Méthodes de programmation en nombres mixtes pour l'optimisation parcimonieuse en traitement du signal

Ramzi Ben Mhenni

THÈSE DE DOCTORAT DE

ÉCOLE CENTRALE DE NANTES

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Traitement du signal et des images*

Par
Ramzi BEN MHENNI

Méthodes de programmation en nombres mixtes pour l'optimisation parcimonieuse en traitement du signal

Thèse présentée et soutenue à Nantes, le 13 mai 2020
Unité de recherche : Laboratoire des Sciences du Numérique de Nantes (LS2N)
Thèse N° :

Rapporteurs avant soutenance :

Sonia Cafieri	Professeur, École Nationale de l'Aviation Civile
Matthieu Kowalski	Maître de conférences HDR, Université Paris-Saclay

Composition du Jury :

Président :	Christian Jutten	Professeur émérite, Université Grenoble Alpes
Examinateurs :	Paul Honeine	Professeur, Université de Rouen
	Liva Ralaivola	Directeur de recherche, Criteo AI Lab
Dir. de thèse :	Sébastien Bourguignon	Maître de conférences HDR, École Centrale de Nantes
Co-encadrant :	Jordan Ninin	Enseignant-chercheur, ENSTA Bretagne

TABLE DES MATIÈRES

TABLE DES FIGURES

Liste des tableaux

INTRODUCTION GÉNÉRALE

Contexte et problématique

Au cours des vingt dernières années, l'approximation parcimonieuse a fait l'objet d'un grand intérêt de la part de la communauté de recherche en traitement du signal. Sous un modèle linéaire $\boldsymbol{y} = \mathbf{H}\boldsymbol{x} + \text{bruit}$, le problème d'approximation parcimonieuse consiste à rechercher un vecteur solution $\boldsymbol{x}$ parcimonieux qui comporte un faible nombre de composantes non nulles approchant un vecteur de données $\boldsymbol{y}$. La mesure intuitive pour quantifier la parcimonie est la « norme » ℓ_0, également appelée fonction de comptage, qui compte le nombre de composantes non nulles. Ainsi, la résolution de ce problème peut être décrite par les trois formulations :

— la minimisation de l'erreur d'approximation au sens des moindres carrés sous contrainte d'une borne sur le nombre de coefficients non nuls de $\boldsymbol{x}$:

$$\min_{\boldsymbol{x}} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \text{ sous contrainte (s.c.) } \|\boldsymbol{x}\|_0 \leq K,$$

— la minimisation du nombre de coefficients non nuls de $\boldsymbol{x}$ sous contrainte d'une borne sur l'erreur quadratique d'approximation :

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \text{ s.c. } \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon,$$

— la minimisation de l'erreur quadratique d'approximation pénalisée par le nombre de coefficients non nuls de $\boldsymbol{x}$:

$$\min_{\boldsymbol{x}} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_0.$$

Selon le problème abordé, les trois formulations peuvent être intéressantes à traiter, en fonction des connaissances *a priori* permettant de fixer le paramètre associé. En revanche, la norme ℓ_0 n'étant pas convexe, celles-ci ne sont pas équivalentes. Le problème d'approximation parcimonieuse est généralement considéré comme un problème combinatoire *NP-difficile* [Natarajan, 1995], dans un espace dont la taille croît exponentiellement

avec le nombres d'inconnues et le degré de parcimonie. L'optimisation est généralement approchée par des méthodes rapides, reposant soit sur la relaxation continue de la norme ℓ_0 (notamment la norme ℓ_1), soit sur des stratégies heuristiques gloutonnes. Des conditions suffisantes garantissant l'optimalité de ces approches vis-à-vis du problème initial ont été établies [Tropp and Wright, 2009]. Cependant, elles reposent sur des hypothèses imposant une faible corrélation entre les colonnes du dictionnaire $\mathbf{H}$, qui malheureusement ne s'appliquent pas dans le cadre de nombreux problèmes inverses parcimonieux. Dans le cas de problèmes inverses de taille modérée, des travaux récents [Bourguignon et al., 2016] ont montré que ces problèmes peuvent être reformulés en programme en nombres mixtes (MIP, *Mixed Integer Programs*) et que la solution globale du problème d'optimisation peut être calculée à partir d'algorithmes de type branch-and-bound, issus du domaine de la recherche opérationnelle. Les optima ainsi obtenus fournissent alors de meilleures solutions que les méthodes sous-optimales existantes, mais pour un coût de calcul bien plus élevé.

Les algorithmes branch-and-bound reposent sur la définition d'un arbre de décision pour chaque variable (nulle ou non nulle ?). Une exploration totale de l'arbre est alors mise en œuvre, reposant sur la construction de relaxations efficaces permettant d'élaguer des branches importantes sans recourir à une énumération explicite de l'arbre. Dans [Bourguignon et al., 2016], la résolution de ces programmes en nombres mixtes s'effectue au moyen de solveurs génériques construits pour résoudre des classes très variées de problèmes et donc globalement aveugles vis-à-vis des spécificités du problème initialement formulé. À l'inverse, l'approximation parcimonieuse est un problème très particulier pouvant bénéficier de stratégies dédiées.

Cette thèse propose d'étudier plusieurs leviers permettant de dépasser les limites de tels solveurs afin d'aborder des problèmes plus gros et plus complexes, en exploitant différents niveaux de connaissance du problème du point de vue de l'approximation parcimonieuse et le savoir-faire algorithmique dans la construction des relaxations et dans les stratégies d'exploration de l'arbre, qui sont les deux éléments clés de l'efficacité d'un algorithme branch-and-bound. Les méthodes développées seront évaluées sur deux types de problèmes simulés : 1) la déconvolution de trains d'impulsions, sur des problèmes de petite taille mais difficiles à traiter à cause de la corrélation élevée entre les colonnes du dictionnaire et 2) des problèmes simulés de sélection de variables avec des dictionnaires aléatoires, qui sont de plus grande taille mais plus faciles à résoudre du fait que les colonnes sont moins corrélées. Nous abordons également, de manière séparée, le problème du démélange spectral parcimonieux en imagerie hyperspectrale, où nous proposons des reformulations

MIP de plusieurs types de contraintes habituellement abordées de manière inexacte dans la littérature (parcimonie structurée et contraintes logiques) afin de gérer la variabilité des spectres.

Contributions

Les contributions concernant l'optimisation parcimonieuse exacte sont présentées dans les chapitres suivants et peuvent être classées en deux axes.

1. **Aspects algorithmiques :**

   Nous avons montré que la relaxation continue permettant d'évaluer chaque nœud de l'arbre de recherche est un problème d'optimisation faisant intervenir la norme ℓ_1 sous contraintes de borne. Nous avons généralisé un algorithme de type homotopie pour le résoudre, ce qui nous a permis de construire un algorithme branch-and-bound rapide permettant d'aborder la résolution des trois formulations susmentionnées d'une manière similaire. Les résultats ont montré que l'évaluation de chaque nœud s'avère plus efficace que l'optimisation quadratique réalisée par le solveur CPLEX.
   Publications associées : [1,10,6].

   Ensuite, nous avons proposé un algorithme de type active-set pour résoudre la relaxation continue. Cet algorithme permet le démarrage à chaud, ce qui est particulièrement opportun dans notre contexte puisqu'il permet d'initialiser le problème de relaxation continue à partir de la solution au nœud parent, généralement proche.
   Publications associées : [5,7].

   Nous avons également développé une règle de sélection de variables pour l'approximation parcimonieuse, basée sur l'optimisation d'un critère en norme ℓ_1, qui se révèle plus efficace que les méthodes classiques mises en œuvre dans les solveurs MIP. Celle-ci a également été à la base de de la construction d'un nouvel algorithme glouton pour l'optimisation parcimonieuse.
   Publication associée : [8].

2. **Modélisation et reformulation en MIP :**

   Nous avons abordé le problème de démélange spectral, qui est un problème inverse classique de séparation de sources en imagerie hyperspectrale où l'on cherche à

décomposer un spectre mesuré en un mélange linéaire de spectres élémentaires purs et à estimer les abondances associées. Dans un premier temps, nous avons proposé une approche d'optimisation globale sous contrainte de parcimonie (norme ℓ_0). Nous avons montré que la résolution exacte est faisable, pour des problèmes de complexité limitée mais réaliste et qu'elle améliore les performances de détection par rapport aux approches classiques.
Publication associée : [2].

Ensuite, nous avons montré qu'on sait prendre en compte d'une manière exacte des contraintes inhabituelles telles que les contraintes des abondances significatives (imposer un seuil minimal aux composantes non nulles) et de parcimonie structurée (exclusivité de groupe), et que la résolution reste faisable pour des problèmes de taille modérée.
Publications associées : [4,3,12].

Liste des publications

Article de journal (en révision)

1. Ramzi Ben Mhenni, Sébastien Bourguignon et Jordan Ninin : « Global Optimization for Sparse Least Squares Minimization Problems », *Journal OMS, Optimization Methods and Software*.

Actes de conférences avec comité de lecture

2. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin et Frédéric Schmidt : « Démélange parcimonieux exact dans une approche supervisée en imagerie hyperspectrale », *Actes du 26ème édition du colloque de GRETSI, Groupement de Recherche en Traitement du Signal et de l'Image*, 2017, Juan-Les-Pins.
3. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin et Frédéric Schmidt : «Spectral unmixing with exact L0-norm sparsity and structural constraints», *18th edition of WHISPERS, IEEE Workshop on Hyperspectral Image and Signal Processing : Evolution in Remote Sensing*, 2018, Amsterdam.
4. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin et Frédéric Schmidt : « Méthodes exactes de démélange spectral en norme L0 et contraintes de parcimo-

nie structurée à l'aide de MIP », *19ème édition du congrès ROADEF, Recherche Opérationnelle et de l'Aide à la Décision en France*, 2018, Lorient.

5. Ramzi Ben Mhenni, Sébastien Bourguignon, Marcel Mongeau, Jordan Ninin et Hervé Carfantan : « Algorithme branch-and-bound pour l'optimisation exacte en norme L0», *Actes du 27ème édition du colloque de GRETSI, Groupement de Recherche en Traitement du Signal et de l'Image*, août 2019, Lille.

6. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin : « Algorithme branch-and-bound pour l'approximation parcimonieuse en traitement du signal et en statistiques », *20ème édition du congrès ROADEF, Recherche Opérationnelle et de l'Aide à la Décision en France*, avril 2020, Montpellier.

7. Ramzi Ben Mhenni, Sébastien Bourguignon, Marcel Mongeau, Jordan Ninin et Hervé Carfantan : « Sparse Branch & Bound For Exact Optimization Of L0-norm penalized Least Squares », *ICASSP, International Conference on Acoustics, Speech, and Signal Processing*, mai 2020, Barcelone.

8. Ramzi Ben Mhenni, Sébastien Bourguignon, Jérôme Idier : « SLS : A Greedy Sparse Approximation Algorithm Based on L1-norm Selection Rules », *ICASSP, International Conference on Acoustics, Speech, and Signal Processing*, mai 2020, Barcelone.

Présentations dans des conférences internationales

9. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin, Marcel Mongeau, et Hervé Carfantan « Optimisation globale multi-objectif pour des problèmes de moindres carrés à faible cardinalité » *JOPT, Journées de l'optimisation*, mai 2019, Montréal.

10. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin : « Global optimization of L0-norm-based sparse approximation criteria with a branch-and-bound algorithm», *International Conference SPARS, Signal Processing with Adaptive Sparse Structured Representations*, juillet 2019, Toulouse.

11. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin «Global Optimization for Sparse Solution of Least Squares Problems», *International conference ICCOPT, International Conference on Continuous Optimization*, août 2019, Berlin.

Présentations dans des conférences nationales

12. Ramzi Ben Mhenni, Sébastien Bourguignon, Jordan Ninin et Frédéric Schmidt : « Démélange parcimonieux et prise en compte de contraintes structurantes par optimisation globale » *Colloque scientifique du Groupe Hyperspectral de la SFPT, Société Française de Photogrammétrie et Télédétection*, juillet 2019, Toulouse.

Organisation du document

Le premier chapitre est une analyse de l'état de l'art des méthodes d'approximation parcimonieuse où nous définissons les différentes formalisations du problème reposant sur la norme ℓ_0, introduisons les différentes notations utilisées par la suite, et présentons les différentes classes de méthodes.

Afin de construire un algorithme branch-and-bound (évaluation et séparation) spécifique au problème de l'approximation parcimonieuse et de développer des stratégies appropriées, nous divisons notre démarche en deux parties. Une première partie, abordée dans les chapitres 2 et 3, concerne l'étape d'évaluation reposant essentiellement sur la relaxation continue.

— Le chapitre 2 contient une description détaillée de la méthode branch-and-bound, à partir de laquelle nous étudions de manière théorique les spécificités engendrées par la norme ℓ_0 et leurs impacts sur les différentes parties de l'algorithme. En particulier, nous montrons que tous les problèmes de relaxation continue intervenant dans l'évaluation peuvent être abordés par l'optimisation de problèmes faisant intervenir la norme ℓ_1, cependant sous une forme non standard : la norme ℓ_1 n'impacte qu'une partie des variables et le problème inclut des contraintes de bornes.

— Le chapitre 3 propose des algorithmes spécifiques dans ce contexte.

Ensuite, une deuxième partie, abordée dans le chapitre 4, concerne l'étape de la séparation, en particulier les règles de branchement dans l'algorithme branch-and-bound. Ce chapitre propose des règles de sélection de variables inspirées des algorithmes gloutons conçus pour le problème d'approximation parcimonieuse.

Dans le chapitre 5, nous évaluons les performances de notre algorithme branch-and-bound en utilisant les différentes parties développées dans les chapitres précédents. Nous le com-

parons à CPLEX, un des meilleurs solveurs MIP actuels, sur les deux types de problèmes parcimonieux : les problèmes de déconvolution impulsionnelle et les problèmes de sélection de variables aléatoires.

Le chapitre 6 est consacré à l'étude du problème de démélange en imagerie hyperspectrale où nous proposons plusieurs reformulations en programmes en nombres mixtes (MIP) de contraintes habituellement prises en compte de manière inexacte pour cette application : des contraintes de parcimonie simple par la norme ℓ_0 et des contraintes de parcimonie structurée (contraintes logiques) telles que les contraintes de variabilité.

Enfin, le chapitre 7 propose une conclusion générale et plusieurs perspectives et pistes de recherche autour de cette thèse.

Chapitre 1

Approximation parcimonieuse en traitement de signal

Contents

1.1 Introduction

Un grand nombre de phénomènes physiques sont difficiles à observer et nécessitent la résolution d'un problème inverse pour retrouver les informations d'intérêt. Autrement dit, on observe des données ayant subi un processus de mesure, qui en général fournit des informations indirectes sur le phénomène observé. Il s'agit alors, étant donné un modèle sur le processus d'observation, d'inverser ce dernier pour remonter au phénomène ayant généré les données. De tels problèmes sont souvent mal posés [Idier, 2008] et les données ne contiennent pas suffisamment d'information permettant d'estimer de manière satisfaisante les quantités d'intérêt. Pour y remédier, on peut notamment les régulariser par la mise en valeur de toute information traduisant une connaissance *a priori* sur l'information recherchée. Une démarche largement utilisée en traitement du signal depuis une vingtaine d'années consiste à imposer la parcimonie (*sparsity* en anglais) de la solution [Elad, 2010], ce qui signifie au sens le plus strict du terme que seuls quelques coefficients dans l'ensemble des valeurs recherchées sont non nuls. La fonction de comptage, appelée aussi « norme » ℓ_0, est la mesure intuitive pour quantifier cette parcimonie. Elle consiste à compter le nombre d'éléments non nuls d'un vecteur, c'est-à-dire, pour tout $\boldsymbol{x} \in \mathbb{R}^n$,

$$\|\boldsymbol{x}\|_0 := \mathrm{Card}(\{q \ |x_q \neq 0\}).$$

Il est à noter que la fonction $\|\boldsymbol{x}\|_0$ ne constitue pas une norme puisque la propriété d'homogénéité n'est pas vérifiée : $\forall \alpha \in \mathbb{R}$, $\alpha \neq 0$, $|\alpha| \neq 1$, on a $\|\alpha\boldsymbol{x}\|_0 \neq |\alpha|\|\boldsymbol{x}\|_0$. Cependant, nous conserverons dans la suite cet abus de langage, afin d'être cohérent avec la littérature.

La parcimonie est naturellement présente dans plusieurs applications en traitement du signal (par exemple pour la déconvolution impulsionnelle [Zala, 1992] et le démélange spectral [Singer and McCord, 1979]) ainsi qu'en économie (par exemple pour la sélection de portefeuilles [Li et al., 2006; Shaw et al., 2008; Bertsimas and Shioda, 2009]). Elle constitue un a priori donnant lieu à ce que l'on appelle l'approximation parcimonieuse, souvent étudié sous le nom de sélection de sous-ensembles [Tibshirani, 1996; Osborne et al., 2000; Miller, 2002] en statistique. La résolution de ce dernier peut être décrite par la minimisation de l'erreur d'approximation entre les données et un modèle ainsi que de la norme ℓ_0 de la solution, qui formule intrinsèquement des questions d'optimisation combinatoire de complexité élevée [Natarajan, 1995]. Elle est généralement approchée par des méthodes rapides, reposant soit sur la relaxation continue de la norme ℓ_0 (notamment

la relaxation convexe par la norme ℓ_1), soit sur des stratégies heuristiques gloutonnes [voir par exemple Tropp and Wright, 2010]. Cependant, aucune d'elles ne garantit l'optimalité vis-à-vis du problème en norme ℓ_0 dans le cadre des problèmes inverses mal posés.

Afin de mieux comprendre l'importance de la parcimonie et la difficulté pour la prendre en compte d'une façon exacte, nous commençons, en section 1.2, par introduire quelques exemples d'applications en traitement du signal. Nous y présentons les différentes formulations utilisées pour le problème d'approximation parcimonieuse. Dans la section 1.3, nous présentons les méthodes de résolution approchées les plus connues dans la littérature, notamment les méthodes heuristiques gloutonnes et une relaxation continue de la norme ℓ_0. Dans la section 1.4, nous terminons ce chapitre en faisant le point sur des reformulations du problème sous la forme d'un programme en nombres mixtes [Bourguignon et al., 2016], ainsi que sur les méthodes de résolution associées [Bienstock, 1996; Bertsimas and Shioda, 2009], issues du domaine de la recherche opérationnelle, permettant une optimisation exacte du problème en norme ℓ_0.

1.2 Problèmes inverses parcimonieux

Le problème d'approximation parcimonieuse a pour but d'estimer, à partir de données bruitées collectées dans un vecteur $\boldsymbol{y} \in \mathbb{R}^m$ et sous un modèle linéaire généralement sous-déterminé (comportant moins d'équations que d'inconnues) :

$$\boldsymbol{y} = \mathbf{H}\boldsymbol{x} + \boldsymbol{\varepsilon} = \sum_{q=1}^{n} \boldsymbol{h}_q x_q + \boldsymbol{\varepsilon}, \tag{1.1}$$

un vecteur $\boldsymbol{x} \in \mathbb{R}^n$ parcimonieux (*i.e.,* dont seuls quelques coefficients ont des valeurs non nulles). La matrice $\mathbf{H} \in \mathbb{R}^{m \times n}$ (avec généralement $m < n$) représente un dictionnaire de m atomes $\boldsymbol{h}_q \in \mathbb{R}^m$ (la q-ème colonne de la matrice $\mathbf{H}$) et $\boldsymbol{\varepsilon} \in \mathbb{R}^m$ est un bruit additif. Dans cette thèse, nous nous intéressons particulièrement aux problèmes inverses en traitement du signal, dont nous présentons ci-après quelques exemples d'applications qui seront abordés dans les chapitres suivants. Notons que des problèmes similaires sont formulés dans le domaine de la statistique [Tibshirani, 1996; Osborne et al., 2000; Efron et al., 2004; Miller, 2002] et en économie [Li et al., 2006; Shaw et al., 2008; Bertsimas and Shioda, 2009; Cui et al., 2013].

1.2.1 Exemples d'applications en traitement de signal

Il existe plusieurs exemples d'applications de l'approximation parcimonieuse en traitement du signal. Nous présentons ici deux problèmes "historiques". La première application est la déconvolution de trains d'impulsions [Zala, 1992] rencontrée en géophysique [Taylor et al., 1979; Mendel, 1986], en contrôle non destructif (CND) par ultrasons [Zala, 1992; O'Brien et al., 1994] et en astronomie [Starck et al., 2002]. La deuxième application est le démélange spectral parcimonieux [Singer and McCord, 1979; Iordache et al., 2011; Drumetz et al., 2019] rencontré dans plusieurs domaines comme l'imagerie hyperspectrale et la spectroscopie.

1.2.1.1 Déconvolution de trains d'impulsions parcimonieuse

La déconvolution de trains d'impulsions est un problème inverse parcimonieux classique en traitement du signal. Le contrôle non destructif par ultrasons (CND) [Zala, 1992; O'Brien et al., 1994] est un exemple d'application utilisant la déconvolution parcimonieuse, où l'on cherche, après émission d'une onde ultrasonore par un instrument de mesure à la surface d'une pièce, à obtenir des informations sur le milieu de propagation en se basant sur l'onde réfléchie (en temps continu) $y(t)$, dans le but de détecter et de localiser des défauts. Ce signal peut alors se modéliser comme le produit de convolution entre une onde (la réponse impulsionnelle $h(t)$) et la séquence de réflectivité du milieu (un train d'impulsions $x(t)$), auquel s'ajoute un terme de bruit $\varepsilon(t)$ (voir Figure 1.1).

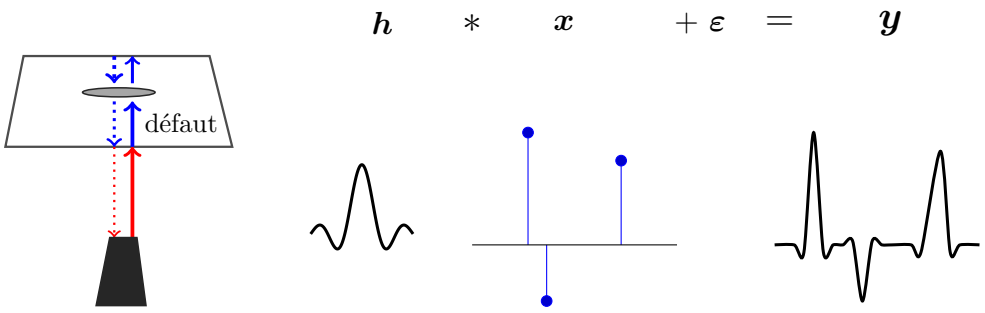


FIGURE 1.1 – La déconvolution impulsionnelle dans le contexte du contrôle non destructif (CND) par ultrasons.

La nature parcimonieuse de $x(t)$ traduit alors le fait que les défauts sont peu nombreux dans un matériau globalement homogène. Le modèle convolutif s'écrit donc de la façon suivante :

$$y(t) = (h * x)(t) + \varepsilon(t). \tag{1.2}$$

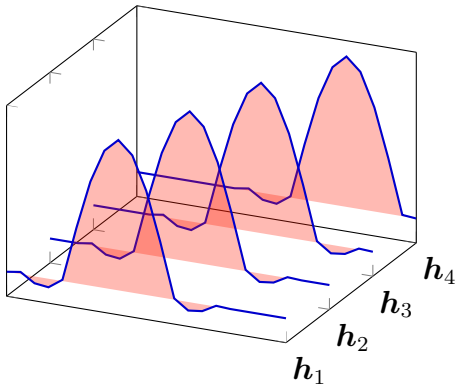


FIGURE 1.2 – Quelques colonnes successives de la matrice $\mathbf{H}$ pour un problème de déconvolution.

Après échantillonnage et discrétisation du modèle convolutif, le modèle discret de données s'écrit :

$$y_i = \sum_{q=1}^{n} x_q h_{i-q} + \boldsymbol{\varepsilon}_i. \tag{1.3}$$

Le problème sous la forme matricielle s'écrit $\boldsymbol{y} = \mathbf{H}\boldsymbol{x} + \boldsymbol{\varepsilon}$ où $\mathbf{H}$ est une matrice de Toeplitz dont les colonnes sont des versions décalées de la réponse impulsionnelle $\boldsymbol{h}$ (voir Figure 1.2).

Clairement, les colonnes voisines sont très corrélées entre elles. Au niveau fréquentiel, le système convolutif se comporte comme un filtre qui rend les données insensibles à une partie des fréquences contenues dans la séquence $\boldsymbol{x}$ (notamment les hautes fréquences), ce qui se traduit numériquement par le mauvais conditionnement de $\mathbf{H}$. Ainsi, les solutions basées sur le seul ajustement du modèle (1.3) donnent un résultat dominé par le bruit, montrant le caractère mal posé du problème [Idier, 2008]. Cependant, comme nous l'avons mentionné plus haut, les défauts recherchés dans la pièce inspectée ne sont pas nombreux, et il semble naturel de régulariser le problème en recherchant des solutions parcimonieuses.

1.2.1.2 Démélange spectral parcimonieux

Une image hyperspectrale est une série d'images d'une scène prises dans un grand nombre de longueurs d'onde. Chacun des pixels constituant cette image est représenté par un vecteur de mesures des *réflectances*, qui désignent la proportion de lumière réfléchie, correspondant au spectre observé. Un problème inverse classique en imagerie hyperspectrale est le démélange spectral [Singer and McCord, 1979], où l'on cherche à décomposer chaque spectre observé $\boldsymbol{y} \in \mathbb{R}^m$ acquis dans m bandes spectrales en un mélange linéaire de

spectres élémentaires (composants purs ou *endmembers*) et à estimer les proportions (abondances) de chacun de ces composants. Dans une approche *supervisée*, on ne s'intéresse qu'à l'estimation des abondances, les spectres purs étant supposés connus, soit par des mesures spectroscopiques en laboratoire des différents minéraux susceptibles d'être présents, soit *via* leur estimation à partir des mêmes données hyperspectrales, par exemple par des approches géométriques ou statistiques [Bioucas-Dias et al., 2012]. Si on considère que le modèle de mélange est linéaire [Singer and McCord, 1979], alors, on peut écrire :

$$\boldsymbol{y} = \sum_{q=1}^{n} a_q \boldsymbol{s}_q + \boldsymbol{\varepsilon} = \mathbf{S}\boldsymbol{a} + \boldsymbol{\varepsilon}, \tag{1.4}$$

où $\boldsymbol{s}_q \in \mathbb{R}^n$ représente le $i^{\text{ème}}$ spectre pur, $\mathbf{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n]$ est le dictionnaire de spectres purs, $a_q \in \mathbb{R}^+$ est l'abondance associée au $q^{\text{ème}}$ composant et $\boldsymbol{\varepsilon}$ représente le bruit et l'erreur de modélisation.

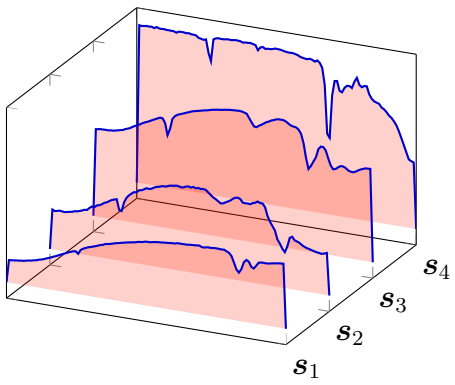


FIGURE 1.3 – Quelques colonnes typiques de la matrice $\mathbf{S}$ pour le problème de démélange spectral. Les spectres sont issus de la base de données du *United States Geological Survey* [Clark et al., 2003a].

Puisque les abondances représentent des pourcentages, des contraintes de positivité et de somme à un sont ajoutées aux inconnues. Le conditionnement de la matrice $\mathbf{S}$ est également très mauvais dans ce cas, car $\mathbf{S}$ contient un nombre élevé de spectres qui sont très corrélés (voir Figure 1.3). Même si les contraintes de positivité créent une certaine parcimonie dans la solution, elles peuvent s'avérer insuffisantes pour ce problème et il peut sembler naturel de rechercher des solutions plus parcimonieuses [Iordache et al., 2011].

1.2.2 Parcimonie et formulations du problème

Le problème d'approximation parcimonieuse peut être formulé mathématiquement comme un problème d'optimisation bi-objectif décrit par la minimisation d'un terme de moindres carrés $\frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 = \frac{1}{2}\sum_{q=1}^{n}(y_q - \boldsymbol{h}_q^T\boldsymbol{x})^2$ pour l'attache aux données, et de la norme ℓ_0 de $\boldsymbol{x}$:

$$\mathcal{P} : \qquad \min_{\boldsymbol{x}\in\mathbb{R}^n}\{\tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2, \|\boldsymbol{x}\|_0\}. \tag{1.5}$$

Le choix de la norme ℓ_2 sur l'erreur d'approximation est justifié dans un cadre statistique par une hypothèse d'un bruit $\boldsymbol{\varepsilon}$ gaussien de moyenne nulle et de covariance proportionnelle à l'identité [Idier, 2008]. Contrairement à l'optimisation mono-objectif, une solution à ce problème bi-objectif est plus un concept qu'une définition. En général, il n'existe pas de solution globale unique car les deux critères sont contradictoires, au sens où une solution considérée bonne pour l'un va être mauvaise pour l'autre. Pour pallier ce problème, on peut s'intéresser à la construction du front de Pareto [Pareto, 1906], qui consiste à trouver l'ensemble des solutions optimales pour lesquelles il n'existe aucune autre solution permettant une réduction simultanée des deux objectifs [Marler and Arora, 2004].

Une autre façon de procéder est de se ramener à une formulation mono-objectif en faisant un choix *a priori*. Plusieurs formulations mono-objectif du problème d'approximation parcimonieuse ont été abordées dans la littérature. La première est la forme contrainte par la norme ℓ_0, appelée aussi contrainte de cardinalité :

$$\mathcal{P}_{2/0} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \text{ sous contrainte (s.c.) } \|\boldsymbol{x}\|_0 \leq K, \tag{1.6}$$

avec K fixé *a priori*. Ce problème a fait l'objet de beaucoup de travaux dans de nombreux domaines d'application, dont l'optimisation de portefeuille [Li et al., 2006; Shaw et al., 2008; Bertsimas and Shioda, 2009; Cui et al., 2013] et la sélection de sous-ensembles en statistique [Tibshirani, 1996; Osborne et al., 2000; Efron et al., 2004; Miller, 2002] où l'on peut disposer d'informations permettant de régler le nombre de composantes non nulles. Cependant, dans certaines applications, on peut préférer contrôler l'erreur d'approximation et résoudre le problème sous contrainte d'erreur [Natarajan, 1995; Miller, 2002; Tropp and Wright, 2010] :

$$\mathcal{P}_{0/2} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}\|_0 \text{ s.c. } \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon. \tag{1.7}$$

Une telle formulation peut être plus judicieuse en traitement du signal et en statistique, où $\boldsymbol{y}$ représente les données observées qui suivent approximativement un modèle linéaire.

Dans ce cas, le paramètre $\epsilon \geq 0$ contrôle le niveau d'approximation. On cherche alors l'approximation la plus parcimonieuse compatible avec un niveau de bruit donné (ou une certaine précision du modèle de prédiction). Finalement, certains travaux (appliqués principalement aux problèmes inverses) tels que rencontrés en géophysique [Mendel, 1986] ou en contrôle non destructif par ultrasons [Zala, 1992] abordent plutôt le problème pénalisé :

$$\mathcal{P}_{2+0} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_0, \tag{1.8}$$

où $\mu > 0$ règle le compromis entre l'erreur d'approximation et la parcimonie. Dans le cadre statistique bayésien, le terme de pénalisation en ℓ_0 correspond à une hypothèse préalable d'un modèle *a priori* Bernoulli-gaussien sur chacune des variables x_q et le paramètre μ dépend alors à la fois du niveau du bruit et du degré de parcimonie de $\boldsymbol{x}$ prévu [Soussen et al., 2011]. Cette forme est très répandue dans la communauté du traitement du signal, car elle relève de l'optimisation sans contrainte, pour laquelle des méthodes dédiées d'optimisation locale ont été développées (voir par exemple [Kormylo and Mendel, 1982; Zala, 1992; Soussen et al., 2011]).

Il est à noter que, si les trois problèmes $\mathcal{P}_{2/0}$, $\mathcal{P}_{0/2}$ et $\mathcal{P}_{2+0}$ sont intéressants, ils ne sont pas équivalents, dans le sens où la solution de l'un n'est pas forcément obtenue en résolvant un des deux autres, en raison de leur non-convexité. Ils sont considérés comme des problèmes *NP-difficiles* [Natarajan, 1995; Bienstock, 1996]. En *théorie de la complexité*, un problème NP-difficile (c'est-à-dire un problème difficile pour la classe *NP*) est un problème de décision pour lequel on ne peut ni trouver rapidement une solution ni vérifier l'optimalité d'une solution proposée en temps polynomial. Les trois problèmes $\mathcal{P}_{2/0}$, $\mathcal{P}_{0/2}$ et $\mathcal{P}_{2+0}$ sont des problèmes d'optimisation combinatoire, dans un espace dont la taille croît exponentiellement avec le nombre d'inconnues et le degré de parcimonie. Trouver la meilleure solution à K composantes non nulles avec une approche de force brute, qui revient à explorer toutes les combinaisons possibles $\binom{K}{n}$, devient très vite inenvisageable dans des cas pratiques. Pour résoudre de tels problèmes, il existe deux catégories de méthodes : les méthodes approchées et les méthodes exactes. Les méthodes approchées ont pour but de trouver une solution approchée rapidement, mais pas nécessairement optimale. En revanche, les méthodes exactes nous permettent de trouver la solution optimale, mais elles sont souvent beaucoup plus coûteuses quand il s'agit de problèmes de grande taille.

Dans la suite de ce chapitre, nous passons en revue quelques méthodes proposées dans la littérature pour l'approximation parcimonieuse. Étant donné la grande quantité de

travaux réalisés sur le sujet au cours des dernières années, l'état de l'art est loin d'être exhaustif et nous renvoyons le lecteur aux références des études mentionnées.

1.3 Stratégies d'optimisation inexactes

Différentes stratégies d'optimisation ont été développées pour la résolution approchée des problèmes parcimonieux $\mathcal{P}_{2/0}$, $\mathcal{P}_{0/2}$ et $\mathcal{P}_{2+0}$. Celles-ci reposent soit sur une relaxation de la norme ℓ_0 par une autre fonction continue favorisant la parcimonie de la solution, soit sur la construction d'une méthode itérative dite gloutonne (*greedy*).

1.3.1 Relaxation convexe de la norme ℓ_0

Un vecteur parcimonieux au sens le plus strict du terme signifie que seuls quelques uns de ses coefficients ont des valeurs non nulles. Étant donnée cette définition, il est clair que la norme ℓ_0, qui peut être également vue comme la limite de la « norme » [1] ℓ_p quand p tend vers zéro :

$$\|\boldsymbol{x}\|_0 = \lim_{p\to 0} \|\boldsymbol{x}\|_p^p = \lim_{p\to 0} \sum_{q=1}^{n} |x_q|^p, \tag{1.9}$$

est la fonction la plus adaptée pour mesurer la parcimonie. Le caractère discret de la norme ℓ_0 rend souvent la résolution combinatoire et très difficile, ce qui a donné lieu à plusieurs mesures alternatives de parcimonie. Il a été montré que le remplacement de la norme ℓ_0 par une fonction de la forme $\sum_{q=1}^{n} \varphi(|x_q|)$ produisait des solutions parcimonieuses, *i.e.,* comprenant des valeurs nulles, si et seulement si φ est strictement croissante en 0 [Moulin and Liu, 1999]. C'est le cas de la norme ℓ_p avec $0 < p \leq 1$. Si le cas $p < 1$, générant des problèmes d'optimisation difficiles car non convexes, a été étudié dans plusieurs travaux [Lai and Wang, 2011; Xu et al., 2010], la norme ℓ_1 ($\|\boldsymbol{x}\|_1 := \sum_q |x_q|$) reste la mesure de parcimonie la plus fréquemment utilisée et la plus facile à prendre en compte en raison de sa nature convexe (voir Figure 1.4).

Le problème faisant intervenir une erreur quadratique et la norme ℓ_1, est connu en statistique sous le nom de LASSO pour *Least Absolute Shrinkage and Selection Operator* [Tibshirani, 1996]) :

$$\mathcal{P}_{2/1} : \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \ \text{ s.c. } \ \|\boldsymbol{x}\|_1 \leq \tau. \tag{1.10}$$

1. Le terme de « norme » ℓ_p pour $0 < p < 1$ est également un abus de langage : cette fonction, ne vérifiant pas l'inégalité triangulaire, est seulement une quasi-norme.

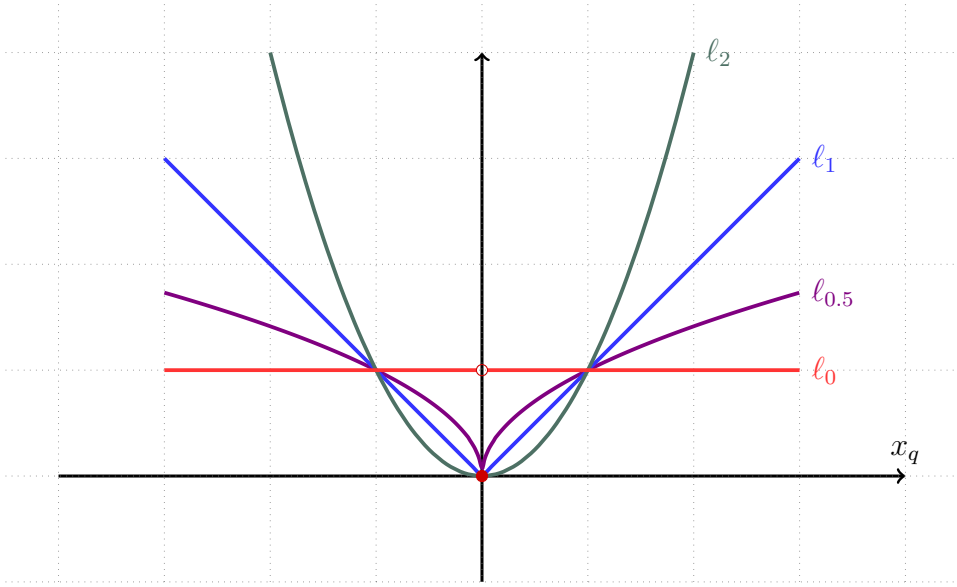


FIGURE 1.4 – Fonctions de la variable scalaire $\varphi(x_q)$ intervenant dans la définition de différentes normes ℓ_p, avec $\|\boldsymbol{x}\|_p^p = \sum_q \varphi(|x_q|)$, pour $p = 0$, 0.5, 1 et 2.

Le problème d'optimisation devenant alors convexe, les minimiseurs des problèmes $\mathcal{P}_{2/1}$,

$$\mathcal{P}_{1/2} : \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}\|_1 \quad \text{s.c.} \quad \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon; \tag{1.11}$$

et de la forme pénalisée

$$\mathcal{P}_{2+1} : \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_1 \tag{1.12}$$

se trouvent sur les surfaces des contraintes pour $\mathcal{P}_{2/1}$ et $\mathcal{P}_{1/2}$. Ainsi les trois problèmes sont équivalents, c'est-à-dire que pour tout $\tau \geq 0$, il existe $\mu^{(\tau)} \geq 0$ tel que $\mathcal{P}_{2/1}$ et $\mathcal{P}_{2+1}$ ont la même solution, et réciproquement. De même, pour tout $\epsilon \geq 0$, il existe $\mu^{(\epsilon)} \geq 0$ tel que les solutions de $\mathcal{P}_{1/2}$ et $\mathcal{P}_{2+1}$ soient identiques (voir Figure 1.5). Cependant, en général, il n'y a pas de correspondance explicite entre les trois paramètres.

L'optimisation faisant intervenir une erreur quadratique et la norme ℓ_1 a fait l'objet de nombreuses recherches au cours des dernières années. De nombreux algorithmes d'optimisation dédiés ont été développés (voir par exemple [Tropp and Wright, 2009; Eldar and Kutyniok, 2012] et leurs références). La plupart des travaux abordent la forme pénalisée, qui relève de l'optimisation sans contrainte. Nous décrivons dans cette section le principe de fonctionnement d'une méthode, que nous utiliserons plus tard, connue par sa capacité

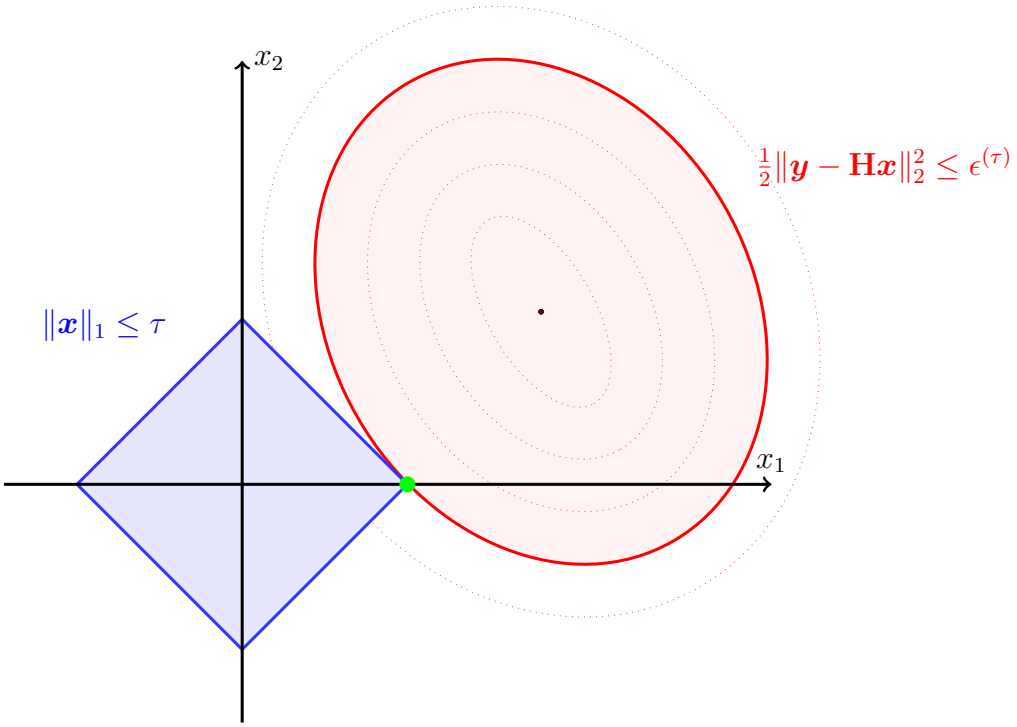
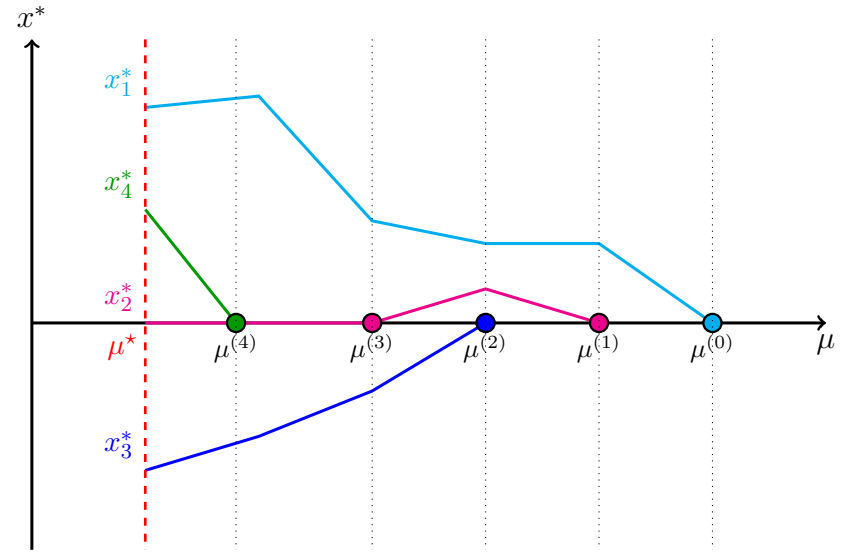


FIGURE 1.5 – Interaction de l'espace admissible des solutions (l'ensemble des points vérifiant les contraintes) des deux problèmes équivalents $\mathcal{P}_{2/1}$ (en bleu) et $\mathcal{P}_{1/2}$ (en rouge). Le point d'intersection (en vert) est la solution du problème.

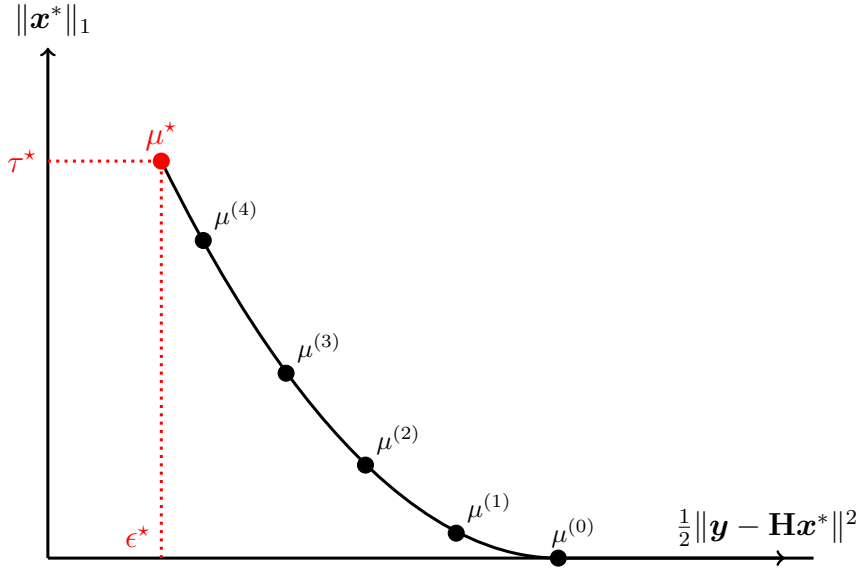
à résoudre les trois formulations $\mathcal{P}_{2/1}$, $\mathcal{P}_{2+1}$ et $\mathcal{P}_{1/2}$: c'est la méthode homotopique.

Méthode homotopique

La méthode homotopique pour l'optimisation en norme ℓ_1 [Tibshirani, 1996; Osborne et al., 2000; Efron et al., 2004; Donoho and Tsaig, 2008] considère la forme pénalisée $\mathcal{P}_{2+1}$, et exploite le fait que le chemin de solution est linéaire par morceaux en fonction de μ. À partir de $\mu^{(0)} = \|\mathbf{H}^T\boldsymbol{y}\|_\infty$ (tel que la solution est nulle $\forall \mu \geq \mu^{(0)}$), la méthode calcule itérativement toutes les solutions en diminuant de façon continue le paramètre μ jusqu'à atteindre la valeur cible. Par conséquent, elle peut également résoudre les problèmes $\mathcal{P}_{2/1}$ ou $\mathcal{P}_{1/2}$, en s'arrêtant lorsque la valeur correspondante $\tau^\star$ ou $\epsilon^\star$ est atteinte. La figure 1.6 montre un chemin de solution typique (partie en haut) et l'évolution correspondante sur le front de Pareto (partie en bas). Cette méthode est exacte (converge en un nombre fini d'itérations) et elle est connue pour être d'autant plus performante que la solution recherchée est parcimonieuse. Cela s'explique par le fait qu'en général lors de la construction du chemin



Chemin de solutions pour $\mathcal{P}_{2+1}$



Front de Pareto correspondant

FIGURE 1.6 – Méthode d'homotopie : exemple montrant le chemin de solution $\boldsymbol{x}^*(\mu) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_1$ en fonction de μ (haut), et l'ensemble correspondant $\left(\frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^*\|_2^2, \|\boldsymbol{x}^*\|_1\right)$ en fonction de μ (bas).

de solution, il y a beaucoup plus de variables qui deviennent non nulles (activation) que de variables qui deviennent nulles (inactivation), et le nombre d'itérations de la méthode homotopique est défini par le nombre d'activation et d'inactivation de variables.

