the Royal Statistical Society Series B : Statistical Methodology* 58.1 (1996), p. 267-288 (cf. p. 28, 55).

[Til+24] Andreas M TILLMANN, Daniel BIENSTOCK, Andrea LODI et al. « Cardinality minimization, constraints, and regularization : a survey ». In : *SIAM Review* 66.3 (2024), p. 403-477 (cf. p. 26, 29).

[TP14] Andreas M. TILLMANN et Marc E. PFETSCH. « The Computational Complexity of the Restricted Isometry Property, the Nullspace Property, and Related Concepts in Compressed Sensing ». In : *IEEE Trans. Inform. Theory* 60.2 (2014), p. 1248-1259 (cf. p. 143).

[Tol+21] Ilya O TOLSTIKHIN, Neil HOULSBY, Alexander KOLESNIKOV et al. « Mlp-mixer : An all-mlp architecture for vision ». In : *Advances in neural information processing systems* 34 (2021), p. 24261-24272 (cf. p. 131).

[Tou+17] Robin TOURNEMENNE, Jean-François PETIOT, Bastien TALGORN et al. « Brass instruments design using physics-based sound simulation models and surrogate-assisted derivative-free optimization ». In : *Journal of Mechanical Design* 139.4 (2017), p. 041401 (cf. p. 41).

[TK23] Asher TROCKMAN et J Zico KOLTER. « Patches Are All You Need? » In : *Transactions on Machine Learning Research* (2023). Featured Certification. ISSN : 2835-8856 (cf. p. 131).

[Tro04] Joel A TROPP. « Greed is good : Algorithmic results for sparse approximation ». In : *IEEE Transactions on Information theory* 50.10 (2004), p. 2231-2242 (cf. p. 30).

[Vap13] Vladimir VAPNIK. *The nature of statistical learning theory*. Springer science & business media, 2013 (cf. p. 53).

[VLW24] Gaël VAROQUAUX, Alexandra Sasha LUCCIONI et Meredith WHITTAKER. « Hype, Sustainability, and the Price of the Bigger-is-Better Paradigm in AI ». In : *arXiv preprint arXiv :2409.14160* (2024) (cf. p. 17).

[Vas17] A VASWANI. « Attention is all you need ». In : *Advances in Neural Information Processing Systems* (2017) (cf. p. 131).

[Vel+17] Lionel VELUT, Christophe VERGEZ, Joël GILBERT et al. « How well can linear stability analysis predict the behaviour of an outward-striking valve brass instrument model? » In : *Acta Acustica united with Acustica* 103.1 (2017), p. 132-148 (cf. p. 40).

[Vir+20] Pauli VIRTANEN, Ralf GOMMERS, Travis E. OLIPHANT et al. « SciPy 1.0 : Fundamental Algorithms for Scientific Computing in Python ». In : *Nature Methods* 17 (2020), p. 261-272 (cf. p. 73, 109).

[Wai09] Martin J WAINWRIGHT. « Sharp thresholds for High-Dimensional and noisy sparsity recovery using ℓ_1-Constrained Quadratic Programming (Lasso) ». In : *IEEE Trans. Inform. Theory* 55.5 (2009), p. 2183-2202 (cf. p. 78, 105).

[Yin+19] Penghang YIN, Jiancheng LYU, Shuai ZHANG et al. « Understanding Straight-Through Estimator in Training Activation Quantized Neural Nets ». In : *International Conference on Learning Representations*. 2019 (cf. p. 78).

[YA23] Chunyu YUAN et Sos S AGAIAN. « A comprehensive review of binary neural network ». In : *Artificial Intelligence Review* 56.11 (2023), p. 12949-13013 (cf. p. 26, 36).

[YLZ16] Xiaotong YUAN, Ping LI et Tong ZHANG. « Exact Recovery of Hard Thresholding Pursuit ». In : *Adv. Neural Inf. Process. Syst.* T. 29. Barcelona, Spain, déc. 2016, p. 3558-3566 (cf. p. 78, 105).

[Zei12] MD ZEILER. « ADADELTA : An Adaptive Learning Rate Method ». In : *arXiv preprint arXiv :1212.5701* (2012) (cf. p. 138).

[ZY06] Peng ZHAO et Bin YU. « On model selection consistency of Lasso ». In : *J. Mach. Learn. Res.* 7 (2006), p. 2541-2563 (cf. p. 78, 105).