1.3.2 Méthodes heuristiques gloutonnes

Les méthodes gloutonnes construisent une approximation parcimonieuse en sélectionnant des variables d'une manière itérative partant d'un ensemble vide. Ces algorithmes suivent une stratégie de sélection de variables dite « progressive » (*forward selection*), répétant deux opérations itérativement jusqu'à atteindre un critère d'arrêt selon la formulation :

— si le degré de parcimonie K est atteint pour $\mathcal{P}_{2/0}$,

— si l'erreur résiduelle est inférieure au seuil donné ϵ pour $\mathcal{P}_{0/2}$,

— si on ne peut plus améliorer la solution pour $\mathcal{P}_{2+0}$.

Ces algorithmes peuvent donc être utilisés indifféremment pour aborder les trois problèmes.

La première opération consiste à sélectionner une variable et la deuxième consiste à mettre à jour le modèle, sachant que, si une variable est sélectionnée, elle ne peut pas être enlevée. Ces algorithmes se prêtent directement aux trois problèmes, en particulier $\mathcal{P}_{2/0}$ et $\mathcal{P}_{0/2}$ qui sont plus faciles à réglé en pratique que $\mathcal{P}_{2+0}$. À l'inverse, il existe d'autres algorithmes plus performants, mettant en œuvre des stratégies plus sophistiquées avec possibilité de retrait, qui donnent de meilleures solutions avec un coût de calcul plus élevé, tel que Single Best Replacement (SBR) [Soussen et al., 2011], conçu pour le problème pénalisé $\mathcal{P}_{2+0}$.

Nous détaillons par la suite le principe de fonctionnement et la différence entre les trois algorithmes de type *forward selection* les plus connus : Matching Pursuit (MP) [Mallat and Zhang, 1993], Orthogonal Matching Pursuit (OMP) [Pati et al., 1993] et Orthogonal Least Squares (OLS) [Chen et al., 1989].

1.3.2.1 Matching Pursuit (MP) & Orthogonal Matching Pursuit (OMP)

Soit $\mathbb{S}_1$ l'indice des variables sélectionnées pour être non nulles (le support actif de la solution) à une itération quelconque, $\mathbf{H}_{\mathbb{S}_1}$ la matrice composée des colonnes de $\mathbf{H}$ indexées par $\mathbb{S}_1$ et $\boldsymbol{x}_{\mathbb{S}_1}$ le vecteur composé des éléments de $\boldsymbol{x}$ indexés par $\mathbb{S}_1$. Similairement, soit $\mathbb{S}$ l'indice des variables non sélectionnées et $\mathbf{H}_{\mathbb{S}}$ la matrice composée des colonnes de $\mathbf{H}$

indexées par $\mathbb{S}$. Sans perte de généralité, les atomes (c'est à dire les colonnes) $\boldsymbol{h}_q$ sont supposés normalisés $\|\boldsymbol{h}_q\| = 1$. Pour les deux algorithmes Matching Pursuit (MP) et Orthogonal Matching Pursuit (OMP), la sélection de l'élément à rajouter se fait de la manière suivante. À une itération donnée, l'amplitude $\hat{a}_j$ optimale qui minimise l'erreur entre le résidu courant $\boldsymbol{r} = \boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}$ et une colonne $\boldsymbol{h}_j$ est donnée par :

$$\hat{a}_j = \arg\min_{a\in\mathbb{R}} \|\boldsymbol{r} - a_j\boldsymbol{h}_j\|_2^2 = \boldsymbol{r}^T\boldsymbol{h}_j, \tag{1.13}$$

où la seconde égalité s'obtient après des manipulations triviales. La coordonnée $\hat{j}$ sélectionnée est alors celle qui minimise cette erreur, c'est-à-dire qui maximise en valeur absolue le produit scalaire entre une colonne de $\mathbf{H}_{\mathbb{S}}$ et le résidu $\boldsymbol{r}$:

$$\hat{j} = \arg\max_{j\in\mathbb{S}} |\boldsymbol{r}^T\boldsymbol{h}_j|. \tag{1.14}$$

La seule différence entre les deux algorithmes réside au niveau de la mise à jour de l'estimée. Tandis que MP à chaque itération modifie une seule composante (celle qui vient d'être sélectionnée) en ajoutant son amplitude optimale à sa valeur à l'itération précédente sans remettre en cause les amplitudes des autres composantes sélectionnées auparavant, OMP a la meilleure approximation de $\boldsymbol{y}$ dans l'espace engendré par $\mathbf{H}_{\mathbb{S}_1}$ au sens des moindres carrés. Il met à jour l'estimée $\hat{\boldsymbol{x}}_{\mathbb{S}_1}$, à chaque itération, en calculant la projection orthogonale de $\boldsymbol{y}$ sur $\mathbf{H}_{\mathbb{S}_1}$:

$$\hat{\boldsymbol{x}}_{\mathbb{S}_1} = (\mathbf{H}_{\mathbb{S}_1}^T\mathbf{H}_{\mathbb{S}_1})^{-1}\mathbf{H}_{\mathbb{S}_1}^T\boldsymbol{y}. \tag{1.15}$$

Ainsi, OMP produit une meilleure approximation (résidu plus faible) que MP, au prix d'un coût de calcul plus important dû à la projection. Le fonctionnement de OMP est détaillé dans l'algorithme 1.

Initialisation : $\mathbb{S}_1 = \emptyset \quad \boldsymbol{r} = \boldsymbol{y}$
tant que $|\mathbb{S}_1| < K$ (*ou critère d'arrêt pour* $\mathcal{P}_{2/0}$ *et* $\mathcal{P}_{2+0}$) **faire**
  $\hat{j} = \arg\max_{j\in\mathbb{S}} |\boldsymbol{r}^T\boldsymbol{h}_j|$; % Choix de variable %
  $\mathbb{S}_1 \leftarrow \mathbb{S}_1 \cup \{\hat{j}\}$; % Mise à jour du support %
  $\boldsymbol{x}_{\mathbb{S}_1} = (\mathbf{H}_{\mathbb{S}_1}^T\mathbf{H}_{\mathbb{S}_1})^{-1}\mathbf{H}_{\mathbb{S}_1}^T\boldsymbol{y}$; % Mise à jour de l'estimée %
  $\boldsymbol{r} = \boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}$; % Mise à jour du résidu %
fin
Résultat : solution $\boldsymbol{x}_{\mathbb{S}_1}$ et le support $\mathbb{S}_1$

Algorithme 1 : Orthogonal Matching Pursuit (OMP)

1.3.2.2 Orthogonal Least Squares (OLS)

Orthogonal Least Squares (OLS) [Chen et al., 1989] est un autre algorithme glouton. Il possède la même structure que OMP, constitué par les deux étapes citées précédemment : la sélection de la variable et la mise à jour de l'estimée. Le point qui change par rapport à OMP réside dans l'étape de sélection de variable. À chaque itération, OLS cherche à minimiser l'erreur d'approximation en résolvant Card($\mathbb{S}$) problèmes de moindres carrés, qui correspondent à inclure chaque possible nouvelle composante. L'élément sélectionné $\hat{j}$ est celui qui conduit à la plus faible erreur d'approximation pour tout $j \in \mathbb{S}$,

$$\hat{j} = \arg\min_{j \in \mathbb{S}} \tfrac{1}{2} \|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1 \cup \{j\}} \hat{\boldsymbol{x}}_{\mathbb{S}_1 \cup \{j\}}\|_2^2, \tag{1.16}$$

où $\hat{\boldsymbol{x}}_{\mathbb{S}_1 \cup \{j\}}$ est donné par la projection orthogonale de $\boldsymbol{y}$ sur $\mathbf{H}_{\mathbb{S}_1 \cup \{j\}}$:

$$\hat{\boldsymbol{x}}_{\mathbb{S}_1 \cup \{j\}} = (\mathbf{H}_{\mathbb{S}_1 \cup \{j\}}^T \mathbf{H}_{\mathbb{S}_1 \cup \{j\}})^{-1} \mathbf{H}_{\mathbb{S}_1 \cup \{j\}}^T \boldsymbol{y}. \tag{1.17}$$

L'algorithme Orthogonal Least Squares (OLS) est meilleur que l'algorithme Orthogonal Matching Pursuit (OMP) pour traiter des problèmes inverses parcimonieux, au prix d'un coût de calcul bien plus élevé dû à sa règle de sélection plus coûteuse (voir les comparaisons dans [Soussen et al., 2011]). Nous tenons à préciser aussi qu'on peut utiliser des implémentations astucieuses comme la factorisation de Cholesky ou le lemme d'inversion des matrices partitionnées pour un calcul récursif et plus rapide de la solution.

1.4 Optimisation globale

Les trois problèmes $\mathcal{P}_{2/0}$, $\mathcal{P}_{0/2}$ et $\mathcal{P}_{2+0}$ sont généralement très difficiles et à ce jour il n'existe pas d'algorithme, avec une complexité polynomiale, permettant de les résoudre efficacement sans restrictions particulières sur la matrice $\mathbf{H}$. Les auteurs dans [Bourguignon et al., 2016] ont proposé de reformuler ces problèmes d'une façon exacte comme des programmes en nombres mixtes (MIP, *Mixed Integer Programs*) qui proviennent du domaine de la recherche opérationnelle et de l'informatique théorique, dans lequel on considère des problèmes d'optimisation avec contraintes d'intégrité sur une partie des variables. L'optimisation en nombres mixtes est considérée comme *NP-difficile*. Sa résolution demande des techniques particulières tel qu'un algorithme de type branch-and-bound [Land and Doig, 1960; Laughhunn, 1970] dont nous allons décrire le principe de fonctionnement dans

cette section. Avant cela, nous discutons de l'hypothèse, connue sous le nom de *BigM*, utilisée par exemple dans [Bienstock, 1996; Bourguignon et al., 2016], pour reformuler les problèmes.

1.4.1 Reformulation en MIP

Pour reformuler les problèmes $\mathcal{P}_{2/0}$, $\mathcal{P}_{0/2}$ et $\mathcal{P}_{2+0}$ sous la forme de programmes en nombres mixtes (MIP), les auteurs dans [Bourguignon et al., 2016] ont introduit des variables de décision binaires b_q où chaque variable traduit une contrainte logique sur le nullité des variables $x_q = 0 \Leftrightarrow b_q = 0$. La norme ℓ_0 devient alors une fonction linéaire $\|\boldsymbol{x}\|_0 = \sum_{q=1}^{n} b_q$. En utilisant l'hypothèse classique de *BigM* [Bienstock, 1996], supposant que les solutions d'intérêt satisfont $\forall q, |x_q| \leq M$ pour une valeur connue M (autrement dit $\|\boldsymbol{x}\|_\infty \leq M$), la contrainte logique peut s'écrire comme $-Mb_q \leq x_q \leq Mb_q$ (voir Figure 1.7).

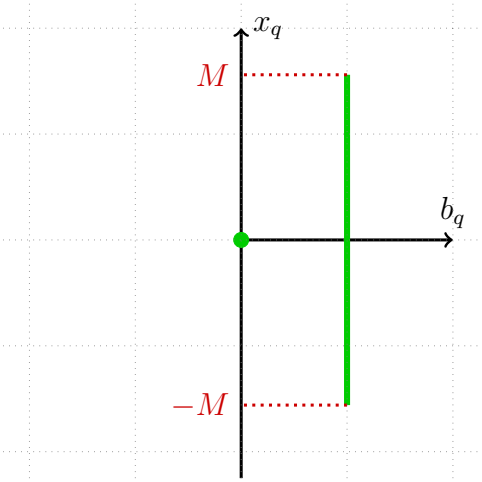


FIGURE 1.7 – Espace admissible de solutions (en vert) pour la contrainte $-Mb_q \leq x_q \leq Mb_q$.

Les extensions triviales d'une telle hypothèse sont : $-M_q^\ell \leq x_q \leq M_q^u$, avec $M_q^\ell, M_q^u > 0$, mais dans ce qui suit nous gardons $|\boldsymbol{x}_q| \leq M$ (ou $\|\boldsymbol{x}\|_\infty \leq M$) pour simplifier les écritures. Cette astuce permet alors d'écrire la norme ℓ_0 de façon linéaire comme suit :

$$\|\boldsymbol{x}\|_0 = \sum_{q=1}^{n} b_q \quad \text{avec } |x_q| \leq Mb_q.$$

Les trois problèmes sont ensuite reformulés sous la forme de programmes en nombres

mixtes (voir par exemple [Bourguignon et al., 2016]) donnés dans le tableau 1.1. Pour être plus précis, notons que les trois problèmes n'appartiennent pas à la même classe. Les problèmes $\hat{\mathcal{P}}_{2/0}$ et $\hat{\mathcal{P}}_{2+0}$ sont reformulés en programmes quadratiques en nombres mixtes (MIQP *Mixed Integer Quadratic Program*) où les contraintes sont linéaires et la fonction objectif est quadratique. À l'inverse, le problème $\hat{\mathcal{P}}_{0/2}$ est reformulé sous la forme d'un programme en nombres mixtes avec contraintes quadratiques (MIQCP *Mixed Integer Quadratically Constrained Program*) qui est généralement plus compliqué à résoudre en raison de la difficulté liée aux contraintes quadratiques du point de vue de l'optimisation. Tous ces programmes MIP peuvent être résolus en utilisant des solveurs d'optimisation en nombres mixtes tels que CPLEX [2] ou GUROBI [3].

Problèmes initiaux	Reformulations en MIP
$\hat{P}_{2/0}: \min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2$	$\min_{\boldsymbol{b}\in\{0,1\}^n,\ \boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2$
s.c. $\|\boldsymbol{x}\|_0 \leq K$ $\|\boldsymbol{x}\|_\infty \leq M$	s.c. $\sum_{q=1}^n b_q \leq K$ $\|\boldsymbol{x}\| \leq M\boldsymbol{b}$
$\hat{P}_{2/0}: \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}\|_0$	$\min_{\boldsymbol{b}\in\{0,1\}^n,\ \boldsymbol{x}\in\mathbb{R}^n} \sum_{q=1}^n b_q$
s.c. $\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon$ $\|\boldsymbol{x}\|_\infty \leq M$	s.c. $\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon$ $\|\boldsymbol{x}\| \leq M\boldsymbol{b}$
$\hat{P}_{2/0} \min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_0$	$\min_{\boldsymbol{b}\in\{0,1\}^n,\ \boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 + \mu\sum_{q=1}^n b_q$
s.c. $\|\boldsymbol{x}\|_\infty \leq M$	s.c. $\|\boldsymbol{x}\| \leq M\boldsymbol{b}$

TABLE 1.1 – Problèmes initiaux (à gauche) et leurs reformulations en MIP (à droite).

1.4.2 Discussion sur l'hypothèse de borne (*BigM*)

La contrainte de *BigM* est une contrainte artificielle qui permet de reformuler exactement la norme ℓ_0. Le réglage de cette borne, de façon générale, est reconnu comme un point critique [Bertsimas et al., 2016] et sa valeur peut fortement affecter l'efficacité de l'optimisation. En particulier, des valeurs grandes de M donnent des temps de résolution

2. www.ibm.com/fr-fr/analytics/cplex-optimizer
3. www.gurobi.com

beaucoup plus élevés (expliqué par le fait que la qualité des relaxations continues dans la procédure branch-and-bound dépend fortement de la valeur de M, ce que nous détaillerons au Chapitre 2).

Soit $\hat{\mathcal{P}}_{2/0}$ le problème borné et $\mathcal{P}_{2/0}$ le problème non borné. Le choix d'une valeur de M qui garantit d'avoir l'optimum global du problème non borné ($\min \hat{\mathcal{P}}_{2/0} = \min \mathcal{P}_{2/0}$) et qui n'est pas trop élevée, reste un problème ouvert. En particulier, une question importante est de savoir, pour une valeur de M donnée, si la solution optimale de $\hat{\mathcal{P}}_{2/0}$ est la solution optimale globale du problème initial $\mathcal{P}_{2/0}$.

La réponse est négative. Les auteurs dans [Bourguignon et al., 2016] ont utilisé une technique qui garantit d'avoir trouvé au moins un minimiseur local au problème initial $\mathcal{P}_{2/0}$. La valeur de la borne M a été réglée de façon heuristique. La technique utilisée consiste à augmenter la valeur de M tant qu'une composante de la solution du problème $\hat{\mathcal{P}}_{2/0}$ atteint la borne. Autrement dit, si la borne M est atteinte en la solution, on conclut que la formulation n'est pas valide et on relance l'optimisation en prenant M plus grand (voir Algorithme 2).

Initialisation : $M = 1.1\|\mathbf{H}^T\boldsymbol{y}\|_\infty$;
Résolution du problème
$\hat{\boldsymbol{x}} = \arg\min \hat{\mathcal{P}}_{2/0}$;
tant que $\|\hat{\boldsymbol{x}}\|_\infty = M$ **faire**
    $M = 1.1 \times M$;
    Résolution du problème : $\hat{\boldsymbol{x}} = \arg\min \hat{\mathcal{P}}_{2/0}$;
fin

Algorithme 2 : Algorithme utilisé par [Bourguignon et al., 2016] pour la résolution du problème $\mathcal{P}_{2/0}$.

Cette technique permet de vérifier que la solution de $\hat{\mathcal{P}}_{2/0}$ est bien un minimiseur local de $\mathcal{P}_{2/0}$. En revanche, elle n'apporte aucune garantie d'avoir le minimiseur global, le raisonnement derrière cette règle n'étant valide que pour les problèmes *convexes* : même si la solution du problème $\hat{\mathcal{P}}_{2/0}$ n'atteint pas la borne, cela n'implique pas qu'il s'agit de la solution optimale au problème $\mathcal{P}_{2/0}$. Pour mieux comprendre cela, la figure 1.8 montre un exemple où, selon la valeur de M choisie, le minimiseur de $\hat{\mathcal{P}}_{2/0}$ n'est pas le minimiseur global du problème initial $\mathcal{P}_{2/0}$, et pourtant la solution n'atteint pas la borne. Dans cet exemple, le problème initial est le suivant :

$$\mathcal{P}_{2/0} : \min_{\boldsymbol{x}\in\mathbb{R}^2} f(\boldsymbol{x}) \text{ s.c. } \|\boldsymbol{x}\|_0 \leq 1, \tag{1.18}$$

avec $f(x_1, x_2) = 2x_1^2 - 12x_1 + 17x_2^2 - 34x_2 + 18$. Comme on ne recherche qu'une seule variable non nulle, le problème peut s'écrire comme suit :

$$\mathcal{P}_{2/0} : \min_{\boldsymbol{x}\in\mathbb{R}^2} f(\boldsymbol{x}) \text{ avec } f(\boldsymbol{x}) = \begin{cases} f_1(x_1) \text{ si } \ x_2 = 0 \\ f_2(x_2) \text{ si } \ x_1 = 0 \end{cases} \tag{1.19}$$

avec $f_1(x_1) = 2x_1^2 - 12x_1 + 18$ et $f_2(x_2) = 17x_2^2 - 34x_2 + 18$. Maintenant, soit $\hat{\mathcal{P}}_{2/0}$ le problème borné par la contrainte *BigM*, avec $M = 2$:

$$\hat{\mathcal{P}}_{2/0} : \min_{\boldsymbol{x}\in\mathbb{R}^2} f(\boldsymbol{x}) \text{ s.c } \begin{cases} \|\boldsymbol{x}\|_0 \leq 1 \\ \|\boldsymbol{x}\|_\infty \leq 2 \end{cases} .$$

La figure 1.8 montre que le minimum global du problème $\mathcal{P}_{2/0}$ vaut 0 et est atteint en la solution ($x_1^\star = 3$, $x_2^\star = 0$). En revanche, pour une valeur de $M = 2$, le minimum du problème $\hat{\mathcal{P}}_{2/0}$ vaut 1 et est atteint en la solution ($x_1^\star = 0$, $x_2^\star = 1$), qui cependant n'atteint pas la borne. Ce dernier point n'est donc qu'un minimiseur local par rapport au problème initial $\mathcal{P}_{2/0}$.

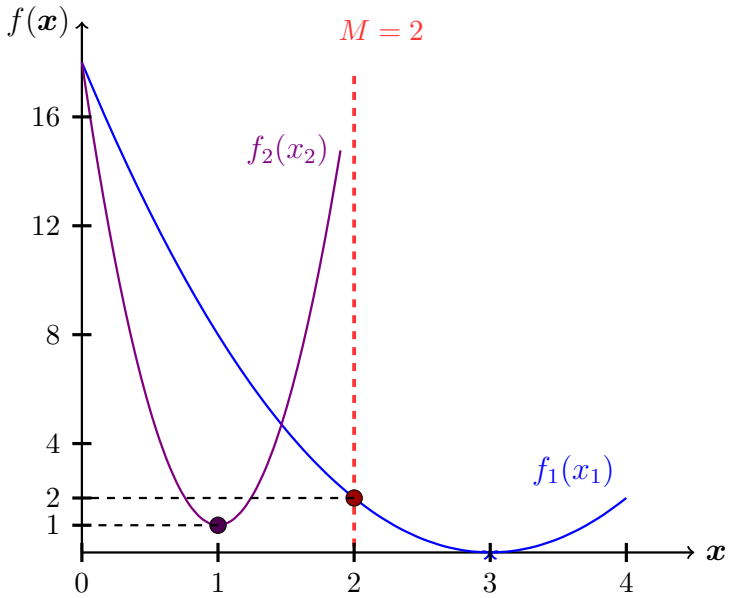


FIGURE 1.8 – Exemple de cas où la borne M (hypothèse de *BigM*) n'est pas atteinte en $\hat{\boldsymbol{x}} = \arg\min \hat{\mathcal{P}}_{2/0}$ et où la solution obtenue pour $\min \hat{\mathcal{P}}_{2/0}$ (1, atteinte en 1) n'est qu'un minimum local pour $\min \mathcal{P}_{2/0}$ (0, atteint en 3).

Dans le cas général, on ne sait donc pas régler la valeur de M de manière automatique. Cependant, pour les problèmes inverses, il est raisonnable de supposer connaître une borne naturelle M résultant de contraintes physique. C'est le cas pour les exemples abordés dans ce manuscrit :

— En déconvolution de trains d'impulsions pour le contrôle non destructif par ultrasons, les ondes émises sont générées grâce à un transducteur électro-acoustique. L'onde émise se propage alors dans le milieu et crée une onde retour à chaque changement d'impédance rencontré, causé par des défauts sur la surface du matériau. Le coefficient de réflexion, qui indique le rapport entre la pression réfléchie et la pression incidente, se calcule en fonction des impédances acoustiques des matériaux (voir par exemple [Carcreff, 2014, chap. 1]). Ce coefficient est toujours inférieur à 1, ce qui fait que l'amplitude de l'onde réfléchie est toujours inférieure (en valeur absolue) à celle de l'onde émise.

— En démélange spectral, le problème est naturellement borné puisque les variables sont des pourcentages. Les contraintes de positivité et de somme à un sur les abondances donnent naturellement des bornes inférieures et supérieures :

$$\left.\begin{array}{l} a_q \geq 0 \\ \sum_{q=1}^{n} a_q \leq 1 \end{array}\right\} \Rightarrow \quad 0 \leq a_q \leq 1. \tag{1.20}$$

Dans la suite du manuscrit, nous supposons que le problème d'approximation parcimonieuse est borné et que M est connu.

1.4.3 Méthode de branch-and-bound pour la programmation en nombres mixtes

La méthode de séparation et évaluation ou *branch-and-bound* [Land and Doig, 1960] est souvent utilisée dans les problèmes d'*optimisation discrète*, où l'espace de recherche est complexe à explorer. Nous utilisons le terme branch-and-bound pour tout le reste du manuscrit. La méthode branch-and-bound pour nos problèmes (cf. Table 1.1) sera détaillée au Chapitre 2. Nous en expliquons ici le principe général, consistant à parcourir l'ensemble des solutions réalisables d'une manière intelligente, en le restreignant progressivement, et en alternant une étape dite de « séparation » et une étape d'« évaluation ».

— La séparation consiste à diviser un problème complexe en sous-problèmes disjoints plus simples à résoudre, constituant un arbre de recherche. Une méthode simple et in-

tuitive existante est la *séparation dichotomique*, où à chaque fois, on divise le domaine réalisable d'une variable pour obtenir deux nœuds fils (deux sous-problèmes disjoints dont l'espace de recherche est plus petit, et donc plus simples), voir Figure 1.9.

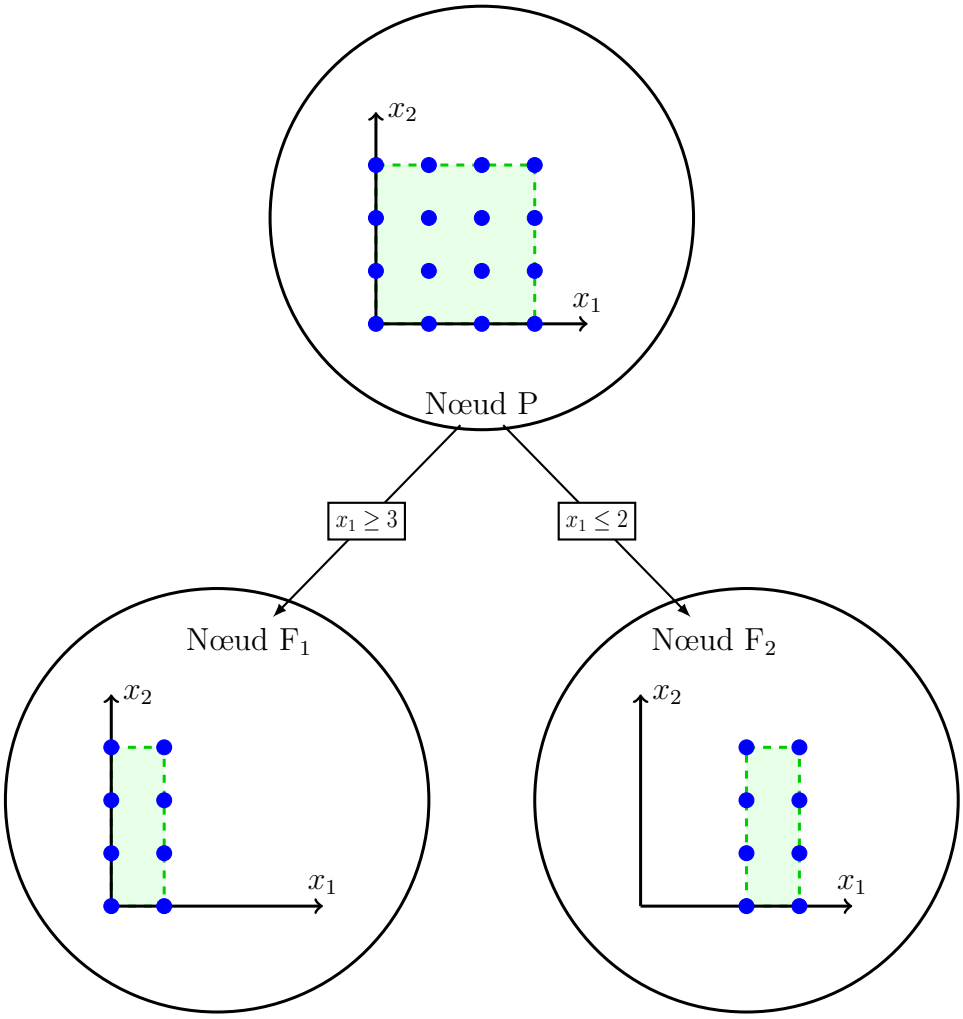


FIGURE 1.9 – Principe de séparation : création de deux nœuds fils (F_1) et (F_2) (avec un espace de recherche plus petit) à partir du nœud parent (P) pour un problème d'optimisation discrète (x_1 et x_2 entiers naturels). Les solutions entières réalisables sont représentées par des points bleus et l'espace admissible de solution est en vert. Dans cet exemple, la séparation (branchement) a été effectuée sur la variable x_1.

— L'évaluation est une étape très importante au sein de l'algorithme branch-and-bound. Elle permet de réduire l'espace de recherche en élaguant les sous-problèmes n'ayant pas la solution optimale, permettant d'éviter une *énumération explicite* de toutes les solutions du problème. On parle d'*énumération implicite* parce que toutes les

solutions admissibles sont évaluées bien qu'elles ne soient pas nécessairement toutes explorées. Elle est basée sur ke calcul de borne inférieure (en cas de minimisation) sur un sous-problème donné. Celui-ci qui permet, en comparant cette borne inférieure avec la borne supérieure définie par la meilleure solution réalisable connue, de déduire si ce sous-problème peut contenir la solution optimale ou non. L'efficacité de cette étape est liée directement à la qualité de la borne calculée et aussi au temps consommé pour son calcul.

Finalement, l'optimalité est assurée une fois que tous les sous-problèmes ont été évalués, ce qui fait de cet algorithme une méthode exacte. Un tel algorithme a plusieurs choix (règles) qui peuvent directement affecter son efficacité, tels que :

— le choix de la variable de branchement (choix de x_1 ou x_2 dans l'exemple de la figure 1.9),

— le choix du nœud à explorer en priorité (choix de F_1 ou F_2 dans l'exemple de la figure 1.9),

— le type de relaxation pour avoir des bornes inférieures (ainsi que le choix de l'algorithme pour la relaxation).

Ces choix peuvent avoir un impact important sur le nombre de nœuds à examiner dans l'arbre de branch-and-bound et il n'y a pas bien entendu de choix optimal pour tous types de problèmes. Généralement, les solveurs d'optimisation en nombres mixtes utilisent la méthode branch-and-bound avec des règles et stratégies très génériques pour couvrir le plus grand nombre de problèmes possibles. Dans les chapitres 2 à 5, nous étudions plusieurs leviers permettant de dépasser de tels solveurs génériques afin d'aborder des problèmes plus gros et plus complexes.

Chapitre 2

Méthode branch-and-bound et Relaxation continue

Contents

2.1 Introduction

Les problèmes d'approximation parcimonieuse peuvent être reformulés en programmes en nombres mixtes (MIP, *Mixed Integer Program* en anglais) (voir Section 1.4.1). Rappelons les trois reformulations :

$$\begin{array}{rcll} \hat{\mathcal{P}}_{2/0} & : & \min\limits_{\boldsymbol{b}\in\{0,1\}^n,\boldsymbol{x}\in\mathbb{R}^n} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \\ & & \texttt{s.c.} & \sum_{q=1}^n b_q \leq K \\ & & & |\boldsymbol{x}| \leq M\boldsymbol{b} \end{array} \tag{2.1}$$

la reformulation du problème contraint par la norme ℓ_0,

$$\begin{array}{rcll} \hat{\mathcal{P}}_{0/2} & : & \min\limits_{\boldsymbol{b}\in\{0,1\}^n,\boldsymbol{x}\in\mathbb{R}^n} & \sum_{q=1}^n b_q \\ & & \texttt{s.c.} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon \\ & & & |\boldsymbol{x}| \leq M\boldsymbol{b} \end{array} \tag{2.2}$$

la reformulation du problème sous contrainte d'erreur d'approximation et

$$\begin{array}{rcll} \hat{\mathcal{P}}_{2+0} & : & \min\limits_{\boldsymbol{b}\in\{0,1\}^n,\boldsymbol{x}\in\mathbb{R}^n} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 + \mu\sum_{q=1}^n b_q \\ & & \texttt{s.c.} & |\boldsymbol{x}| \leq M\boldsymbol{b}, \end{array} \tag{2.3}$$

la reformulation du problème pénalisé. La résolution de ces MIPs à l'aide d'un algorithme branch-and-bound peut s'effectuer au moyen des solveurs génériques tels que CPLEX (voir par exemple [Bourguignon et al., 2016; Liu et al., 2019]), qui sont globalement aveugles vis-à-vis des spécificités du problème et qui utilisent des techniques assez génériques pour tout type de programme en nombres mixtes (MIP).

Suivant les travaux de [Bienstock, 1996; Bertsimas and Shioda, 2009] sur le problème contraint par la cardinalité, notre motivation est basée sur le fait que les problèmes reformulés de l'approximation parcimonieuse $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$ sont des MIPs spécifiques, qui pourraient être résolus de façon globale efficacement par un algorithme branch-and-bound dédié. En particulier :

1. ils sont reformulés en MIP mêlant des variables continues et autres binaires avec un simple couplage reliant chaque variable continue à une binaire, ce qui est déjà une spécificité comparé à un programme MIP générique avec des variables entières.

2. la parcimonie de la solution (norme ℓ_0) est une contrainte qui donne une structure

particulière à l'arbre de recherche.

Afin de construire un algorithme efficace et de développer des stratégies appropriées, nous étudions de manière théorique les spécificités des trois problèmes introduits plus haut et leurs impacts sur les différentes parties de l'algorithme. Pour cela, nous divisons l'analyse en deux parties.

1. Une première partie, détaillée en Section 2.3, concerne l'étape d'évaluation de chaque sous-problème par la relaxation continue des variables binaires dans l'intervalle $[0, 1]$. En particulier, nous montrons que tous les problèmes relâchés se ramènent à des problèmes d'optimisation avec un terme en norme ℓ_1 sur une partie des variables et des contraintes de borne sur l'ensemble des variables. Ensuite, des algorithmes permettant de résoudre ces problèmes relâchés seront proposés au chapitre 3.
2. Une deuxième partie, concernant l'étape de séparation, sera étudiée au chapitre 4. Nous y proposons des stratégies de branchement et de parcours adaptées au problème, notamment inspirées des heuristiques définissant les algorithmes gloutons.

Avant cela, nous commençons par une description détaillée de la méthode branch-and-bound mise en œuvre pour le problème d'approximation parcimonieuse.

2.2 Méthode de branch-and-bound

La méthode branch-and-bound est basée sur une recherche arborescente d'une solution optimale en divisant successivement le domaine réalisable et en créant ainsi des sous-problèmes plus simples à résoudre, constituant un arbre de recherche. Afin d'éviter une énumération implicite, il faut donc que chaque sous-problème soit évalué. Cette évaluation permet principalement de prouver que le sous-problème n'est pas intéressant en ceci qu'aucune solution optimale ne s'y trouvera. Dans ce cas particulier, le nœud est élagué et la recherche est interrompue dans cette partie de l'arbre.

2.2.1 Procédure de séparation

La procédure de séparation est au centre du fonctionnement de la méthode de branch-and-bound, c'est elle qui permet d'explorer l'ensemble admissible de solutions en formulant de nouveaux sous-problèmes. Pour nos problèmes $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{2+0}$ et $\hat{\mathcal{P}}_{0/2}$ reformulés en MIP, c'est la fixation d'une ou de plusieurs variables binaires qui va permettre de partitionner l'ensemble admissible de solutions, que l'on pourra alors représenter graphiquement sous

la forme d'un arbre de recherche. La Figure 2.1 donne une représentation schématique d'un tel arbre.

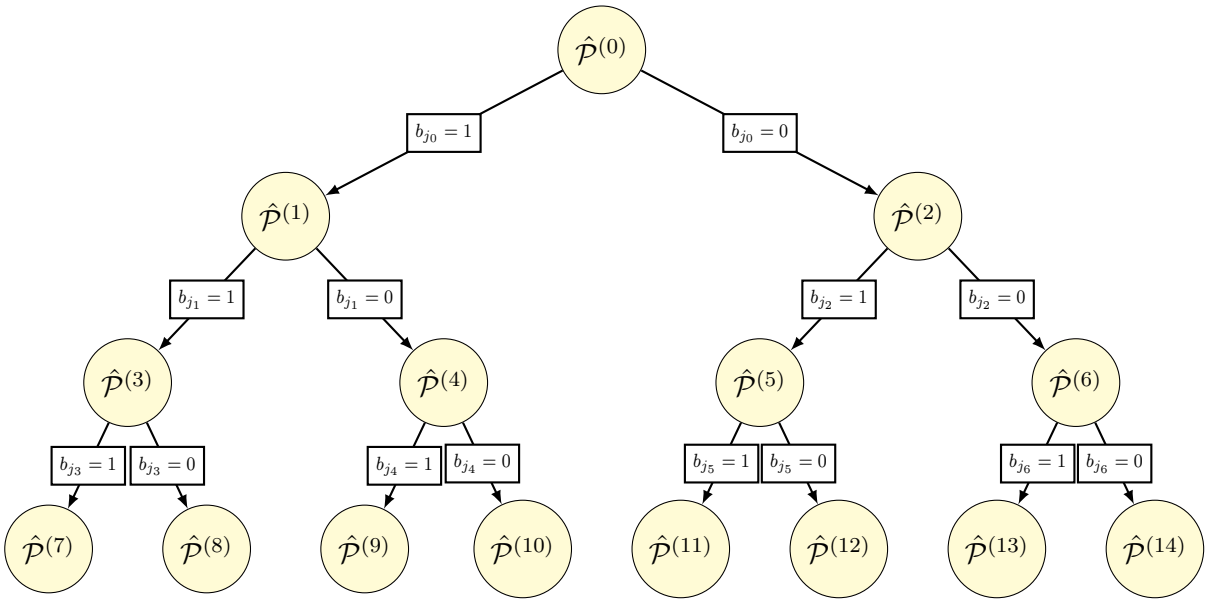


FIGURE 2.1 – Arbre binaire de décision : chaque nœud est divisé en deux nœuds fils obtenus en contraignant une variable à être nulle ou non-nulle.

La racine de cet arbre de recherche représente alors le problème initial $\hat{\mathcal{P}}^{(0)} := \hat{\mathcal{P}}$, que l'on cherche à résoudre, tandis que chaque nœud suivant représente un sous-problème dans lequel certaines variables sont fixées à 0 ou 1. La procédure de séparation se décompose en deux éléments essentiels qui peuvent avoir une grande influence sur le nombre de nœuds explorés dans l'arbre de recherche : la stratégie de branchement et la stratégie de parcours de l'arbre de recherche. Ces deux éléments seront étudiés au chapitre 4 où nous proposons des stratégies spécifiques au problème d'approximation parcimonieuse.

Pour chaque nœud i, et donc pour chaque sous-problème, les variables binaires sont donc partitionnées en trois classes, pour lesquelles nous introduisons les notations suivantes :

— les variables binaires fixées à 0 sont indexées par $\mathbb{S}_0$;

— les variables binaires fixées à 1 sont indexées par $\mathbb{S}_1$;

— les variables non fixées (que nous dénommons par la suite variables libres) sont indexées par $\bar{\mathbb{S}}$.

2.2.2 Procédure d'évaluation

L'évaluation dans la méthode branch-and-bound est essentielle car c'est elle qui va permettre d'éviter une énumération explicite de toutes les nœuds de l'arbre de recherche. Elle repose sur le calcul, d'une part, d'une borne inférieure $z^{R(i)}$ sur un sous-problème donné $\hat{\mathcal{P}}^{(i)}$, et d'autre part d'une borne supérieure globale z_U associée à la meilleure solution réalisable trouvée.

Borne inférieure. Le calcul d'une bonne inférieure consiste à définir, à un nœud i de l'arbre de recherche, une borne inférieure de la valeur optimale du sous-problème d'optimisation $\hat{\mathcal{P}}^{(i)}$ associé. Souvent, pour calculer cette borne, on utilise la *relaxation continue* qui consiste à relâcher les contraintes d'intégrité sur les variables contraintes à être entières, générant un problème d'optimisation continue plus facile à résoudre, que nous appelons $\hat{\mathcal{P}}^{R(i)}$. Dans notre cas, la relaxation continue est obtenue en relâchant les contraintes de binarité :

$$\boldsymbol{b} \in \{0,1\}^n \rightarrow \boldsymbol{b} \in [0,1]^n.$$

Par exemple, pour le problème $\hat{\mathcal{P}}_{2/0}$ contraint par la parcimonie, la relaxation continue au niveau du nœud racine s'écrit ainsi :

$$\hat{\mathcal{P}}^R_{2/0} : \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \text{ s.c. } \begin{cases} \sum_{q=1}^n b_q \leq K \\ |\boldsymbol{x}| \leq M\boldsymbol{b} \end{cases} . \tag{2.4}$$

La solution de ce problème relâché et la valeur de la borne inférieure sur le nœud racine sont alors :

$$\begin{cases} z^R = \min\limits_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R_{2/0} \\ (\boldsymbol{x}^R, \boldsymbol{b}^R) = \arg \min\limits_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R_{2/0} \end{cases} . \tag{2.5}$$

Borne supérieure. La borne supérieure est définie comme la meilleure (*i.e.,* la plus petite) valeur de la fonction de coût du problème connue à une étape donnée de la résolution. Par définition, c'est bien une borne supérieure sur la valeur de l'optimum global du problème. Elle peut être initialisée au début de l'algorithme par la valeur $+\infty$ ou par la valeur de la fonction objectif d'une solution réalisable obtenue par une heuristique quelconque. Une bonne initialisation de la borne supérieure z_U permet parfois une amélioration significative de la qualité et de la rapidité de l'exploration de l'arbre de recherche. La borne supérieure est en général mise à jour durant l'exploration de l'arbre par la valeur de

la fonction objectif obtenue par la meilleure solution réalisable trouvée. Ainsi, une solution $(\boldsymbol{b}^{R(i)}, \boldsymbol{x}^{R(i)}) = \arg\min \hat{\mathcal{P}}^{R(i)}$ devient une nouvelle solution potentiellement optimale du MIP, si elle est réalisable pour le sous-problème $\hat{\mathcal{P}}^{(i)}$ (c'est-à-dire si $\boldsymbol{b}^{R(i)}$ est binaire) et si sa valeur optimale $z^{R(i)}$ est inférieure à la borne supérieure actuelle z_U. Cette solution sera appelée une solution potentiellement optimale du MIP. En effet, celle-ci n'est pas (encore) dominée par une autre solution réalisable, mais pourra l'être plus tard par une solution trouvée ultérieurement.

Élagage d'un nœud. En utilisant la borne inférieure et la borne supérieure, l'algorithme branch-and-bound cherche à éviter une énumération explicite de toutes les solutions réalisables dans l'espace de recherche. L'exploration d'un nœud évalué i sera ainsi arrêtée (le nœud i sera élagué) si l'une des conditions suivantes est vérifiée :

— le sous-problème relâché associé $\hat{\mathcal{P}}^{R(i)}$ n'a pas de solution. Dans ce cas, $\hat{\mathcal{P}}^{(i)}$ n'a pas de solution non plus puisque l'ensemble de solutions réalisables pour $\hat{\mathcal{P}}^{(i)}$ est inclus dans l'ensemble de solutions réalisables pour $\hat{\mathcal{P}}^{R(i)}$.

— La valeur de sa borne inférieure $z^{R(i)} = \min \hat{\mathcal{P}}^{R(i)}$ est supérieure ou égale à la borne supérieure z_U. En effet, si le sous-problème $\hat{\mathcal{P}}^{(i)}$ contient la solution optimale du MIP, sa valeur $\min \hat{\mathcal{P}}^{(i)}$ est forcément supérieure ou égale à $z^{R(i)}$, ce qui nous permet de conclure que si $z^{R(i)} \geq z_U$, alors aucune solution réalisable du sous-problème $\hat{\mathcal{P}}^{(i)}$ n'est meilleure que celle déjà trouvée et le sous-problème sera élagué.

2.2.3 Algorithme branch-and-bound

Nous présentons dans l'Algorithme 3 le schéma générique de la méthode branch-and-bound pour la résolution d'un des trois MIP. Nous utilisons les notations suivantes :

— L : l'ensemble des sous-problèmes actifs ;

— $(\hat{\boldsymbol{b}}, \hat{\boldsymbol{x}})$: la meilleure solution réalisable connue ;

— z_U : la borne supérieure courante ;

— $\hat{\mathcal{P}}^{R(i)}$: sous-problème i relâché (*relaxation continue* du sous-problème $\hat{\mathcal{P}}^{(i)}$) ;

— $z^{R(i)} = \min \hat{\mathcal{P}}^{R(i)}$: la borne inférieure sur la valeur du sous-problème i ;

— $(\boldsymbol{x}^{R(i)}, \boldsymbol{b}^{R(i)})$: la solution du sous-problème relâché i ;

— z_L : la borne inférieure globale du problème ; $z_L = \min_{i \in L} z^{R(i)}$.

L'algorithme génère et parcourt simultanément l'arbre de recherche jusqu'à atteindre la condition d'arrêt.

Initialisation : $L \leftarrow \{\text{Problème initial}\}$; $z_U \leftarrow +\infty$;

Condition d'arrêt : Si $L = \emptyset$, alors $(\hat{\boldsymbol{b}}, \hat{\boldsymbol{x}})$ est la solution optimale ;

Sélection du nœud : Choisir un sous-problème i dans L et l'éliminer de L ;

Évaluation du nœud :

début
  Résoudre le problème relâché $\hat{\mathcal{P}}^{R(i)}$;
  si $\hat{\mathcal{P}}^{R(i)}$ *n'a pas de solution* **alors**
    retourner à l'étape Optimalité ; ▷ Élagage
  sinon
    poser $z^{R(i)}$ et $(\boldsymbol{b}^{R(i)}, \boldsymbol{x}^{R(i)})$ la valeur et la solution optimales de $\hat{\mathcal{P}}^{R(i)}$;
  fin
  si $z^{R(i)} \geq z_U$ **alors**
    retourner à l'étape Optimalité. ▷ Élagage
  sinon
    si $\boldsymbol{b}^{R(i)}$ *est binaire* **alors**
      mise à jour de la solution potentiellement optimale du MIP : $\hat{\boldsymbol{b}} \leftarrow \boldsymbol{b}^{R(i)}$ et $\hat{\boldsymbol{x}} \leftarrow \boldsymbol{x}^{R(i)}$;
      mise à jour de la borne supérieure $z_U \leftarrow z^{R(i)}$;
      éliminer de L tous les sous-problèmes j tels que $z^{R(j)} \geq z_U$; ▷ Élagage
      mise à jour de la borne inférieure globale $z_L = \min_{i \in L} z^{R(i)}$;
      retourner à l'étape Optimalité ; ▷ Élagage
    sinon
      aller à l'étape Branchement ;
    fin
  fin
fin

Branchement : Choisir une variable binaire b_q^i qui n'est pas encore sélectionnée et subdiviser le problème i à partir de cette variable en deux sous-problèmes. Ajouter les nouveaux sous-problèmes à L.

Algorithme 3 : Algorithme branch-and-bound

2.2.4 Condition d'arrêt

L'algorithme se termine lorsque tous les nœuds dans L ont été évalués. Ainsi le *saut d'intégrité Gap*, c'est-à-dire l'écart relatif entre la valeur de la borne supérieure z_U et la valeur de la borne inférieure globale z_L :

$$Gap = \frac{z_U - z_L}{z_U}, \tag{2.6}$$

devient nul. La Figure 2.2 illustre la convergence des bornes calculées par l'algorithme de branch-and-bound. En pratique, et puisque numériquement les valeurs que l'on manipule durant le calcul des bornes inférieures sont parfois imprécises, les solveurs d'optimisation MIP considèrent en général une tolérance d'optimalité sur ce saut d'intégrité (par exemple 10^{-4} par défaut dans le solveur CPLEX).

Il est important de noter, cependant, que le réglage de ce saut d'intégrité peut être critique. Dans l'ensemble de nos simulations dans le cadre des problèmes inverses mal posés, le réglage par défaut s'est avéré insuffisant : en raison de la très forte corrélation entre les colonnes du dictionnaire, les solutions obtenues à chaque nœud exploré peuvent fournir une modélisation très proche de la solution optimale, ce qui provoque souvent l'arrêt prématuré de l'algorithme. Il y a donc un intérêt primordial à disposer d'algorithmes d'optimisation dits *exacts* pour le calcul des bornes inférieures, qui convergent en un nombre fini d'itérations et ainsi plus stables numériquement.

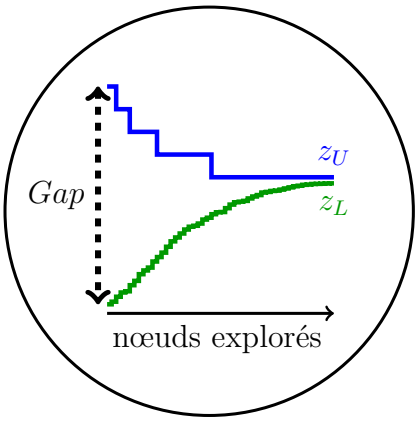


FIGURE 2.2 – Gap : l'écart relatif entre la valeur de la borne supérieure z_U et la valeur de la borne inférieure globale z_L décroît au fur et à mesure de l'exploration des nœuds.

2.3 Borne inférieure : Relaxation continue

Le calcul d'une relaxation de bonne qualité est un élément essentiel pour que la méthode branch-and-bound) soit efficace. Le but est de fournir une borne inférieure sur un sous-problème donné $z^{R(i)} = \min \hat{\mathcal{P}}^{R(i)}$ la plus proche de la valeur optimale $\min \hat{\mathcal{P}}^{(i)}$. Cependant, il y a un compromis à définir entre la qualité de la relaxation et le temps nécessaire pour la calculer.

La relaxation la plus simple utilisée par les différents solveurs MIP est la relaxation continue. Pour des problèmes contraints par la cardinalité (et avec des contraintes supplémentaires de positivité des inconnues $\boldsymbol{x}$), Bienstock [1996] a été le premier à proposer un algorithme branch-and-bound avec une modélisation permettant de résoudre la relaxation continue sans variables binaires en remplaçant la contrainte de cardinalité par $\sum_q x_q/M \leq K$:

$$\begin{array}{c} \text{contraintes de (2.4)} \\ \\ \text{positivité} \end{array} \left\{ \begin{array}{c} x_q \leq M b_q \\ \sum_q b_q \leq K \\ b_q \in [0,1] \\ \boldsymbol{x} \geq 0 \end{array} \right\} \Rightarrow \begin{cases} \sum_q x_q/M \leq K \\ \boldsymbol{x} \geq 0 \end{cases} .$$

Ce résultat permet de réduire le nombre de variables et par conséquent d'envisager une résolution plus rapide. Nous exploitons cette idée et montrons dans cette section que, pour les trois problèmes $\hat{\mathcal{P}}^R_{2/0}$, $\hat{\mathcal{P}}^R_{0/2}$ et $\hat{\mathcal{P}}^R_{2+0}$, la relaxation continue pour chaque sous-problème (un nœud dans l'arbre de recherche, voir Figure 2.1) revient à un problème d'optimisation sans variables binaires, faisant intervenir la norme ℓ_1 des variables continues. Nous analysons tout d'abord les relaxations impliquées au nœud racine de ces 3 problèmes, puis nous généralisons ces propriétés pour tout nœud de l'arbre.

2.3.1 Relaxation continue au niveau du nœud racine

Au nœud racine, aucune décision n'a été prise concernant une variable binaire. Rappelons que la relaxation continue du problème $\hat{\mathcal{P}}_{2/0}$ (donnée par l'équation (2.4)) est alors la suivante :

$$\hat{\mathcal{P}}^R_{2/0} : \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \text{ s.c. } \begin{cases} \sum_{q=1}^n b_q \leq K \\ |\boldsymbol{x}| \leq M\boldsymbol{b} \end{cases} .$$

Soit $(\boldsymbol{b}^R, \boldsymbol{x}^R)$ un minimiseur de $\hat{\mathcal{P}}^R_{2/0}$. Alors, si la contrainte $\sum_{q=1}^n b_q \leq K$ est inactive (non

saturée), on peut dire à l'optimum que :

$$\min \hat{\mathcal{P}}^R_{2/0} = \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \;\text{ s.c. } ||\boldsymbol{x}||_\infty \leq M \;. \tag{2.7}$$

Maintenant, si la contrainte $\sum_{q=1}^n b_q \leq K$ est active, c'est à dire si le minimiseur se trouve à la borne de la contrainte $\sum_{q=1}^n b_q^R = K$, nous montrons que la relaxation continue des variables binaires peut être réduite à un problème d'optimisation en norme ℓ_1.

Proposition 1. *La solution du problème $\hat{\mathcal{P}}^R_{2/0}$ est donnée par :*

$$(\boldsymbol{b}^R, \boldsymbol{x}^R) = \arg \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R_{2/0} \Leftrightarrow \begin{cases} \boldsymbol{b}^R = |\boldsymbol{x}^R|/M \\ \boldsymbol{x}^R = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}_{2/1} \end{cases}, \tag{2.8}$$

avec

$$\hat{\mathcal{P}}_{2/1} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \;\; s.c. \begin{cases} \|\boldsymbol{x}\|_1 = KM \\ \|\boldsymbol{x}\|_\infty \leq M \end{cases}. \tag{2.9}$$

Démonstration. Soit $(\boldsymbol{b}^R, \boldsymbol{x}^R)$ un minimiseur de $\hat{\mathcal{P}}^R_{2/0}$ et $\boldsymbol{x}^1$ un minimiseur de $\hat{\mathcal{P}}_{2/1}$. Soit $\boldsymbol{b}^1 := \frac{1}{M}|\boldsymbol{x}^1|$. Alors, $(\boldsymbol{b}^1, \boldsymbol{x}^1)$ est réalisable pour $\hat{\mathcal{P}}^R_{2/0}$, donc $\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^R\|_2^2 \leq \|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^1\|_2^2$. Réciproquement, $\boldsymbol{x}^R$ est réalisable pour $\hat{\mathcal{P}}_{2/1}$ car $\|\boldsymbol{x}^R\|_1 \leq M\|\boldsymbol{b}^R\|_1 = M\sum_{q=1}^n b_q^R \leq KM$ et $\|\boldsymbol{x}^R\|_\infty \leq M\|\boldsymbol{b}^R\|_\infty \leq M$. Par conséquent, $\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^1\|_2^2 \leq \|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^R\|_2^2$. □

L'argument est d'ailleurs souvent avancé que la norme ℓ_1 est la relaxation convexe de la norme ℓ_0. Ceci est vrai sous des contraintes supplémentaires de borne, comme le montre la Figure 2.3.

Un résultat similaire concerne la relaxation continue $\hat{\mathcal{P}}^R_{0/2}$ du problème sous contrainte d'erreur d'approximation $\hat{\mathcal{P}}_{0/2}$:

$$\hat{\mathcal{P}}^R_{0/2} : \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \sum_{q=1}^n b_q \;\text{ s.c. } \begin{cases} \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon \\ |\boldsymbol{x}| \leq M\boldsymbol{b} \end{cases}. \tag{2.10}$$

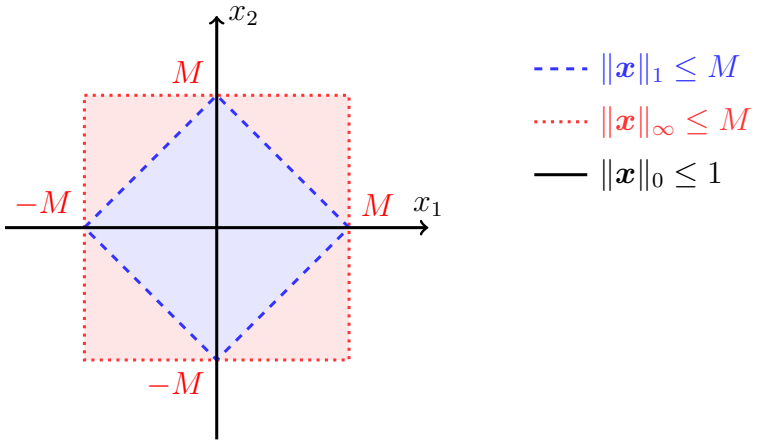


FIGURE 2.3 – Relaxation convexe de la norme ℓ_0 par la norme ℓ_1 sous contrainte de borne.

Proposition 2. *La solution du problème $\hat{\mathcal{P}}^R_{0/2}$ peut être calculée par :*

$$(\boldsymbol{b}^R, \boldsymbol{x}^R) = \arg \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R_{0/2} \Leftrightarrow \begin{cases} \boldsymbol{b}^R = |\boldsymbol{x}^R|/M \\ \boldsymbol{x}^R = \arg \min_{\boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}_{1/2} \end{cases} \tag{2.11}$$

avec

$$\hat{\mathcal{P}}_{1/2} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{M}\|\boldsymbol{x}\|_1 \quad s.c. \begin{cases} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \leq \epsilon \\ \|\boldsymbol{x}\|_\infty \leq M \end{cases} . \tag{2.12}$$

Démonstration. Nous montrons tout d'abord que $|\boldsymbol{x}^R| = M\boldsymbol{b}^R$. Supposons que $|x^R_{q_0}| < Mb^R_{q_0}$ pour une composante de coordonnée q_0. Soit $\boldsymbol{b}' := \frac{1}{M}|\boldsymbol{x}^R|$, avec donc $b'_{q_0} < b^R_{q_0}$. Alors, $(\boldsymbol{b}', \boldsymbol{x}^R)$ est réalisable pour $\hat{\mathcal{P}}^R_{0/2}$, avec $\sum_{q=1}^n b'_q < \sum_{q=1}^n b^R_q$, qui est en contradiction avec la définition de $(\boldsymbol{b}^R, \boldsymbol{x}^R)$. □

Enfin, considérons la relaxation continue des variables binaires pour le problème pénalisé $\hat{\mathcal{P}}_{2+0}$ ci-dessous :

$$\hat{\mathcal{P}}^R_{2+0} : \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \mu \sum_{q=1}^n b_q \text{ s.c. } |\boldsymbol{x}| \leq M\boldsymbol{b}. \tag{2.13}$$

Proposition 3. *La solution du problème $\hat{\mathcal{P}}^R_{2+0}$ est alors donnée par :*

$$(\boldsymbol{b}^R, \boldsymbol{x}^R) = \arg \min_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R_{2+0} \Leftrightarrow \begin{cases} \boldsymbol{x}^R = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}_{2+1} \\ \boldsymbol{b}^R = |\boldsymbol{x}^R|/M \end{cases} \tag{2.14}$$

où

$$\hat{\mathcal{P}}_{2+1} : \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 + \lambda_c \|\boldsymbol{x}\|_1 \quad s.c. \quad \|\boldsymbol{x}\|_\infty \leq M. \tag{2.15}$$

avec $\lambda_c = \frac{\mu}{M}$.

Démonstration. La preuve est similaire au cas précédent. □

2.3.2 Relaxation continue au niveau d'un nœud quelconque

Nous considérons maintenant un nœud quelconque de l'algorithme branch-and-bound et la relaxation continue du sous-problème correspondant pour les trois problèmes $\hat{\mathcal{P}}^R_{2/0}$, $\hat{\mathcal{P}}^R_{0/2}$ et $\hat{\mathcal{P}}^R_{2+0}$. Rappelons que $\mathbb{S}_0$ (respectivement, $\mathbb{S}_1$) définit l'ensemble des indices des variables binaires fixées à 0 (respectivement, à 1), et que $\bar{\mathbb{S}}$ définit l'ensemble des indices des variables binaires non déterminées (voir Figure 2.4) :

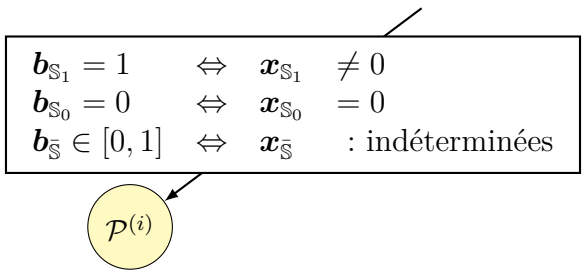


FIGURE 2.4 – Configuration en un nœud i quelconque.

Afin de simplifier le problème, nous retirons les variables $\boldsymbol{x}_{\mathbb{S}_0}$, qui sont fixées à zéro, des problèmes d'optimisation associés à l'évaluation du nœud. Pour le problème $\hat{\mathcal{P}}_{2/0}$, la

relaxation continue des variables $\boldsymbol{b}_{\bar{\mathbb{S}}}$ du sous-problème correspondant se réduit à :

$$\hat{\mathcal{Q}}^R_{2/0}: \min_{\substack{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1}\\ \boldsymbol{b}_{\bar{\mathbb{S}}}\in[0,1]^{\bar{n}}\\ \boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}} \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \text{ s.c. } \begin{cases}\sum_{q\in\bar{\mathbb{S}}} b_q \leq K-n_1\\ |\boldsymbol{x}_{\bar{\mathbb{S}}}| \leq M\boldsymbol{b}_{\bar{\mathbb{S}}}\\ \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M\end{cases}, \tag{2.16}$$

où n_1 et $\bar{n}$ représentent respectivement la taille de $\mathbb{S}_1$ et de $\bar{\mathbb{S}}$. Ensuite, de manière similaire aux développements du § 2.3.1, si $(\boldsymbol{x}^R_{\mathbb{S}_1}, \boldsymbol{b}^R_{\bar{\mathbb{S}}}, \boldsymbol{x}^R_{\bar{\mathbb{S}}})$ est un minimiseur de $\hat{\mathcal{Q}}^R_{2/0}$, alors on a $\boldsymbol{b}^R_{\bar{\mathbb{S}}} = \frac{1}{M}|\boldsymbol{x}^R_{\bar{\mathbb{S}}}|$. Ainsi, la solution du problème $\hat{\mathcal{Q}}^R_{2/0}$ peut être donnée par :

$$(\boldsymbol{x}^R_{\mathbb{S}_1}, \boldsymbol{b}^R_{\bar{\mathbb{S}}}, \boldsymbol{x}^R_{\bar{\mathbb{S}}}) = \arg \min_{\substack{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1}\\ \boldsymbol{b}_{\bar{\mathbb{S}}}\in[0,1]^{\bar{n}}\\ \boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}} \hat{\mathcal{Q}}^R_{2/0} \Leftrightarrow \begin{cases}(\boldsymbol{x}^R_{\bar{\mathbb{S}}}, \boldsymbol{x}^R_{\mathbb{S}_1}) = \arg \min_{\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}, \boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1}} \hat{\mathcal{Q}}_{2/1}\\ \boldsymbol{b}^R_{\bar{\mathbb{S}}} = |\boldsymbol{x}^R_{\bar{\mathbb{S}}}|/M\\ \boldsymbol{b}^R_{\mathbb{S}_1} = \mathbf{1}\end{cases} \tag{2.17}$$

où

$$\hat{\mathcal{Q}}_{2/1}: \min_{\substack{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1}\\ \boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}} \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \text{ s.c. } \begin{cases}\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \leq \tau_c\\ \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \leq M\\ \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M\end{cases}. \tag{2.18}$$

avec $\tau_c = M(K-n_1)$. En appliquant un raisonnement similaire aux deux autres formulations, on obtient par la suite les problèmes équivalents résumés dans la Table 2.1.

2.4 Conclusion

Dans ce chapitre, nous avons présenté les différentes parties de l'algorithme branch-and-bound. Nous avons principalement étudié la relaxation continue impliquée dans l'étape d'évaluation de l'algorithme, où nous avons montré deux propriétés :

— quelle que soit la formulation (contrainte ou pénalisée), tous les sous-problèmes relâchés $\hat{\mathcal{P}}^R_{2/0}$, $\hat{\mathcal{P}}^R_{0/2}$ et $\hat{\mathcal{P}}^R_{2+0}$ impliqués dans l'évaluation de chaque nœud de l'algorithme branch-and-bound peuvent être reformulés sans variable binaire.

Relaxation continue			Problème équivalent sans variables binaires		
$\hat{\mathcal{Q}}^R_{2/0}$:	$\min\limits_{\boldsymbol{b}\in[0,1]^n,\,\boldsymbol{x}\in\mathbb{R}^n}$	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2$	$\hat{\mathcal{Q}}_{2/1}$:	$\min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}$	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2$
	s.c.	$\sum_{q=1}^n b_q \le K$		s.c.	$\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \le \tau_c$
		$\|\boldsymbol{x}\| \le M\boldsymbol{b}$			$\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_1} = 1$			$\|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_0} = 0$		avec	$\tau_c = M(K-n_1)$
$\hat{\mathcal{Q}}^R_{0/2}$:	$\min\limits_{\boldsymbol{b}\in[0,1]^n,\,\boldsymbol{x}\in\mathbb{R}^n}$	$\sum_{q=1}^n b_q$	$\hat{\mathcal{Q}}_{1/2}$:	$\min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}$	$\frac{1}{M}\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 + n_1$
	s.c.	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 \le \epsilon$		s.c.	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \le \epsilon$
		$\|\boldsymbol{x}\| \le M\boldsymbol{b}$			$\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_1} = 1$			$\|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_0} = 0$			
$\hat{\mathcal{Q}}^R_{2+0}$:	$\min\limits_{\boldsymbol{b}\in[0,1]^n,\,\boldsymbol{x}\in\mathbb{R}^n}$	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2$ $+\mu\sum_{q=1}^n b_q$	$\hat{\mathcal{Q}}_{2+1}$:	$\min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}$	$\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2$ $+\lambda_c\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 + \mu\, n_1$
	s.c.	$\|\boldsymbol{x}\| \le M\boldsymbol{b}$		s.c.	$\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_1} = 1$			$\|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \le M$
		$\boldsymbol{b}_{\mathbb{S}_0} = 0$		avec	$\lambda_c = \frac{\mu}{M}$

TABLE 2.1 – Problèmes de relaxation continue à un nœud quelconque de la méthode branch-and-bound (à gauche), et problèmes équivalents sans variables binaires impliquant la norme ℓ_1 (à droite), pour les trois formulations considérées.

— Ils se réduisent tous à des problèmes d'optimisation mélangeant une fonction des moindres carrés, des termes en norme ℓ_1 impliquant seulement une partie des variables (les variables pour lesquelles la décision nulle/non-nulle n'a pas encore été prise), ainsi que des contraintes de borne sur l'ensemble des variables.

Ces propriétés présentent un intérêt majeur pour notre travail. Dans le chapitre suivant (Chapitre 3), nous allons développer des méthodes appropriées et rapides pour résoudre les sous-problèmes relâchés $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$, dans le but de construire un algorithme branch-and-bound efficace.

Chapitre 3

Calcul de la borne inférieure : Méthodes spécifiques pour l'optimisation en norme ℓ_1

Contents

3.1 Introduction

Dans le chapitre précédent, nous avons étudié la relaxation continue impliquée dans l'évaluation de chaque nœud de l'algorithme branch-and-bound et nous avons montré que les problèmes relâchés quelle que soit la formulation :

$$\begin{array}{rll} \hat{\mathcal{Q}}_{2/1}: & \min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \\ & \text{s.c.} & \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \leq \tau_c \\ & & \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \leq M,\ \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M \end{array} \tag{3.1}$$

et

$$\begin{array}{rll} \hat{\mathcal{Q}}_{1/2}: & \min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}} & \frac{1}{M}\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 + n_1 \\ & \text{s.c.} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \leq \epsilon \\ & & \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \leq M, \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M \end{array} \tag{3.2}$$

ou la formulation pénalisée

$$\begin{array}{rll} \hat{\mathcal{Q}}_{2+1}: & \min\limits_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 + \lambda_c\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \\ & \text{s.c.} & \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \leq M \\ & & \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M \end{array}, \tag{3.3}$$

se réduisent tous à des problèmes comprenant une fonction des moindres carrés, des termes en norme ℓ_1 impliquant seulement une partie des variables et des contraintes de borne. Rappelons que $\mathbb{S}_1$, de taille n_1, dénote l'ensemble des indices des variables non nulles ($\boldsymbol{b}_{\mathbb{S}_1} = \mathbf{1}$ dans la formulation MIP) et que $\bar{\mathbb{S}}$, de taille $\bar{n}$, dénote l'ensemble des indices des variables non déterminées ($\boldsymbol{b}_{\bar{\mathbb{S}}} \in [0,1]^{\bar{n}}$ dans la formulation MIP), les variables nulles ayant été retirées du sous-problème correspondant.

L'optimisation faisant intervenir une erreur quadratique et la norme ℓ_1 (connue en statistique sous le nom de LASSO pour *Least Absolute Shrinkage and Selection Operator* [Tibshirani, 1996]) a fait l'objet de nombreuses recherches et de nombreux algorithmes d'optimisation dédiés ont été développés (voir par exemple [Tropp and Wright, 2009; Eldar and Kutyniok, 2012] et leurs références). Dans ce chapitre, nous allons développer des algorithmes dédiés permettant une résolution efficace des problèmes $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$, qui ne sont pas des problèmes en norme ℓ_1 standard en raison des contraintes de borne et du fait que la norme ℓ_1 n'est prise que sur une partie des variables.

Nous commençons dans la section 3.2 par une étude sur les conditions d'optimalité du

problème pénalisé $\hat{\mathcal{Q}}_{2+1}$. Ensuite, nous étudions deux classes d'algorithmes exacts (*i.e.,* pour lesquels la solution est obtenue en un nombre fini d'itérations), permettant de garantir les bornes inférieures calculées dans la procédure de *branch-and-bound*. En Section 3.3, nous proposons une généralisation de la méthode homotopique [Osborne et al., 2000; Efron et al., 2004; Malioutov et al., 2005; Donoho and Tsaig, 2008] pour résoudre les trois problèmes $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$. Cette approche semble appropriée à notre problème étant donné sa capacité à résoudre les trois problèmes avec la même efficacité – en particulier le problème avec contraintes quadratiques $\hat{\mathcal{Q}}_{1/2}$, que peu d'algorithmes connus peuvent aborder efficacement. Ensuite en Section 3.4, nous proposons une généralisation d'un algorithme de type *ensemble actif*, appelé *feature-sign search* dans [Lee et al., 2007] pour résoudre le problème $\hat{\mathcal{Q}}_{2+1}$. Cette approche semble appropriée parce qu'elle permet le démarrage à chaud lors de l'exploration de l'arbre de recherche. Ces méthodes sont évaluées en Section 3.5 et sont comparées à la résolution de la formulation initiale du problème de relaxation continue par le solveur CPLEX. Une conclusion et l'évocation de quelques perspectives spécifiques clôturent ce chapitre en Section 3.6.

3.2 Conditions d'optimalité du problème pénalisé $\hat{\mathcal{Q}}_{2+1}$

Nous nous intéressons d'abord au problème $\hat{\mathcal{Q}}_{2+1}$, le problème relâché impliqué dans la forme pénalisée. Afin de simplifier les notations, nous considérons de manière équivalente le problème d'optimisation suivant :

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & F(\boldsymbol{x},\lambda) := J(\boldsymbol{x}) + \lambda p(\boldsymbol{x}) \;\text{ s.c. }\; g_q(\boldsymbol{x}) \leq 0 \quad \forall q = 1,\dots,n, \\ \text{avec} \quad & J(\boldsymbol{x}) := \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2, \\ & p(\boldsymbol{x}) := \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1, \\ & g_q(\boldsymbol{x}) := |x_q| - M. \end{aligned} \tag{3.4}$$

où, les variables nulles ayant été retirées du problèmes, nous redéfinissons tout au long de ce chapitre $\boldsymbol{x} := \begin{bmatrix} \boldsymbol{x}_{\mathbb{S}_1} \\ \boldsymbol{x}_{\bar{\mathbb{S}}} \end{bmatrix}$, $\mathbf{H} := [\mathbf{H}_{\mathbb{S}_1}, \mathbf{H}_{\bar{\mathbb{S}}}]$ et $n := n_1 + \bar{n}$. La fonction objectif $F(\boldsymbol{x},\lambda)$ est convexe puisqu'elle est la somme de deux fonctions convexes. Le Lagrangien associé s'écrit :

$$\mathcal{L}(\boldsymbol{x},\lambda,\boldsymbol{\pi}) = J(\boldsymbol{x}) + \lambda p(\boldsymbol{x}) + \sum_{q=1}^{n} \pi_q g_q(\boldsymbol{x}), \tag{3.5}$$

avec $\boldsymbol{\pi} \in \mathbb{R}^n$ le vecteur des multiplicateurs de Lagrange associés aux contraintes $g_q(\boldsymbol{x}) \leq 0$.

La fonction J est une fonction lisse continûment différentiable en tout point et son gradient est :

$$\nabla J(\boldsymbol{x}) = -\mathbf{H}^T(\boldsymbol{y} - \mathbf{H}\boldsymbol{x}). \tag{3.6}$$

Les fonctions p et g_q ne sont pas lisses puisque la fonction valeur absolue n'est pas différentiable en 0. Son sous-différentiel s'écrit :

$$\partial|x| = \begin{cases} -1 & \text{si } x < 0 \\ +1 & \text{si } x > 0 \\ [-1, 1] & \text{si } x = 0 \end{cases}. \tag{3.7}$$

La Figure 3.1 représente graphiquement le sous-différentiel de $|x|$ en 0.

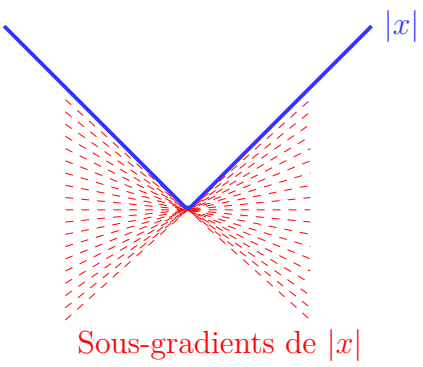


FIGURE 3.1 – Sous-différentiel de $|x|$: la fonction $|x|$ pour $x \in \mathbb{R}$ est affichée en bleu et les droites orientées par les éléments du sous-différentiel en 0 sont en pointillé rouge.

Les sous-différentiels de $p(\boldsymbol{x})$ et de $g_q(\boldsymbol{x})$ sont alors respectivement :

$$\partial p(\boldsymbol{x}) = \left\{ \boldsymbol{z} \in \mathbb{R}^n \;\middle|\; \begin{array}{ll} z_q = 0 & \text{si } q \in \mathbb{S}_1 \\ z_q = \operatorname{sgn}(x_q) & \text{si } q \in \bar{\mathbb{S}} \text{ et } x_q \neq 0 \\ z_q \in [-1, 1] & \text{si } q \in \bar{\mathbb{S}} \text{ et } x_q = 0 \end{array} \right\} \tag{3.8}$$

et

$$\partial g_q(\boldsymbol{x}) = \left\{ \boldsymbol{z} \in \mathbb{R}^n \;\middle|\; \begin{array}{ll} z_\ell = 0 & \text{pour } \ell \neq q \\ z_q = \operatorname{sgn}(x_q) & \text{si } x_q \neq 0 \\ z_q \in [-1, 1] & \text{si } x_q = 0 \end{array} \right\}, \tag{3.9}$$

où $\operatorname{sgn}(x_q)$ est le signe de x_q (1 si $x_q > 0$, -1 si $x_q > 0$). Le vecteur $\boldsymbol{x}^*$ est un minimiseur de (3.4) si et seulement s'il existe $\boldsymbol{\pi}^* \in \mathbb{R}^n$ tel que $(\boldsymbol{x}^*, \boldsymbol{\pi}^*)$ satisfait les conditions d'optimalité de Karush-Kuhn-Tucker appliquées aux fonctions continues, convexes et non

différentiables (voir par exemple [Rockafellar, 1970], Chapitre 28) :

$$\begin{cases} \mathbf{0} \in \nabla J(\boldsymbol{x}^*) + \lambda \partial p(\boldsymbol{x}^*) + \partial \sum_{q=1}^{n} \pi_q^* g_q(\boldsymbol{x}^*) & \text{(3.10a)} \\ g_q(\boldsymbol{x}^*) \leq 0 \ \ \forall q = 1, \ldots, n & \text{(3.10b)} \\ \pi_q^* \geq 0 \ \ \forall q = 1, \ldots, n & \text{(3.10c)} \\ \pi_q^* g_q(\boldsymbol{x}^*) = 0 \ \ \forall i = 1, \ldots, n. & \text{(3.10d)} \end{cases}$$

Les points particuliers sont ceux qui activent les contraintes de borne ($x_q^* = \pm M$ pour $q \in \bar{\mathbb{S}} \cup \mathbb{S}_1$) ou les points de non-différentiabilité ($x_q^* = 0$ pour $q \in \bar{\mathbb{S}}$). Par conséquent, nous divisons les indices des variables en cinq cas possibles (que nous appelons le *support*) :

$$\begin{cases} \left.\begin{array}{lcl} \bar{\mathbb{S}}_0 & := \{q \in \bar{\mathbb{S}} & \mid \ |x_q^*| = 0\} \\ \bar{\mathbb{S}}_\square & := \{q \in \bar{\mathbb{S}} & \mid \ |x_q^*| = M\} \\ \mathbb{S}_\square & := \{q \in \mathbb{S}_1 & \mid \ |x_q^*| = M\} \end{array}\right\} \text{Support fixé} \\ \left.\begin{array}{lcl} \bar{\mathbb{S}}_{in} & := \{q \in \bar{\mathbb{S}} & \mid \ 0 < |x_q^*| < M\} \\ \mathbb{S}_{in} & := \{q \in \mathbb{S}_1 & \mid \ 0 \leq |x_q^*| < M\} \end{array}\right\} \text{Support actif} \end{cases} \tag{3.11}$$

Nous détaillons maintenant les conditions d'optimalité en séparant les variables selon leur appartenance à l'un de ces cinq cas de support. Soit :

$$\boldsymbol{c} := -\nabla J(\boldsymbol{x}^*) = \mathbf{H}^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}^* - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}^*). \tag{3.12}$$

- Cas 1 : Conditions d'optimalité de $\boldsymbol{x}_{\bar{\mathbb{S}}_0}^*$

  À partir de l'équation (3.10d), on a $\boldsymbol{\pi}_{\bar{\mathbb{S}}_0}^* = \mathbf{0}$ pour les variables nulles (ces variables ne saturent pas les contraintes de borne). L'équation (3.10a) s'écrit alors :

  $$\mathbf{0} \in \nabla J(\boldsymbol{x}_{\bar{\mathbb{S}}_0}^*) + \lambda \partial p(\boldsymbol{x}_{\bar{\mathbb{S}}_0}^*)$$

  et, par l'équation (3.8), on a $\partial p(\boldsymbol{x}_{\bar{\mathbb{S}}_0}^*) = [-1, 1]^{\bar{n}}$. Les conditions d'optimalité de $\boldsymbol{x}_{\bar{\mathbb{S}}_0}^*$ deviennent donc :

  $$|\boldsymbol{c}_{\bar{\mathbb{S}}_0}| < \boldsymbol{\lambda}, \tag{3.13a}$$

  où $\boldsymbol{\lambda}$ est le vecteur colonne de taille appropriée dont chaque coordonnée vaut λ et l'inégalité est à considérer variable par variable.

- Cas 2 : Conditions d'optimalité de $\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}$

À partir de l'équation (3.10c), on a $\boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square} \geq \mathbf{0}$ pour les variables, indexées par $\bar{\mathbb{S}}_\square$, qui sont à la borne $\pm M$. L'équation (3.10a) s'écrit alors :

$$\mathbf{0} \in \nabla J(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) + \lambda \partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) + \partial \sum_{q \in \bar{\mathbb{S}}_\square} \pi^*_q g_q(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square})$$

et, par les équations (3.8) et (3.9), on a :

$$\begin{cases} \partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) & = \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) \\ \partial g_q(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) & = \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) \end{cases}.$$

Le sous-différentiel $\partial \sum_{q \in \bar{\mathbb{S}}_\square} \pi^*_q g_q(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square})$ peut alors s'écrire sous la forme vectorielle suivante

$$\partial \sum_{q \in \bar{\mathbb{S}}_\square} \pi^*_q g_q(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) = \boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}),$$

où $\odot$ désigne le produit de Hadamard (le vecteur résultant du produit terme à terme). Ainsi, les conditions d'optimalité de $\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}$ deviennent :

$$\begin{aligned} \boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square} \geq 0, \quad \text{avec} \quad & -\boldsymbol{c}_{\bar{\mathbb{S}}_\square} + \lambda \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) + \boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) = 0 \\ & \Leftrightarrow \ \boldsymbol{c}_{\bar{\mathbb{S}}_\square} = \lambda \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) + \boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) \\ & \Leftrightarrow \ \boldsymbol{c}_{\bar{\mathbb{S}}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) = \boldsymbol{\lambda} + \boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square}. \end{aligned} \tag{3.13b}$$

La positivité de $\boldsymbol{\pi}^*_{\bar{\mathbb{S}}_\square}$ s'écrit alors :

$$\boldsymbol{c}_{\bar{\mathbb{S}}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square}) \geq \boldsymbol{\lambda}. \tag{3.13c}$$

- Cas 3 : Conditions d'optimalité de $\boldsymbol{x}^*_{\mathbb{S}_\square}$

À partir de l'équation (3.10a), on a $\boldsymbol{\pi}^*_{\mathbb{S}_\square} \geq 0$ pour les variables non nulles qui sont à la borne $\pm M$. L'équation (3.9) s'écrit alors :

$$\mathbf{0} \in \nabla J(\boldsymbol{x}^*_{\mathbb{S}_\square}) + \lambda \partial p(\boldsymbol{x}^*_{\mathbb{S}_\square}) + \partial \sum_{q \in \mathbb{S}_\square} \pi^*_q g_q(\boldsymbol{x}^*_{\mathbb{S}_\square})$$

et, par les équations (3.8) et (3.9), on a

$$\begin{cases} \partial p(\boldsymbol{x}^*_{\mathbb{S}_\square}) & = \mathbf{0} \\ \partial g_q(\boldsymbol{x}^*_{\mathbb{S}_\square}) & = \operatorname{sgn}(\boldsymbol{x}^*_{\mathbb{S}_\square}) \end{cases}.$$

Les conditions d'optimalité de $\boldsymbol{x}^*_{\mathbb{S}_\square}$ deviennent donc :

$$\begin{aligned} \boldsymbol{\pi}^*_{\mathbb{S}_\square} \geq 0, \quad \text{avec} \qquad & -\boldsymbol{c}_{\mathbb{S}_\square} + \boldsymbol{\pi}^*_{\mathbb{S}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\mathbb{S}_\square}) = 0 \\ & \Leftrightarrow \boldsymbol{c}_{\mathbb{S}_\square} = \boldsymbol{\pi}^*_{\mathbb{S}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\mathbb{S}_\square}) \\ & \Leftrightarrow \boldsymbol{c}_{\mathbb{S}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\mathbb{S}_\square}) = \boldsymbol{\pi}^*_{\mathbb{S}_\square}. \end{aligned} \tag{3.13d}$$

La positivité de $\boldsymbol{\pi}^*_{\mathbb{S}_\square}$ s'écrit alors :

$$\boldsymbol{c}_{\mathbb{S}_\square} \odot \operatorname{sgn}(\boldsymbol{x}^*_{\mathbb{S}_\square}) \geq 0. \tag{3.13e}$$

- Cas 4 : Conditions d'optimalité de $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$

  À partir de l'équation (3.10d), on a $\boldsymbol{\pi}^*_{\bar{\mathbb{S}}_{in}} = \mathbf{0}$ pour les variables non nulles qui ne sont pas à la borne. L'équation (3.10a) s'écrit :

$$\mathbf{0} \in \nabla J(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) + \lambda \partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})$$

  et, par l'équation (3.8), on a $\partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})$. Ainsi, les conditions d'optimalité de $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$ deviennent :

$$\boldsymbol{c}_{\bar{\mathbb{S}}_{in}} = \lambda \operatorname{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}). \tag{3.13f}$$

- Cas 5 : Conditions d'optimalité de $\boldsymbol{x}^*_{\mathbb{S}_{in}}$

  À partir de l'équation (3.10d), on a $\boldsymbol{\pi}^*_{\mathbb{S}_{in}} = 0$ pour les variables non nulles qui ne sont pas à la borne. L'équation (3.10a) s'écrit

$$\mathbf{0} \in \nabla J(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) + \lambda \partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})$$

  et, d'après l'équation (3.8), on a $\partial p(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = \mathbf{0}$. Ainsi, les conditions d'optimalité de $\boldsymbol{x}^*_{\mathbb{S}_{in}}$ deviennent :

$$\boldsymbol{c}_{\mathbb{S}_{in}} = 0. \tag{3.13g}$$

Notons que les équations (3.13f) et (3.13g) sont des systèmes linéaires couplés en $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$ et

$\boldsymbol{x}^*_{\mathbb{S}_{in}}$, que l'on peut réécrire sous la forme suivante :

$$\begin{cases} \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}\boldsymbol{x}^*_{\mathbb{S}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = \lambda \text{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) \\ \mathbf{H}^T_{\mathbb{S}_{in}}(\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}\boldsymbol{x}^*_{\mathbb{S}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0 \end{cases},$$

en ayant introduit le vecteur constant $\underline{\boldsymbol{r}} := \boldsymbol{y} - \mathbf{H}_{\bar{\mathbb{S}}_0}\boldsymbol{x}^*_{\bar{\mathbb{S}}_0} - \mathbf{H}_{\bar{\mathbb{S}}_\square}\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square} - \mathbf{H}_{\mathbb{S}_\square}\boldsymbol{x}^*_{\mathbb{S}_\square}$, représentant le résidu engendré par les variables fixées (les variables dans $\bar{\mathbb{S}}_\square$ et $\mathbb{S}_\square$ sont fixées à $\pm M$ et les variables dans $\bar{\mathbb{S}}_0$ sont nulles). La solution de ces systèmes est alors donnée par (les détails de calcul sont présentés dans l'annexe A) :

$$\begin{cases} \boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} = \left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I} - \mathbf{P}^{\mathbb{S}_{in}})\mathbf{H}_{\bar{\mathbb{S}}_{in}}\right)^{-1}\left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I} - \mathbf{P}^{\mathbb{S}_{in}})\underline{\boldsymbol{r}} - \lambda\ \text{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})\right) & \text{(3.14a)} \\ \boldsymbol{x}^*_{\mathbb{S}_{in}} = (\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}(\mathbf{H}^T_{\mathbb{S}_{in}}\underline{\boldsymbol{r}} - \mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}), & \text{(3.14b)} \end{cases}$$

où $\mathbf{P}^{\mathbb{S}_{in}} := \mathbf{H}_{\mathbb{S}_{in}}(\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}\mathbf{H}^T_{\mathbb{S}_{in}}$ est la matrice de projection sur le sous-espace engendré par les colonnes de $\mathbf{H}_{\mathbb{S}_{in}}$ et $\mathbf{I}$ est la matrice identité de taille appropriée.

3.3 Méthode homotopique pour la résolution des trois problèmes relâchés $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$

La méthode homotopique [Osborne et al., 2000; Efron et al., 2004; Malioutov et al., 2005; Donoho and Tsaig, 2008], telle que décrite dans sa version standard pour le problème du LASSO au § 1.3.1, considère la forme pénalisée. Nous adaptons ici l'algorithme d'homotopie pour résoudre les problèmes $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$. La méthode d'homotopie pour le problème du LASSO avec contraintes de borne a été récemment proposée dans [Liang and Wang, 2017], mais l'inclusion de variables "libres" (les variables qui ne sont pas pénalisées par la norme ℓ_1) nécessite des tests supplémentaires à effectuer à chaque itération et a également un impact sur l'initialisation.

La méthode homotopique, pour notre problème pénalisé $\hat{\mathcal{Q}}_{2+1}$ (reformulé avec les notations de l'équation (3.4)), permet d'obtenir les solutions pour toutes les valeurs possibles du paramètre de régularisation λ :

$$\boldsymbol{x}^{*(\lambda)} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n}\ F(\boldsymbol{x}, \lambda) \quad \text{s.c } \|\boldsymbol{x}\|_\infty \leq M. \tag{3.15}$$

Cet ensemble de solutions est souvent appelé chemin de solution (ou chemin de régula-

risation). La méthode d'homotopie exploite le fait que le chemin de solution est linéaire par morceaux en fonction de λ. En effet, pour une configuration donnée du support $\{\bar{\mathbb{S}}_{in}, \mathbb{S}_{in}, \bar{\mathbb{S}}_{\square}, \mathbb{S}_{\square}, \bar{\mathbb{S}}_0\}$, la solution du problème (3.4) est linéaire en λ – les équations (3.14a) et (3.14b) le montrent clairement pour les variables $\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}$ et $\boldsymbol{x}_{\mathbb{S}_{in}}$, les autres variables étant constantes. Nous détaillons ci-après les différentes étapes permettant de généraliser la procédure d'homotopie à notre contexte.

3.3.1 Initialisation

Lorsque $\lambda \to +\infty$, les variables $\boldsymbol{x}_{\bar{\mathbb{S}}}^*$ pénalisées par la norme ℓ_1 sont nulles. Dans ce cas, les autres variables $\boldsymbol{x}_{\mathbb{S}_1}^*$ sont obtenues en résolvant le problème des moindres carrés sous contrainte de borne : $\min_{-M \leq \boldsymbol{x}_{\mathbb{S}_1} \leq M} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}\|_2^2$. Notons $\boldsymbol{x}^{(0)}$ le vecteur défini par

$$\begin{cases} \boldsymbol{x}_{\mathbb{S}_1}^{(0)} := \arg\min_{\boldsymbol{x}_{\mathbb{S}_1}} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}\|_2^2 \ \text{ s.c } \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M & \text{(3.16a)} \\ \boldsymbol{x}_{\bar{\mathbb{S}}}^{(0)} := \mathbf{0}. & \text{(3.16b)} \end{cases}$$

Plus précisément, l'équation (3.13a) montre que $\boldsymbol{x}_{\bar{\mathbb{S}}}^* = \mathbf{0}$ lorsque :

$$|\mathbf{H}_{\bar{\mathbb{S}}}^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}^{(0)})| < \lambda.$$

Ainsi, $\boldsymbol{x}^{(0)}$ est le minimiseur de $F(\boldsymbol{x}, \lambda)$ quelque soit $\lambda \geq \lambda^{(0)}$, avec :

$$\lambda^{(0)} := \|\mathbf{H}_{\bar{\mathbb{S}}}^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1}^{(0)})\|_\infty. \tag{3.16c}$$

À partir du point initial $\lambda^{(0)}$, la méthode d'homotopie construit récursivement le chemin de solutions en identifiant itérativement les différents points de rupture qui conduisent à des changements dans la configuration du support (voir l'équation (3.11) pour la définition du support).

3.3.2 Mise à jour récursive de la solution

Quand la valeur de λ devient inférieure à $\lambda^{(0)}$, les indices $j \in \bar{\mathbb{S}}$ pour lesquels $|\boldsymbol{h}_j^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}^T\boldsymbol{x}_{\mathbb{S}_1}^{(0)})| = \lambda^{(0)}$ quittent $\bar{\mathbb{S}}_0$ pour créer le nouveau sous-ensemble $\bar{\mathbb{S}}_{in}$. Cette nouvelle configuration de support reste valide pour tout $\lambda \in [\lambda^{(1)}, \lambda^{(0)}]$, où $\lambda^{(1)}$ définit le prochain point de rupture, *etc.* Une séquence décroissante $\{\lambda^{(k)}\}_k$ est ainsi construite

itérativement (avec les solutions optimales associées $\boldsymbol{x}^{*(\lambda^{(k)})}$ simplifiées en $\boldsymbol{x}^{(k)}$ par la suite), en testant tous les changements possibles qui peuvent se produire sur la configuration du support et en sélectionnant les changements correspondant à la plus petite diminution de λ. Ensuite, la configuration du support est mise à jour, et un nouveau point de rupture est recherché. Puisque le chemin de la solution est linéaire par morceaux en fonction de λ, la solution $\boldsymbol{x}^{(k)}$ au $k^{\text{ème}}$ point de rupture peut s'écrire :

$$\begin{cases} \boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \gamma^{(k)}\boldsymbol{d}^{(k)} & \text{(3.17a)} \\ \text{et } \lambda^{(k)} = \lambda^{(k-1)} - \gamma^{(k)}, & \text{(3.17b)} \end{cases}$$

où $\boldsymbol{d}^{(k)}$ représente le vecteur des changements de pente et $\gamma^{(k)} > 0$ est la longueur de l'intervalle $[\lambda^{(k)}, \lambda^{(k-1)}]$. À partir des équations (3.14a) et (3.14b), la direction $\boldsymbol{d}^{(k)}$ est obtenue par :

$$\begin{cases} \boldsymbol{d}^{(k)}_{\bar{\mathbb{S}}_{in}} = (\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I} - \mathbf{P}^{\mathbb{S}_{in}})\mathbf{H}_{\bar{\mathbb{S}}_{in}})^{-1}\text{sgn}(\boldsymbol{x}^{(k-1)}_{\bar{\mathbb{S}}_{in}}) & \text{(3.18a)} \\ \boldsymbol{d}^{(k)}_{\mathbb{S}_{in}} = -(\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{d}^{(k)}_{\bar{\mathbb{S}}_{in}} & \text{(3.18b)} \\ d^{(k)}_q = 0 \ \forall q \notin \{\bar{\mathbb{S}}_{in} \cup \mathbb{S}_{in}\}, & \text{(3.18c)} \end{cases}$$

où la dernière égalité concerne les variables qui sont fixées à zéro ou à $\pm M$ et $\mathbf{P}^{\mathbb{S}_{in}}$ est la matrice de projection sur le sous-espace engendré par les colonnes de $\mathbf{H}_{\mathbb{S}_{in}}$, définie en fin de Section 3.2.

3.3.3 Calcul de longueur de pas

La longueur de pas $\gamma^{(k)}$ est obtenue comme la plus petite valeur positive $\gamma > 0$ telle que $\boldsymbol{x}^{(k-1)} + \gamma\boldsymbol{d}^{(k)}$ atteint un nouveau point de rupture. Ces points de rupture se produiront à des valeurs spécifiques de λ, pour lesquelles (au moins) l'une des conditions des équations (3.13a)–(3.13g) est violée. Nous introduisons les notations suivantes :

$$\begin{aligned} \boldsymbol{t}^{(k-1)} &:= \boldsymbol{y} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}^{(k-1)}_{\bar{\mathbb{S}}} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}^{(k-1)}_{\mathbb{S}_1}, \\ \boldsymbol{v}^{(k-1)} &:= \mathbf{H}^T\boldsymbol{t}^{(k-1)}, \\ \boldsymbol{u}^{(k)} &:= \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{d}^{(k)}_{\bar{\mathbb{S}}_{in}} + \mathbf{H}_{\mathbb{S}_1}\boldsymbol{d}^{(k)}_{\mathbb{S}_1}, \\ \boldsymbol{w}^{(k)} &:= \mathbf{H}^T\boldsymbol{u}^{(k)}. \end{aligned} \tag{3.19}$$

Cinq cas différents peuvent se produire, que nous détaillons ci-après.

1. *Une composante d'indice $\ell \in \bar{\mathbb{S}}_{in}$ devient nulle.* D'après l'équation (3.17a),

$$x_\ell^{(k)} = x_\ell^{(k-1)} + \gamma d_\ell^{(k)} = 0$$

peut se produire lorsque :

$$\gamma = \frac{-x_\ell^{(k-1)}}{d_\ell^{(k)}}. \tag{3.20a}$$

2. *Une composante d'indice $\ell \in \bar{\mathbb{S}}_{in}$ ou $\ell \in \mathbb{S}_{in}$ atteint la borne M ou $-M$, selon son signe courant* (par construction, il n'y a pas de changement de signe entre $x_\ell^{(k-1)}$ et $x_\ell^{(k)}$). À partir de (3.17a),

$$x_\ell^{(k)} = x_\ell^{(k-1)} + \gamma d_\ell^{(k)} = M\mathrm{sgn}(x_\ell^{(k-1)})$$

peut se produire lorsque :

$$\gamma = \frac{M\mathrm{sgn}(x_\ell^{(k-1)}) - x_\ell^{(k-1)}}{d_\ell^{(k)}}. \tag{3.20b}$$

3. *Une composante d'indice $\ell \in \bar{\mathbb{S}}_0$ devient non nulle* lorsque l'égalité dans l'équation (3.13a) est atteinte. En insérant les équations (3.14a) et (3.14b) dans l'équation (3.13a), on peut montrer qu'elle peut devenir positive (respectivement, négative) quand :

$$\gamma = \frac{\lambda^{(k-1)} + v_\ell^{(k-1)}}{1 - w_\ell^{(k)}} \left(\text{respectivement, quand } \gamma = \frac{-\lambda^{(k-1)} + v_\ell^{(k-1)}}{-1 - w_\ell^{(k)}}\right). \tag{3.20c}$$

Les détails de calcul sont présentés dans l'annexe A.

4. *La contrainte de borne pour une composante d'indice $\ell \in \mathbb{S}_\square$ devient inactive.* Cela se produit lorsque le multiplicateur de Lagrange correspondant π_ℓ dans l'équation (3.13d) devient nul, ce qui donne :

$$\gamma = \frac{-v_\ell^{(k-1)}}{w_\ell^{(k)}}. \tag{3.20d}$$

Les détails de calcul sont présentés dans l'annexe A.

5. *La contrainte de borne pour une composante avec indice $\ell \in \bar{\mathbb{S}}_\square$ devient inactive.* Cela se produit lorsque le multiplicateur de Lagrange correspondant π_ℓ dans l'équa-

tion (3.13b), devient nul, ce qui donne :

$$\gamma = \frac{\operatorname{sgn}(x_\ell^{(k-1)})\lambda^{(k-1)} - v_\ell^{(k-1)}}{\operatorname{sgn}(x_\ell^{(k-1)}) - w_\ell^{(k)}}. \tag{3.20e}$$

Les détails de calcul sont présentés dans l'annexe A.

Le pas choisi $\gamma^{(k)}$ est alors le pas positif le plus petit parmi tous les pas possibles définis par les équations (3.20a)–(3.20e). En théorie, celui-ci peut être obtenu par plusieurs conditions simultanément ; si cela se produit, la configuration du support est mise à jour en conséquence.

3.3.4 Solution pour le problème pénalisé $\hat{\mathcal{Q}}_{2+1}$

Pour résoudre le problème $\hat{\mathcal{Q}}_{2+1}$ de l'équation (3.3), l'algorithme s'arrête lorsque la valeur cible du paramètre de pénalisation, λ_c est atteinte, c'est-à-dire après la $k^{\text{ème}}$ itération telle que $\lambda_c \in [\lambda^{(k)}, \lambda^{(k-1)}]$. Ensuite, la solution étant linéaire en λ sur cet intervalle, la solution optimale $\boldsymbol{x}^R$ pour $\lambda = \lambda_c$ est déterminée par :

$$\boldsymbol{x}^R = \boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}, \tag{3.21}$$

avec $\gamma^\star = \lambda_c - \lambda^{(k)}$.

Notre algorithme d'homotopie est résumé dans l'algorithme 4. La figure 3.2 montre un chemin de solution typique pour un exemple avec 5 variables : $\bar{\mathbb{S}} = \{1, 2, 3\}$ et $\mathbb{S}_1 = \{4, 5\}$.

Soit $k = 0$. Initialiser $\lambda^{(0)}$, $\boldsymbol{x}^{(0)}$ par les équations (3.16a)–(3.16c).

tant que $\lambda^{(k)} > \lambda_c$ **faire**

- $k \leftarrow k + 1$
- Mettre à jour $\boldsymbol{d}^{(k)}$ par les équations (3.18a)–(3.18b).
- Déterminer la taille de pas $\gamma^{(k)}$ comme la plus petite valeur positive calculée à partir des équations (3.20a)–(3.20e).
- Calculer $(\boldsymbol{x}^{(k)}, \lambda^{(k)})$ par les équations (3.17a)–(3.17b).
- Mettre à jour les ensembles d'indices $\{\bar{\mathbb{S}}_{in}, \mathbb{S}_{in}, \bar{\mathbb{S}}_\square, \mathbb{S}_\square, \bar{\mathbb{S}}_0\}$.

fin

Calculer $\boldsymbol{x}^R$ par l'équation (3.21).

Algorithme 4 : Algorithme homotopique pour résoudre le problème $\hat{\mathcal{Q}}_{2+1}$ avec $\lambda = \lambda_c$.

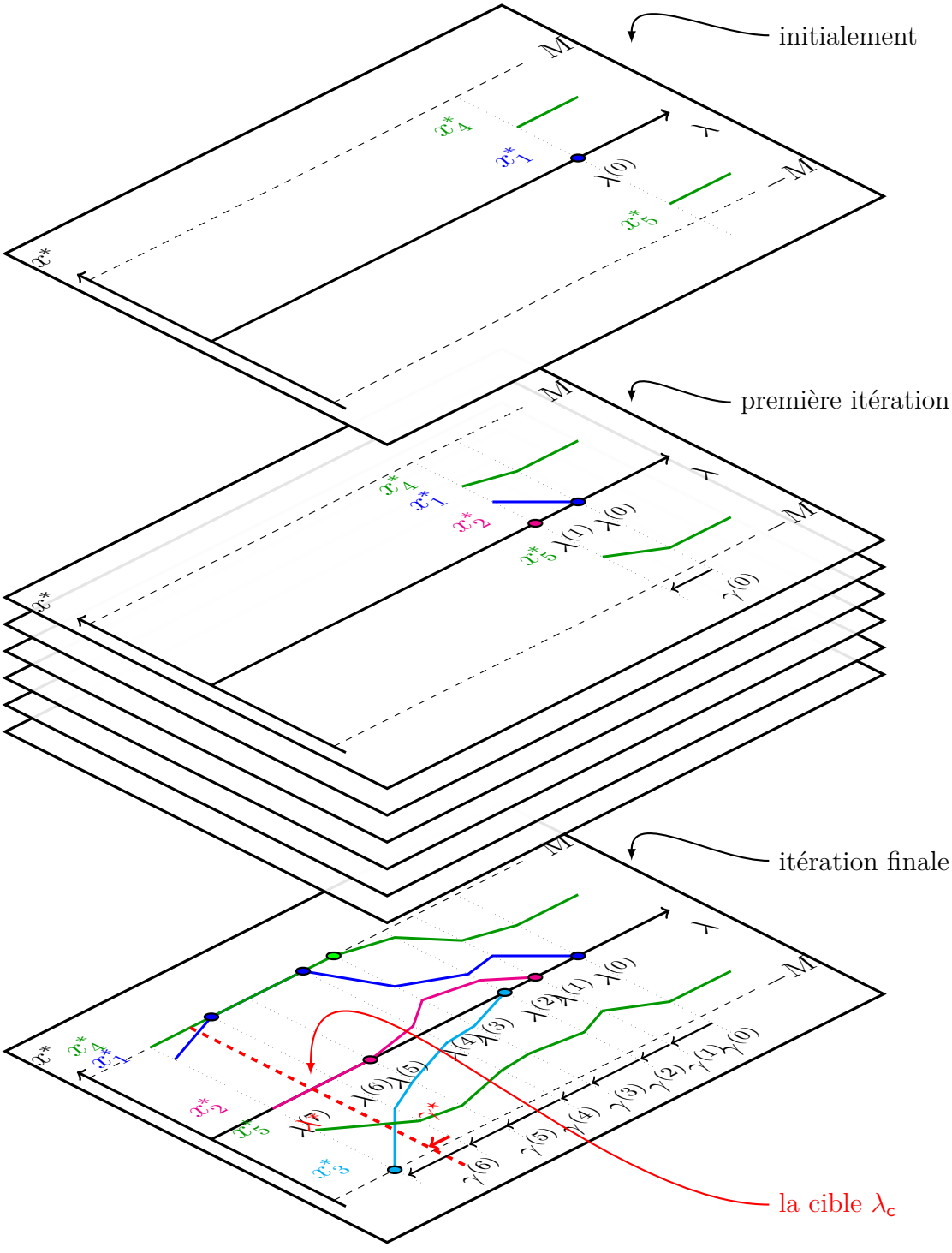


FIGURE 3.2 – Exemple de chemin de solution donné par la solution $\boldsymbol{x}^*$ du problème (3.4) en fonction de λ, avec 5 variables : $\bar{\mathbb{S}} = \{1, 2, 3\}$ et $\mathbb{S}_1 = \{4, 5\}$. Chaque plan correspond à une nouvelle partie du chemin $\lambda \in [\lambda^{(k)}, \lambda^{(k-1)}]$. Les cercles représentent les événements provoquant un changement dans la configuration du support. Les lignes pointillées verticales représentent les points de rupture.

3.3.5 Solutions pour les problèmes contraints $\hat{\mathcal{Q}}_{2/1}$ et $\hat{\mathcal{Q}}_{1/2}$

Les deux termes de la fonction objectif $\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}^*_{\bar{\mathbb{S}}}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}^*_{\mathbb{S}_1}\|_2^2$ et $\|\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_1$ étant convexes, lorsque λ diminue de façon continue, la norme ℓ_1 des variables pénalisées $\|\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_1$ augmente continûment et la fonction des moindres carrés $\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}^*_{\bar{\mathbb{S}}}-\mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}^*_{\mathbb{S}_1}\|_2^2$ diminue continûment. Pour cette raison, la méthode homotopique peut également résoudre les problèmes contraints $\hat{\mathcal{Q}}_{2/1}$ et $\hat{\mathcal{Q}}_{1/2}$, en s'arrêtant lorsque la valeur du paramètre correspondant τ_c ou ϵ_c est atteinte (voir Figure 3.3). Plus précisément :

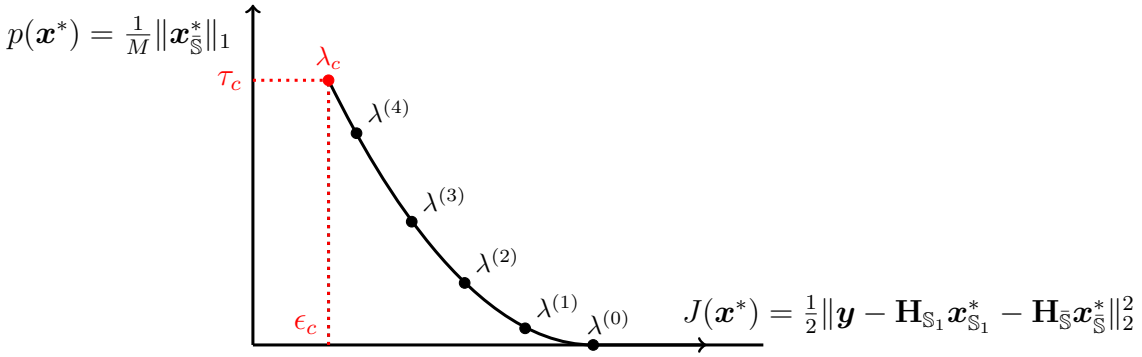


FIGURE 3.3 – Front de Pareto : Ensemble de solutions optimales $\left(\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^*\|_2^2,\ \frac{1}{M}\|\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_1\right)$ en fonction de λ, et illustration d'un critère d'arrêt pour l'algorithme d'homotopie.

— Pour $\hat{\mathcal{Q}}_{2/1}$, l'algorithme s'arrête au premier point de rupture tel que la norme ℓ_1 des variables pénalisées $\|\boldsymbol{x}^{(k)}_{\bar{\mathbb{S}}}\|_1$ dépasse la valeur $\tau_c := M(K-n_1)$. Ensuite, dans l'intervalle correspondant $[\lambda^{(k)}, \lambda^{(k-1)}]$, la solution est donnée par l'équation (3.21). Par construction, il n'y a pas de changement de signe entre $\boldsymbol{x}^{(k-1)}$ et la solution optimale $\boldsymbol{x}^R$ telle que $\|\boldsymbol{x}^R\|_1 = \tau_c$. Ainsi, la valeur de $\gamma^\star$ telle que

$$\|\boldsymbol{x}^R\|_1 = \tau_c \quad \Leftrightarrow \|\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}\|_1 = \tau_c \quad ,$$

est donnée par :

$$\gamma^\star := \frac{\tau_c - \|\boldsymbol{x}^{(k-1)}\|_1}{\operatorname{sgn}(\boldsymbol{x}^{(k-1)})^T \boldsymbol{d}^{(k)}}. \tag{3.22}$$

Les détails de calcul sont présentés dans l'annexe A.

— De même, pour $\hat{\mathcal{Q}}_{1/2}$, l'algorithme s'arrête au premier point de rupture tel que $\frac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}^{(k)}\|_2^2 \leq \epsilon_c$. En remplaçant l'équation (3.21) dans l'expression des moindres

carrés, la valeur de $\gamma^\star$ telle que

$$\tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^R\|_2^2 = \epsilon_c \Leftrightarrow \|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^{(k-1)} + \gamma^\star \mathbf{H}\boldsymbol{d}^{(k)}\|_2^2 = 2\epsilon_c$$

peut être trouvée en résolvant une équation quadratique scalaire (les détails de calcul sont présentés dans l'annexe A), dont la solution est :

$$\gamma^* := \frac{\boldsymbol{t}^{(k-1)^T}\boldsymbol{u}^{(k)} - \sqrt{\left(\boldsymbol{t}^{(k-1)^T}\boldsymbol{u}^{(k)}\right)^2 - \boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}(\boldsymbol{t}^{(k-1)^T}\boldsymbol{t}^{(k-1)} - 2\epsilon_c)}}{\boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}}, \tag{3.23}$$

où $\boldsymbol{t}^{(k-1)}$ et $\boldsymbol{u}^{(k)}$ sont définis dans l'équation (3.19).

3.3.6 Mise en œuvre et aspects pratiques

Cette section détaille quelques aspects pratiques de la mise en œuvre numérique de l'algorithme d'homotopie.

Mise à jour récursive. À chaque itération de la procédure, la majeure partie du temps de calcul est consacrée à la résolution des systèmes linéaires donnés par les équations (3.18a)–(3.18b), dont la taille correspond respectivement à celle de $\bar{\mathbb{S}}_{in}$ et $\mathbb{S}_{in}$. Puisque la configuration de support ne change que par une composante entre deux points de rupture, la matrice inverse des équations (3.18a)–(3.18b) peut être calculée de manière récursive en effectuant des mises à jour de rang 1. Pour cela, nous utilisons une stratégie rapide basée sur le lemme d'inversion des matrices partitionnées [Hager, 1989]. Cette stratégie nous est apparue la plus efficace par rapport à d'autres techniques comme la factorisation de Cholesky, où il est plus compliqué de gérer les retraits (voir Annexe A).

Réduction du nombre de tests. Nous notons également que, pour chaque composante non nulle avec indice $\ell \in \bar{\mathbb{S}}_{in}$, seul un des deux tests définis par les équations (3.20a) et (3.20b) est nécessaire : selon la signe de la pente associée, la composante peut soit atteindre la borne $\pm M$ soit devenir nulle. Par exemple, si $x_\ell^{(k-1)} > 0$ et si l'amplitude augmente lorsque λ diminue ($d_\ell^{(k)} > 0$), alors le seul changement possible du support est lorsque x_ℓ atteint la borne supérieure M : cette composante ne peut devenir nulle. Ainsi, au paragraphe 3.3.3, seul le cas (2) doit être envisagé et non le cas (1).

Cas particulier de $\hat{\mathcal{Q}}_{1/2}$ dans l'algorithme branch-and-bound. Nous concluons cette section par deux remarques importantes concernant la résolution du problème $\hat{\mathcal{Q}}_{1/2}$ par la méthode homotopique. Rappelons que le problème $\hat{\mathcal{Q}}_{1/2}$ est la relaxation continue permettant de calculer la borne inférieure (notée z_L) au sein de l'algorithme branch-and-bound pour la reformulation en MIP $\hat{\mathcal{P}}_{0/2}$ (voir Section 2.3). Dans l'algorithme branch-and-bound, cette borne inférieure est comparée à la borne supérieure globale z_U, et si z_L est supérieure à z_U, alors le nœud sera élagué (voir Section 2.2 pour le principe d'élagage de la méthode branch-and-bound). Par conséquent :

1. durant la résolution de $\hat{\mathcal{Q}}_{1/2}$ par homotopie, puisque $\|\boldsymbol{x}^{(k)}\|_1$ augmente au fil des itérations, on peut arrêter et élaguer le nœud dès que la valeur de $\|\boldsymbol{x}^{(k)}\|_1$ dépasse la borne supérieure z_U. Il est donc inutile de continuer jusqu'à la valeur cible ($\|\boldsymbol{x}^{(k)}\|_1 = \tau_c$) puisque la valeur de $\|\boldsymbol{x}^R\|_1$ sera toujours plus grande que z_U et le nœud sera élagué dans tous les cas.

2. Étant donné que z_U est entier (z_U est la norme ℓ_0 de $\boldsymbol{x}$), il est clair que le nœud peut être élagué dès que la borne inférieure dépasse $z_U - 1$.

La figure 3.4 illustre ces deux points.

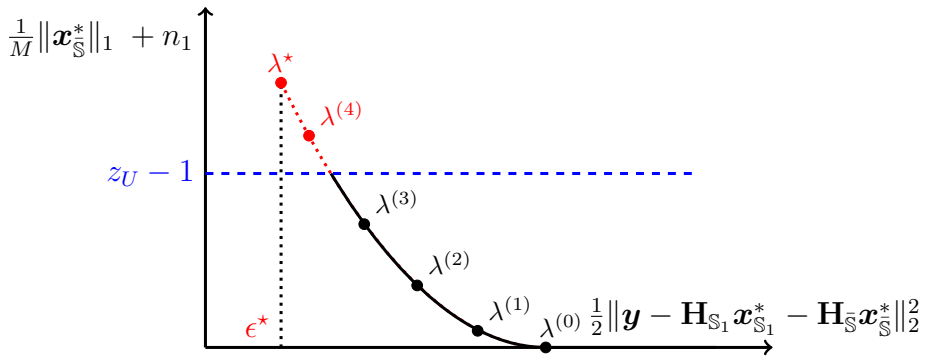


FIGURE 3.4 – Ensemble de solutions optimales $\left(\frac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}^*_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_2^2, \frac{1}{M}\|\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_1 + n_1\right)$ en fonction de λ, et illustration d'un critère d'arrêt avec l'algorithme d'homotopie pour la formulation $\hat{\mathcal{Q}}_{1/2}$. L'algorithme peut être arrêté (et le nœud correspondant sera élagué) dès que $\frac{1}{M}\|\boldsymbol{x}^*_{\bar{\mathbb{S}}}\|_1 + n_1 \geq z_U - 1$, où z_U est la borne supérieure (la norme ℓ_0 de la meilleure solution connue de $\hat{\mathcal{P}}_{0/2}$ à une itération donnée de l'algorithme branch-and-bound).

3.4 Algorithme d'ensemble actif pour le problème relâché $\hat{\mathcal{Q}}_{2+1}$ avec redémarrage à chaud

Le principe des méthodes de type *ensemble actif* pour résoudre un problème d'optimisation convexe sous contraintes [voir Nocedal and Wright, 2006, Section 16.5], est d'optimiser itérativement des sous-problèmes avec un ensemble de contraintes actives (appelé support des contraintes actives). La configuration du support est mise à jour à chaque itération dans une direction minimisant le critère, ce qui va assurer la convergence de l'algorithme. Pour le problème standard en norme ℓ_1 (voir l'équation (1.10)), de nombreux algorithmes d'optimisation de type *ensemble actif* dédiés ont été développés (voir par exemple la thèse [Loth, 2011] et ses références). Nous proposons de résoudre le problème relâché impliqué dans la forme pénalisée $\hat{\mathcal{Q}}_{2+1}$, par une généralisation d'un algorithme de type *ensemble actif*, appelé *feature-sign search* dans [Lee et al., 2007], initialement proposé pour le cas standard des moindres carrés pénalisés par la norme ℓ_1. L'avantage de cet algorithme est sa capacité de démarrage à chaud en partant de n'importe quelle solution, contrairement à la procédure homotopique qui ne peut être initialisée qu'en considérant $\boldsymbol{x}_{\bar{\mathbb{S}}} = \mathbf{0}$. En revanche, il est dédié au problème pénalisé (1.12) et ainsi le même algorithme ne permet pas de résoudre les formulations (1.10) et (1.11).

3.4.1 Principe de fonctionnement

Afin de lever toute ambiguïté, nous appelons ici *ensemble actif* l'ensemble des *variables actives* dans le problème, *i.e.,* non nulles et n'atteignant pas la borne (en effet, ces algorithmes sont parfois présentés en considérant l'ensemble des *contraintes actives* qui, à l'inverse, se réfèrent à l'ensemble des variables inactives). Plus précisément, notre algorithme d'ensemble actif considère séparément l'ensemble dit *actif* composé des variables non fixées pénalisées par la norme ℓ_1 (*i.e.*, $\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}$) et des variables non fixées non pénalisées (*i.e.*, $\boldsymbol{x}_{\mathbb{S}_{in}}$) et l'ensemble dit *inactif* composé des variables *fixées* ($\boldsymbol{x}_{\bar{\mathbb{S}}_{\square}}$, $\boldsymbol{x}_{\mathbb{S}_{\square}}$ et $\boldsymbol{x}_{\bar{\mathbb{S}}_0}$). En n'importe quel point $\boldsymbol{x}$, on définit, de la même façon que dans l'équation (3.11), un support comme suit :

$$\left\{\begin{array}{lll}
\left.\begin{array}{lll}
\boldsymbol{x}_{\bar{\mathbb{S}}_{in}} & := \{x_q \mid & \forall q \in \bar{\mathbb{S}}, \quad 0 < |x_q| < M\} \\
\boldsymbol{x}_{\mathbb{S}_{in}} & := \{x_q \mid & \forall q \in \mathbb{S}_1, \quad 0 \leq |x_q| < M\}
\end{array}\right\} \text{ensemble actif} \\
\left.\begin{array}{lll}
\boldsymbol{x}_{\bar{\mathbb{S}}_0} & := \{x_q \mid & \forall q \in \bar{\mathbb{S}}, \quad |x_q| = 0\} \\
\boldsymbol{x}_{\bar{\mathbb{S}}_{\square}} & := \{x_q \mid & \forall q \in \bar{\mathbb{S}}, \quad |x_q| = M\} \\
\boldsymbol{x}_{\mathbb{S}_{\square}} & := \{x_q \mid & \forall q \in \mathbb{S}_1, \quad |x_q| = M\}
\end{array}\right\} \text{ensemble inactif/fixé}
\end{array}\right. \tag{3.24}$$

L'optimisation se déroule en alternant entre deux étapes : *inactivation* et *activation de variable* jusqu'à trouver la solution optimale. Nous détaillons ci-après le fonctionnement de l'algorithme proposé. Rappelons le problème $\hat{\mathcal{Q}}_{2+1}$ que nous cherchons à résoudre :

$$\hat{\mathcal{Q}}_{2+1} : \min_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1},\boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}} \quad F(\boldsymbol{x}) := \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 + \lambda_c\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \quad \text{s.c.} \quad \|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_\infty \leq M \quad \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M \quad . \tag{3.25}$$

3.4.1.1 Recherche en ligne discrète et inactivation de variable

Considérons un point courant quelconque $\boldsymbol{x}$ et notons $\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}$ le vecteur signe (de valeurs -1 ou 1) des variables non nulles intervenant dans le terme en norme ℓ_1, de sorte que $\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 = \boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}^T\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}$. À partir des variables actives $\boldsymbol{x}_{\bar{\mathbb{S}}_{in}\cup\mathbb{S}_{in}}$, notre algorithme cherche à trouver la meilleure solution réalisable (*i.e.,* satisfaisant les contraintes de borne du problème $\hat{\mathcal{Q}}_{2+1}$), dont le signe est cohérent avec le vecteur signe $\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}$. Pour cela, on considère le critère à signe fixé :

$$F_s(\boldsymbol{x},\boldsymbol{\theta}) := \tfrac{1}{2}\|\underline{\boldsymbol{r}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}_{\bar{\mathbb{S}}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}_{\mathbb{S}_{in}}\|_2^2 + \lambda_c\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}^T\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}$$

où $\underline{\boldsymbol{r}} := \boldsymbol{y} - \mathbf{H}_{\bar{\mathbb{S}}_\square}\boldsymbol{x}_{\bar{\mathbb{S}}_\square} - \mathbf{H}_{\mathbb{S}_\square}\boldsymbol{x}_{\mathbb{S}_\square}$ est le vecteur constant représentant le résidu engendré par les variables fixées (les variables dans $\bar{\mathbb{S}}_\square$ et $\mathbb{S}_\square$ sont fixées à $\pm M$, et les variables dans $\bar{\mathbb{S}}_0$ sont nulles). Nous commençons par résoudre le problème non contraint en les variables actives :

$$(\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}^{\text{new}}, \boldsymbol{x}_{\mathbb{S}_{in}}^{\text{new}}) = \arg\min_{\boldsymbol{x}_{\bar{\mathbb{S}}_{in}},\boldsymbol{x}_{\mathbb{S}_{in}}} F_s(\boldsymbol{x},\boldsymbol{\theta}). \tag{3.26}$$

En exploitant les conditions d'optimalité décrites en Section 3.2 (en particulier, le système d'équations (3.14a) et (3.14b), les détails des calculs sont dans l'annexe A), la solution de ce problème est donnée par :

$$\left\{\begin{array}{rcl} \boldsymbol{x}_{\bar{\mathbb{S}}_{in}}^{\text{new}} &=& \left(\mathbf{H}_{\bar{\mathbb{S}}_{in}}^T\mathbf{B}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\right)^{-1}\left(\mathbf{H}_{\bar{\mathbb{S}}_{in}}^T\mathbf{B}\underline{\boldsymbol{r}} - \lambda_c\,\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}\right) \\ \boldsymbol{x}_{\mathbb{S}_{in}}^{\text{new}} &=& (\mathbf{H}_{\mathbb{S}_{in}}^T\mathbf{H}_{\mathbb{S}_{in}})^{-1}(\mathbf{H}_{\mathbb{S}_{in}}^T\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}^T\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}^{\text{new}}) \end{array}\right. \tag{3.27}$$

où $\mathbf{B} := \mathbf{I} - \mathbf{H}_{\mathbb{S}_{in}}(\mathbf{H}_{\mathbb{S}_{in}}^T\mathbf{H}_{\mathbb{S}_{in}})^{-1}\mathbf{H}_{\mathbb{S}_{in}}^T$. Nous complétons alors $\boldsymbol{x}^{\text{new}}$ en y adjoignant les variables inactives du point courant : $\boldsymbol{x}_{\bar{\mathbb{S}}_0}^{\text{new}} = \boldsymbol{x}_{\bar{\mathbb{S}}_0}$, $\boldsymbol{x}_{\bar{\mathbb{S}}_\square}^{\text{new}} = \boldsymbol{x}_{\bar{\mathbb{S}}_\square}$ et $\boldsymbol{x}_{\mathbb{S}_\square}^{\text{new}} = \boldsymbol{x}_{\mathbb{S}_\square}$.

Soient les deux conditions suivantes :

C_1 : le signe de $\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}^{\text{new}}$ est cohérent avec le vecteur signe $\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}$ $(\text{sgn}(\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}^{\text{new}}) = \boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}})$

C_2 : $\boldsymbol{x}^{\text{new}}$ satisfait la contrainte de borne ($\|\boldsymbol{x}^{\text{new}}\|_\infty \leq M$).

Nous distinguons alors deux cas de figure.

— Si les deux conditions (C_1) et (C_2) sont vérifiées, alors par construction $F(\boldsymbol{x}^{\text{new}}) \leq F(\boldsymbol{x})$ et $\boldsymbol{x}^{\text{new}}$ est réalisable pour le problème (3.25) ; $\boldsymbol{x}^{\text{new}}$ devient alors la solution actuelle.

— Sinon (on a alors $\boldsymbol{x}^{\text{new}} \neq \boldsymbol{x}$), une recherche en ligne est effectuée dans la direction de $\boldsymbol{x}$ à $\boldsymbol{x}^{\text{new}}$, reposant sur la Proposition suivante.

Proposition 4. *Il existe une solution*

$$\boldsymbol{x}^t := \boldsymbol{x} + t(\boldsymbol{x}^{new} - \boldsymbol{x}) \tag{3.28}$$

avec $t \in [0, 1]$, *telle que*

$$F(\boldsymbol{x}^t) < F(\boldsymbol{x}).$$

Démonstration. Par définition, pour $\boldsymbol{\theta}$ fixé, on a $F_s(\boldsymbol{x}^{\text{new}}, \boldsymbol{\theta}) \leq F_s(\boldsymbol{x}, \boldsymbol{\theta})$ puisque $\boldsymbol{x}^{\text{new}}$ est la solution optimale du problème (3.26). D'autre part, la fonction $t \to F_s(\boldsymbol{x}^t, \boldsymbol{\theta})$ est quadratique en t, minimale en $t = 1$ et décroissante en $t = 0$ et tant que $\text{sgn}(\boldsymbol{x}^t) = \boldsymbol{\theta}$ (le signe de $\boldsymbol{x}^t$ ne change pas), $F_s(\boldsymbol{x}^t, \boldsymbol{\theta}) = F(\boldsymbol{x}^t)$ (le vrai critère). Ainsi, F diminue en $t = 0$, et une recherche en ligne dans la direction de $\boldsymbol{x}^{\text{new}}$ réduit forcément le vrai critère. □

Recherche en ligne discrète

Nous décrivons ci-après l'étape de recherche en ligne, qui va parcourir l'ensemble des points $\boldsymbol{x}^t = \boldsymbol{x} + t(\boldsymbol{x}^{\text{new}} - \boldsymbol{x})$ où une coordonnée subit un changement de signe, pour $t \in [0, t^{\max}]$, où $t^{\max} \leq 1$ définit la valeur maximale de t telle que $\|\boldsymbol{x}^t\|_\infty \leq M$, que nous allons d'abord expliciter. On peut distinguer deux cas possibles :

1. Si $\text{sgn}(x_q) = \text{sgn}(x_q^{\text{new}})$, alors on a $\text{sgn}(x_q) = \text{sgn}(x_q^t)$. Ainsi

$$\begin{aligned} |x_q^t| = M \Leftrightarrow \quad x_q^t = M\text{sgn}(x_q) &\Leftrightarrow \quad x_q + t(x_q^{\text{new}} - x_q) = M\text{sgn}(x_q) \\ &\Leftrightarrow \quad t = (M\text{sgn}(x_q) - x_q)/(x_q^{\text{new}} - x_q). \end{aligned}$$

2. Si $\text{sgn}(x_q^{\text{new}}) \neq \text{sgn}(x_q)$, alors on a $\text{sgn}(x_q^t) = -\text{sgn}(x_q)$. Ainsi

$$\begin{aligned} x_q^t = -M\text{sgn}(x_q) \Leftrightarrow \quad & x_q + t(x_q^{\text{new}} - x_q) = -M\text{sgn}(x_q) \\ \Leftrightarrow \quad & t = (-M\text{sgn}(x_q) - x_q)/(x_q^{\text{new}} - x_q). \end{aligned}$$

La borne de l'intervalle $t^{\max}$ est donc obtenue par

$$t^{\max} = \min_{q \in \{\bar{\mathbb{S}}_{in}, \mathbb{S}_{in}\}} \min{}^+\{\frac{M - x_q}{x_q^{\text{new}} - x_q}, \frac{-M - x_q}{x_q^{\text{new}} - x_q}\}, \tag{3.29}$$

où $\min^+$ est la fonction minimum restreinte aux valeurs positives :

$$\min{}^+\{a, b\} = \begin{cases} \min\{a, b\} & \text{si} \quad a > 0 \text{ et } b > 0 \\ a & \text{si} \quad a > 0 \text{ et } b < 0 \\ b & \text{si} \quad a < 0 \text{ et } b > 0 \\ \emptyset & \text{si} \quad a < 0 \text{ et } b < 0 \end{cases}.$$

Notons $f(t)$ la valeur de la fonction objectif $F(\boldsymbol{x}^t)$. Par construction, f est continue et quadratique par morceaux sur $[0, t^{\max}]$ et sa forme change lorsqu'une composante de $\boldsymbol{x}^t$ change de signe. Une illustration est donnée en Figure 3.5. Il y a donc un nombre fini de points d'intérêt (où au moins un coefficient dans le support actif devient nul) où évaluer $f(t)$. Parmi ceux-ci, celui dont la valeur objective est la plus faible, soit $\boldsymbol{x}^{t^\star}$ avec

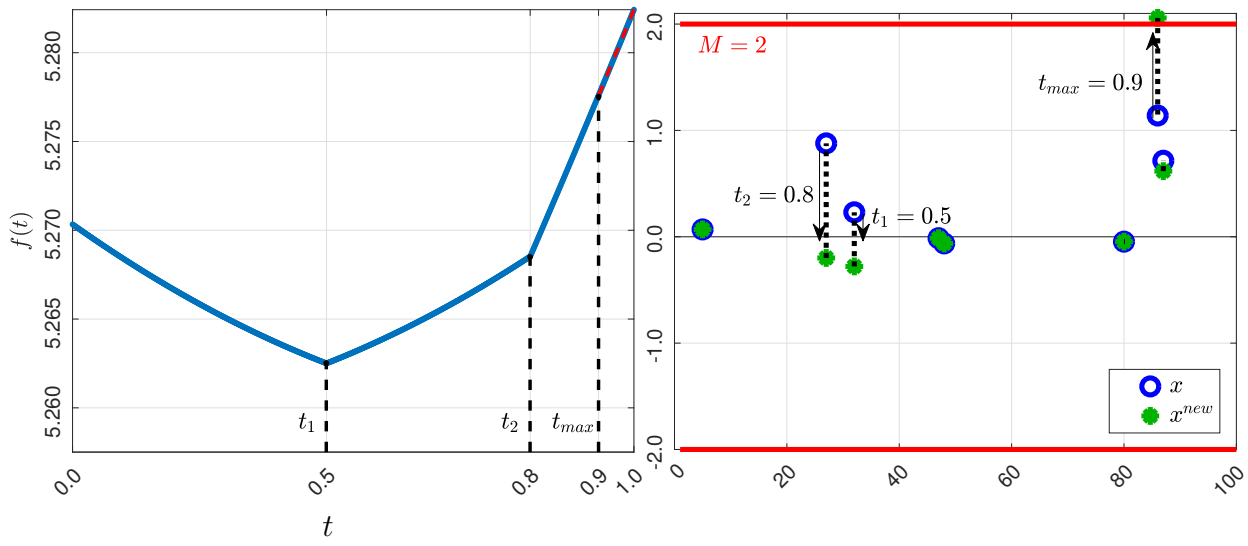
$$t^\star = \arg \min_{t \in [0, t^{\max}]} f(t),$$

est choisi comme nouvelle solution. Ensuite, les éléments qui sont devenus nuls (ou qui ont atteint la borne, si $t^\star = t^{\max}$), sont supprimés de l'ensemble actif.

Cette étape est répétée jusqu'à trouver un ensemble actif $\boldsymbol{x}_{\bar{\mathbb{S}}_{in}}$ ayant un signe $\boldsymbol{\theta}_{\bar{\mathbb{S}}_{in}}$ cohérent avec le signe de $\boldsymbol{x}^{\text{new}}_{\bar{\mathbb{S}}_{in}}$, la solution du problème non contraint. Une fois le problème résolu pour ce sous-ensemble actif et lorsque la solution vérifie la condition d'optimalité (C_2), alors l'ensemble actif est agrandi par l'étape *d'activation de variable*.

3.4.1.2 Activation de variable

Cette étape consiste à choisir l'élément de l'ensemble inactif violant le plus fortement les conditions d'optimalité des variables *fixées* ($\boldsymbol{x}_{\bar{\mathbb{S}}_0}$, $\boldsymbol{x}_{\bar{\mathbb{S}}_\square}$ et $\boldsymbol{x}_{\mathbb{S}_\square}$) pour l'ajouter à l'ensemble actif. Nous considérons séparément les trois cas.



a) fonction $f(t)$ b) chemin de solution $\boldsymbol{x}^t$

FIGURE 3.5 – Exemple de recherche en ligne : a) la fonction quadratique par morceaux $f(t)$ (en bleu) sur $[0, t^{\max}]$ avec $t^{\max} = 0.9$; b) le chemin de solution associé $\boldsymbol{x}^t$ (en pointillé) qui relie la solution actuelle $\boldsymbol{x}$ (en bleu) à la solution du problème non contraint $\boldsymbol{x}^{\text{new}}$ (en vert).

— Pour les variables $\boldsymbol{x}_{\bar{\mathbb{S}}_0}$ fixées à 0, l'élément violant le plus les conditions d'optimalité (3.13a) est choisi par :

$$j_0 = \arg\max{}^{+}_{j \in \bar{\mathbb{S}}_0} \ (|c_j| - \lambda_c). \tag{3.30}$$

où $\max^+$ est la fonction maximum sur les valeurs positives :

$$\max{}^{+}\{a, b\} = \begin{cases} \max\{a, b\} & \text{si} \quad a > 0 \text{ et } b > 0 \\ a & \text{si} \quad a > 0 \text{ et } b < 0 \\ b & \text{si} \quad a < 0 \text{ et } b > 0 \\ \emptyset & \text{si} \quad a < 0 \text{ et } b < 0 \end{cases}$$

et, pour rappel, $\boldsymbol{c} = \mathbf{H}^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}})$.

— Pour les variables pénalisées $\boldsymbol{x}_{\mathbb{S}_\square}$ fixées à la borne $\pm M$, l'élément violant le plus les conditions d'optimalité (3.13c) est choisi par :

$$j_1 = \arg\max_{j \in \bar{\mathbb{S}}_\square}{}^{+} \ (-\text{sgn}(x_j)c_j + \lambda_c). \tag{3.31}$$

— Pour les variables non pénalisées $\boldsymbol{x}_{\mathbb{S}_\square}$ fixées à la borne, l'élément violant le plus les conditions d'optimalité (3.13e) est choisi par :

$$j_2 = \arg\max_{j \in \mathbb{S}_\square}{}^{+} \ (-\text{sgn}(x_j)c_j). \tag{3.32}$$

Enfin, la variable d'indice $j^* = j_0 \ j_1$ ou j_2, engendrant l'écart maximal parmi les trois équations (3.30), (3.31) et (3.32), est choisie pour intégrer l'ensemble actif. La dernière étape consiste maintenant à *deviner* le signe associé à la nouvelle composante activée.

3.4.1.3 Signe de la variable activée

Pour les variables non nulles fixées à la borne $\pm M$, le signe correspondant est bien défini. Ainsi, si la variable d'indice j^* choisie pour intégrer l'ensemble actif appartient à $\bar{\mathbb{S}}_\square$ ou $\mathbb{S}_\square$, la variable garde son signe et $\theta_{j^*} = \text{sgn}(x_{j^*})$. Maintenant, le problème se pose lorsque la variable choisie j^* appartient à l'ensemble inactif des variables fixées à zéro, $\bar{\mathbb{S}}_0$. Dans ce cas, l'activation de cette variable nécessite une estimation de son signe. Nous reprenons une heuristique proposée dans l'algorithme *feature-sign search* de Lee et al. [2007], qui exploite le signe du gradient $\nabla J(x_{j^*})$ en x_{j^*} pour deviner son signe. En résumé, le signe de la variable activée est le suivant :

$$\theta_{j^*} = \begin{cases} \text{sgn}(x_{j^*}) & \text{si } j^* \in \{\bar{\mathbb{S}}_\square \text{ ou } \mathbb{S}_\square\} \\ \text{sgn}(-c_{j^*}) & \text{si } j^* \in \{\bar{\mathbb{S}}_0\} \end{cases} . \tag{3.33}$$

3.4.2 Redémarrage à chaud

Le démarrage à chaud signifie généralement l'utilisation d'une solution initiale choisie de manière à accélérer la résolution d'un problème donné. Dans la procédure de branch-and-bound, il s'agit d'initialiser la résolution du problème relâché à chaque nœud de l'arbre. Par construction, l'algorithme d'ensembles actifs décrit précédemment peut être initialisé en tout point satisfaisant les contraintes du problème. Dans le cas du problème pénalisé $\hat{\mathcal{Q}}_{2+1}$, les seules contraintes à prendre en compte sont les contraintes de borne, ainsi, à chaque nœud, tout point $\boldsymbol{x}^{(0)}$ tel que

$$\begin{cases} |\boldsymbol{x}^{(0)}_{\mathbb{S}}| & \leq & M \\ |\boldsymbol{x}^{(0)}_{\mathbb{S}_1}| & \leq & M \\ \boldsymbol{x}^{(0)}_{\mathbb{S}_0} & = & \mathbf{0} \end{cases}$$

peut être utilisé comme point de départ.

Souvent, il est utile de se servir des méthodes permettant le démarrage à chaud lorsque nous devons résoudre une famille de problèmes connexes. Si nous considérons que l'algorithme d'ensemble actif, pour la résolution de la relaxation continue, comme un schéma itératif, alors la progression de l'algorithme peut être mesurée approximativement comme la « distance par rapport à l'optimalité », c'est-à-dire le degré de violation des conditions d'optimalité. Plus le point de départ est proche de la solution recherchée, plus on peut espérer que la résolution et la convergence de l'algorithme seront rapides. De ce point de vue, nous choisissons d'*utiliser la solution du problème relâché du nœud parent comme initialisation pour les nœuds fils* car ce sont des problèmes très proches, voir le schéma de la méthode branch-and-bound en Section 2.2 : les deux nœuds fils ne diffèrent en effet du nœud parent que par une seule contrainte qui porte sur la variable choisie pour la séparation. Cette initialisation permet également d'exploiter des informations supplémentaires obtenues en sortie du nœud parent, comme les inverses de matrices intervenant dans les calculs de l'équation (3.27), qui seront utilisées à la première itération de l'algorithme.Nous testerons et illustrerons l'efficacité de cette idée dans la section 3.5.

3.4.3 Synthèse de l'algorithme

Le fonctionnement de notre algorithme d'ensemble actif est décrit dans l'Algorithme 5.

Par construction, l'algorithme converge vers un minimum global en un nombre fini d'itérations. Une preuve complète peut être trouvée dans [Lee et al., 2007] pour le cas standard du LASSO (moindres carrés avec pénalisation ℓ_1), qui s'étend sans problème aux spécificités de $F(\boldsymbol{x})$.

À chaque itération, la partie majeure du temps de calcul est consacrée à la résolution des systèmes linéaires dans (3.27), dont la taille correspond respectivement à celle de $\bar{\mathbb{S}}_{in}$ et $\mathbb{S}_{in}$ pour calculer $\boldsymbol{x}^{\text{new}}$. Puisque la configuration de support ne change que par une composante entre deux itérations, nous utilisons une implémentation astucieuse basée sur le lemme d'inversion des matrices partitionnées pour un calcul récursif plus rapide des matrices inverses comme pour l'algorithme d'homotopie (voir Section 3.3.6).

Initialisation : $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$; $\boldsymbol{\theta} \leftarrow \text{sgn}(\boldsymbol{x}^0)$; $\{\bar{\mathbb{S}}_{in}, \mathbb{S}_{in}, \bar{\mathbb{S}}_{\square}, \mathbb{S}_{\square}, \bar{\mathbb{S}}_0\}$ par (3.24).

Étape 1 :
début
  répéter
    calculer $\boldsymbol{x}^{\text{new}}$ par (3.27) ;
    calculer $t^\star = \arg\min_{t \in [0, t^{\max}]} f(t)$;
    mettre à jour la solution $\boldsymbol{x} \leftarrow \boldsymbol{x} + t^\star(\boldsymbol{x}^{\text{new}} - \boldsymbol{x})$;
    mettre à jour le signe $\boldsymbol{\theta} \leftarrow \text{sgn}(\boldsymbol{x})$;
    mettre à jour le support par (3.24) ;
  jusqu'à $sgn(\boldsymbol{x}) = sgn(\boldsymbol{x}^{new})$ *et* $\|\boldsymbol{x}^{new}\|_\infty \leq M$;
fin

Étape 2 :
début
  si *les conditions d'optimalité* (3.13a),(3.13e) *et* (3.13c) *sont satisfaites,* **alors**
    retourner $\boldsymbol{x}$ la solution optimale.
  sinon
    choisir un indice $j^* \in \bar{\mathbb{S}}_0 \cup \bar{\mathbb{S}}_{\square} \cup \mathbb{S}_{\square}$, à partir de (3.30),(3.32) et (3.31), correspondant à une condition d'optimalité violée ;
    mettre à jour le signe θ_{j^*} de la variable choisie :
    si $j^* \in \bar{\mathbb{S}}_0$ **alors**
      $\theta_{j^*} \leftarrow \text{sgn}\ (-c_{j^*})$
    sinon
      $\theta_{j^*} \leftarrow \text{sgn}\ (x_{j^*})$
    fin
    $\bar{\mathbb{S}}_{in} \leftarrow \bar{\mathbb{S}}_{in} \cup \{j^*\}$;
    retourner à l'étape 1.
  fin
fin

Algorithme 5 : Méthode *d'ensemble actif* pour résoudre $\hat{\mathcal{Q}}_{2+1}$.

3.5 Résultats expérimentaux

Dans cette section, nous nous concentrons sur la performance des algorithmes proposés dans ce chapitre pour le calcul de relaxations continues dans un schéma de branch-and-bound (cf. Section 2.2). Pour pouvoir comparer les méthodes dans les mêmes conditions, nous utilisons un schéma d'algorithme branch-and-bound commun, où nous fixons les mêmes stratégies de branchement et de parcours d'arbre. Nous utilisons une stratégie de branchement qui consiste à sélectionner la variable de la relaxation continue $b_{j \in \bar{\mathbb{S}}}$ ayant la

valeur maximale [Bertsimas and Shioda, 2009] et une stratégie de parcours favorisant les branches de l'arbre du côté $b_j = 1$ en premier. Nous insérons l'algorithme d'homotopie (cf. Section 3.3) et d'ensemble actif (cf. Section 3.4) dans ce schéma de branch-and-bound, que nous appelons B&B$_{\text{R-HOM}}$ et B&B$_{\text{R-Act}}$. Ensuite, nous effectuons plusieurs tests :

1. Nous testons B&B$_{\text{R-HOM}}$ sur les trois formulations $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, puisque le même algorithme d'homotopie proposé (cf. Section 3.3) permet la résolution, de manière similaire, des trois problèmes relâchés $\hat{\mathcal{Q}}_{2/1}$, $\hat{\mathcal{Q}}_{1/2}$ et $\hat{\mathcal{Q}}_{2+1}$ définissant la relaxation continue. La méthode homotopique ne permet en revanche aucun redémarrage à chaud puisqu'elle impose une initialisation de la résolution du problème relâché par la solution donnée dans l'équation (3.16).

2. Nous testons B&B$_{\text{R-Act}}$ sur la formulation pénalisée $\hat{\mathcal{P}}_{2+0}$, étant donné que l'algorithme d'ensemble actif proposé dans la section 3.4 ne se transpose pas simplement à la résolution des problèmes relâchés $\hat{\mathcal{Q}}_{2/1}$ et $\hat{\mathcal{Q}}_{1/2}$. En revanche, il permet une initialisation par n'importe quel point réalisable. Ainsi, nous réalisons des tests sans redémarrage à chaud, en partant à chaque nœud du même point donné par l'équation (3.16) (initialisation de la procédure homotopique) et avec redémarrage à chaud en initialisant la relaxation continue d'un nœud courant par la solution de son nœud parent, comme expliqué au § 3.4.2.

Ensuite, nous comparons les performances de ces algorithmes à la même stratégie d'exploration branch-and-bound, où la relaxation continue est résolue avec le solveur de programmation quadratique CPLEX v12.8 (noté B&B$_{\text{R-CPLEX}}$). Toutes les méthodes sont implémentées en C++ et exécutées sur une machine UNIX équipée de 31,1 Go de RAM et de quatre processeurs centraux (CPU) Intel Core i7-6600U cadencés à 2,60 GHz. Les calculs sont limités à un seul cœur afin de se concentrer sur la performance des algorithmes (capacités de parallélisation désactivées) ; le temps CPU maximum est fixé à 1 000 secondes.

Nous étudions d'abord la performance des algorithmes en temps de calcul (tous les algorithmes sont exacts et donc la solution sera la même) sur des problèmes simulés de déconvolution parcimonieuse, puis sur des problèmes simulés de sélection de variables générés sous forme aléatoire.

3.5.1 Problèmes de déconvolution parcimonieuse

La déconvolution impulsionnelle est problème inverse classique en traitement du signal (cf. § 1.2.1.1), qui vise à estimer une séquence parcimonieuse $\boldsymbol{x}$ à partir d'une observation bruitée sous un modèle de la forme $\boldsymbol{y} = \mathbf{H}\boldsymbol{x} + \boldsymbol{\varepsilon}$, où $\mathbf{H}$ est une matrice de convolution discrète et $\boldsymbol{\varepsilon}$ est un terme aléatoire qui représente le bruit et les erreurs de modèle. Nous considérons les exemples de problèmes [1] proposés dans [Bourguignon et al., 2016], où $\boldsymbol{y} \in \mathbb{R}^{120}$, $\mathbf{H} \in \mathbb{R}^{120\times 100}$. Les données sont contaminées par un bruit blanc gaussien $\boldsymbol{\varepsilon}$ de rapport signal sur bruit RSB = 10 dB et $M = 1.1\|\mathbf{H}^T\boldsymbol{y}\|_\infty$, comme dans [Bourguignon et al., 2016]. Les colonnes de $\mathbf{H}$ sont normalisées, et le nombre de composantes non nulles K varie de 5 à 9 pour les problèmes $\hat{\mathcal{P}}_{2/0}$. Pour les problèmes $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, les paramètres respectifs ϵ et μ sont réglés de façon statistique en fonction de l'écart-type σ du bruit et du degré de parcimonie : ϵ est accordé de telle sorte que la probabilité $P(\|\boldsymbol{\varepsilon}\|_2^2 \leq \epsilon) = 95\ \%$, et $\mu = 2\sigma^2 \log(1/\rho - 1)$, où σ^2 est la variance du bruit et $\rho = K/n$ (voir exemple [Soussen et al., 2011] pour ce réglage). Malgré leur faible taille, ces problèmes sont difficiles à résoudre en raison de la forte corrélation entre les colonnes de la matrice $\mathbf{H}$.

Les résultats de temps de calcul moyennés sur 50 instances de chaque problème sont consignés dans la Table 3.1. Pour tous les problèmes, les trois algorithmes explorent le même nombre de nœuds, puisqu'ils utilisent la même stratégie branch-and-bound. Ainsi, reflètent directement les performances des algorithmes de relaxation continue intégrées sur l'ensemble du parcours de l'arbre de recherche. Nous comparons les temps de calcul séparément pour chacune des trois formulations du problème d'approximation parcimonieuse.

1. **Pour les problèmes** $\hat{\mathcal{P}}_{2/0}$: B&B$_{\text{R-HOM}}$ résout plus d'instances dans le temps maximal autorisé (1 000 s). Ainsi, pour $K = 7$, deux instances ne sont pas résolues par B&B$_{\text{R-CPLEX}}$ en temps autorisé, alors que toutes les instances sont bien résolues par B&B$_{\text{R-HOM}}$. L'algorithme B&B$_{\text{R-HOM}}$ est plus efficace que B&B$_{\text{R-CPLEX}}$ sur toutes les instances ($K = 5$, $K = 7$ et $K = 9$), avec une résolution de 10 fois (pour $K = 9$) à 12 fois plus rapide (pour $K = 7$).
2. **Pour les problèmes** $\hat{\mathcal{P}}_{0/2}$: les résultats sont beaucoup encore plus marqués, B&B$_{\text{R-HOM}}$ surpassant largement B&B$_{\text{R-CPLEX}}$ sur toutes les instances en temps de calcul. En particulier, B&B$_{\text{R-HOM}}$ a un temps de 60 à 110 fois plus rapide que B&B$_{\text{R-CPLEX}}$. Il arrive aussi à prouver l'optimalité pour toutes les instances, tandis que B&B$_{\text{R-CPLEX}}$ échoue avec $K = 9$, où 18 instances sont non résolues en 1 000 s.

1. Les données sont en ligne à l'adresse `pagesperso.ls2n.fr/~bourguignon-s/download_MIP.html`

Problème		B&B$_{\text{R-HOM}}$			B&B$_{\text{R-CPLEX}}$			B&B$_{\text{R-Act}}$					
								avec redémarrage à chaud			sans redémarrage à chaud		
		Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E
$\hat{\mathcal{P}}_{2/0}$	$K=5$	**0.7**	1.28	0	**7.7**	1.28	0						
	$K=7$	**11.6**	17.89	0	**141.9**	17.89	2						
	$K=9$	**43.5**	57.37	9	**448.1**	57.46	30						
$\hat{\mathcal{P}}_{0/2}$	$K=5$	**0.1**	0.21	0	**6.0**	0.21	0						
	$K=7$	**0.9**	2.32	0	**85.2**	2.32	0						
	$K=9$	**2.5**	5.22	0	**296.9**	5.22	18						
$\hat{\mathcal{P}}_{2+0}$	$K=5$	**1.8**	2.01	0	**32.6**	2.02	0	**0.7**	2.02	0	**1.9**	2.02	0
	$K=7$	**7.3**	10.20	0	**187.3**	10.22	7	**4.4**	10.22	0	**15.9**	10.22	0
	$K=9$	**25.6**	31.80	5	**470.7**	31.87	28	**17.1**	31.88	4	**36.5**	31.88	6

TABLE 3.1 – Efficacité algorithmique pour des problèmes de déconvolution de taille $m = 120$ et $n = 100$ en fonction du nombre de variables non nulles $K \in \{5, 7, 9\}$. Temps de calcul, nombre de nœuds explorés (Nds) et nombre de problèmes non résolus en 1 000 s (E). Les moyennes ne sont faites que sur les instances qui ont pu être résolues en moins de 1 000 s par les trois algorithmes.

3. **Pour les problèmes $\hat{\mathcal{P}}_{2+0}$** : B&B$_{\text{R-HOM}}$ est toujours meilleur que B&B$_{\text{R-CPLEX}}$ (environ 20 fois plus rapide). En revanche, le temps de calcul est en faveur de B&B$_{\text{R-Act}}$ avec redémarrage à chaud, qui est environ deux fois plus rapide que B&B$_{\text{R-HOM}}$ et résout plus d'instances en moins de 1 000 s. On remarque aussi l'efficacité du redémarrage à chaud pour l'algorithme d'ensemble actif, permettant d'accélérer la résolution d'un facteur deux (pour $K = 9$) à quatre (pour $K = 7$).

De manière générale, on peut dire que l'algorithme B&B$_{\text{R-HOM}}$ est meilleur sur les problèmes $\hat{\mathcal{P}}_{2/0}$ et $\hat{\mathcal{P}}_{0/2}$, alors que pour le problème $\hat{\mathcal{P}}_{2+0}$, l'algorithme B&B$_{\text{R-Act}}$ s'est avéré plus avantageux.

3.5.2 Problèmes de sélection de variables

Nous considérons également des problèmes simulés de sélection de variables avec des données générées aléatoirement (voir [Bertsimas and Shioda, 2009; Zheng et al., 2014] pour des simulations similaires), où $\mathbf{H} \in \mathbb{R}^{m \times n}$, avec $n = 2m$. Les coefficients de $\mathbf{H}$ sont gaussiens, le choix des coefficients non nuls est aléatoire et les amplitudes associées sont générées selon $u + \text{sgn}(u)$, où u est gaussien centré de variance unitaire, afin d'éviter des valeurs arbitrairement faibles. En raison de la nature aléatoire de $\mathbf{H}$ (il y a alors une faible corrélation entre les colonnes), ce type de problème est plus facile à résoudre, c'est pourquoi nous pouvons monter en dimension : nous considérons ici $n = 500$ et $n = 1\,000$, et la cardinalité varie de $K = 5$ à $K = 15$.

Comme dans le cas précédent, les données sont contaminées par un bruit blanc gaussien de rapport signal sur bruit 10 dB. Le paramètre M et les paramètres ϵ et μ sont réglés comme en Section 3.5.1. Les résultats moyennés sur 50 instances sont consignés dans la Table 3.2.

Comme pour les problèmes précédents, les trois algorithmes explorent le même nombre de nœuds. Cependant, le nombre de nœuds est bien inférieur sur ces instances que pour les problèmes de déconvolution, pourtant de plus petite taille, confirmant la relative « simplicité » de cette deuxième classe de problèmes.

Nous remarquons tout d'abord que nos algorithmes sont maintenant beaucoup plus efficaces que B&B$_{\text{R-CPLEX}}$ sur tous les formulations en termes de temps d'exécution : les temps de calcul de B&B$_{\text{R-HOM}}$ et B&B$_{\text{R-Act}}$ sont inférieurs à ceux obtenus pour la déconvolution, alors que les performances de B&B$_{\text{R-CPLEX}}$ s'écroulent en raison de l'augmentation de la dimension.

Problème			B&B$_{\text{R-HOM}}$			B&B$_{\text{R-CPLEX}}$			B&B$_{\text{R-Act}}$					
									avec redémarrage à chaud			sans redémarrage à chaud		
			Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E
$n = 500$	$\hat{\mathcal{P}}_{2/0}$	$K = 5$	**0.4**	0.03	0	**23.5**	0.03	0						
		$K = 10$	**3.8**	0.26	0	**249.4**	0.26	12						
		$K = 15$	**7.6**	0.59	24	**538.7**	0.59	48						
	$\hat{\mathcal{P}}_{0/2}$	$K = 5$	**0.03**	0.01	0	**85.8**	0.01	2						
		$K = 10$	**0.5**	0.07	0	**552.6**	0.07	20						
		$K = 15$	**1.5**	0.17	7	**809.1**	0.17	49						
	$\hat{\mathcal{P}}_{2+0}$	$K = 5$	**0.4**	0.13	0	**205.0**	0.13	1	**3.4**	0.12	0	**1.4**	0.12	0
		$K = 10$	**1.5**	0.29	0	**442**	0.29	21	**14.7**	0.29	1	**6.9**	0.29	0
		$K = 15$	**7.3**	0.63	26	**626.9**	0.63	49	**21.5**	0.63	35	**22.7**	0.63	34
$n = 1\,000$	$\hat{\mathcal{P}}_{2/0}$	$K = 5$	**1.1**	0.02	0	**109.2**	0.02	0						
		$K = 10$	**6.2**	0.06	0	**437.7**	0.06	13						
		$K = 15$	**33.9**	0.38	7	-	-	50						
	$\hat{\mathcal{P}}_{0/2}$	$K = 5$	**0.2**	0.01	0	**99.3**	0.01	0						
		$K = 10$	**2.6**	0.04	0	**462.9**	0.04	6						
		$K = 15$	**113**	1.02	0	-	-	50						
	$\hat{\mathcal{P}}_{2+0}$	$K = 5$	**0.6**	0.06	0	**275.5**	0.06	10	**2.3**	0.06	0	**1.8**	0.06	0
		$K = 10$	**2.5**	0.13	0	**508.2**	0.13	32	**8.9**	0.13	0	**6.1**	0.11	0
		$K = 15$	**19.5**	0.52	11	-	-	50	**100**	0.51	20	**93.6**	0.51	19

TABLE 3.2 – Efficacité algorithmique pour des problèmes de sélection de variables, avec les tailles respectives $n = \{500, 1\,000\}$ et $m = \{250, 500\}$, en fonction du nombre de variables non nulles $K = \{5, 10, 15\}$. Temps de calcul, nombre de nœuds explorés (Nds) et nombre de problèmes non résolus en 1 000 s (E). Les moyennes ne sont faites que sur les instances qui ont pu être résolues en moins de 1 000 s par les trois algorithmes.

1. **Pour les problèmes** $\hat{\mathcal{P}}_{2/0}$: B&B$_{\text{R-HOM}}$ résout plus d'instances dans le temps maximal autorisé (1 000 s) : ainsi, sur les instances les plus difficiles avec ($K = 10, n = 500$), 48 instances sur 50 ne sont pas résolues par B&B$_{\text{R-CPLEX}}$ en 1 000 s, alors que seulement 24 instances (presque la moitié) ne sont pas résolues par B&B$_{\text{R-HOM}}$. Pour le même nombre de variables non nulles $K = 10$ avec une dimension plus grande $n = 1\,000$, tous les instances ne sont pas résolues par B&B$_{\text{R-CPLEX}}$ en temps autorisé, alors que seulement 7 instances ne sont pas résolues par B&B$_{\text{R-HOM}}$. Le temps de calcul de B&B$_{\text{R-HOM}}$ est ainsi 50 fois ($K = 5, n = 500$) à 100 fois ($K = 5, n = 1\,000$) plus faible que celui de B&B$_{\text{R-CPLEX}}$.
2. **Pour les problèmes** $\hat{\mathcal{P}}_{0/2}$: comme pour les problèmes de déconvolution, B&B$_{\text{R-HOM}}$ surpasse largement B&B$_{\text{R-CPLEX}}$ sur toutes les instances en temps de calcul. En particulier pour $K = 5$, B&B$_{\text{R-HOM}}$ a un temps 495 fois plus rapide que B&B$_{\text{R-CPLEX}}$ avec $n = 1\,000$ et 2800 fois plus rapide pour $n = 500$. En plus, il arrive aussi à résoudre toutes les instances pour $n = 1\,000$ et $K = 15$, tandis que B&B$_{\text{R-CPLEX}}$ n'arrive à résoudre aucune instance dans le temps maximal autorisé.
3. **Pour les problèmes** $\hat{\mathcal{P}}_{2+0}$: B&B$_{\text{R-HOM}}$ est également toujours bien meilleur que B&B$_{\text{R-CPLEX}}$. Il arrive à résoudre plus de problèmes en 1 000 s et est beaucoup plus rapide (environ 400 fois pour $K = 5$ et 200 fois pour $K = 10$). Cependant, contrairement aux problèmes de déconvolution, le temps de calcul est en faveur de B&B$_{\text{R-HOM}}$ par rapport à B&B$_{\text{R-Act}}$, où on remarque que le redémarrage à chaud dégrade même légèrement le temps de résolution.

3.6 Conclusion et perspectives

En interprétant l'évaluation de chaque nœud comme un problème en norme ℓ_1, nous avons pu proposer une résolution dédiée et exacte permettant de garantir les bornes inférieures calculées dans la procédure de branch-and-bound, en exploitant le savoir-faire développé en optimisation ℓ_1. Deux algorithmes ont été construits pour résoudre les problèmes de relaxation continue impliqués dans n'importe quel nœud de l'arbre de recherche. Le premier est inspiré du principe de l'homotopie et peut être appliqué dans une procédure de branch-and-bound avec la même efficacité pour les trois formulations $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{2+0}$ et $\hat{\mathcal{P}}_{0/2}$. Sur l'ensemble des simulations réalisées, cette stratégie s'est avérée plus efficace que le calcul de la relaxation continue du MIP initial par le solveur CPLEX. En particulier, elle permet de résoudre efficacement la formulation contrainte par une

borne sur l'erreur quadratique de modélisation, qui présente un intérêt majeur dans de nombreuses applications, alors que le solveur CPLEX s'est révélé beaucoup moins efficace sur cette classe de problèmes. Un deuxième algorithme développé, de type ensemble actif, a permis de résoudre encore plus efficacement les problèmes difficiles de déconvolution parcimonieuse, formulés sous la forme pénalisée $\hat{\mathcal{P}}_{2+0}$, notamment grâce à la mise en œuvre d'une stratégie de démarrage à chaud. Sur des problèmes plus gros, cependant, c'est à nouveau la méthode homotopique qui s'est montrée la plus efficace.

Plusieurs perspectives en relation avec ce chapitre peuvent être identifiées. Tout d'abord, un algorithme d'ensemble actif pourrait également être développé pour aborder les relaxations continues du problème $\hat{\mathcal{P}}_{2/0}$. Leur reformulation $\hat{\mathcal{Q}}_{2/1}$, contrainte par un terme en norme ℓ_1, fait intervenir des problèmes avec des contraintes linéaires, pour lesquelles la méthodologie développée reste envisageable. L'exploitation de stratégies d'ensembles actifs pour le résolution de problèmes à contraintes quadratiques de type $\hat{\mathcal{Q}}_{1/2}$ nous semble en revanche plus délicate. Ensuite, en ce qui concerne le redémarrage à chaud, nous avons utilisé un redémarrage qui consiste à initialiser la solution de la relaxation continue du nœud courant par la solution de son nœud parent. Une étude sur une meilleure initialisation peut être intéressante afin de se rapprocher de la solution optimale et ainsi réduire le temps de convergence. Enfin, le choix du meilleur algorithme pour le calcul de la relaxation continue en fonction de la profondeur dans l'arbre de recherche ou du degré de parcimonie recherché peut également être mis en perspective.

Dans le chapitre suivant, nous nous concentrons sur l'étape de séparation, où nous étudions les stratégies de sélection de variables (branchement) et nous élaborons une nouvelle règle de sélection. Nous analysons aussi la structure de l'arbre de recherche et l'impact de la parcimonie sur quelques règles de parcours afin de définir des stratégies spécifiques au problème d'approximation parcimonieuse.

Chapitre 4

Stratégies d'exploration et règle de branchement

Contents

4.1 Introduction

La méthode branch-and-bound [Land and Doig, 1960], décrite au Chapitre 2, est basée sur la décomposition itérative du problème MIP complexe en sous-problèmes disjoints plus simples à résoudre (avec un espace de solution plus petit que le problème parent), constituant un arbre de recherche (voir Section 2.2). La manière dont on divise le problème est assurée par la *règle de branchement* qui définit comment choisir la variable binaire sur laquelle on va brancher au niveau du nœud ayant été sélectionné pour la séparation (voir Figure 4.1). Ce chapitre est consacré aux règles de branchement et à la stratégie de parcours de l'arbre de recherche.

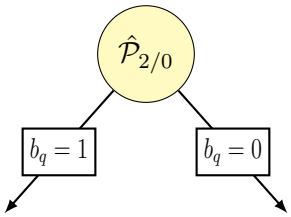


FIGURE 4.1 – Branchement sur une variable b_q

Les règles de branchement sont des éléments clés qui affectent fortement le nombre de nœuds explorés de l'algorithme branch-and-bound, puisqu'un changement de la règle de branchement implique un parcours différent dans l'espace des solutions. L'objectif est alors de minimiser le nombre de nœuds tout en ayant une règle de branchement rapide. La construction de règles de branchement avancées est au cœur de tout solveur moderne (voir par exemple [Achterberg and Wunderling, 2013]). Un certain nombre de règles très génériques ont été élaborées et présentées dans la littérature afin d'aborder la plus grande variété de problèmes possibles [Benichou et al., 1971; Borchers and Mitchell, 1994; Achterberg et al., 2005; Alvarez et al., 2017]. Les stratégies de branchement classiques se divisent en deux grandes catégories.

— D'une part, les approches de séparation forte (SF) ou branchement fort (*strong branching*) [Linderoth and Savelsbergh, 1999; Applegate et al., 1998; Achterberg et al., 2005] testent toutes les variables à chaque nœud et choisissent celle provoquant un changement maximal des bornes inférieures locales afin de réduire l'écart avec la borne supérieure (valeur de la meilleure solution possible). Bien que simple en apparence, le branchement fort est la stratégie de branchement la plus efficace

afin de réduire le nombre de nœuds dans l'arbre de recherche [Achterberg et al., 2005]. Cependant, cette efficacité se fait au détriment du temps de calcul par nœud, entraînant un temps de calcul total très élevé, ce qui fait que la séparation forte n'est généralement utilisée en pratique qu'au niveau du nœud racine (nœud initial de l'arbre).

— D'autre part, on trouve des stratégies de branchement qui sont conçues pour imiter la séparation forte en réduisant le temps de calcul par nœud, permettant de définir un compromis entre le nombre de nœuds et le temps total pour résoudre un MIP. Nous citons à titre d'exemple les pseudo-coûts [Benichou et al., 1971], le branchement hybride [Achterberg and Berthold, 2009], qui combine les pseudo-coûts et la séparation forte, et des approches qui consistent à imiter les décisions prises par le branchement fort avec des techniques d'apprentissage automatique à partir d'un ensemble de décisions de branchement observées et prises par la séparation forte [voir par exemple Alvarez et al., 2017].

À notre connaissance, les quelques travaux préalables à cette thèse sur la construction d'un algorithme branch-and-bound dédié au problème d'approximation parcimonieuse (toujours abordé dans sa formulation contrainte par la cardinalité) utilisent des règles de branchement basées sur la solution de la relaxation continue du MIP. Bienstock [1996] a ainsi utilisé la règle d'*infaisabilité maximale* (IM). Celle-ci consiste à choisir, dans la solution de relaxation continue, la variable binaire la plus éloignée de ses bornes, ce qui revient à choisir la variable binaire la plus proche de 0,5. Par la suite, une autre règle dite *amplitude maximale* (AM) a été proposée par Bertsimas and Shioda [2009], qui consiste à choisir la variable continue de valeur absolue maximale.

Nous analysons d'abord en Section 4.2 le comportement de la règle de séparation forte ainsi que les deux règles de branchement infaisabilité maximale (IM) et amplitude maximale (AM), que nous étudions à travers les propriétés des problèmes de relaxation continue établies au chapitre 2, qui relient chaque variable binaire à la variable continue associée à l'optimum. Nous proposons ensuite une règle de sélection spécifique au problème d'approximation parcimonieuse, nommée LPS pour ℓ_1 *Path Selection*, qui exploite le chemin de solutions de problèmes en norme ℓ_1 construit par la méthode homotopique lors du calcul de la relaxation continue (voir Section 3.3). Dans la section 4.3, nous présentons une stratégie de parcours exploitant la structure non symétrique de notre arbre de recherche permettant de trouver rapidement des solutions réalisables. Enfin, une étude comparative des différentes règles mentionnées ci-dessus est présentée en Section 4.4. Pour illustrer

nos propos, nous considérons dans ce chapitre la forme contrainte par la parcimonie $\hat{\mathcal{P}}_{2/0}$ (voir l'équation (2.1)). Les choix qui en résultent seront également appliqués pour résoudre les problèmes $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$.

4.2 Règles de branchement

La règle de branchement (règle de sélection de variable pour l'algorithme branch-and-bound) définit comment l'espace de recherche doit être partitionné et quelle variable va être choisie pour le branchement. Afin de simplifier les notations, nous notons ici $\hat{\mathcal{P}}$ le sous-problème associé à un nœud quelconque de l'arbre de recherche, $\hat{\mathcal{P}}^R$ le problème de relaxation continue associée, z^R la valeur de la relaxation continue et $(\boldsymbol{x}^R, \boldsymbol{b}^R)$ la solution de la relaxation continue :

$$\begin{cases} z^R = \min \hat{\mathcal{P}}^R \\ (\boldsymbol{b}^R, \boldsymbol{x}^R) = \arg \min\limits_{\boldsymbol{b}\in[0,1]^n, \boldsymbol{x}\in\mathbb{R}^n} \hat{\mathcal{P}}^R. \end{cases}$$

Nous rappelons tout d'abord quelques notations qui seront utilisées dans ce chapitre. Comme nous l'avons vu en Section 2.3, la relaxation continue se réduit à un problème faisant intervenir la norme ℓ_1. Celui-ci s'écrit comme suit au niveau du nœud racine :

$$\begin{cases} z^R = \min\limits_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \text{ s.c. } \begin{cases} \|\boldsymbol{x}\|_1 \leq KM \\ \|\boldsymbol{x}\|_\infty \leq M \end{cases} \\ \boldsymbol{b}^R = |\boldsymbol{x}^R|/M \end{cases} . \tag{4.1}$$

Pour le problème relâché dans un nœud i quelconque, les variables binaires indexées par $\mathbb{S}_1$ sont fixées à 1 ($\boldsymbol{b}_{\mathbb{S}_1} = \mathbf{1}$), les variables binaires non fixées sont indexées par $\bar{\mathbb{S}}$ et les variables binaires indexées par $\mathbb{S}_0$ sont fixées à 0 ($\boldsymbol{b}_{\mathbb{S}_0} = \mathbf{0}$) et sont retirées du problème afin de simplifier les écritures (on a alors $\boldsymbol{x}_{\mathbb{S}_0} = \mathbf{0}$). Le problème de relaxation continue associé s'écrit alors :

$$z^R = \min_{\substack{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1} \\ \boldsymbol{b}_{\bar{\mathbb{S}}}\in[0,1]^{\bar{n}} \\ \boldsymbol{x}_{\bar{\mathbb{S}}}\in\mathbb{R}^{\bar{n}}}} \frac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 \text{ s.c. } \begin{cases} \sum_{q\in\bar{\mathbb{S}}} b_q \leq K - n_1 \\ |\boldsymbol{x}_{\bar{\mathbb{S}}}| \leq M\boldsymbol{b}_{\bar{\mathbb{S}}} \\ \|\boldsymbol{x}_{\mathbb{S}_1}\|_\infty \leq M \end{cases} \tag{4.2}$$

où $n_1 = \text{Card}(\mathbb{S}_1)$ et $\bar{n} = \text{Card}(\bar{\mathbb{S}})$. Étant donné que les variables de décision sont binaires, le branchement sur une variable non fixée b_q, $q \in \bar{\mathbb{S}}$, engendre deux sous-problèmes : un en ajoutant la contrainte d'égalité $b_q = 1$ (appelé le sous-problème gauche ou nœud fils gauche, noté $\hat{\mathcal{P}}_q^{(\text{g})}$) et l'autre en ajoutant la contrainte d'égalité $b_q = 0$ (appelé le sous-problème droit ou le nœud fils droit, noté $\hat{\mathcal{P}}_q^{(\text{d})}$). Ce branchement revient donc respectivement soit à déplacer son indice q de l'ensemble des indices des variables non fixées $\bar{\mathbb{S}}$ à l'ensemble des indices des variables non nulles ($\mathbb{S}_1 \leftarrow \mathbb{S}_1 \cup \{q\}$) ou à ceux des variables nulles ($\mathbb{S}_0 \leftarrow \mathbb{S}_0 \cup \{q\}$), auquel cas on peut le retirer du problème relâché. La Figure 4.2 illustre ce branchement.

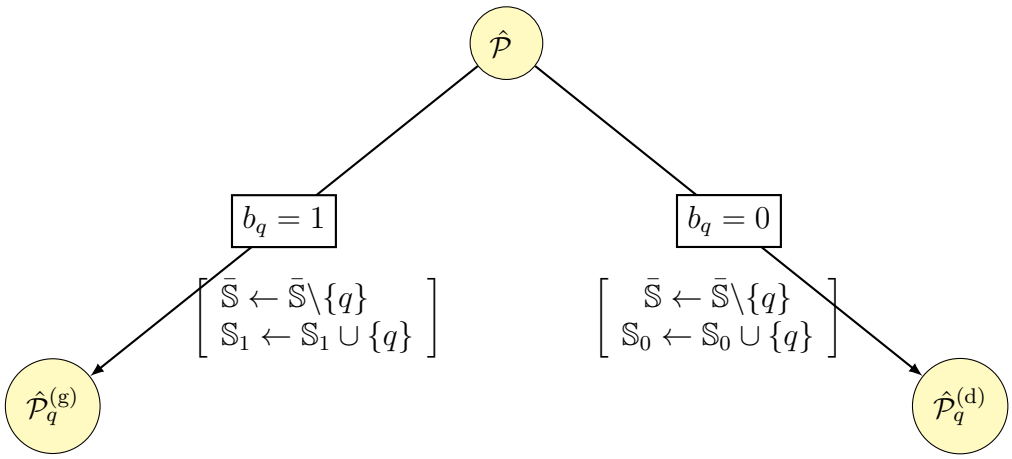


FIGURE 4.2 – Exemple de branchement sur la variable b_q et création des nœuds fils : fils gauche $\hat{\mathcal{P}}_q^{(\text{g})}$ et fils droit $\hat{\mathcal{P}}_q^{(\text{d})}$.

Nous notons respectivement $\hat{\mathcal{P}}_q^{R(\text{g})}$ et $\hat{\mathcal{P}}_q^{R(\text{d})}$ les problèmes de relaxation continue associés aux deux nœuds fils et $z_q^{R(\text{g})}$ et $z_q^{R(\text{d})}$ les valeurs respectives à l'optimum de ces relaxations (qui constituent donc les bornes inférieures de l'ensemble des sous-problèmes du nœud fils gauche et du nœud fils droit). L'objectif de la règle de branchement est d'évaluer les variables candidates (les variables binaires non fixées encore), et de sélectionner la « meilleure » variable sur laquelle brancher afin de minimiser le nombre de nœuds à évaluer. Nous commençons par une présentation de la règle de *Séparation Forte* (SF), qui est l'une des meilleures règles de branchement connues pour minimiser le nombre de nœuds explorés par l'algorithme branch-and-bound.

4.2.1 La Séparation Forte (SF) pour le problème d'approximation parcimonieuse

L'idée générale de la règle de la *Séparation Forte* (SF) est de trouver la variable la plus « influente » sur la borne inférieure. Dans notre cas, la règle SF résout, pour toutes les variables candidates b_q, $q \in \bar{\mathbb{S}}$, les sous-problèmes relâchés des deux nœuds fils $\hat{\mathcal{P}}_q^{R(\mathrm{g})}$ et $\hat{\mathcal{P}}_q^{R(\mathrm{d})}$: elle résout ainsi $2\bar{n}$ problèmes relâchés en fixant à chaque fois b_q à 1 puis à 0. Pour choisir la variable la plus influente sur la borne inférieure, un score est alors attribué à chaque composante q en fonction de l'écart relatif à la borne inférieure du nœud courant z^R [voir par exemple Achterberg et al., 2005]. Soient les écarts :

$$\begin{cases} \Delta_q^{(\mathrm{g})} := z_q^{R(\mathrm{g})} - z^R & \text{pour le fils gauche,} \\ \Delta_q^{(\mathrm{d})} := z_q^{R(\mathrm{d})} - z^R & \text{pour le fils droit.} \end{cases}$$

Le score d'une variable q est calculé en fonction des deux écarts comme suit :

$$\text{Score}_{\,\mathrm{SF}}(q) = u\Delta_q^{(\mathrm{d})} + (1-u)\Delta_q^{(\mathrm{g})}. \tag{4.3}$$

où $u \in [0,1]$ est un paramètre compris entre 0 et 1 (indépendant de q), généralement déterminé empiriquement ou ajusté dynamiquement au cours de la résolution La règle SF sélectionne la variable $\hat{q}$ maximisant la fonction de score associée :

$$\hat{q} = \arg\max_{q \in \bar{\mathbb{S}}} \ \text{Score}_{\,\mathrm{SF}}(q). \tag{4.4}$$

Des tests réalisés lors de cette thèse ont montré que, pour nos problèmes d'approximation parcimonieuse, le réglage empirique du paramètre u a donné de meilleurs résultats en termes de nombre de nœuds en choisissant $u = 1$. Ainsi, la fonction score de la règle de branchement SF, pour une variable d'indice q, se réduit à :

$$\text{Score}_{\,\mathrm{SF}}(q) = \Delta_q^{(\mathrm{d})}. \tag{4.5}$$

Ce choix revient à sélectionner la variable provoquant un changement maximal sur les bornes inférieures du côté droit de l'arbre $\hat{\mathcal{P}}_q^{R(\mathrm{d})}$ $(b_q = 0)$ *i.e.,* à sélectionner la variable x_q qui, une fois enlevée du problème, fait remonter le plus la borne inférieure. Une explication possible réside dans la structure particulière de l'arbre qui est naturellement déséquilibré à cause de la contrainte de parcimonie, où le sous-arbre du côté gauche $\hat{\mathcal{P}}_q^{(\mathrm{g})}$ $(b_q = 1)$ est

plus petit que celui du côté droit. En effet, en présence de la contrainte de parcimonie $\sum_{q=1}^{n} b_q \leq K$, la profondeur de l'arbre de recherche est limitée à K variables fixées du côté $b_q = 1$. Ainsi, les sous-arbres du côté gauche $\hat{\mathcal{P}}_q^{(\mathrm{g})}$ ($b_q = 1$) sont souvent moins profonds que ceux du côté droit $\hat{\mathcal{P}}_q^{(\mathrm{d})}$ ($b_q = 0$). La figure 4.3 montre un exemple d'arbre de recherche pour $K = 2$.

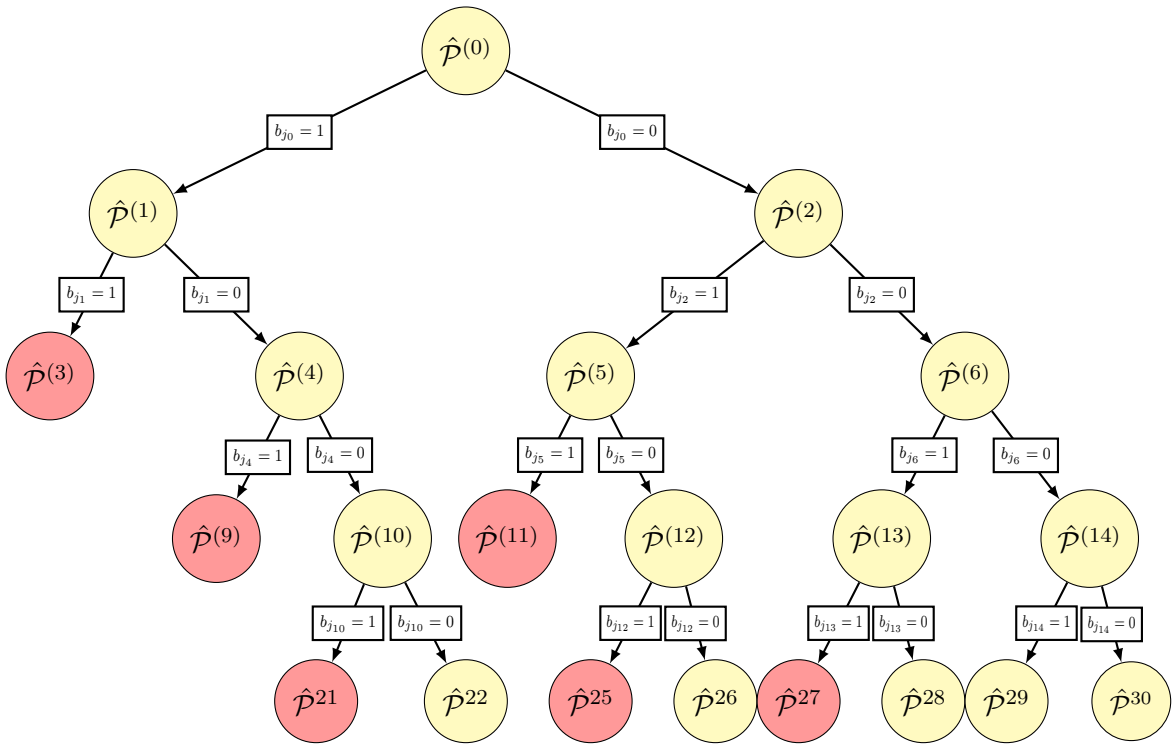


FIGURE 4.3 – Exemple d'arbre binaire de recherche pour le problème $\hat{\mathcal{P}}_{2/0}$ en présence de la contrainte de parcimonie avec $\sum_{q=1}^{n} b_q \leq 2$. Les branches du côté $b_q = 1$ (respectivement, $b_q = 0$) sont représentées à gauche (respectivement, à droite). La contrainte de parcimonie engendre obligatoirement une solution réalisable après fixation de deux variables binaires à 1 (nœuds en rouge).

Remarquons enfin que l'écart $\Delta_q^{(\mathrm{d})}$ du côté nœud fils droit $\hat{\mathcal{P}}_q^{(\mathrm{d})}$ (créé par l'ajout de la contrainte $b_q = 0$), qui représente le côté le plus profond de l'arbre, est faible quand le branchement est effectué sur une variable de faible amplitude.

Proposition 5. *La valeur de $\Delta_q^{(d)}$ est nulle si le branchement est effectué sur une variable b_q dont la valeur dans le problème relâché b_q^R est nulle :*

$$b_q^R = 0 \Leftrightarrow \Delta_q^{(d)} = 0. \tag{4.6}$$

Autrement dit, si la solution de la relaxation continue b_q^R est nulle, l'ajout de la contrainte $b_q = 0$ ne change pas la solution de la relaxation continue au niveau du nœud fils $\hat{\mathcal{P}}_q^R$.

Démonstration. Soit $(\boldsymbol{b}^R, \boldsymbol{x}^R)$ le minimiseur de $\hat{\mathcal{P}}^R$ et b_q la variable choisie pour le branchement. Soit $(\boldsymbol{b}^{R(\mathrm{d})}, \boldsymbol{x}^{R(\mathrm{d})})$ le minimiseur de $\hat{\mathcal{P}}_q^{R(\mathrm{d})}$ (créé par l'ajout de la contrainte $b_q = 0$). Si $b_q^R = 0$, alors $(\boldsymbol{b}^{R(\mathrm{d})}, \boldsymbol{x}^{R(\mathrm{d})})$ est clairement réalisable pour $\hat{\mathcal{P}}_q^{R(\mathrm{d})}$, donc $\boldsymbol{b}^R = \boldsymbol{b}^{R(\mathrm{d})}$ et par conséquent $\Delta_q^{(\mathrm{d})}$ est nulle. Réciproquement, si $\Delta_q^{(\mathrm{d})} = 0$, alors $\boldsymbol{b}^R = \boldsymbol{b}^{R(\mathrm{d})}$ et $\boldsymbol{x}^R = \boldsymbol{x}^{R(\mathrm{d})}$. La contrainte ajoutée n'a donc pas changé la solution du nœud fils $(\boldsymbol{b}^{R(\mathrm{d})}, \boldsymbol{x}^{R(\mathrm{d})})$ et ainsi $b_q^R = 0$. □

Nous pouvons aussi supposer que l'écart $\Delta_q^{(\mathrm{d})}$ est moins important pour les variables b_q telles que la solution b_q^R de la relaxation continue est proche de zéro. Cette remarque sera utilisée plus tard pour expliquer l'inefficacité de certaines règles de branchement génériques sur nos problèmes d'approximation parcimonieuse.

4.2.2 Infaisabilité Maximale et Infaisabilité Minimale

Les règles de branchement *Infaisabilité Maximale* et *Infaisabilité Minimale* sont des règles génériques que l'on peut trouver dans la plupart des solveurs MIP. La règle de branchement *Infaisabilité Maximale* consiste à choisir la variable entière qui, dans le problème relâché, possède la plus grande partie fractionnaire. Autrement dit, elle sélectionne la variable relâchée la plus éloignée d'un nombre entier. Pour notre problème avec des variables binaires, cette règle va ainsi sélectionner la variable fractionnaire du problème relâché $b_q^R \in [0, 1]$ la plus proche de 0.5. La fonction score de cette règle s'écrit alors :

$$\text{Score}_{\,\mathrm{IM}}(q) = -|0.5 - b_q^R|. \tag{4.7}$$

Nous pouvons l'écrire aussi en fonction de $\boldsymbol{x}_q^R$. Étant donné qu'à l'optimum du problème relâché on a $|x_q^R| = Mb_q^R$ (voir l'équation (4.1)), la fonction score de la règle IM s'écrit :

$$\text{Score}_{\text{IM}}(q) = -|\tfrac{M}{2} - |x_q^R|\ |,$$

et dans ce cas, la règle IM va sélectionner la variable x_q la plus proche de $\frac{M}{2}$.

La règle de branchement *Infaisabilité Maximale* est l'opposée de la règle *Infaisabilité Minimale*. Cette dernière cherche à sélectionner la variable fractionnaire la plus proche d'un nombre entier. Pour notre problème, la règle *Infaisabilité Minimale* va sélectionner la variable binaire b_q la plus proche de 0 ou de 1. Cela revient à considérer la variable continue x_q la plus proche de 0 ou de M. En pratique, cependant, la relaxation continue qui est un problème en norme ℓ_1, contient généralement de nombreuses valeurs très proches de 0. Cette règle de branchement va donc surtout avoir tendance à sélectionner des composantes non nulles de faible amplitude dans le problème relâché. Or, sélectionner des variables proches de 0 n'est pas un bon choix puisqu'il ne permet pas d'améliorer significativement les bornes inférieures, comme nous l'avons montré dans Proposition 5. La règle *Infaisabilité Minimale* est donc plutôt inefficace pour notre problème, contrairement à la règle de branchement IM qui a été déjà utilisée par Bienstock [1996].

4.2.3 Amplitude Maximale (AM)

La règle de branchement *Amplitude maximale* (AM) a été utilisée par Bertsimas and Shioda [2009] pour le problème d'approximation parcimonieuse. Elle consiste simplement à choisir, dans le problème relâché, la variable fractionnaire b_q ayant la valeur maximale c'est-à-dire la variable b_q la plus proche de 1). La fonction score de cette règle s'écrit donc

$$\text{Score}_{\text{AM}}(q) = b_q^R. \tag{4.8}$$

Nous pouvons également éclairer cette règle à l'aide des propriétés établies au chapitre 2 : étant donné qu'à l'optimum du problème relâché on a $|x_q^R| = Mb_q^R$ (voir l'équation (4.1)), cette règle revient à choisir la variable continue x_q^R du problème relâché dont la valeur absolue est maximale :

$$\text{Score}_{\text{AM}}(q) = |x_q^R|.$$

Cette règle est très proche de IM, puisque la variable ayant la valeur maximale est celle la plus proche de la borne M, ce qui lui donne un comportement très similaire à IM.

4.2.4 Chemin de Solution ℓ_1 (LPS)

Les règles AM et IM, qui sont basées sur la solution de la relaxation continue, sont efficaces lorsque celle-ci est assez parcimonieuse (proche de la solution recherchée du problème MIP). Alors que la solution de la relaxation continue, qui est un problème en norme ℓ_1, est très sensible au conditionnement de $\mathbf{H}$ et à la valeur de *BigM* (voir Section 2.3). Pour donner plus de stabilité, nous proposons une nouvelle règle de sélection de variables, nommée LPS pour ℓ_1 *Path Selection*, basée sur le chemin de régularisation en norme ℓ_1. Ce dernier est défini par l'ensemble des solutions optimales du critère pénalisé (voir Chapitre 3, Équation (3.4)) en fonction de λ :

$$\boldsymbol{x}^{*(\lambda)} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \ \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \mathbf{H}_{\bar{\mathbb{S}}}\boldsymbol{x}_{\bar{\mathbb{S}}}\|_2^2 + \lambda\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 \quad \text{s.c } \|\boldsymbol{x}\|_\infty \leq M. \tag{4.9}$$

Rappelons en effet que, dans le chapitre précédent, nous avons proposé une méthode homotopique (voir Algorithme 4) permettant la construction du chemin de régularisation associé au problème de la relaxation continue à chaque nœud, reformulé sous la forme de l'équation (4.9).

L'idée de la règle LPS est de sélectionner la variable avec la valeur absolue maximale intégrée sur toute la plage $\lambda \in [\lambda_c, \lambda^{(0)}]$, de la solution telle que $\boldsymbol{x}_{\bar{\mathbb{S}}} = \mathbf{0}$ pour $\lambda = \lambda^{(0)}$ (voir l'équation (3.16b)) à la solution de relaxation continue pour $\lambda = \lambda_c$. À titre de rappel, pour le problème contraint par la parcimonie considéré dans ce chapitre, la cible λ_c est atteinte lorsque $\|\boldsymbol{x}_{\bar{\mathbb{S}}}\|_1 = (K - \text{Card}(\mathbb{S}_1))M$ (pour les autres formulations, voir les sections 3.3.5 et 3.3.4). Autrement dit, nous choisissons la variable la plus présente tout au long du chemin de régularisation, ce qui apporte plus de stabilité dans la sélection par rapport aux règles AM et IM. Pour cela, nous définissons la fonction de score suivante :

$$\text{Score}_{\text{LPS}}(q) = \sum_{k=0} |x_q^{*(\lambda^k)}|. \tag{4.10}$$

Le calcul de cette fonction peut être réalisé récursivement à chaque itération k de l'algorithme homotopique au cours de la résolution de la relaxation continue. Cette règle ne génère donc aucun coût supplémentaire par rapport au calcul effectué dans chaque nœud de l'arbre de recherche durant la résolution par l'algorithme branch-and-bound. La figure 4.4 présente un exemple illustratif de la sélection de variables par la règle SLS ainsi que par les règles AM, IM. Les règles AM et IM sont toutes les deux basées uniquement sur

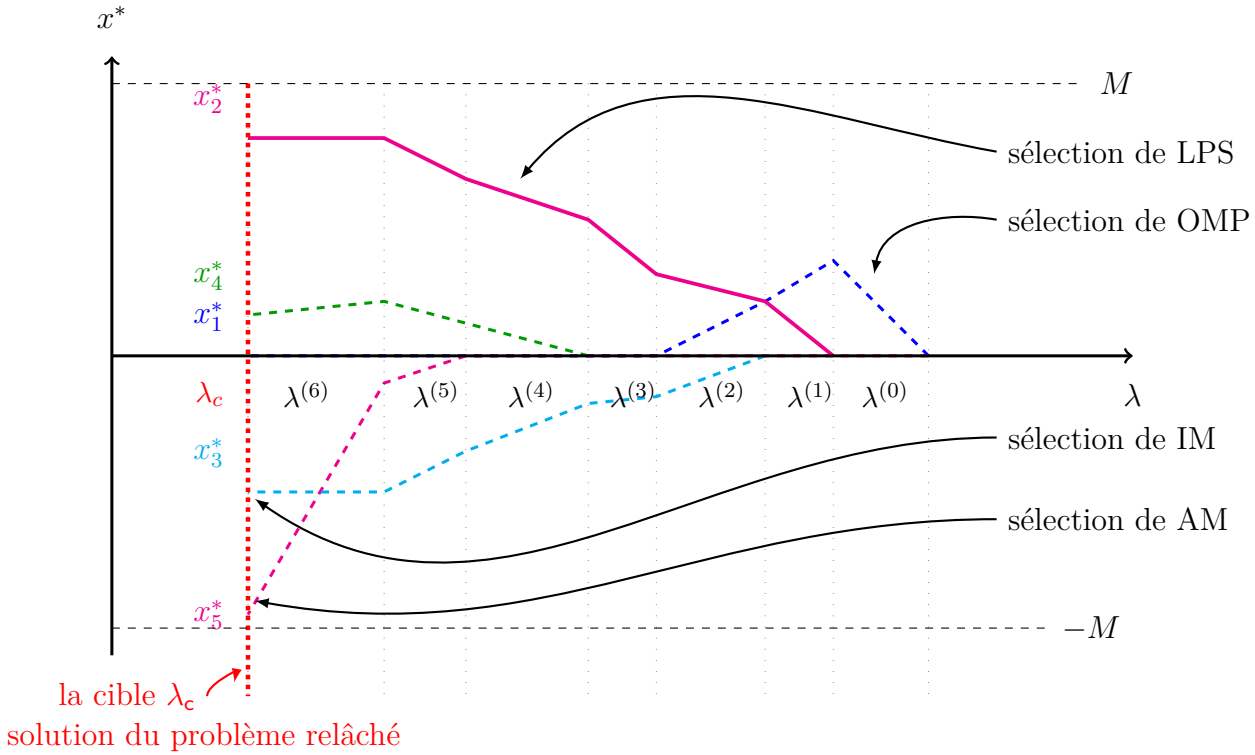


FIGURE 4.4 – Exemple de chemin de régularisation en norme ℓ_1 en fonction de $\lambda \in [\lambda_c, \lambda^{(0)}]$, avec 5 variables : $\bar{\mathbb{S}} = \{1, ..., 5\}$ et règles de sélection de variable associés. La variable x_2^* est la plus présente le long du chemin de solution. La ligne pointillée verticale rouge représentent la cible λ_c.

la solution de la relaxation continue, *i.e.*, pour $\lambda = \lambda_c$ (voir la ligne verticale rouge). Sur cet exemple, AM sélectionne la variable binaire associée à x_5 ayant la valeur absolue la plus proche de la borne M et IM sélectionne la variable binaire associée à x_3 ayant la valeur absolue la plus proche de $M/2$. En revanche, la règle SLS sélectionne la variable binaire associée à x_2, qui est la variable la plus présente tout au long du chemin. La figure montre également la sélection de la règle résultant de l'algorithme glouton OMP qui choisit la première variable devenant non nulle lorsque λ décroît en-dessous de $\lambda^{(0)}$ (cette règle sera discutée au § 4.4.1.2, où nous présentons les règles de sélection de quelques algorithmes gloutons).

En pratique, comme les règles AM et IM, la règle LPS sélectionne une variable binaire dont la valeur à l'optimum du problème relâché est non nulle, ce qui permet d'améliorer les bornes inférieures des nœuds fils. Cependant, la règle de branchement dépend fortement

de la règle de parcours de l'arbre qui définit quel nœud va être exploré en premier. Nous proposons maintenant une stratégie de parcours de l'arbre, dite *Recherche en profondeur du côté branche supérieure d'abord*, qui sera utilisée par la suite.

4.3 Stratégie de parcours *Recherche en profondeur du côté branche supérieure d'abord (DUFS)*

Dans l'algorithme branch-and-bound (voir l'Algorithme 3), à chaque itération, il faut choisir le prochain nœud à évaluer (de quelle manière l'arbre doit être exploré). Cette étape est définie par les stratégies de parcours de l'arbre appelées aussi stratégies de sélection des nœuds. Il existe plusieurs stratégies standard en optimisation MIP (voir par exemple Wolsey and Nemhauser [1999]), les plus couramment utilisées étant :

— la recherche en profondeur d'abord (*DFS, Depth-First Search*) est une stratégie qui explore en priorité les nœuds les plus éloignés de la racine,

— la recherche en largeur d'abord (*BFS, Breadth-First Search*) est l'inverse de la recherche DFS : elle explore en priorité les nœuds les moins éloignés de la racine,

— la recherche en meilleur d'abord (*BFS, Best-First Search*) traite le nœud ayant la borne inférieure la plus faible.

Dans notre cas nous proposons d'utiliser la stratégie de recherche dite *en profondeur du côté branche supérieure d'abord* (*DUFS, Depth-Up First Search*). Elle consiste à explorer en priorité les nœuds les plus éloignés de la racine en commençant par le côté $b_q = 1$ d'abord. La figure 4.5 illustre un exemple de parcours en profondeur d'abord (à gauche) et un parcours en largeur d'abord (à droite) sur un même arbre de recherche.

La stratégie de recherche en profondeur du côté branche supérieure d'abord, exploite la structure non symétrique de l'arbre de recherche (voir Figure 4.3) pour trouver rapidement des solutions réalisables (*i.e.*, des solutions avec K composantes non nulles). Cette stratégie qui vise à activer en priorité les variables est cohérente avec la règle de branchement proposée au § 4.2.4 : dans le même esprit que les algorithmes gloutons dits *Forward selection* pour l'approximation parcimonieuse (voir le § 1.3.2), elle permet d'avoir une solution réalisable après au maximum K nœuds explorés (K variables binaires ont alors été fixées à 1). En conséquence, cela permet d'obtenir très rapidement une borne supérieure de qualité définie

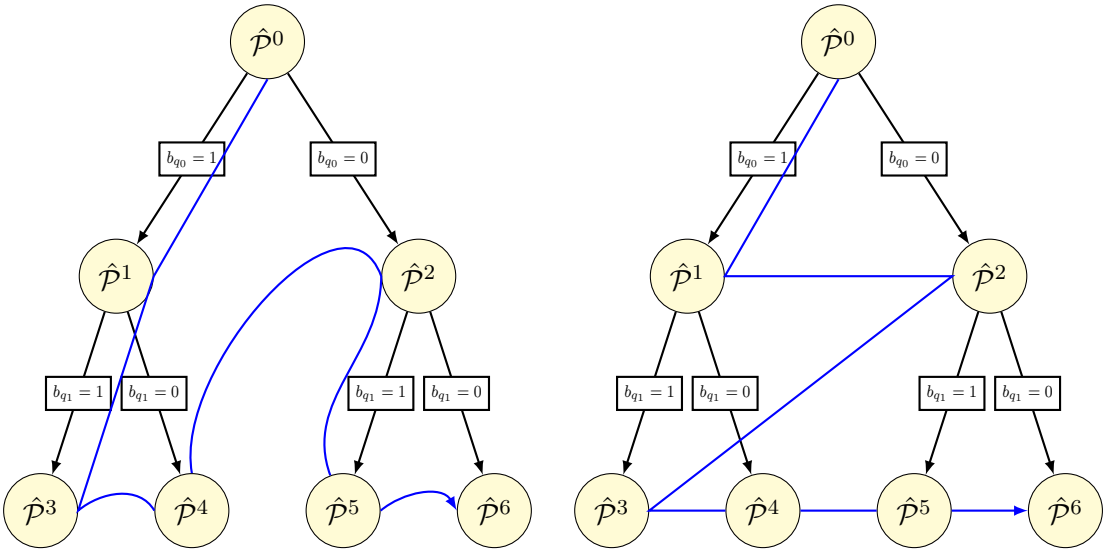


FIGURE 4.5 – Exemple de parcours avec la stratégie de *recherche en profondeur du côté branche supérieure d'abord* (à gauche) vs parcours en *largeur d'abord* (à droite)

par la valeur de la meilleure solution réalisable trouvée et ainsi d'espérer réduire le nombre de nœuds parcourus.

Cette stratégie de parcours permet aussi d'économiser l'espace mémoire puisque l'exploration sera faite branche par branche. Sa complexité spatiale (le nombre maximal de nœuds actifs dans la file d'attente) est de l'ordre du degré de parcimonie recherché ($O(K)$). Enfin, avec cette stratégie, le passage du nœud parent au nœud fils permet également de transmettre rapidement toutes les informations calculées au nœud parent et qui peuvent être utiles pour faire le redémarrage à chaud (voir Section 3.4.2) au niveau du nœud fils.

4.4 Comparaison de différentes règles de branchement

Dans cette section, nous évaluons la qualité des règles de branchement sur deux aspects :

i) **aspect local :** nous étudions leur capacité à fournir des bonnes solutions réalisables pour le problème MIP (solution à K composantes non nulles) permettant d'améliorer la borne supérieure,

ii) **aspect global :** nous étudions leur efficacité à réduire le nombre de nœuds explorés dans l'algorithme branch-and-bound.

4.4.1 Recherche de solutions de bonne qualité

Effectuer K sélections successives de variables binaires fixées à 1 permet de construire la première solution réalisable à K composantes. On explore ainsi la première branche de l'arbre jusqu'à sa profondeur maximale. Plus la solution est bonne, plus la borne supérieure initiale est basse, ce qui va favoriser l'élagage dans le parcours des nœuds suivants. Afin de focaliser sur la capacité des règles de branchement à trouver des bonnes solutions rapidement, nous évaluons, dans un premier temps, la qualité de la première solution réalisable en utilisant LPS par rapport à celles obtenues par les règles de branchement AM et IM. Cette stratégie de parcours s'apparentant à une sélection, en K itérations, des « meilleures variables » explicatives dans le dictionnaire, nous la comparons ensuite à la solution trouvée par les algorithmes gloutons classiques en l'approximation parcimonieuse.

4.4.1.1 Comparaison avec les règles de branchement AM et IM

Dans cette partie, nous illustrons la capacité des règles de branchement LPS, AM et IM à trouver des bonnes solutions réalisables sur des problèmes simulés de déconvolution impulsionnelle en utilisant la stratégie de recherche DUFS décrite en Section 4.3. Nous évaluons la qualité de la première solution trouvée, en considérant les mesures d'erreur suivantes, où $\bar{\boldsymbol{x}}$ et $\widehat{\boldsymbol{x}}$ dénotent respectivement la solutions trouvée et la solution globale du problème en norme ℓ_0, calculée par une procédure branch-and-bound :

— E_Q : l'erreur quadratique $\|\boldsymbol{y} - \mathbf{H}\bar{\boldsymbol{x}}\|_2^2$, *i.e.,* la valeur de la fonction de coût obtenue par la solution K-parcimonieuse obtenue ;

— E_S : l'erreur de support $E_S = \|\bar{\boldsymbol{b}} - \widehat{\boldsymbol{b}}\|_0$, où $b_q = 1$ (respectivement, $b_q = 0$) si $x_q \neq 0$ (respectivement, si $x_q = 0$).

Nous considérons les mêmes instances de problèmes abordés dans la section 3.5.1, où les données $\boldsymbol{y} \in \mathbb{R}^{120}$ et le dictionnaire $\mathbf{H} \in \mathbb{R}^{120\times 100}$. Les résultats sont moyennés sur 50 instances dans chaque cas. La figure 4.6 montre les erreurs E_Q et E_S des solutions trouvées par les règles introduites précédemment, en fonction de la parcimonie de la solution. Nous constatons que les erreurs E_Q et E_S augmentent naturellement avec K (le problème étant plus difficile à résoudre), et que les trois règles ont un comportement presque similaire pour les deux erreurs E_Q et E_S. En moyenne, la solution trouvée par notre règle LPS est meilleure que celle de AM et IM en termes d'erreur quadratique E_Q et d'erreur de support E_S, et cet écart de performance augmente avec K.

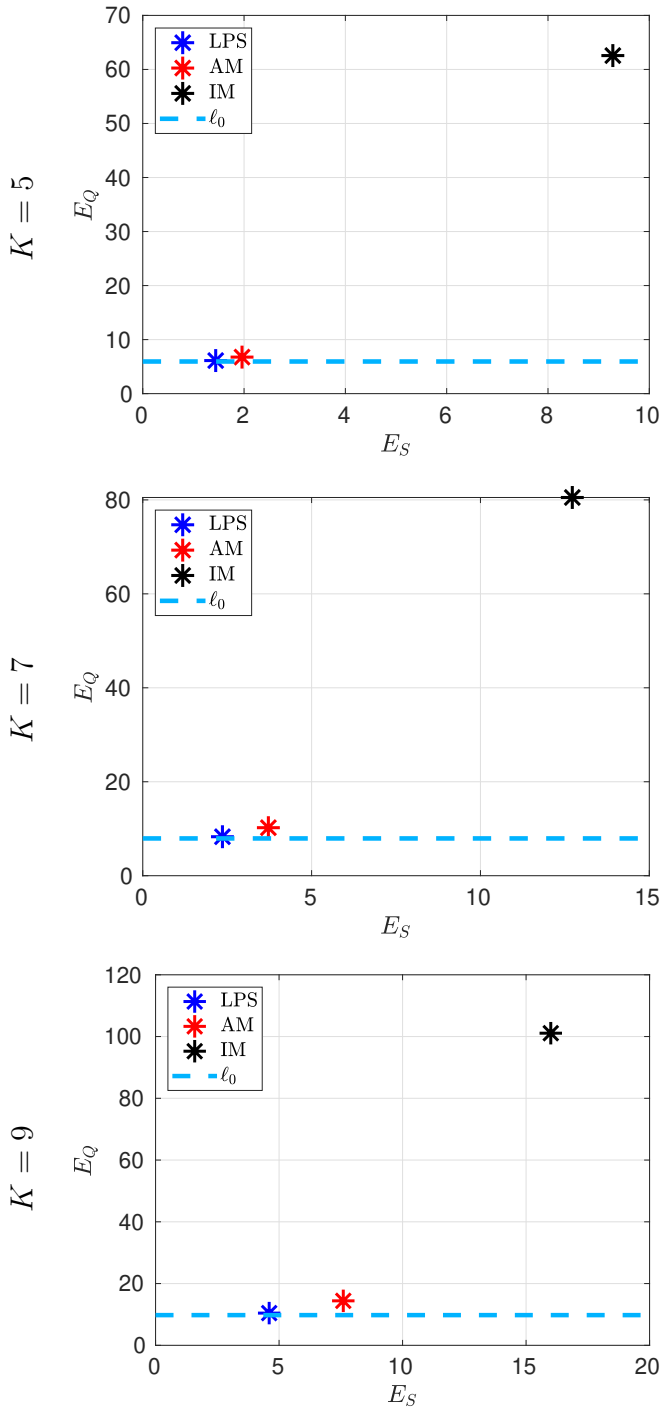


FIGURE 4.6 – Performances de la règle LPS (*) comparée à AM (*) et IM (*) en qualité de première solution trouvée après la fixation de K composantes à 1 : erreur quadratique E_Q et erreur de support E_S. Pour une meilleure visualisation, la solution optimale du problème en norme ℓ_0, qui est un point d'abscisse 0, est représentée en pointillée (−−). Les résultats sont moyennés sur 50 réalisations de problèmes de déconvolution impulsionnelle de taille $n = 100$ inconnues et $m = 120$ données et avec un rapport signal sur bruit de SNR $= 10\,\text{dB}$.

4.4.1.2 Règles de sélection de variables classiques des méthodes sous-optimales

Pour une comparaison supplémentaire de la qualité de la solution trouvée par la règle LPS, nous la comparons ici avec les solutions obtenues par les algorithmes gloutons conçus pour le problème d'approximation parcimonieuse.

Rappelons que les méthodes gloutonnes (voir Section 1.3.2) construisent une approximation parcimonieuse en sélectionnant des variables d'une manière itérative. La différence principale entre les différents algorithmes réside dans la *règle de sélection des variables*. Le but de cette dernière est de choisir une variable qui mène à une solution de bonne qualité. Plusieurs règles de sélection de variables ont été proposées dans la littérature. Deux règles bien connues sont celles des algorithmes Orthogonal Matching Pursuit (OMP) [Pati et al., 1993] et Orthogonal Least Squares (OLS) [Chen et al., 1989]. L'algorithme OMP utilise une règle de sélection de variables qui consiste à choisir la variable minimisant l'erreur d'approximation du résidu courant et une colonne de $\mathbf{H}_{\bar{\mathbb{S}}}$, en adaptant l'amplitude. Sa fonction de score revient à calculer le produit scalaire :

$$\text{Score}_{\text{OMP}}(q) = |\boldsymbol{h}_q^T(\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1})|. \tag{4.11}$$

En particulier, la règle d'OMP choisit la même variable que notre règle LPS si l'algorithme homotopique était arrêté à la première itération (comme illustré sur la Figure 4.4), puisque à l'initialisation de l'algorithme homotopique on retrouve la mème équation (voir l'équation (3.16)), et la première variable qui devient non nulle en $\lambda^{(0)}$, est bien la variable donnée par $\text{Score}_{\text{OMP}}(q)$.

Pour OLS, la règle de sélection de variable a un coût de calcul plus élevé que OMP. À chaque itération, OLS cherche à sélectionner la variable minimisant l'erreur d'approximation tout en remettant en cause les amplitudes des variables qui sont déjà sélectionnées. La fonction de score est calculée comme suit :

$$\text{Score}_{\text{OLS}}(q) = -\min_{\boldsymbol{x}_{\mathbb{S}_1}\in\mathbb{R}^{n_1}, x_q\in\mathbb{R}} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}_{\mathbb{S}_1}\boldsymbol{x}_{\mathbb{S}_1} - \boldsymbol{h}_q x_q\|_2^2 \tag{4.12}$$

Une limitation des algorithmes gloutons avec les règles classiques mentionnées ci-dessus réside dans leur vision *locale* de la sélection de variables, qui peut facilement se tromper dans le cas des problèmes inverses où le dictionnaire $\mathbf{H}$ est corrélé. Les règles classiques comme celles de OMP et OLS, qui cherchent à minimiser à chaque itération

un critère incluant une seule nouvelle composante, peuvent alors sélectionner des fausses composantes. La figure 4.7 présente un exemple type d'une fausse détection sur des problèmes de déconvolution parcimonieuse, où les interférences entre les deux composantes recherchées créent un pic au milieu (voir les figures 4.7 a) et b)), rendant la détection difficile. Nous pouvons voir sur la figure 4.7 c) que la fonction de score à la première itération de OMP, qui est également la même fonction pour OLS à la première itération :

$$\text{Score}_{\text{OMP}}(q) = \text{Score}_{\text{OLS}}(q) = |\boldsymbol{h}_q^T \boldsymbol{y}|,$$

va donc sélectionner cette composante erronée. Ainsi, les algorithmes OMP et OLS font tous les deux une fausse détection dès la première sélection, ce qui va ensuite affecter la seconde sélection. En revanche, sur cet exemple, la règle LPS permet une meilleure détection. On peut voir sur la figure 4.7 e) que la fonction score de LPS est plus élevée aux positions des deux vraies composantes que sur la fausse composante du milieu ; ainsi, les deux vraies composantes sont bien localisées. Au final, l'approximation fournie par la règle LPS donne donc une meilleure approximation que OMP, comme on peut voir sur les figure 4.7 d) et f).

Afin d'évaluer de manière plus quantitative la performance de la règle proposée LPS, nous comparons la solution trouvée, après fixation des K composantes b_q à 1, à la solution donnée par les algorithmes gloutons les plus connus pour l'approximation parcimonieuse : OMP, OLS, Single Best Replacement (SBR) [Soussen et al., 2011], $A^\star$ Orthogonal Matching Pursuit ($A^\star$OMP) [Karahanoglu and Erdogan, 2012] et Iterative Hard Thresholding (IHT) [Blumensath and Davies, 2008], mais aussi à la régularisation en norme ℓ_1 ou Basis Pursuit (BP). Nous nous intéressons toujours que à la formulation $\mathcal{P}_{2/0}$ contrainte par la norme ℓ_0 et tous les algorithmes sont réglés afin d'obtenir une solution avec le même degré de parcimonie recherché K. Nous illustrons le comportement des algorithmes gloutons sur les mêmes problèmes simulés de déconvolution impulsionnelle introduits en Section 3.5.1 et utilisons les mêmes mesures d'erreur qu'au § 4.4.1.1.

Les résultats moyennés sur 50 instances sont présentés en figure 4.8. Tout d'abord, nous constatons qu'aucun algorithme glouton ne parvient à trouver les solutions optimales en norme ℓ_0, ce qui montre la difficulté de ces problèmes malgré leur petite taille. Nous remarquons que la solution trouvée par LPS fournit toujours l'erreur de support la plus faible quelle que soit la valeur de K. Pour l'erreur quadratique, seul l'algorithme A*OMP

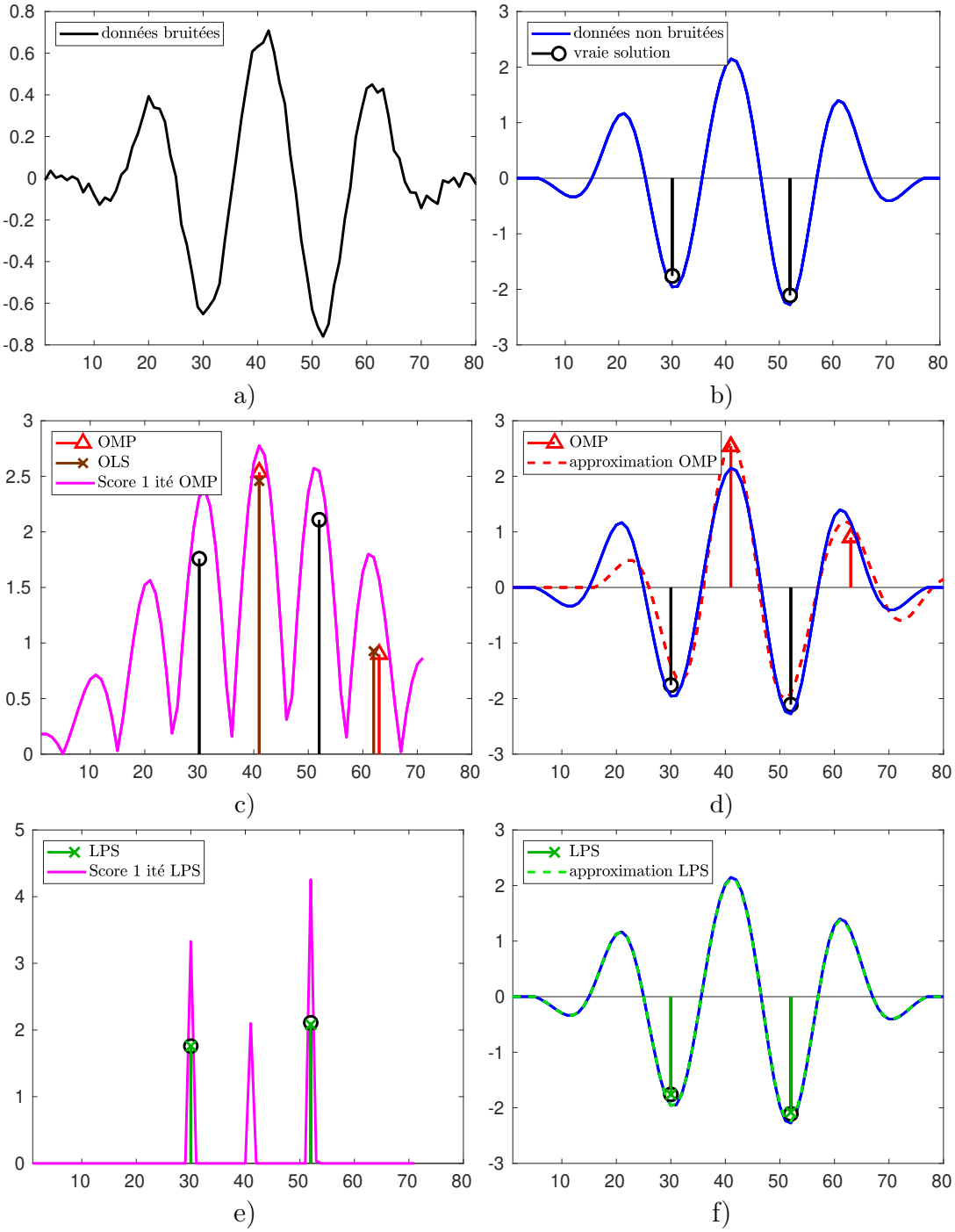


FIGURE 4.7 – Illustration typique d'une fausse détection par des algorithmes gloutons sur un problème simulé de déconvolution. a) données bruitées $\boldsymbol{y}$ (−) ; b) données non bruitées (−) et vraie solution (∘) ; c) solution de OMP (△), solution de OLS (□) et fonction score associée à la 1ère itération de OMP/OLS (−) ; d) solution de OMP (△) et approximation associée (- -) ; e) fonction score associée à la 1ère itération de LPS (−) ; f) solution de LPS (×) et approximation associée (- -).

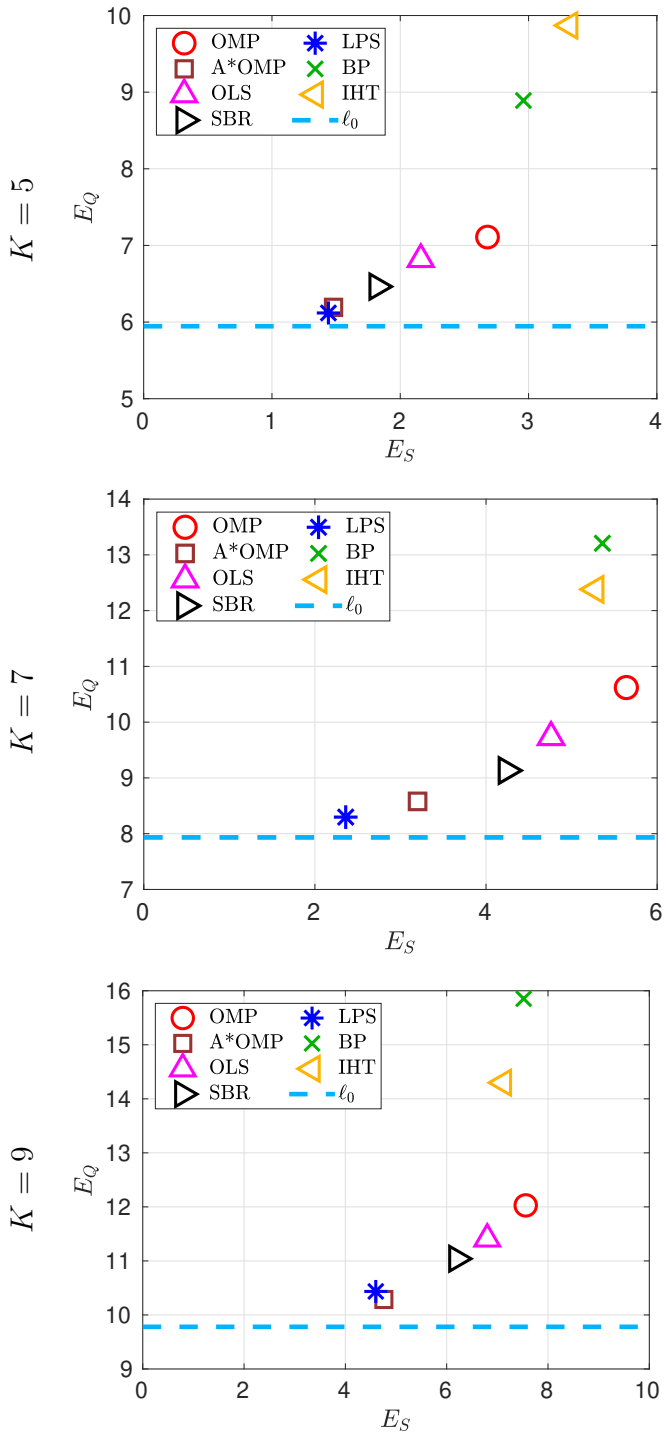


FIGURE 4.8 – Performances des algorithmes LPS, OLS, OMP, $A^\star$OMP, SBR, BP et IHT. Les résultats sont moyennés sur 50 réalisations pour l'erreur quadratique E_Q et l'erreur de Support E_S sur des problèmes de déconvolution impulsionnelle de taille $n = 100$ inconnues et $m = 120$ données et avec un rapport signal sur bruit de SNR $= 10\,\text{dB}$. La solution optimale du problème en norme ℓ_0 est représentée en pointillé (−−).

parvient à fournir une erreur légèrement inférieure pour $K = 9$.

En conclusion, il est clair que notre règle de sélection de variable proposée LPS semble judicieuse et très intéressante pour trouver des solutions de bonne qualité pour le problème de l'approximation parcimonieuse, d'autant plus qu'elle n'induit aucun surcoût dans la procédure branch-and-bound si la relaxation continue est calculée par la méthode homotopique.

4.4.2 Comparaison de différentes règles de branchement dans l'algorithme de branch-and-bound

Dans cette partie, nous évaluons les performances des différentes règles de branchement discutées précédemment dans un algorithme branch-and-bound : la règle de *Séparation Forte* (SF), la règle *Infaisabilité Maximale* (IF), la règle *Amplitude Maximale* (AM) et notre règle *Chemin de Solution* ℓ_1 (LPS), pour le problème contraint par la parcimonie $\hat{\mathcal{P}}_{2/0}$. Nous utilisons la stratégie de parcours *Recherche en profondeur du côté branche supérieure d'abord* (DUFS). Toutes les méthodes sont implémentées en C++ et le temps de calcul maximum est fixé à 1 000 secondes. Les séries d'expérimentations sont réalisées sur les mêmes instances de problèmes simulés de déconvolution impulsionnelle introduits en Section 3.5.1.

Afin de comparer les différentes règles en temps de calcul et en nombre de nœuds, nous utilisons les profils de performance [Dolan and Moré, 2002]. Ces derniers sont des outils visuels devenus un standard pour la comparaison d'algorithmes d'optimisation qui s'appuient sur des statistiques obtenues à partir d'un grand nombre d'instances de problèmes. Le profil de performance prend en compte le nombre de problèmes résolus en un temps donné maximum ainsi que le coût de la résolution d'un problème (le coût peut être le temps de calcul nécessaire ou le nombre de nœuds explorés), qui est évalué par rapport au meilleur algorithme. Plus précisément, étant donné un ensemble de problèmes P et un ensemble d'algorithmes A, soit $c_{a,p}$ le coût de la résolution du problème $p \in P$ par l'algorithme $a \in A$. Si l'algorithme ne parvient pas à résoudre le problème p, on définit $c_{a,p} = +\infty$. Supposons qu'au moins un algorithme résout le problème p, le meilleur algorithme pour un problème donné est celui qui le résout avec le coût minimum, ainsi le

coût relatif ou *rapport de performance* de l'algorithme sur un problème s'écrit :

$$\alpha_{a,p} = \frac{c_{a,p}}{\min_{a \in A} c_{a,p}}, \quad \text{avec } \alpha_{a,p} \geq 1.$$

La valeur $\alpha_{a,p} = 1$ signifie donc que l'algorithme a est le meilleur pour le problème p. Enfin, la *fonction de performance* de l'algorithme a, qui sera présentée sur le profil, est la fonction de répartition empirique de $\alpha_{a,p}$, donnée par

$$F_a(t) = \frac{\text{Card}(\{p \in P \mid \alpha_{a,p} \leq t\})}{\text{Card}(P)}.$$

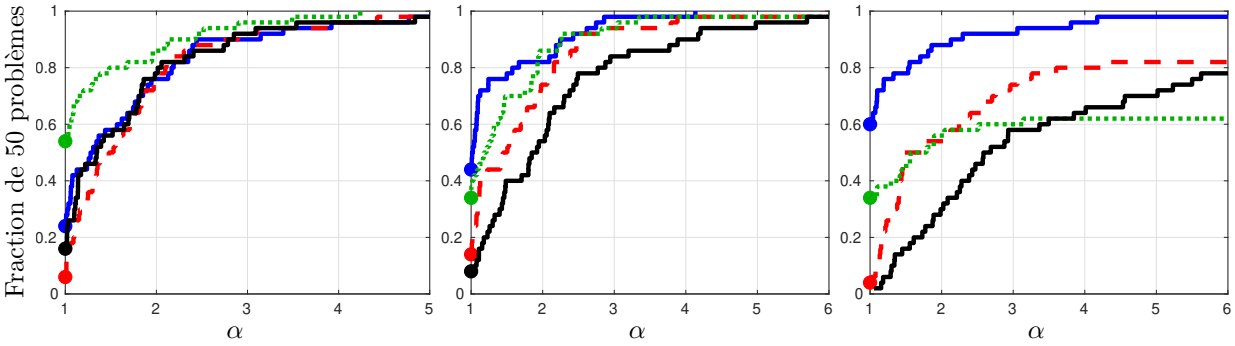
On constate que $F_a(1)$ est le pourcentage de problèmes tels que $\alpha_{a,p} = 1$, c'est-à-dire le pourcentage de problèmes pour lesquels l'algorithme a est le meilleur. En outre, $F_a(\alpha_{\max})$ est la fraction de problèmes résolus par l'algorithme a, où

$$\alpha_{\max} = \max_{a \in A, p \in P} \alpha_{a,p}.$$

La valeur $F_a(1)$ est appelée l'*efficacité* de l'algorithme a et $F_a(\alpha_{\max})$ est la *robustesse*.

Tout d'abord, nous nous concentrons sur la comparaison des différentes règles de branchement *en termes de nombre de nœuds explorés*. Les profils de performance associés (où $c_{a,p}$ compte le nombre de nœuds explorés par l'algorithme a sur l'instance p), obtenus sur 50 instances considérées du problème $\hat{\mathcal{P}}_{2/0}$, sont présentés dans la figure 4.9 pour $K = 5$, 7 et 9. La proportion de problèmes résolus est représentée en fonction du rapport de performance α. L'algorithme le plus performant est celui qui se trouve en haut à gauche, et l'algorithme le plus robuste (*i.e.,* qui résout le plus de problèmes) est celui qui se trouve en haut à droite sur le profil de performance.

Nous remarquons déjà que les trois règles LPS, AM et IM sont compétitives avec SF. Pour $K = 5$, tous les problèmes sont résolus en moins de 1 000 s pour les différents algorithmes (toutes les courbes sur le profil de performance ont atteint la valeur de 1. La règle SF explore le plus petit nombre de nœuds sur 57% des problèmes, et notre règle LPS sur 25% des problèmes (points à gauche pour $\alpha = 1$ des profils de performance sur la figure 4.9 à gauche). Les règles IM et AM explorent le plus petit nombre de nœuds sur respectivement 18% et 5% des problèmes. Pour les problèmes un peu plus difficiles avec $K = 7$, la règle LPS explore le plus petit nombre de nœuds sur 44% des problèmes, puis nous trouvons successivement les règles SF sur 36% des problèmes, AM sur 15% des problèmes et IM sur 9% des problèmes. Enfin, pour les cas les plus difficiles (avec $K = 9$),



Rapport de performance en nombre de nœuds pour $K = 5$

Rapport de performance en nombre de nœuds pour $K = 7$

Rapport de performance en nombre de nœuds pour $K = 9$

FIGURE 4.9 – Profils de performance en nombre de nœuds obtenus sur 50 problèmes simulés de déconvolution impulsionnelle pour la formulation $\hat{\mathcal{P}}_{2/0}$, avec les règles de branchement LPS (−), SF (· · ·), AM (−−) et IM (−).

la règle LPS explore le plus petit nombre de nœuds sur 60% des problèmes et elle est plus efficace et plus robuste que les autres puisque elle a permis de résoudre plus de problèmes en 1 000 s (98% des problèmes sont résolus par LPS en 1 000 s, ensuite 82% sont résolus par AM, 78% par IM et seulement 62% par SF).

La figure 4.10 montre enfin les profils de performance *en temps de calcul* sur ces mêmes problèmes. Nous pouvons voir que la règle SF, malgré son efficacité à réduire le nombre de nœuds, reste très coûteuse et inefficace en pratique. Ce comportement était prévisible, le temps de calcul par nœud étant très élevé. La règle LPS a permis d'avoir un algorithme plus rapide que AM et IM sur 50% des problèmes pour $K = 5$, sur 80% des problèmes pour $K = 7$ et sur 88% des problèmes pour $K = 9$. Enfin, elle est plus efficace et plus robuste, ayant permis de résoudre plus de problèmes en 1 000 s pour les différentes valeurs de K. Ainsi, pour le cas le plus difficile avec $K = 9$, la règle LPS a résolu plus de 98% des problèmes, alors que SF n'a résolu que 26% des problèmes.

4.5 Conclusions et perspectives

Dans ce chapitre, nous avons discuté des stratégies de séparation dans un algorithme branch-and-bound (sélection de variables et parcours de l'arbre de recherche) pour le problème d'approximation parcimonieuse contraint par la parcimonie. Nous avons proposé une stratégie de parcours, nommée DUFS pour *Depth-Up First Search*, privilégiant l'acti-

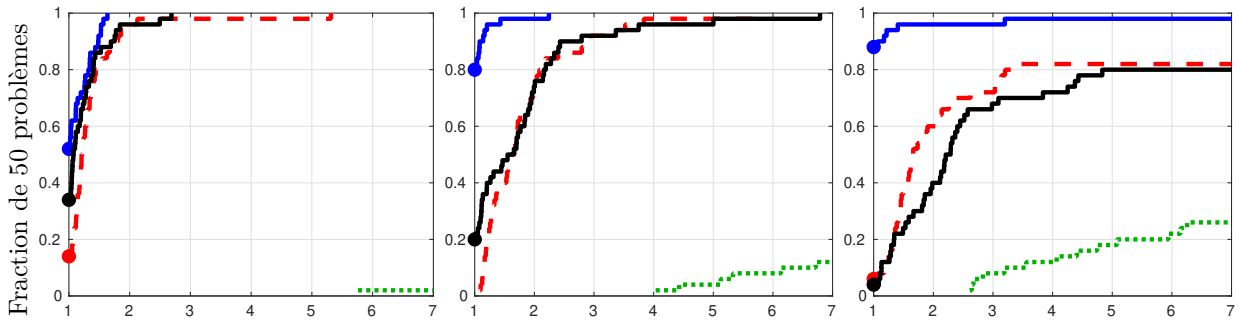


FIGURE 4.10 – Profils de performance en temps de calcul obtenus sur 50 problèmes simulés de déconvolution impulsionnelle pour la formulation $\hat{\mathcal{P}}_{2/0}$, avec les règles de branchement LPS (−), SF (· · ·), AM (−−) et IM (−).

vation des variables dans le modèle. Ensuite, nous avons proposé une règle de branchement appelée LPS pour ℓ_1 *Path Selection*, basée sur le chemin de régularisation en norme ℓ_1. Ce dernier étant déjà calculé par la méthode homotopique lors du calcul de la résolution du problème de relaxation continue à chaque nœud, la règle LPS l'exploite et ne génère ainsi aucun coût supplémentaire à la résolution. La particularité de notre règle de branchement LPS combinée à DUFS est sa capacité à orienter la recherche vers des solutions de bonne qualité, permettant donc une amélioration plus rapide de la borne supérieure. Plusieurs tests ont été effectués visant à évaluer la qualité des solutions trouvées à l'aide de la règle LPS sur des problèmes inverses difficiles de déconvolution impulsionnelle, où nous avons comparé la première solution réalisable trouvée (après fixation de K variables binaires à 1) :

— à la solution trouvée par les règles de branchement utilisées dans les travaux antérieurs proposant des algorithmes branch-and-bound pour l'approximation parcimonieuse, telles que la règle *Infaisabilité Maximale* (IM) et la règle *Amplitude Maximale* (AM),

— aux solutions des algorithmes classiques d'estimation parcimonieuse tels que OMP, OLS, SBR, A⋆OMP, IHT et BP.

Dans les deux cas, les solutions trouvées par LPS sont souvent de meilleure qualité.

Insérée dans l'algorithme branch-and-bound, la règle de branchement LPS, a montré de bonnes performances et s'avère beaucoup plus efficace que les autres règles de branchement plus génériques telles que SF, AM et IM. Dans un travail non directement lié aux questions de branchement dans un algorithme de branch-and-bound, nous avons d'ailleurs proposé

de construire une méthode gloutonne dont la sélection de variable est basée sur la règle LPS, s'avérant très compétitif par rapport aux approches existantes [Ben Mhenni et al., 2020].

Plusieurs perspectives s'ouvrent autour des stratégies de branchement et de parcours de l'arbre, ainsi que de l'amélioration de la borne supérieure au fil de la résolution par l'algorithme branch-and-bound. Dans ce chapitre, nous avons abordé une stratégie de recherche très simple, qui consiste à faire une recherche en profondeur du côté supérieur d'abord. On pourrait cependant imaginer des règles hybrides plus sophistiquées, qui consistent à mélanger la recherche en profondeur d'abord et la recherche meilleur d'abord afin de bénéficier des avantages des deux stratégies. Dans l'idée d'améliorer la borne supérieure, outre la règle de branchement et son efficacité à guider la recherche vers des solutions réalisables de bonne qualité, il serait également possible de faire appel aux heuristiques gloutonnes lors de la résolution par l'algorithme branch-and-bound et à différents endroits de l'arbre de recherche, pour construire des solutions réalisables au niveau d'un nœud donné (en prenant en compte les choix qui ont déjà été fixés). Afin d'optimiser le coût du calcul des heuristiques gloutonnes, une étude sur le meilleur algorithme glouton ainsi que sur la fréquence des appels ou les endroits intéressants dans l'arbre de recherche est également une perspective intéressante à étudier.

Chapitre 5

Évaluation des performances de l'algorithme résultant

Contents

5.1 Introduction

Dans cette partie, nous évaluons les performances de notre algorithme branch-and-bound pour la résolution des trois problèmes d'approximation parcimonieuse $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, en intégrant les différentes parties développées dans les chapitres précédents, à savoir les algorithmes pour le calcul de la relaxation continue du Chapitre 3, et les stratégies de branchement et d'exploration du Chapitre 4.

Nous comparons notre algorithme branch-and-bound au solveur MIP CPLEX v12.8 [1], reconnu comme l'un des meilleurs solveurs MIP actuels, en utilisant ses paramètres par défaut. Toutes les méthodes sont mises en œuvre en C++ et exécutées sur une machine UNIX équipée de quatre processeurs centraux (CPU) Intel Core i7-6600U cadencés à 2,60 GHz. Les calculs sont limités à un seul cœur afin de se concentrer sur la performance de l'algorithme hors capacités de parallélisation et, pour chaque problème, le temps CPU maximum est fixé à 1 000 secondes. Dans ce chapitre, nous ne regardons pas la qualité des solutions par rapport aux méthodes parcimonieuses classiques. Il a été déjà montré dans [Bourguignon et al., 2016], que l'optimisation exacte en norme ℓ_0 donne une meilleure qualité de solution sur des problèmes inverses difficiles. Nous ne comparons pas non plus les solutions entre notre algorithme branch-and-bound et CPLEX, qui sont les mêmes puisque les deux méthodes sont exactes. Nous nous concentrons plutôt sur le temps de calcul, le nombre de nœuds explorés et le nombre de problèmes résolus dans le temps autorisé.

Comme en Section 3.5, nous réalisons l'étude comparative sur deux types de problèmes. Nous commençons notre comparaison sur des problèmes de déconvolution parcimonieuse qui sont des problèmes de petite taille mais difficiles. Ensuite, nous regardons des problèmes simulés de sélection de variables, où le dictionnaire est une matrice aléatoire. En raison de la faible corrélation entre les éléments du dictionnaire, cette deuxième classe de problèmes est plus facile à résoudre ; nous pouvons alors envisager la résolution exacte de problèmes de plus grande taille.

5.2 Problèmes de déconvolution parcimonieuse

Nous considérons les exemples de problèmes de déconvolution parcimonieuse issus de [Bourguignon et al., 2016] et abordés dans la section 3.5.1. Notre algorithme branch-

1. https ://www.ibm.com/support/pages/cplex-optimization-studio-v128

and-bound, que nous appelons ici B&B, utilise l'algorithme d'homotopie (cf. Section 3.3) pour calculer les relaxations continues des problèmes $\hat{\mathcal{P}}_{2/0}$ et $\hat{\mathcal{P}}_{0/2}$, combiné avec la règle de branchement *Chemin de Solution* ℓ_1 (LPS) décrite dans la section 4.2.4. Pour le problème $\hat{\mathcal{P}}_{2+0}$, notre algorithme B&B utilise l'algorithme d'ensemble actif (cf. Section 3.4) pour le calcul de la relaxation continue ainsi que la règle de branchement *Amplitude maximale* (AM) décrite dans la section 4.2.3. Ces choix sont ceux ayant débouché sur les meilleurs résultats pour chaque classe de problèmes. Pour la règle du parcours de l'arbre, nous utilisons la stratégie de *recherche en profondeur d'abord du côté branche supérieure*, décrite en Section 4.3.

Les résultats de temps de calcul, moyennés sur 50 instances de chaque problème, sont consignés dans la Table 5.1, dont nous tirons quelques conclusions. D'une manière générale, on retrouve la complexité d'ordre combinatoire où tous les temps de calcul augmentent rapidement avec K, de même pour le nombre de nœuds explorés. Nous comparons maintenant les résultats du temps de calculs pour les trois formulations séparément :

1. **Pour les problèmes** $\hat{\mathcal{P}}_{2/0}$: B&B résout plus d'instances dans le temps maximal autorisé (1 000 s) : pour le cas le plus difficile avec $K = 9$, six instances ne sont pas résolues par CPLEX, alors qu'une seule est non résolue par B&B. Notre algorithme B&B est plus efficace que CPLEX sur toutes les instances ($K = 5$, $K = 7$ et $K = 9$), avec une résolution deux fois (pour $K = 9$) à six fois plus rapide (pour $K = 5$) en moyenne. Nous remarquons également que le nombre de nœuds explorés est toujours plus faible pour notre algorithme, confirmant la pertinence de la stratégie de branchement LPS et à sa capacité à guider la recherche vers des solutions de bonne qualité. Le temps de calcul rapporté au nombre de nœuds est également en faveur de B&B (plus de trois fois plus rapide pour $K = 5$), grâce à la rapidité de la méthode homotopique pour le calcul de la relaxation continue.
2. **Pour les problèmes** $\hat{\mathcal{P}}_{0/2}$: les résultats sont beaucoup plus marqués en faveur de notre algorithme, où B&B surpasse largement CPLEX sur toutes les instances en temps de calcul. En particulier, B&B a un temps moyen de 100 (pour $K = 9$) à 250 fois (pour $K = 5$) plus petit que CPLEX. Il résout également toutes les instances, tandis que CPLEX échoue en partie. Ainsi, pour $K = 9$, 17 instances ne sont pas résolues en 1 000 s alors que le temps moyen de B&B est de 3,4 s. On remarque aussi que le nombre de nœuds explorés par CPLEX est plus important sur cette formulation. Cela est probablement dû aux techniques de linéarisation utilisées par

Problème		B&B			CPLEX		
		Temps (s)	Nds (10^3)	E	Temps (s)	Nds (10^3)	E
$\hat{\mathcal{P}}_{2/0}$	$K = 5$	**0.5**	1.03	0	**3.0**	1.71	0
	$K = 7$	**8.0**	15.23	0	**21.0**	28.80	0
	$K = 9$	**115.3**	198.26	1	**232.9**	369.96	6
$\hat{\mathcal{P}}_{0/2}$	$K = 5$	**0.1**	0.23	0	**25.7**	6.71	0
	$K = 7$	**0.8**	2.54	0	**114.8**	49.54	2
	$K = 9$	**3.4**	8.04	0	**346.3**	107.78	17
$\hat{\mathcal{P}}_{2+0}$	$K = 5$	**0.7**	2.02	0	**3.2**	1.98	0
	$K = 7$	**11.6**	24.38	0	**11.6**	19.23	0
	$K = 9$	**87.5**	186.21	4	**78.2**	142.50	2

TABLE 5.1 – Comparaison de B&B et CPLEX pour des problèmes de déconvolution parcimonieuse en fonction du nombre K de variables non nulles. Temps de calcul, nombre de nœuds explorés (Nds) et nombre de problèmes non résolus en 1 000 s (E). Les moyennes ne sont faites que sur les instances qui ont pu être résolues en moins de 1000 s par les deux algorithmes.

CPLEX pour gérer les contraintes quadratiques, rendant les relaxations calculées moins performantes. D'autre part, pour cette formulation ayant une fonction objectif discrète (la norme ℓ_0). L'élagage d'un nœud peut être raffiné en utilisant une méthode de coupe simple, décrite en section 3.3.6. Cette coupe qui consiste à élaguer un nœud dès que l'écart entre sa borne inférieure et la borne supérieure globale devient inférieure à un, va forcément réduire le nombre total de nœuds explorés.

Enfin, le temps de calcul par nœud est ici sept à huit fois plus faible dans notre cas, ce qui était prévisible, la méthode homotopique gérant aussi efficacement les relaxations pour cette formulation que pour les précédentes – ce qui n'est pas le cas des méthodes génériques utilisées par CPLEX.

3. **Pour les problèmes $\hat{\mathcal{P}}_{2+0}$** : nous remarquons, contrairement aux autres problèmes, que le nombre de nœuds explorés est légèrement plus faible pour CPLEX que pour notre algorithme. En revanche, B&B reste meilleur que CPLEX pour $K = 5$ (environ 4,5 fois plus rapide) et les deux algorithmes sont comparables pour $K = 7$, à nouveau en raison d'un calcul plus efficace des relaxations à chaque nœud. Enfin, les instances des problèmes $\hat{\mathcal{P}}_{2+0}$ avec $K = 9$ constituent la seule classe de problèmes pour laquelle CPLEX est (légèrement) plus rapide que B&B et résout plus d'instances en moins de 1 000 s.

Pour une comparaison complémentaire aux résultats de la Table 5.1, nous utilisons les profils de performance [Dolan and Moré, 2002] introduits en Section 4.4.2, qui présentent la proportion de problèmes résolus en fonction du *rapport de performance* : pour chaque instance, on calcule le rapport entre le temps de calcul de chaque algorithme et le temps de calcul de l'algorithme le plus rapide. Ensuite, les profils de performance représentent les fonctions de répartition de ces ratios. Nous présentons les résultats de chaque problème séparément.

Les profils de performance obtenus sur les 50 instances considérées pour le problème $\hat{\mathcal{P}}_{2/0}$ pour différentes valeurs de $K = 5$, 7 et 9 sont présentés dans la partie du haut de la Figure 5.1. Nous retrouvons le résultat que B&B et CPLEX résolvent avec succès tous les problèmes pour $K = 5$ et $K = 7$ puisque leurs courbes représentatives atteignent la valeur 1 (pour B&B, la courbe est représentée par un seul point car l'algorithme a pu résoudre toutes les instances en étant le plus rapide). Pour $K = 9$, B&B résout 98% et CPLEX résout 88% des problèmes. Notre algorithme B&B est toujours meilleur que CPLEX pour $K = 5$ et 7, et sur 85% des problèmes pour $K = 9$ (points à gauche de chaque courbe). De

plus CPLEX a besoin de 5,5 fois (respectivement 26 fois) le temps de B&B pour résoudre toutes les instances pour $K = 7$ (respectivement pour $K = 5$).

Pour $\hat{\mathcal{P}}_{2/0}$, nous pouvons voir à partir de partie du milieu de la figure 5.1 que notre algorithme B&B résout avec succès tous les problèmes pour les différentes valeurs de K durant le temps autorisé, alors que CPLEX n'en résout que 96% pour $K = 7$ et 62% pour $K = 9$. CPLEX est loin d'être compétitif sur cette classe de problème, il a besoin de plus de 2000 fois le temps de B&B pour résoudre tous les problèmes pour $K = 5$, presque 400 fois le temps de B&B pour résoudre 90% des problèmes pour $K = 7$ et presque 400 fois le temps de B&B pour résoudre 60% des problèmes pour $K = 9$.

Enfin pour le problème pénalisé $\hat{\mathcal{P}}_{2+0}$, les profils de performance sont représentés dans la partie du bas de la figure 5.1. Nous remarquons que les deux algorithmes B&B et CPLEX résolvent avec succès tous les problèmes pour $K = 5$ et $K = 7$ dans le temps autorisé. En revanche, CPLEX est moins rapide sur toutes les instances pour $K = 5$ et sur 90% pour $K = 7$. De plus, il met plus de 27 fois (respectivement 14 fois) le temps de B&B pour résoudre tous les problèmes pour $K = 5$ (respectivement pour $K = 7$). Pour $K = 9$, les performances des deux algorithmes sont très proches avec un avantage mineur pour CPLEX, qui résout un peu plus de problèmes (96% de problèmes sont résolus pour CPLEX et seulement 92% pour notre algorithme B&B).

5.3 Problèmes de sélection de variables

Nous considérons à présent des problèmes simulés de sélection de variables similaires à ceux de la section 3.5.2, avec des données générées aléatoirement, où la matrice $\mathbf{H} \in \mathbb{R}^{m \times n}$ et le nombre d'inconnues vaut $n = 2m$. Les coefficients de $\mathbf{H}$ sont indépendants et identiquement distribués selon une loi normale centrée de variance unitaire, le choix des coefficients non nuls est aléatoire et les amplitudes associées sont générées selon $u + \mathrm{sgn}(u)$, où u est gaussien centré de variance unitaire, afin d'éviter des valeurs arbitrairement faibles. En raison de la nature aléatoire de $\mathbf{H}$ (faible corrélation entre les colonnes), ce type de problème est plus facile à résoudre, ce qui nous permet de monter en dimension : nous considérons ici le nombre de variables $n = 500$ et $n = 1\,000$, avec une cardinalité qui varie de $K = 5$ à $K = 15$. Les données sont contaminées par un bruit blanc gaussien de rapport signal sur bruit égal à 10 dB. Le paramètre M et les paramètres ϵ et μ sont réglés comme en Section 5.2.

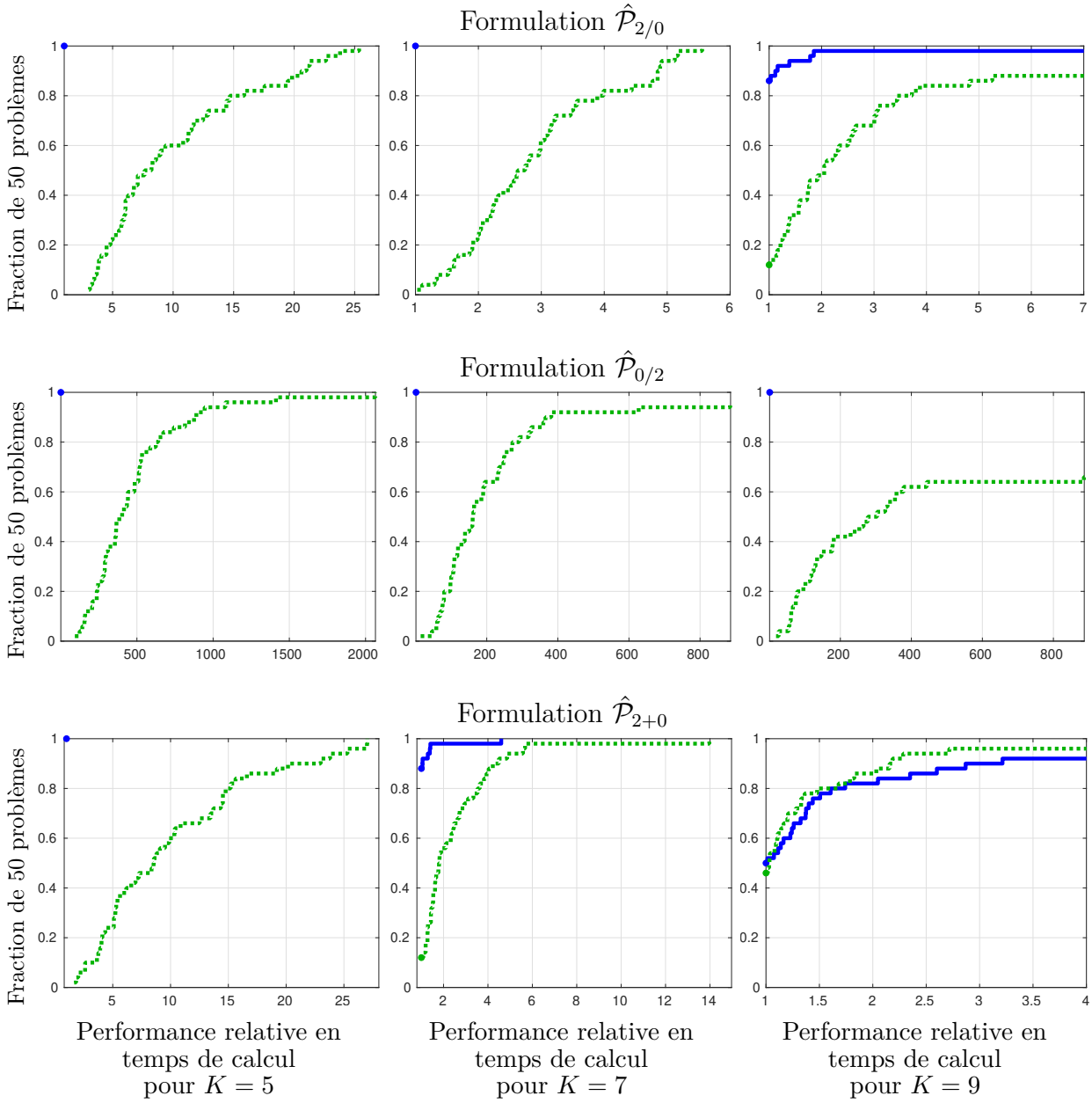


FIGURE 5.1 – Profils de performance en temps de calcul obtenus sur 50 problèmes simulés de déconvolution impulsionnelle pour les trois formulations en fonction de K, avec les algorithmes B&B (−) et CPLEX (...).

Sur ces problèmes, les meilleurs résultats de notre algorithme B&B, présentés ci-après, ont été obtenus avec l'algorithme d'homotopie (voir Section 3.3) pour le calcul des relaxations continues des trois problèmes $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{2/0}$ et $\hat{\mathcal{P}}_{2+0}$, combiné avec la règle de branchement *Amplitude Maximale* (AM) et la stratégie de *recherche en profondeur du côté branche supérieure d'abord* (DUFS) décrite en Section 4.3. Les résultats moyennés sur 50 instances sont consignés dans la Table 5.2. Nous pouvons d'abord remarquer que ces problèmes sont plus faciles que les problèmes de déconvolution puisque le nombre de nœuds explorés est largement inférieur. Surtout, les performances de notre algorithme sont maintenant très nettement supérieures à celles de CPLEX, quelle que soient la formulation et la complexité du problème. Nous analysons maintenant ces performances de manière détaillée.

1. **Pour les problèmes** $\hat{\mathcal{P}}_{2/0}$: commençons par les résultats avec $n = 500$, notre algorithme B&B est plus efficace que CPLEX pour les différentes valeurs de K. Il est 42 fois plus rapide que CPLEX pour $K = 5$, 12 fois pour $K = 10$ et 14 fois pour $K = 15$. Il résout aussi plus d'instances dans le temps maximal autorisé : pour les instances difficiles $K = 15$, 45 instances sur 50 ne sont pas résolues par CPLEX en 1 000 s, et seulement 24 instances (presque la moitié) ne sont pas résolues par B&B. Le temps de calcul par nœud est aussi en faveur de notre algorithme B&B grâce à la rapidité de la méthode homotopique : il est 31 fois plus rapide que CPLEX pour $K = 5$, 9 fois pour $K = 10$ et 10 fois pour $K = 15$. Le rapport devient plus important en plus grande dimension avec $n = 1\,000$, on passe à 64 fois plus rapide pour $K = 5$, 37 fois pour $K = 10$ et 19 fois pour $K = 15$. De même pour le nombre d'instances non résolues, par exemple pour les instances difficiles $K = 15$, 42 ne sont pas résolues par CPLEX en temps autorisé, alors que seulement 7 instances ne sont pas résolues par B&B.

2. **Pour les problèmes** $\hat{\mathcal{P}}_{0/2}$: tout comme dans le cas de la déconvolution, les résultats sont encore plus favorables à notre algorithme B&B, qui surpasse largement CPLEX sur toutes les instances en temps de calcul et en nombre de nœuds. Ainsi, pour les instances les plus faciles avec $K = 5$, B&B est 1 800 fois à 1 900 fois plus rapide pour respectivement $n = 1\,000$ et $n = 500$. De plus, B&B résout toutes les instances, tandis que CPLEX résout seulement 33 instances sur 50 pour $n = 500$ et 22 pour $n = 1\,000$. Pour les instances les plus difficiles avec $K = \{10, 15\}$, que ce soit avec $n = 500$ ou $n = 1\,000$, B&B résout toutes les instances, alors que CPLEX n'arrive à résoudre aucune d'elles dans le temps maximal autorisé.

Problème			B&B			CPLEX		
			Temps (s)	Nds (10^3)	F	Temps (s)	Nds (10^3)	F
$n = 500$	$\mathcal{P}_{2/0}$	$K = 5$	**0.4**	0.03	0	**16.8**	0.04	0
		$K = 10$	**11.6**	0.65	0	**148.2**	0.91	5
		$K = 15$	**32.5**	2.09	24	**451.6**	2.81	45
	$\mathcal{P}_{0/2}$	$K = 5$	**0.1**	0.01	0	**191.1**	4.28	17
		$K = 10$	**3.5**	0.32	0	-	15.89	50
		$K = 15$	**155.1**	10.28	7	-	12.28	50
	$\mathcal{P}_{2+0}$	$K = 5$	**0.6**	0.14	0	**20.2**	0.16	0
		$K = 10$	**10.9**	0.97	0	**169.7**	2.43	3
		$K = 15$	**16.7**	1.34	26	**234.2**	4.94	47
$n = 1\,000$	$\mathcal{P}_{2/0}$	$K = 5$	**1.1**	0.02	0	**70.7**	0.03	0
		$K = 10$	**11.1**	0.13	0	**411.1**	0.38	5
		$K = 15$	**33.9**	0.38	7	**665.3**	0.72	42
	$\mathcal{P}_{0/2}$	$K = 5$	**0.2**	0.01	0	**360.9**	0.28	28
		$K = 10$	**3.6**	0.05	0	-	0.18	50
		$K = 15$	**113.1**	1.02	0	-	0.18	50
	$\mathcal{P}_{2+0}$	$K = 5$	**3.2**	0.14	0	**154.0**	0.32	0
		$K = 10$	**32.0**	0.44	0	**389.3**	1.09	3
		$K = 15$	**19.5**	0.52	11	**665.3**	2.15	40

TABLE 5.2 – Efficacité de B&B et CPLEX pour des problèmes de sélection de variables aléatoire en fonction du nombre K de variables non nulles. Temps de calcul, nombre de nœuds explorés (Nds) et nombre de problèmes non résolus en 1 000 s (E). Les moyennes ne sont faites que sur les instances qui ont pu être résolues en moins de 1 000 s par les deux algorithmes.

3. **Pour les problèmes** $\hat{\mathcal{P}}_{2+0}$: de manière similaire aux résultats obtenus pour les problèmes $\hat{\mathcal{P}}_{2/0}$, B&B est toujours meilleur que CPLEX : il résout plus de problèmes en 1 000 s et est toujours beaucoup plus rapide (de 14 à 33 fois pour $n = 500$ et de 12 à 48 fois pour $n = 1\,000$).

Nous présentons enfin en Figure 5.2 les profils de performance obtenus en regroupant les 450 instances considérées pour les problèmes $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, et pour les différentes valeurs de K (5, 10 et 15), pour $n = 500$ (à gauche) et $n = 1\,000$ (à droite).

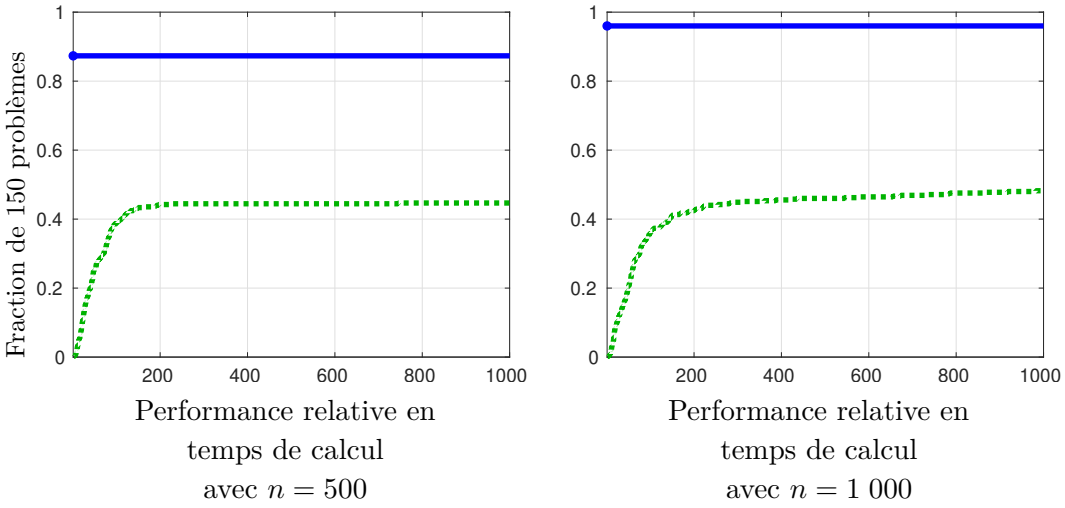


FIGURE 5.2 – Profils de performance en temps de calcul obtenus sur 150 problèmes simulés des trois formulations mélangées $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, avec les algorithmes B&B (−) et CPLEX (...).

Les résultats rejoignent ceux présentés dans la Table 5.2, où nous pouvons voir que notre algorithme B&B surpasse largement CPLEX sur l'ensemble des problèmes $\hat{\mathcal{P}}_{2/0}$, $\hat{\mathcal{P}}_{0/2}$ et $\hat{\mathcal{P}}_{2+0}$, que ce soit pour $n = 500$ ou pour $n = 1\,000$. Notre algorithme B&B est plus rapide que CPLEX sur la totalité des instances et il résout également plus de problèmes durant le temps autorisé : pour $n = 500$, 88% des problèmes sont résolus par B&B alors que CPLEX n'atteint pas les 45% ; pour $n = 1\,000$, la marge devient plus importante où plus de 95% des problèmes sont résolus par B&B tandis que CPLEX n'a résolu que 50% des problèmes.

5.4 Conclusion

Nous avons présenté dans ce chapitre une étude des performances de notre algorithme branch-and-bound, en assemblant les différentes parties développées dans les chapitres

précédents, en comparaison au solveur MIP CPLEX, sur deux types de problèmes d'approximation parcimonieuse.

— Pour des problèmes de déconvolution, notre algorithme surpasse généralement CPLEX que se soit en temps de calcul total ou en nombre de nœuds explorés. En particulier, pour la formulation $\hat{\mathcal{P}}_{0/2}$ avec des contraintes quadratiques, CPLEX s'avère très peu performant alors que, à l'inverse, c'est sur cette formulation que notre algorithme est le plus rapide. Nous avons montré que ces performances résultaient à la fois des stratégies de branchement et de parcours de l'arbre de recherche proposées et de l'efficacité des méthodes développées pour le calcul des relaxations continues à chaque nœud.

— Sur des problèmes simulés de sélection de variables avec un dictionnaire aléatoire, de plus grande dimension, notre algorithme s'avère beaucoup plus efficace que CPLEX pour les trois formulations, avec également un rapport plus élevé pour la formulation $\hat{\mathcal{P}}_{0/2}$. Ces derniers résultats illustrent la capacité de notre algorithme à monter en dimension, grâce à nouveau au développement de stratégies dédiées, alors que les performances du solveur générique s'écroulent.

Chapitre 6

Démélange Spectral : Optimisation exacte en norme ℓ_0 et contraintes de parcimonie structurée à l'aide de MIP

Contents

6.1 Introduction : Démélange spectral parcimonieux

L'imagerie hyperspectrale est une technique désormais très répandue dans de nombreuses applications telles que la télédétection, les géosciences, la planétologie ou encore l'étude des écosystèmes [Chang, 2003]. Cette technique combine l'imagerie et la spectroscopie, où chaque image est prise dans une bande étroite du spectre électromagnétique : à titre comparatif, si l'œil humain peut voir la lumière en trois bandes (rouge, vert et bleu), l'imagerie hyperspectrale permet de visualiser dans un large domaine de bandes, ce qui permet d'avoir plus d'informations. Cependant, en raison de la faible résolution spatiale de la plupart des images hyperspectrales qui sont souvent prises de loin (par des instruments à bord de satellite, ou de drones), l'information dans un pixel résulte souvent du mélange de plusieurs composantes, ce qui peut causer des difficultés d'interprétation en pratique. De ce fait, il est important de bien décomposer les spectres mesurés pour avoir une bonne interprétation de l'image hyperspectrale. Ce problème est connu sous le nom de démélange spectral [Singer and McCord, 1979; Keshava and Mustard, 2002; Bioucas-Dias et al., 2012].

Sous l'hypothèse d'un modèle de mélange linéaire [Adams et al., 1986], deux grandes familles d'approches ont été discutées dans la littérature : l'approche dite non supervisée et l'approche supervisée. Dans une approche non supervisée, l'opération de démélange se compose de deux étapes : l'extraction des spectres « purs » des composants de référence *(endmembers)* qui sont supposés être une représentation des matériaux purs présentés dans l'image, et l'estimation des proportions d'abondance qui sont leurs pourcentages respectifs dans chaque pixel (voir par exemple [Parra et al., 2000]). Dans une approche supervisée (voir par exemple [Bioucas-Dias et al., 2012]), le problème du démélange spectral se concentre sur l'estimation des proportions d'abondance, à partir d'un ensemble de spectres constituant un dictionnaire de spectres de référence. Ce dictionnaire est une collection de signatures spectrales pures qui sont généralement acquises par des mesures spectroscopiques en laboratoire. Avoir une bibliothèque spectrale avec un grand nombre de spectres de référence est un problème critique, en particulier lorsque ceux-ci sont fortement corrélés [Bioucas-Dias et al., 2012; Iordache et al., 2011]. De plus, les bibliothèques spectrales risquent de contenir des spectres de minéraux très semblables les uns aux autres car ils représentent des minéraux dont les propriétés de réflectance sont très proches. Dans ces cas, l'approche de résolution standard basée sur les moindres carrés sous contraintes de positivité et éventuellement de somme à un [Heinz et al., 2001] peut être inefficace et

l'incorporation de contraintes plus fortes peut servir à régulariser le problème. Dans ce chapitre, nous étudions différents types de contraintes visant à améliorer les modèles de mélange.

— La première est la **parcimonie** : en pratique, le mélange ne peut contenir que quelques composantes actives extraites du dictionnaire, c'est-à-dire que le vecteur d'abondances correspondantes a une faible norme ℓ_0. Des contraintes explicites de parcimonie pour le démélange spectral ont été proposées avec la norme ℓ_1 [Guo et al., 2009; Iordache et al., 2011] et la norme $\ell_{p,\,p<1}$ [Tuia et al., 2016], ou encore avec une méthode itérative par déflation [Greer, 2012]. Bien qu'elles soient intéressantes en termes de coût de calcul, ces approches restent sous-optimales pour le problème en norme ℓ_0.

— La deuxième est la **parcimonie structurée** : il est également intéressant d'imposer des contraintes structurelles sur le dictionnaire afin de gérer la variabilité spectrale [Zare and Ho, 2014]. Par exemple, les données expérimentales sur un minéral sont généralement acquises dans des conditions diverses, et le dictionnaire peut inclure des spectres qui correspondent à plusieurs variantes d'un même minéral, bien qu'au plus un seul soit réellement utilisé dans le démélange (contraintes d'exclusivité de groupe).

— Enfin, imposer une **valeur minimale** sur chaque abondance non nulle est physiquement légitime. C'est également un moyen de créer de la parcimonie sur la solution : associée à la positivité et à la contrainte de somme à un, cette contrainte permet de concentrer la solution en quelques composantes, sans imposer directement un nombre maximal d'abondances non nulles.

À notre connaissance, ces contraintes n'ont jamais été abordées de façon exacte pour des problèmes de démélange spectral, certainement en raison de leur difficulté où les contraintes de cardinalité ainsi que les contraintes logiques relèvent de l'optimisation combinatoire, généralement NP-difficile. Une des premières contributions de ce chapitre est de les incorporer *via* des formulations dédiées en programmes en nombres mixtes (MIP), qui offrent un cadre naturel pour de telles contraintes. Ensuite, nous étudions d'une part la résolution numérique de ces contraintes dans une approche parcimonieuse exacte sur le problème de démélange spectral. D'autre part, nous évaluons l'apport de cette approche relativement aux méthodes de résolution approchées en termes de qualité de solution. Dans la Section 6.2, nous introduisons les notations et le modèle de mélange linéaire. Dans

la Section 6.3, nous présentons différents types de contraintes permettant d'incorporer de la parcimonie dans la solution recherchée. Dans la Section 6.4, nous proposons une reformulation en programme en nombres mixtes de ces différents problèmes. Des résultats de simulations sont présentés en Section 6.5, puis une conclusion est donnée en Section 6.6.

6.2 Démélange spectral linéaire et optimisation

Le modèle de mélange linéaire est souvent utilisé pour le problème de démélange spectral [Singer and McCord, 1979; Keshava and Mustard, 2002]. Ce modèle suppose que chaque spectre observé (spectre mesuré) est linéairement pondéré par les spectres purs présents dans le mélange :

$$\boldsymbol{y} = \sum_{q=1}^{m} a_q \boldsymbol{s}_q + \boldsymbol{\varepsilon} = \mathbf{S}\boldsymbol{a} + \boldsymbol{\varepsilon},$$

où $\boldsymbol{y}$ est le spectre de réflectance observé acquis dans n bandes spectrales, $\mathbf{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m]$ représente le dictionnaire de spectres purs qui peut contenir un grand nombre d'éléments, $\boldsymbol{a}$ est le vecteur d'abondances où $a_q \in \mathbb{R}^+$ est l'abondance associée au $q^{\text{ème}}$ composant et enfin $\boldsymbol{\varepsilon}$ représente le bruit et l'erreur de modélisation. Une illustration du modèle de mélange est représentée en Figure 6.1.

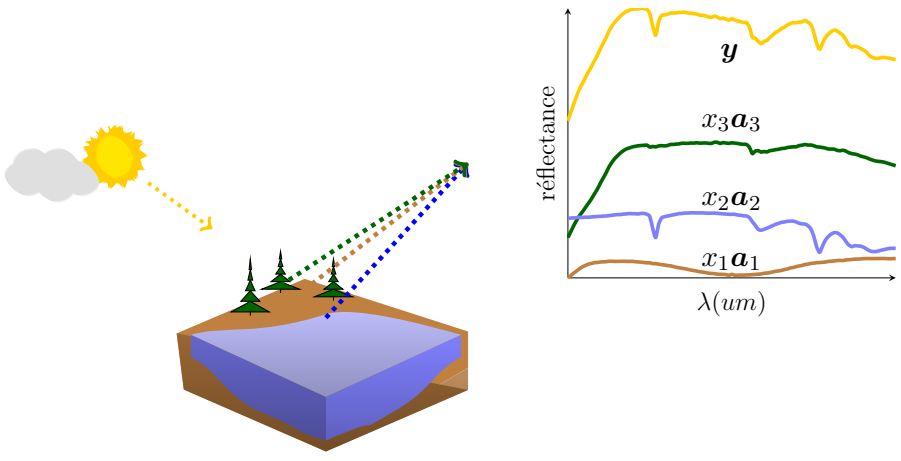


FIGURE 6.1 – Mélange linéaire de spectres de réflectance. La réflectance mesurée est une somme pondérée des rayonnements des minéraux présents. Le spectre mélangé $\boldsymbol{y}$ dans ce cas est une combinaison linéaire de 3 spectres élémentaires $\boldsymbol{s}_1$, $\boldsymbol{s}_2$ et $\boldsymbol{s}_3$.

Dans une approche supervisée, on ne s'intéresse qu'à l'estimation des abondances. Pour traduire des considérations physiques (les abondances représentent des pourcentages), des

contraintes de positivité et de somme à un sont en général ajoutées aux inconnues :

$$\forall q = 1, \ldots, m,\ a_q \geq \mathbf{0} \text{ et } \sum_{q=1}^{m} a_q = 1.$$

Le problème d'estimation au sens des moindres carrés, souvent dénommé FCLS pour *Fully-Constrained Least-Squares* [Heinz et al., 2001], s'écrit alors :

$$\begin{aligned} \text{FCLS} : \quad & \min_{\boldsymbol{a} \in \mathbb{R}^n} \ \tfrac{1}{2} \|\boldsymbol{y} - \mathbf{S}\boldsymbol{a}\|^2 \\ & \text{s.c.} \quad \boldsymbol{a} \geq \mathbf{0} \\ & \qquad \textstyle\sum_{q=1}^{m} a_q = 1 \end{aligned} \tag{6.1}$$

Les bibliothèques spectrales contiennent souvent un grand nombre de spectres de minéraux dont certains se ressemblent beaucoup. Ceci est un problème critique, surtout en présence de bruit. Dans ce cas, l'approche standard basée sur les moindres carrés sous contraintes de positivité et de somme à un [Heinz et al., 2001] peut être inefficace. La figure 6.2 montre ainsi un exemple de solution de FCLS sans bruit et en présence de bruit, sur un problème de démélange spectral simulé on utilisant une bibliothèque d'environ 500 spectres issus de l'*United States Geological Survey* [Clark et al., 2003b]. On peut voir que sans bruit, FCLS trouve parfaitement les composantes recherchées puisque la vraie solution minimise le critère (qui vaut alors 0). En présence de bruit, le problème devient plus compliqué, et la solution FCLS n'est plus satisfaisante : elle contient un grand nombre d'abondances de faible valeur qui perturbent l'interprétation et peuvent empêcher la détection des vraies composantes. Dans un tel contexte, il semble naturel de régulariser davantage le problème en ajoutant des contraintes plus fortes.

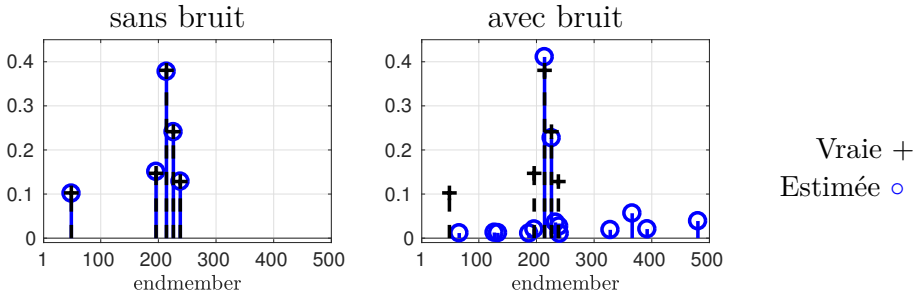


FIGURE 6.2 – Exemples de solutions estimées par FCLS (représentées par des cercles bleus ∘) sans bruit à gauche et en présence de bruit avec $RSB = 45$ dB à droite. Le vrai mélange est représenté en noir par +.

6.3 Contraintes favorisant la parcimonie

La parcimonie est la propriété qui consiste à n'utiliser qu'un faible nombre de spectres élémentaires purs afin de représenter le mélange spectral. Même si la contrainte de positivité dans le problème (6.1) produit des solutions pour lesquelles certains coefficients sont nuls, nous venons de voir qu'elle peut s'avérer insuffisante en présence de bruit. Il peut alors sembler naturel de rechercher des solutions plus parcimonieuses [Iordache et al., 2011; Greer, 2012; Bioucas-Dias et al., 2012]. Dans cette partie, nous présentons plusieurs contraintes inhabituellement abordées de façon exacte en démélange spectral, qui peuvent créer de la parcimonie.

6.3.1 Contrainte de parcimonie en norme ℓ_0

La façon la plus naturelle pour imposer de manière plus explicite cette propriété de parcimonie (un faible nombre de spectres élémentaires pour représenter le mélange) est de contraindre la norme ℓ_0 de la solution. Le problème de démélange linéaire parcimonieux peut alors être formulé comme suit :

$$\begin{aligned} \text{FCLS}_{\ell_0}: \quad \min_{\boldsymbol{a}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{S}\boldsymbol{a}\|^2 \\ \text{s.c.} \quad & \boldsymbol{a}\geq\mathbf{0} \\ & \textstyle\sum_{q=1}^{m} a_q = 1 \\ & \|\boldsymbol{a}\|_0 \leq K. \end{aligned} \tag{6.2}$$

La norme ℓ_0 rend le problème de démélange essentiellement combinatoire. Afin de limiter le coût de calcul (voir chapitre 1, Section 1.3), les approches classiques parcimonieuses reposent soit sur la relaxation de la norme ℓ_0 en norme ℓ_1, soit sur la mise en œuvre d'algorithmes de constructions itératives.

Remplacer la norme ℓ_0 par la norme ℓ_1, convexe, est un usage classique reconnu depuis longtemps par les méthodes d'optimisation parcimonieuse, voir § 1.3.1. Cependant, il n'est pas approprié pour le problème de démélange spectral abordé ici : comme les abondances représentent des proportions, toute formulation reposant sur la norme ℓ_1 pour imposer un certain niveau de parcimonie est inappropriée, puisque la norme ℓ_1 des abondances vaut 1. Par conséquent, la norme ℓ_1 ne peut pas être utilisée pour créer plus de parcimonie que la solution FCLS. En pratique, en présence des erreurs de mesure et de modélisation,

certains auteurs préfèrent parfois contraindre la somme des abondances à une valeur $t < 1$ (voir par exemple [Guo et al., 2009; Iordache et al., 2011]). Le problème devient alors :

$$\begin{aligned} \min_{\boldsymbol{a}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{S}\boldsymbol{a}\|^2 \\ \text{s.c.} \quad & \boldsymbol{a} \geq \mathbf{0} \\ & \textstyle\sum_{q=1}^{m} a_q \leq t \end{aligned} \tag{6.3}$$

Le fait d'imposer que la somme des abondances est inférieure à 1 peut agir comme une régularisation parcimonieuse. Cependant, dans nos expériences (voir Section 6.5, Figure 6.7), la meilleure solution basée sur la norme ℓ_1 a toujours été obtenue avec $\|\boldsymbol{a}\|_1$ très proche de 1, c'est-à-dire la solution de FCLS.

La régularisation non convexe à base de la "norme" ℓ_p, $p < 1$ proposée dans [Tuia et al., 2016] semble être mieux adaptée pour imposer la parcimonie dans (6.1), bien que les stratégies d'optimisation dans [Tuia et al., 2016] ignorent également la contrainte de somme à un. Les algorithmes itératifs gloutons classiques [Tropp, 2004] ne sont pas non plus adaptés pour prendre en compte la contrainte de somme à un puisque les spectres du dictionnaire $\mathbf{S}$ sont des spectres de *réflectance* et ne peuvent pas être normalisés sans affecter l'estimation des abondances [Greer, 2012]. En revanche, il existe une méthode itérative [Greer, 2012] de type *backward selection*, qui peut fournir une solution parcimonieuse à notre problème de démélange. Partant de la solution FCLS, elle consiste à éliminer itérativement les petites composantes non nulles et à répéter la procédure jusqu'à atteindre une condition d'arrêt (par exemple, une solution avec un nombre déterminé d'abondances non nulles).

6.3.2 Contraintes d'exclusivité de groupe

Afin d'améliorer la précision du modèle dans le cadre de démélange spectral à base de dictionnaire, un minéral donné peut être représenté par plusieurs spectres possibles, en fonction de la taille des grains, de la géométrie d'acquisition, des impuretés, *etc.* De plus, les échantillons de minéraux utilisés pour les mesures en laboratoire peuvent provenir de différents endroits sur Terre, ce qui entraîne de légères différences au niveau des impuretés. Certains minéraux (*e.g.,* olivine ou pyroxène) sont en fait des solutions à l'état solide avec un continuum de compositions, créant une *variabilité* dans la forme spectrale (voir Figure 6.3).

Dans de tels cas, la bibliothèque spectrale contient plusieurs variantes d'un même minéral, alors qu'on réalité au plus un minéral parmi ces variantes va être présent dans le

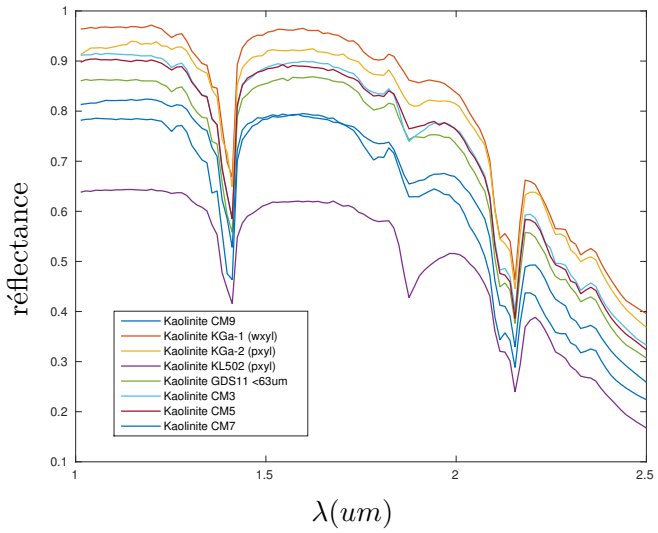


FIGURE 6.3 – Exemple de la variabilité de la Kaolinite extrait de la base de données fournie par le *United States Geological Survey* [Clark et al., 2003b].

mélange. Par conséquent, il peut être intéressant sur le plan pratique d'imposer que le mélange contienne au maximum un seul élément spectral par groupe G_j. Nous appelons ces contraintes *exclusivité de groupe* (EG). Cette contrainte est mentionnée dans [Iordache et al., 2011], mais n'est finalement pas prise en compte de façon exacte. La figure 6.4 illustre ces contraintes, qui sont également « de type ℓ_0 ». Séparons les indices des composantes

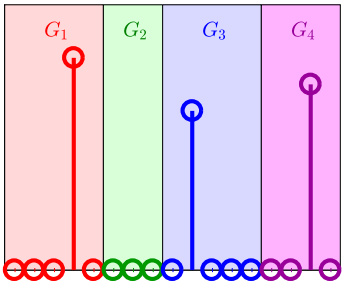


FIGURE 6.4 – Illustration des contraintes d'exclusivité de groupe (EG) : chaque couleur représente un groupe G_j.

en J groupes G_1, ..., G_J, et supposons que $\boldsymbol{a}^{(j)}$ désigne l'ensemble des abondances dans chaque groupe. Les contraintes d'exclusivité de groupe peuvent être formulées en utilisant la norme ℓ_0 sur les composantes de chaque groupe comme suit :

$$\|\boldsymbol{a}^{(j)}\|_0 \leq 1, \quad \forall j = 1, \ldots, J.$$

Le problème d'optimisation associé à cette contrainte est alors le suivant :

$$\begin{aligned} \text{FCLS}_{\text{EG}} : \quad \min_{\boldsymbol{a}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{S}\boldsymbol{a}\|^2 \\ \text{s.c.} \quad & \boldsymbol{a}\geq\mathbf{0} \\ & \textstyle\sum_{q=1}^{m} a_q = 1 \\ & \|\boldsymbol{a}^{(j)}\|_0 \leq 1,\ \forall j = 1,\ldots,J \end{aligned} \tag{6.4}$$

Nous tenons à souligner que plus les groupes sont de grande taille, plus ces contraintes d'exclusivité de groupe deviennent pertinentes et peuvent permettre d'avoir des solutions plus parcimonieuses.

6.3.3 Contrainte des abondances significatives

Avec un dictionnaire spectral fortement corrélé et en présence de bruit, la solution FCLS présente généralement des composantes de faible amplitude (éventuellement nombreuses). De tels artefacts peuvent être facilement éliminés par une étape de post-traitement ; cependant, ces composantes peuvent aussi interférer avec la détection des vraies composantes, comme nous le verrons dans l'exemple de la figure 6.6. Pour éviter de tels artefacts, on peut imposer une valeur minimale aux abondances non nulles :

$$a_q \neq 0 \Rightarrow a_q \geq \tau\ , \quad q = 1,\ldots,n, \quad \text{avec } \tau > 0, \tag{6.5}$$

afin de détecter uniquement les contributions importantes. Nous appelons ces contraintes des abondances significatives (AS) (voir Figure 6.5).

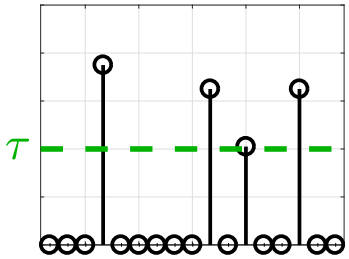


FIGURE 6.5 – Illustration de la contrainte des abondances significatives (AS) : le seuil τ représenté en vert (- -).

Le problème d'optimisation associé à cette contrainte s'écrit alors :

$$\begin{aligned} \text{FCLS}_{\text{AS}} : \quad \min_{\boldsymbol{a}\in\mathbb{R}^n, \boldsymbol{b}\in\{0,1\}^n} \quad & \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{S}\boldsymbol{a}\|^2 \\ \text{s.c.} \quad & \boldsymbol{a} \geq \mathbf{0} \\ & \textstyle\sum_{q=1}^{m} a_q = 1 \\ & (a_q \neq 0 \Rightarrow a_q \geq \tau)\ , q = 1, \ldots, n \end{aligned} \tag{6.6}$$

Cette contrainte d'abondances significatives représente une autre formulation favorisant la parcimonie, qui peut être préférée au problème en norme ℓ_0 (6.2). En effet, en présence de cette contrainte, nous avons au plus $\frac{1}{\tau}$ composantes non nulles puisque la somme des composantes non nulles vaut 1. En pratique, il peut être plus facile de régler le seuil associé τ que de spécifier le nombre d'abondances non nulles recherchées dans le mélange.

Enfin, nous concluons cette partie en mentionnant que les contraintes ci-dessus (de parcimonie en norme ℓ_0, d'exclusivité de groupe et des abondances significatives) peuvent évidemment être mélangées afin de construire des modèles plus sophistiqués.

6.4 Reformulations en MIP

Nous nous intéressons maintenant aux reformulations en MIP du problème de démélange spectral et des contraintes introduites en Section 6.3. Ces reformulations reposent sur l'introduction de variables de décision binaires $b_q,\ \forall q = 1, \ldots, n$, qui modélisent la présence ou l'absence d'un spectre donné dans le mélange :

$$b_q = 0 \Leftrightarrow a_q = 0. \tag{6.7}$$

Notons que pour les problèmes parcimonieux classiques, généralement non bornés, des contraintes artificielles de borne ($|a_q| < M$), dites *big-M*, sont généralement nécessaires afin que la contrainte logique puisse être traduite linéairement par $-Mb_q \leq a_q \leq Mb_q$. Le réglage de M peut être un problème critique puisque sa valeur a un impact très important sur le temps de calcul de la résolution (voir la discussion à ce sujet au § 1.4.2).

Pour le problème de démélange spectral, les contraintes de positivité et de somme à un imposent naturellement des bornes sur les abondances : $0 \leq a_q \leq 1$. Par conséquent, la contrainte logique (6.7) peut être *exactement* traduite par :

$$0 \leq a_q \leq b_q,$$

ce qui donne un ensemble de n contraintes linéaires mêlant des variables binaires et des variables réelles. Nous présentons maintenant les reformulations de problème de démélange intégrant le différentes contraintes favorisant la parcimonie.

— **Contrainte de parcimonie en norme** ℓ_0 : Comme présenté en Section 1.4.1, la norme ℓ_0 s'écrit linéairement : $\|\boldsymbol{a}\|_0 = \sum_q b_q$, et le problème (6.2) peut être reformulé en MIP comme suit :

$$\text{FCLS}_{\ell_0} : \quad \min_{\boldsymbol{a}\in\mathbb{R}^n, \boldsymbol{b}\in\{0,1\}^n} \ \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{S}\boldsymbol{a}\|^2 \quad \text{s.c.} \quad \begin{array}{l} \boldsymbol{a} \geq \mathbf{0} \\ \sum_{q=1}^n a_q = 1 \\ \mathbf{0} \leq \boldsymbol{a} \leq \boldsymbol{b} \\ \sum_{q=1}^n b_q \leq K \end{array} . \tag{6.8}$$

— **Contraintes d'exclusivité de groupe (EG)** :
Cette contrainte, qui n'est autre qu'une borne sur la norme ℓ_0 d'une partie des variables, peut être similairement traduite linéairement par une somme de variables binaires mais opérant seulement sur la partie des variables qui appartiennent au même groupe. Avec les notations de la section 6.3.2, la contrainte peut être reformulée par :

$$\sum_{q\in G_j} b_q \leq 1, \quad \forall j = 1, \ldots, J$$

et le problème MIP s'écrit de la façon suivante :

$$\text{FCLS}_{\text{EG}} : \quad \min_{\boldsymbol{a}\in\mathbb{R}^n, \boldsymbol{b}\in\{0,1\}^n} \ \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{S}\boldsymbol{a}\|^2 \quad \text{s.c.} \quad \begin{array}{l} \boldsymbol{a} \geq \mathbf{0} \\ \sum_{q=1}^n a_q = 1 \\ \mathbf{0} \leq \boldsymbol{a} \leq \boldsymbol{b} \\ \sum_{q\in G_j} b_q \leq 1, \quad \forall j = 1, \ldots, J \end{array} . \tag{6.9}$$

— **Contrainte des abondances significatives (AS)** :
Cette contrainte décrite en Section 6.3.3, qui consiste à imposer une valeur minimale sur les abondances non nulles peut être traduite par :

$$\tau b_q \leq a_q \leq b_q, \ \forall q = 1, \ldots, n$$

et le problème (6.6) peut être reformulé en MIP de la façon suivante :

$$(6.6) \Leftrightarrow \begin{array}{ll} \min\limits_{\boldsymbol{a}\in\mathbb{R}^n, \boldsymbol{b}\in\{0,1\}^n} & \frac{1}{2}\|\boldsymbol{y}-\mathbf{S}\boldsymbol{a}\|^2 \\ \text{s.c.} & \boldsymbol{a}\geq\mathbf{0} \\ & \sum_{q=1}^n a_q = 1 \\ & \tau b_q \leq a_q \leq b_q,\ \forall q = 1,\ldots,n \end{array} . \tag{6.10}$$

6.5 Résultats en imagerie hyperspectrale

Nous présentons maintenant des résultats de simulations numériques. Comme le temps de calcul de la résolution MIP est beaucoup plus élevé que celui des méthodes d'optimisation parcimonieuses classiques (sous-optimales), notre étude est centrée sur la qualité des solutions obtenues en prenant en compte les contraintes ci-dessus, ainsi que sur la faisabilité de leur mise en œuvre numérique appliquée à des problèmes pratiques. Nous utilisons ici le solveur CPLEX. Il est important de noter, cependant, que nous avons abaissé la tolérance sur le saut d'intégrité (voir le Chapitre 2, Section 2.2.4), de 10^{-4} (valeur par défaut de CPLEX) à 10^{-6}, car les réglages par défaut se sont avérés inefficaces : en raison de la très forte corrélation entre les spectres du dictionnaire, les solutions obtenues à chaque nœud exploré fournissent une modélisation très proche de la solution optimale, ce qui provoque l'arrêt prématuré de l'algorithme.

6.5.1 Construction de problèmes simulés de démélange spectral

Nous avons utilisé un dictionnaire composé de spectres en réflectance de minéraux éventuellement présents sur la surface terrestre [Schmidt et al., 2014], fourni par la base de données de l'*United States Geological Survey* [Clark et al., 2003b]). L'échantillonnage spectral représente $m = 106$ longueurs d'ondes de 1 à 2.5 microns. Le dictionnaire contient $n = 481$ spectres, repartis en $J = 85$ groupes de spectres correspondent chacun à un même minéral. Ces groupes sont composés de 2 à 18 spectres de référence et de 146 spectres de référence qui ne sont pas regroupés en groupes.

Les abondances sont générées uniformément au-dessus du seuil $\tau = 0.1$ et leur somme est égale à un. Les composantes spectrales sont choisies aléatoirement, en imposant qu'au plus une composante est sélectionnée dans chaque groupe. Le degré de parcimonie K varie entre 1 et 7, et du bruit gaussien $\boldsymbol{\epsilon}$ est ajouté avec $\mathrm{RSB}_{\mathrm{dB}} = 10\log(\|\mathbf{S}\boldsymbol{a}\|^2/\|\boldsymbol{\epsilon}\|^2)$ qui varie entre 60 et 40 dB. Ces valeurs peuvent sembler particulièrement élevées , mais sont dues à

la positivité des spectres et correspondent en fait à des niveaux de bruits faible à moyen (voir la Figure 6.6 par exemple).

Nous considérons les abondances estimées $\widehat{\boldsymbol{a}}$ obtenues en intégrant les différentes contraintes :

— les contraintes d'exclusivité de groupe (FCLS_{EG}),

— la contrainte de parcimonie en norme ℓ_0 (FCLS_{ℓ_0}),

— la contrainte de parcimonie en norme ℓ_0 et contraintes d'exclusivité de groupe ($\text{FCLS}_{\ell_0+\text{EG}}$),

— les contraintes des abondances significatives et contraintes d'exclusivité de groupe ($\text{FCLS}_{\text{AS+EG}}$).

Le seuil τ utilisé pour les contraintes d'abondances significatives est fixé à 0.1 et le paramètre K pour la contrainte en norme ℓ_0 est fixé à sa valeur réelle. Les solutions sont également comparées à la méthode par déflation proposée dans [Greer, 2012] et explicitée au § 6.3.1.

La figure 6.6 présente un exemple de résultats, obtenus avec RSB = 45 dB et $K = 5$, représentatif pour les problèmes de cette complexité. FCLS ne parvient à détecter correctement que deux spectres et produit de nombreuses détections erronées. Dans cet exemple, l'ajout de la contrainte (EG) n'améliore pas la détection par rapport à FCLS mais on peut voir que la solution est un peu plus parcimonieuse. La méthode par déflation détecte correctement trois spectres, ce qui est un peu mieux que FCLS. En revanche, $\text{FCLS}_{\ell_0+\text{EG}}$ et $\text{FCLS}_{\text{AS+EG}}$ détectent parfaitement les cinq spectres.

6.5.2 Résultats quantitatifs

Afin de focaliser sur la capacité des méthodes à détecter les composantes présentes dans les données, nous considérons à présent des résultats moyennés sur 30 réalisations aléatoires. Nous utilisons les mesures d'erreur suivantes, en notant respectivement $\widehat{\boldsymbol{a}}$ et $\mathring{\boldsymbol{a}}$ les abondances estimées et vraies :

— l'erreur quadratique $E_Q = \|\widehat{\boldsymbol{a}} - \mathring{\boldsymbol{a}}\|_2^2$;

— l'erreur d'estimation du support à K composantes $E_S = \|\text{supp}(\widehat{\boldsymbol{a}}_K) - \text{supp}(\mathring{\boldsymbol{a}})\|_0$, où $\text{supp}(\boldsymbol{a})_q = 1$ si $a_q \neq 0$ et 0 sinon et, pour les solutions non contraintes en norme ℓ_0, $\text{supp}(\widehat{\boldsymbol{a}}_K)$ est obtenu en ne conservant que les K plus grandes composantes de $\widehat{\boldsymbol{a}}$.

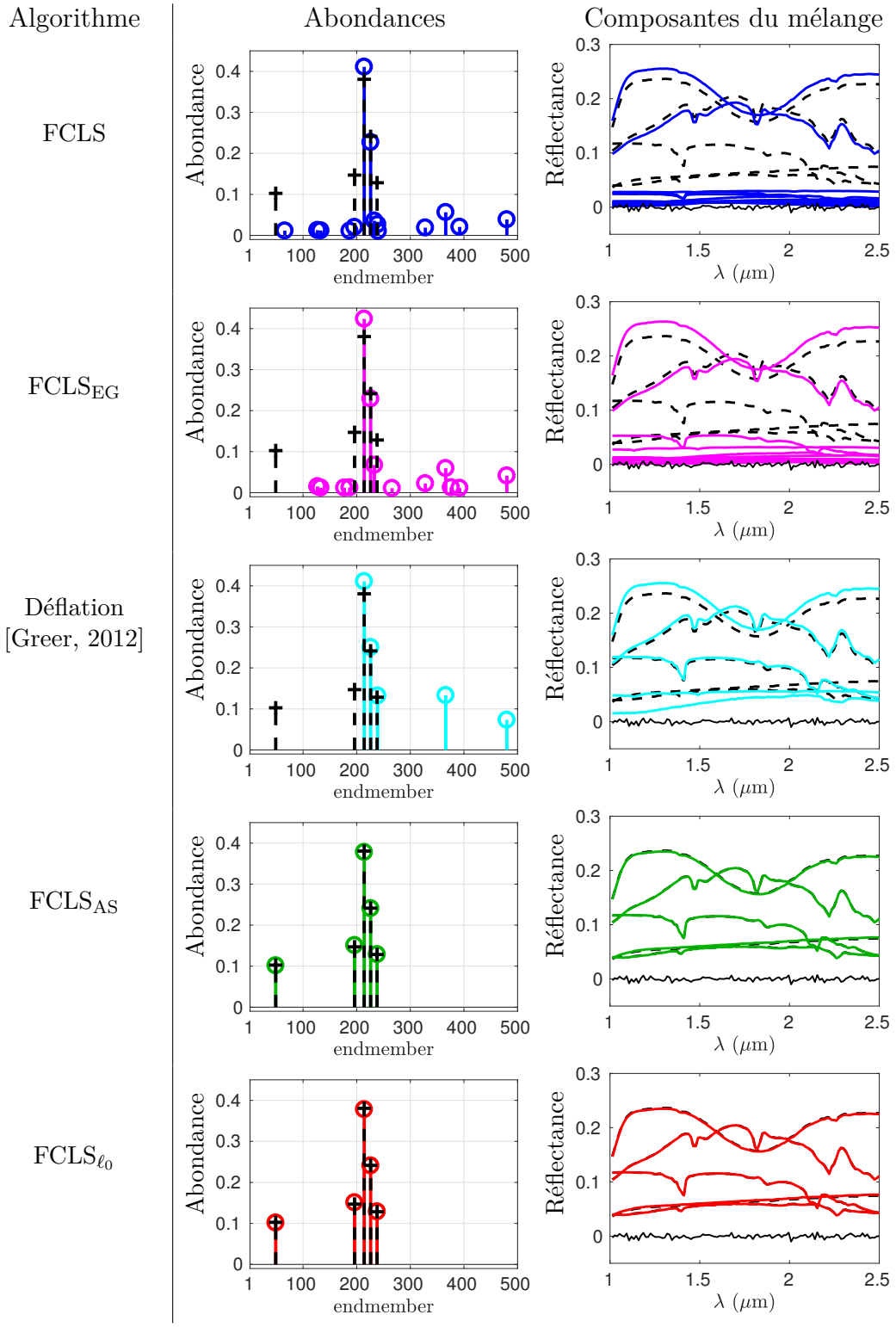


FIGURE 6.6 – Exemple de résultat de démélange avec $K = 5$ spectres, RSB = 45 dB. À gauche : abondances estimées (o) et vraies (+ noirs). À droite : vrais endmembers (lignes noires pointillées) et endmembers estimés (ligne continue), pondérés par leurs abondances. La ligne noire représente le bruit.

Nous comparons tout d'abord la solution obtenue par FCLS et une version similaire « en norme ℓ_1 » où la somme des abondances est contrainte à $t \leq 1$. La figure 6.7 montre ainsi l'erreur E_Q en fonction de t, moyennée sur l'ensemble des valeurs de K, et pour plusieurs niveaux de bruit de 40 à 60 db. Nous pouvons constater que la contrainte en

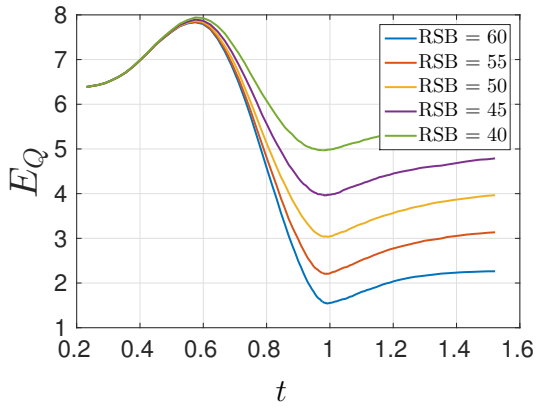


FIGURE 6.7 – Erreur quadratique d'estimation moyennée sur 300 réalisations pour une approche en norme ℓ_1 ($\|\boldsymbol{a}\|_1 \leq t$) en fonction de t, pour plusieurs niveaux de bruit.

norme ℓ_1 n'améliore pas vraiment la solution obtenue par FCLS : la valeur t^* minimisant cette erreur passe ainsi de 0.998 lorsque RSB = 60 dB à 0.98 lorsque RSB = 40 dB. Par conséquent, les solutions obtenues sont toujours très proches de celles de FCLS et nous ne regardons que celle-ci par la suite.

La figure 6.8 présente les résultats obtenus pour l'erreur quadratique E_Q et l'erreur du support E_S, en fonction de K, pour RSB de 55 dB (niveau de bruit faible), 45 dB (niveau de bruit modéré) et 40 dB (niveau de bruit plus fort). Les deux approches basées sur la norme ℓ_0 ($\text{FCLS}_{\ell_0+\text{EG}}$ ou FCLS_{ℓ_0}) donnent toujours les mêmes résultats. Pour alléger la figure, nous ne présentons donc que FCLS_{ℓ_0}.

FCLS_{EG} est meilleure que FCLS, et les deux approches basées sur la norme ℓ_0 ($\text{FCLS}_{\ell_0+\text{EG}}$ et FCLS_{ℓ_0}) donnent toujours les meilleurs résultats, avec une détection parfaite du support dans 100% des cas pour les faibles niveaux de bruit ou les petites valeurs de K. Naturellement, leurs performances diminuent quand la complexité du problème augmente : même pour les cas où l'optimisation en norme ℓ_0 a bien fonctionné et où la solution trouvée est donc garantie d'être le vrai minimiseur en norme ℓ_0, FCLS_{ℓ_0} n'arrive pas à retrouver les éléments ayant généré les données à cause du trop fort niveau de bruit et de la corrélation élevée du dictionnaire. Pour $K \geq 6$ et à faible RSB, la complexité du problème rend la résolution MIP impossible dans le temps maximal autorisé (1000 s).

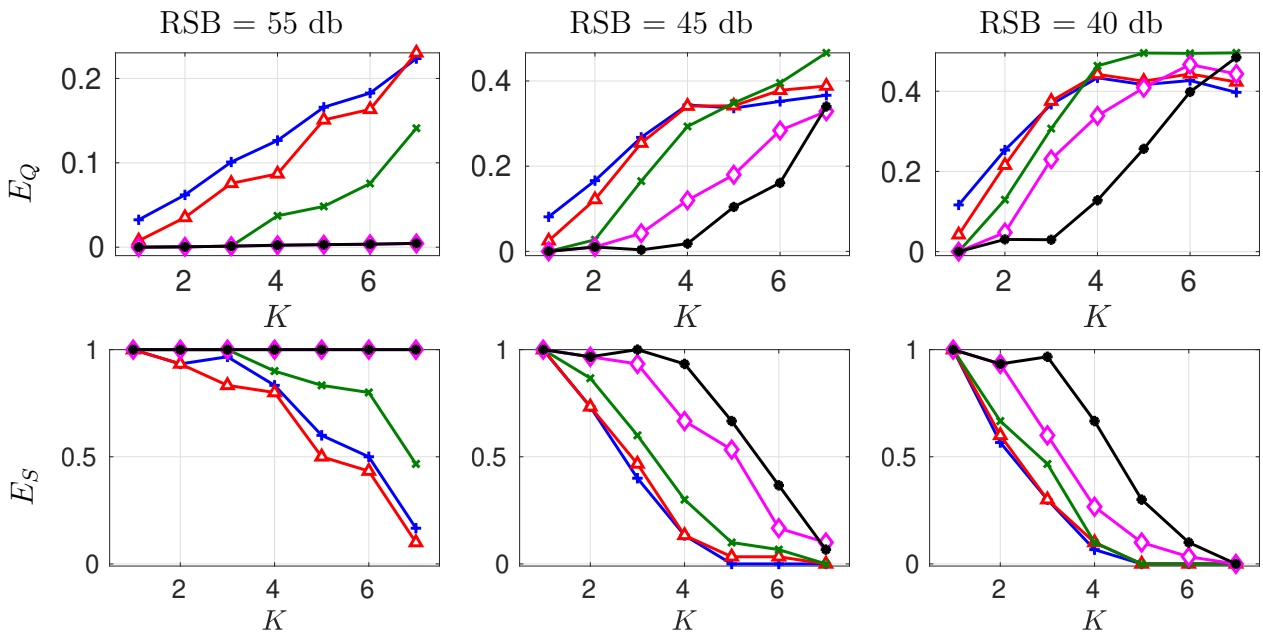


FIGURE 6.8 – Performances d'estimation de FCLS (+), FCLS $_{\text{EG}}$ (△), FCLS $_{\text{EG}+\ell_0}$ (○), FCLS $_{\text{EG+AS}}$ (◇), et la méthode itérative par déflation [Greer, 2012] (×).

Enfin, la Table 6.1 présente le temps de calcul moyen relatif au calcul des solutions FCLS_{ℓ_0}, $\text{FCLS}_{\ell_0+\text{EG}}$ et $\text{FCLS}_{\text{AS+EG}}$. Celui-ci reste relativement raisonnable pour un faible niveau de bruit ou une petite valeur de K, mais augmente fortement avec ces deux facteurs. En particulier, le solveur CPLEX permet de trouver et de garantir la solution optimale de $\text{FCLS}_{\ell_0+\text{EG}}$ en moins de 1000 s, sur toutes les instances jusqu'à $K = 4$ pour un RSB = 45 db, et $K = 6$ pour un RSB plus élevé. Notons encore que, dans toutes nos simulations, FCLS_{ℓ_0} et $\text{FCLS}_{\ell_0+\text{EG}}$ ont toujours donné la même solution. Cependant, l'ajout des contraintes (EG) permet une résolution numérique plus efficace du problème contraint par la norme ℓ_0, en particulier lorsque les problèmes deviennent plus difficiles. Enfin, nous remarquons que la prise en compte des contraintes d'abondances significatives dans $\text{FCLS}_{\text{AS+EG}}$ s'avère bien plus coûteuse que celle d'une contrainte en norme ℓ_0. Ce résultat est logique, car l'espace de recherche associé à la première est moins contraint (pour chaque problème, le nombre maximal de composantes non-nulles est égal de $1/\tau = 10$).

		K		
		2	4	6
FCLS_{ℓ_0}	RSB=60dB	0.91	1.01	1.23
	RSB=55dB	0.94	1.08	1.36
	RSB=50dB	1.14	5.89	$251^{(1)}$
	RSB=45dB	1.32	17	$581^{(4)}$
$\text{FCLS}_{\ell_0+\text{EG}}$	RSB=60dB	1.02	1.17	1.39
	RSB=55dB	1.04	1.46	2.17
	RSB=50dB	1.23	4.25	123
	RSB=45dB	1.41	11.1	$547^{(4)}$
$\text{FCLS}_{\text{AS+EG}}$	RSB=60dB	1.53	1.75	4.4
	RSB=55dB	1.44	2.43	4.7
	RSB=50dB	6.98	16.7	$212^{(1)}$
	RSB=45dB	3.33	$204^{(1)}$	$681^{(5)}$

TABLE 6.1 – Temps de calcul (s) moyen sur 30 instances pour l'optimisation des problèmes MIP en fonction du degré de parcimonie. Entre parenthèses : nombre de réalisations n'ayant par fourni la solution optimale en 1 000 s.

6.6 Conclusion

Le problème de démélange spectral, qui contraint naturellement les abondances entre 0 et 1, est particulièrement adapté aux reformulations en MIP, évitant le recours à une hypothèse artificielle de type *BigM* (voir la discussion en Section 1.4.2). Nous avons pu prendre en compte des contraintes de parcimonie et des contraintes logiques inhabituelles, grâce à la flexibilité offerte par l'introduction des variables binaires de décision. Dans la mesure où ces contraintes sont valables d'un point de vue pratique (selon le problème), elles peuvent être avantageusement utilisées pour améliorer la détection des abondances par rapport à la solution classique FCLS. Nous avons présenté trois formulations à base de contraintes logiques qui peuvent créer de la parcimonie :

— la formulation contrainte par la cardinalité de la solution, qui est très efficace si le nombre de composants dans le mélange est connu ;

— la formulation permettant de prendre en compte la variabilité des spectres dans le dictionnaire, qui peut être efficace pour limiter la complexité des problèmes lorsque le dictionnaire est structuré en groupes et de grande taille ;

— la formulation qui impose une valeur minimale aux abondances non nulles est

également intéressante, où il peut être préférable d'ajuster le seuil correspondant plutôt que de spécifier le nombre d'abondances non nulles.

Enfin, même si l'optimisation globale requiert un temps de calcul bien plus élevé que les méthodes habituelles, nous avons montré qu'elle reste cependant faisable pour des problèmes de démélange de taille raisonnable, limitée par le nombre de composantes recherchées et le niveau de bruit. Une perspective prioritaire afin d'améliorer le temps de calcul de ces approches réside bien évidemment dans le développement d'algorithmes de résolution dédiés en lieu et place de l'utilisation d'un solveur, thématique au centre des chapitres 2 à 5 de ce manuscrit. Nous aborderons cette possibilité dans la conclusion générale.

Chapitre 7

Conclusions et Perspectives

Dans cette thèse, nous avons travaillé sur deux axes principaux. La plus grande partie concerne le développement d'algorithmes branch-and-bound pour la résolution exacte du problème d'approximation parcimonieuse en norme ℓ_0. Ce problème peut être formulé mathématiquement comme un problème d'optimisation bi-objectif, décrit par la minimisation de l'erreur d'approximation quadratique et la norme ℓ_0 de $\boldsymbol{x}$:

$$\mathcal{P} : \qquad \min_{\boldsymbol{x}\in\mathbb{R}^n} \{\tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2, \|\boldsymbol{x}\|_0\}$$

En particulier, nous avons abordé les trois formulations mono-objectif classiques : la formulation contrainte par la norme ℓ_0 ($\mathcal{P}_{2/0}$), la formulation minimisant la norme ℓ_0 sous contrainte d'une borne sur l'erreur d'approximation ($\mathcal{P}_{0/2}$) et la formulation pénalisée par la norme ℓ_0 ($\mathcal{P}_{2+0}$). Afin de pouvoir traiter ces problèmes de manière exacte, nous avons supposé dès le départ que l'espace de recherche est borné, et que les composantes du vecteur $\boldsymbol{x}$ sont inférieures à une certaine valeur M (hypothèse de BigM), qui est totalement justifiée d'un point de vue pratique, où les applications abordées dans cette thèse ont un espace de solution borné.

Le deuxième axe de recherche a été consacré au problème de démélange spectral parcimonieux. Notre travail a surtout concerné la reformulation en programmes en nombres mixtes (MIP) de plusieurs contraintes logiques favorisant la parcimonie, qui sont généralement abordées par des méthodes d'optimisation approchées. Ensuite, nous avons étudié la faisabilité ainsi que la contribution de ces contraintes sur des problèmes inverses difficiles de démélange spectral.

Dans ce dernier chapitre, nous dressons successivement, pour chacune des deux parties, des éléments de conclusion générale et des perspectives de poursuite de ces travaux.

7.1 Méthode branch-and-bound spécifique au problème d'approximation parcimonieuse

La méthode branch-and-bound n'est pas nouvelle en soi, elle est largement utilisée en optimisation combinatoire, et elle est composée de deux étapes essentielles : l'évaluation basée sur les relaxations et la séparation basée sur des heuristiques pour le choix de variables de branchement et les stratégies de parcours d'arbre de recherche. Afin de construire un algorithme branch-and-bound spécifique au problème d'approximation parcimonieuse, nous avons développé des stratégies appropriées à chaque étape, où nous avons étudié de manière théorique les spécificités engendrées par la norme ℓ_0 et leurs impacts sur les différentes parties de l'algorithme.

Dans un premier temps, nous avons étudié la relaxation continue impliquée dans l'étape d'évaluation de chaque nœud de l'arbre de recherche, qui consiste à relâcher les variables binaires. Nous avons montré une propriété qui présente un intérêt majeur pour notre travail : les problèmes relâchés, quelle que soit la formulation (contrainte ou pénalisée), se réduisent tous à des problèmes sans variable binaire, comprenant une fonction des moindres carrés, des termes en norme ℓ_1 impliquant une partie des variables continues et des contraintes de borne sur toutes les variables continues.

En interprétant l'évaluation comme un problème en norme ℓ_1, nous avons proposé une résolution dédiée et exacte permettant de garantir les bornes inférieures, en exploitant le savoir-faire développé en optimisation ℓ_1. Nous avons ainsi construit deux algorithmes spécifiques. Le premier est inspiré du principe de l'homotopie et peut être appliqué dans une procédure de branch-and-bound avec la même efficacité pour les trois formulations. En particulier, il permet de résoudre efficacement la formulation à contraintes quadratiques, qui présente un intérêt majeur dans de nombreuses applications, là où les algorithmes utilisés par les différents solveurs MIP s'avèrent beaucoup moins efficaces. Le deuxième algorithme est de type ensemble actif spécifique à l'optimisation ℓ_1, qui s'est montré efficace pour résoudre les relaxations continues de la formulation pénalisée grâce au démarrage à chaud.

Inspirés par les méthodes gloutonnes de l'approximation parcimonieuse, nous avons développé une stratégie de parcours priorisant le branchement sur les variables non-nulles d'abord, ce qui permet d'avoir des solutions réalisables plus rapidement. Ensuite, nous avons proposé une règle de sélection de variables, nommée LPS pour L_1 *Path Selection*, basée sur le chemin de régularisation en norme ℓ_1 construit par la méthode d'homotopie lors

du calcul de la relaxation continue. Cette règle n'ajoute ainsi aucun coût supplémentaire à la résolution. La règle LPS a montré de meilleures performances pour le problème contraint par la parcimonie par rapport aux règles de branchement classiques.

Les performances de l'algorithme branch-and-bound résultant de l'assemblage des différentes parties développées ont été réalisées en comparaison avec CPLEX, l'un des meilleurs solveurs commerciaux d'optimisation MIP, sur deux types de problèmes d'approximation parcimonieuse :

— pour des problèmes de déconvolution parcimonieuse, qui sont de petite taille mais très difficiles en raison de la corrélation élevée entre les colonnes du dictionnaire, notre algorithme est plus efficace que CPLEX. L'écart de performance est notamment élevé pour la formulation minimisant la norme ℓ_0 sous contraintes de l'erreur d'approximation, CPLEX gérant mal les contraintes quadratiques.

— Pour des problèmes de sélection de variables avec des données aléatoires, qui sont plus grands en taille mais plus faciles à résoudre, notre algorithme s'avère beaucoup plus efficace que CPLEX pour les trois formulations, avec un rapport moyen en temps de calcul de plus de 1500 fois en faveur de notre algorithme pour la formulation sous contraintes quadratiques. Ceci montre de plus la capacité de notre méthode à monter en dimension par rapport aux solveurs génériques.

Même si le solveur MIP CPLEX utilise toute une batterie de techniques qui n'ont pas été abordées dans ces travaux, notre algorithme reste plus performant pour traiter ces problèmes inverses parcimonieux.

Perspectives

Au delà des perspectives qui ont pu être mentionnées en conclusion des différents chapitres, de nombreuses autres perspectives s'ouvrent suite au travail réalisé autour de la méthode branch-and-bound pour le problème d'approximation parcimonieuse.

Problème d'approximation parcimonieuse bi-objectif

En général, contrairement à l'optimisation mono-objectif, il n'existe pas de solution globale unique au problème bi-objectif $\mathcal{P}$, et on s'intéresse souvent à la construction du front de Pareto [Pareto, 1906] afin de trouver l'ensemble des solutions optimales pour

lesquelles il n'existe aucune autre solution permettant une réduction simultanée des deux objectifs Marler and Arora [2004].

Puisque la norme ℓ_0 est discrète, la manière la plus simple pour de construire le front de Pareto consiste à résoudre un ensemble fini de problèmes mono-objectifs sous contrainte d'une borne sur la norme ℓ_0 de $\boldsymbol{x}$ variant de 1 à une valeur maximale, $K_{\max}$:

$$\forall k = 1, \ldots, K_{\max}, \hat{\boldsymbol{x}}(k) = \arg \hat{\mathcal{P}}_{2/0}^{(k)}, \text{ avec } \hat{\mathcal{P}}_{2/0}^{(k)} : \min_{\boldsymbol{x}} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \quad \text{s.c.} \quad \|\boldsymbol{x}\|_0 \leq k.$$

La figure 7.1 illustre ce principe.

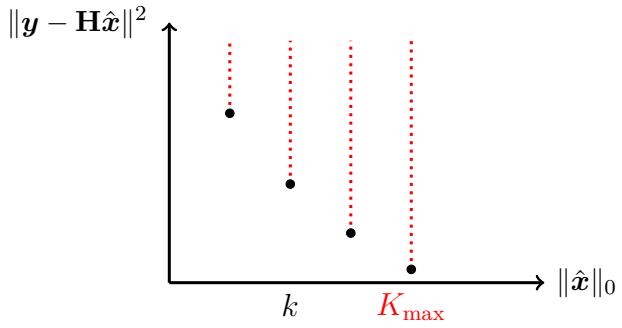


FIGURE 7.1 – Front de Pareto pour le problème $\mathcal{P}$: ensemble fini de solutions.

Nous avons déjà proposé un algorithme branch-and-bound spécifique pour résoudre le problème $\hat{\mathcal{P}}_{2/0}^{(k)}$ à k fixé. Une perspective concerne l'adaptation de cet algorithme pour résoudre simultanément les différents problèmes pour k variant de 1 à $K_{\max}$. En pratique, sur le plan informatique, nous pouvons résoudre tous les problèmes $\hat{\mathcal{P}}_{2/0}^{(k)}$ simultanément en construisant un seul arbre de recherche où, dans chaque nœud, nous calculons et stockons $K_{\max}$ bornes inférieures :

$$z_{(k)}^R = \min_{-M \leq \boldsymbol{x} \leq M} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}\|_2^2 \quad \text{s.c. } \|\boldsymbol{x}\|_1 \leq kM, \quad k = 1, \ldots, K_{\max}$$

sur chaque sous-problème (la borne inférieure est donc un vecteur de taille $K_{\max}$, voir Figure 7.2). Ces bornes inférieures seront comparées aux bornes supérieures associées $z_{L(k)}$ associées (les meilleures solutions trouvées à k composantes), et un nœud sera élagué lorsque $z_{(k)}^R > z_{L(k)}$ $\forall k = 1, \ldots, K_{\max}$.

Or, la résolution de ce problème bi-objectif sans aucun coût de calcul supplémentaire est possible grâce à la méthode homotopique. Rappelons que la méthode homotopique considère la relaxation continue dans un nœud quelconque, et exploite le fait que le chemin

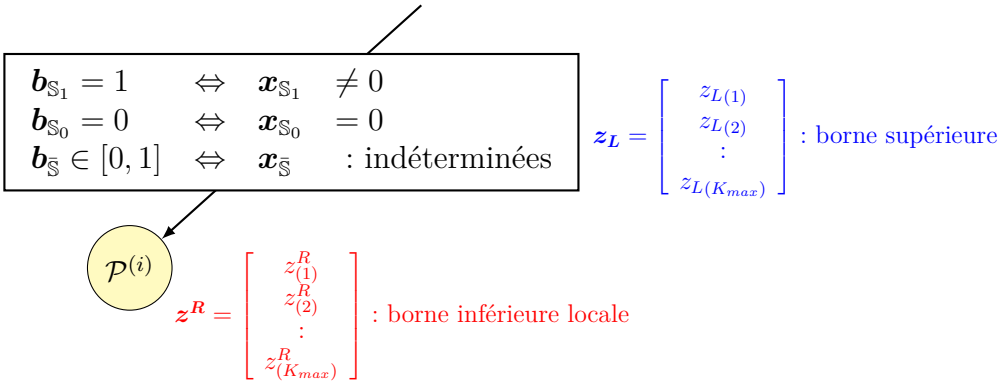


FIGURE 7.2 – Exemple illustrant les bornes inférieures $z^R_{(k)}$ associées à un sous-problème $\hat{\mathcal{P}}_{2/0_{(k)}}$ quelconque de l'algorithme branch-and-bound pour la résolution du problème bi-objectif $\mathcal{P}$; Les bornes supérieures $z_{L(k)}$ en bleu ;

de solution est linéaire par morceaux en fonction du paramètre de régularisation λ. À partir d'un point initial λ^0 tel que les variables dans le terme en norme ℓ_1 sont nulles, la méthode calcule itérativement le chemin des solutions en diminuant de façon continue le paramètre λ jusqu'à trouver la solution de la relaxation continue du problème $\hat{\mathcal{P}}^{(K_{\max})}_{2/0}$. Par conséquent, elle permet également de trouver au passage toutes les solutions de la relaxation continue des problèmes $\hat{\mathcal{P}}^{(k)}_{2/0}$ pour $k = 1, \ldots, K_{\max}$ (voir Figure 7.3).

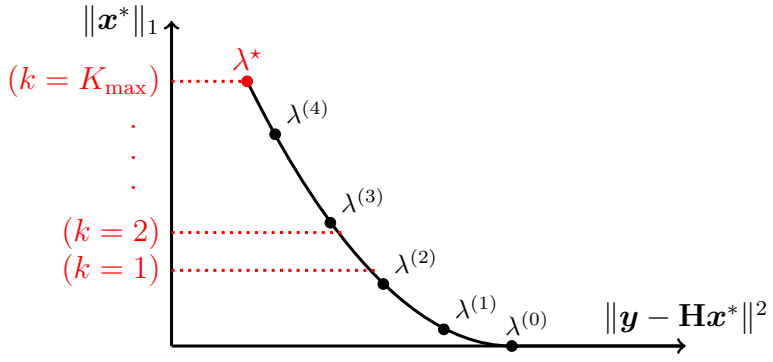


FIGURE 7.3 – Méthode d'homotopie : exemple de Front de Pareto correspondant au chemin de solution de la relaxation continue du problème $\hat{\mathcal{P}}^{(K_{\max})}_{2/0}$.

Relaxation lagrangienne

Pour d'autres problèmes parcimonieux, le problème d'approximation parcimonieuse $\mathcal{P}_{2/0}$ peut être formulé avec un terme supplémentaire de pénalité en norme ℓ_2 :

$$\mathcal{P}_{2d/0} \ : \ \min_{\boldsymbol{x}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{y}-\mathbf{H}\boldsymbol{x}\|_2^2 + d\|\boldsymbol{x}\|_2^2 \ \text{ s.c. } \|\boldsymbol{x}\|_0 \leq K.$$

C'est le cas du modèle Bernoulli-gaussien souvent rencontré en déconvolution parcimonieuse [Kormylo and Mendel, 1982; Soussen et al., 2011]. Des formulations similaires apparaissent également en statistiques, portant le nom d'*elastic net* lorsque la norme ℓ_0 est remplacée par la norme ℓ_1.

Dans ce cas, il existe d'autres types de relaxation que la relaxation continue, qui ont montré une grande efficacité dans la résolution de problèmes contraints par la norme ℓ_0, notamment en finance pour l'optimisation de portefeuille (ces problèmes sont alors assez proches du problème $\mathcal{P}_{2d/0}$, avec en général quelques contraintes linéaires supplémentaires et des contraintes de non-négativité). En utilisant la technique de duplication des variables (voir par exemple Guignard and Kim [1987]; Michelon and Maculan [1991]; Shaw et al. [2008]), le problème $\mathcal{P}_{2d/0}$ peut être reformulé comme suit :

$$\min_{\boldsymbol{x}\in\mathbb{R}^n,\boldsymbol{z}\in\mathbb{R}^n} \tfrac{1}{2}\boldsymbol{z}^T\mathbf{H}^T\mathbf{H}\boldsymbol{z} - \boldsymbol{y}^T\mathbf{H}\boldsymbol{x} + \tfrac{1}{2}\boldsymbol{x}^T\mathbf{D}\boldsymbol{x} \quad \text{s.c.} \quad \begin{cases} \boldsymbol{x} = \boldsymbol{z} \\ \|\boldsymbol{x}\|_0 \leq K \end{cases}$$

où $\mathbf{D} \in \mathbb{R}^{n\times n} := d\,\mathbf{I}$ est une matrice diagonale positive. La dualisation de la contrainte $\boldsymbol{x} = \boldsymbol{z}$ avec le vecteur multiplicateur $\boldsymbol{\alpha} \in \mathbb{R}^n$ donne le problème dual lagrangien :

$$\max\{\mathcal{L}_1(\boldsymbol{\alpha}) + \mathcal{L}_2(\boldsymbol{\alpha})\}$$

où :

$$\begin{cases} \mathcal{L}_1(\boldsymbol{\alpha}) = \min_{\boldsymbol{z}\in\mathbb{R}^n} \tfrac{1}{2}\boldsymbol{z}^T\mathbf{H}^T\mathbf{H}\boldsymbol{z} + \boldsymbol{\alpha}^T\boldsymbol{z} \\ \mathcal{L}_2(\boldsymbol{\alpha}) = \min_{\boldsymbol{x}\in\mathbb{R}^n} (\mathbf{H}^T\boldsymbol{y} - \boldsymbol{\alpha})^T\boldsymbol{x} + \tfrac{1}{2}\boldsymbol{x}^T\mathbf{D}\boldsymbol{x} \quad \text{s.c. } \|\boldsymbol{x}\|_0 \leq K \end{cases}.$$

Chacun des deux problèmes d'optimisation ci-dessus peut alors être résolu facilement (le premier est quadratique, le second est séparable). Dans ce cas, des techniques de relaxation lagrangienne [Li et al., 2006; Shaw et al., 2008; Cui et al., 2013], de reformulation perspective Frangioni and Gentile [2006] et relaxation géométrique [Gao and Li, 2013], peuvent être appliquées et produire une borne inférieure meilleure que la relaxation continue.

Autres perspectives

Plusieurs autres perspectives peuvent être évoquées, telles que l'utilisation d'un algorithme de type branch-and-cut en ajoutant des *méthodes de coupes* spécifiques au problème Wolsey and Nemhauser [1999] ou encore l'exploitation d'autres outils de la parcimonie en traitement du signal, comme les méthodes de *screening*, permettant d'exclure de manière garantie des variables du problème initial (screening ℓ_0) ou des différents problèmes relâchées (screening en norme ℓ_1). Enfin, pour pouvoir exploiter toute la puissance de calcul des ordinateurs actuels comme le font les solveurs MIP commerciaux de nos jours, il serait intéressant d'étudier les possibilités de paralléliser les algorithmes développés.

7.2 Reformulation MIP et problème de démélange parcimonieux

La deuxième partie de cette thèse a concerné le problème de démélange en imagerie hyperspectrale. Ce problème, qui contraint naturellement les abondances entre 0 et 1, est particulièrement adapté aux reformulations en MIP. Nous avons montré que la prise en compte de la contrainte de parcimonie en norme ℓ_0 améliore les performances de détection par rapport à la solution classique FCLS, là où l'utilisation de la norme ℓ_1 est peu utile. La souplesse de la formulation MIP, par l'introduction de variables de décision, a ainsi permis d'envisager la prise en compte de contraintes habituellement abordées de manière approchée, voire jamais abordées, pour le démélange. Par ailleurs, nous avons montré que l'optimisation globale reste envisageable pour des problèmes dont la complexité est limitée par un faible nombre de composantes non nulles et un faible niveau de bruit.

Perspectives

Plusieurs perspectives s'ouvrent au sujet de le reformulation MIP de problèmes de démélange, principalement autour de la construction de méthodes de résolution dédiées.

Branch-and-Bound spécifique pour le démélange spectral

La construction d'un algorithme branch-and-bound est une perspective particulièrement intéressante, sachant que le problème de démélange spectral est un MIP très spécifique, et

que plusieurs points peuvent être exploités afin de construire un algorithme très efficace. Nous recensons ci-dessous quelques éléments identifiés motivant la poursuite de ces travaux.

1. La contrainte de positivité va simplifier la résolution de la relaxation continue, puisque la norme ℓ_1 sera juste la somme de ses valeurs. La résolution du problème relâché sera donc plus simple, que ce soit par la méthode homotopique ou par un algorithme d'ensemble actif par exemple, puisque tous les tests sur la partie négative seront supprimés ; les algorithmes seront donc plus rapides.
2. En présence de la contrainte de positivité et de la contrainte de somme à un, la contrainte de parcimonie devient superflue au niveau de la relaxation continue : comme nous l'avons vu dans cette thèse, la relaxation continue de la contrainte de la norme ℓ_0 s'écrit en fonction de la norme ℓ_1 :

$$\|\boldsymbol{x}\|_0 \leq K \rightsquigarrow \|\boldsymbol{x}\|_1 \leq KM.$$

Or, pour les problèmes de démélange, les amplitudes sont inférieures à 1 ($M = 1$). Par conséquent, $\|\boldsymbol{x}\|_1 = \sum_q x_q \leq 1$, si bien que la contrainte en norme ℓ_1 sera inactive.

$$\|\boldsymbol{x}\|_0 \leq K \rightsquigarrow \begin{cases} \sum_q^n x_q \leq 1 & \text{contrainte active} \\ \|\boldsymbol{x}\|_1 \leq K & \text{contrainte inactive} \end{cases}$$

Cette propriété peut avoir un impact sur la résolution par l'algorithme de branch-and-bound qui sera moins coûteuse, car la solution du problème relâché ne va pas changer.

Autres formulations

Jusqu'à présent, nous avons supposé la connaissance exacte du nombre de composantes non nulles afin de focaliser notre attention sur la capacité de l'approche en norme ℓ_0 à retrouver une solution exactement parcimonieuse. Le réglage automatique de K est bien sûr une perspective, une approche consistant à discriminer parmi les solutions obtenues pour différentes valeurs de K au moyen d'un critère de sélection de modèle. Cette perspective rejoint ainsi le développement, proposé en Section 7.1, d'un algorithme branch-and-bound bi-objectif, mais cette fois dédié aux spécificités du problème de démélange.

Enfin, nous n'avons abordé dans cette partie que la formulation contrainte par la parcimonie $\mathcal{P}_{2/0}$. D'un point de vue pratique, cependant, la formulation $\mathcal{P}_{0/2}$ peut être

préférée, permettant de régler une tolérance sur l'erreur de modèle ou le niveau de bruit plutôt qu'un nombre de composantes dans le mélange. Il serait donc également intéressant d'en étudier la résolution au moyen d'approches dédiées.

BIBLIOGRAPHIE

Achterberg, T. and Berthold, T. (2009). Hybrid Branching. In van Hoeve, W.-J. and Hooker, J. N., editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Lecture Notes in Computer Science, pages 309–311. Springer Berlin Heidelberg.

Achterberg, T., Koch, T., and Martin, A. (2005). Branching rules revisited. *Operations Research Letters*, 33(1) :42–54.

Achterberg, T. and Wunderling, R. (2013). Mixed Integer Programming : Analyzing 12 Years of Progress. In Jünger, M. and Reinelt, G., editors, *Facets of Combinatorial Optimization*, pages 449–481. Springer Berlin Heidelberg, Berlin, Heidelberg.

Adams, J. B., Smith, M. O., and Johnson, P. E. (1986). Spectral mixture modeling : A new analysis of rock and soil types at the Viking Lander 1 Site. *Journal of Geophysical Research : Solid Earth*, 91(B8) :8098–8112.

Alvarez, A. M., Louveaux, Q., and Wehenkel, L. (2017). A Machine Learning-Based Approximation of Strong Branching. *INFORMS Journal on Computing*, 29 :185–195.

Applegate, D., Bixby, R., Cook, W., and Chvatal, V. (1998). *On the Solution of Traveling Salesman Problems*. Springer, Berlin, Heidelberg.

Ben Mhenni, R., Bourguignon, S., and Idier, J. (2020). SLS : A Greedy Sparse Approximation Algorithm Based on L1-norm Selection Rules. In *International Conference on Acoustics, Speech, and Signal Processing*. IEEE.

Benichou, M., Gauthier, J. M., Girodet, P., Hentges, G., Ribiere, G., and Vincent, O. (1971). Experiments in mixed-integer linear programming. *Mathematical Programming*, 1(1) :76–94.

Bertsimas, D., King, A., and Mazumder, R. (2016). Best Subset Selection via a Modern Optimization Lens. *The Annals of Statistics*, 44(2) :813–852.

Bertsimas, D. and Shioda, R. (2009). Algorithm for cardinality-constrained quadratic optimization. *Computational Optimization and Applications*, 43(1) :1–22.

Bienstock, D. (1996). Computational study of a family of mixed-integer quadratic programming problems. *Mathematical Programming*, 74(2) :121–140.

Bioucas-Dias, J. M., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., and Chanussot, J. (2012). Hyperspectral Unmixing Overview : Geometrical, Statistical, and Sparse Regression-Based Approaches. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(2).

Blumensath, T. and Davies, M. E. (2008). Iterative Thresholding for Sparse Approximations. *Journal of Fourier Analysis and Applications*, 14(5) :629–654.

Borchers, B. and Mitchell, J. E. (1994). An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers & Operations Research*, 21(4) :359–367.

Bourguignon, S., Ninin, J., Carfantan, H., and Mongeau, M. (2016). Exact sparse approximation problems via mixed-integer programming : Formulations and computational performance. *IEEE Transactions on Signal Processing*, 64(6) :1405–1419.

Carcreff, E. (2014). *Déconvolution adaptative pour le contrôle non destructif par ultrasons*. PhD thesis, Université du Maine.

Chang, C. (2003). *Hyperspectral Imaging : Techniques for Spectral Detection and Classification*. Number vol. 1 in Hyperspectral Imaging : Techniques for Spectral Detection and Classification. Springer US.

Chen, S., Billings, S., and Luo, W. (1989). Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50(5) :1873–1896.

Clark, R., Swayze, G., Wise, R., Livo, K., Hoefen, T., Kokaly, R., and Sutley, S. (2003a). USGS digital spectral library splib05a. *US Geological Survey, Digital Data Series*, 231.

Clark, R., Swayze, G., Wise, R., Livo, K., Hoefen, T., Kokaly, R., and Sutley, S. (2003b). USGS digital spectral library splib05a. *US Geological Survey, Digital Data Series*, 231.

Cui, X. T., Zheng, X. J., Zhu, S. S., and Sun, X. L. (2013). Convex relaxations and MIQCQP reformulations for a class of cardinality-constrained portfolio selection problems. *Journal of Global Optimization*, 56(4) :1409–1423.

Dolan, E. D. and Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2) :201–213.

Donoho, D. L. and Tsaig, Y. (2008). Fast Solution of ℓ_1 Norm Minimization Problems When the Solution May Be Sparse. *IEEE Transactions on Information Theory*, 54(11) :4789–4812.

Drumetz, L., Meyer, T., Chanussot, J., Bertozzi, A., and Jutten, C. (2019). Hyperspectral Image Unmixing with Endmember Bundles and Group Sparsity Inducing Mixed Norms. *IEEE Transactions on Image Processing*, 28(7) :3435–3450.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32(2) :407–499.

Elad, M. (2010). *Sparse and Redundant Representations. From Theory to Applications in Signal and Image Processing*. Springer-Verlag New York.

Eldar, Y. and Kutyniok, G. (2012). *Compressed Sensing : Theory and Applications*. Cambridge University Press.

Frangioni, A. and Gentile, C. (2006). Perspective cuts for a class of convex 0–1 mixed integer programs. *Mathematical Programming*, 106(2) :225–236.

Gao, J. and Li, D. (2013). Optimal Cardinality Constrained Portfolio Selection. *Operations Research*, 61(3) :745–761.

Greer, J. B. (2012). Sparse Demixing of Hyperspectral Images. *IEEE Transactions on Image Processing*, 21(1) :219–228.

Guignard, M. and Kim, S. (1987). Lagrangean decomposition : A model yielding stronger lagrangean bounds. *Mathematical Programming*, 39(2) :215–228.

Guo, Z., Wittman, T., and Osher, S. (2009). L1 unmixing and its application to hyperspectral image enhancement. In *Proc SPIE Conference on Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV*.

Hager, W. W. (1989). Updating the inverse of a matrix. *SIAM Review*, 31(2) :221–239.

Heinz, D. C. et al. (2001). Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery. *IEEE transactions on geoscience and remote sensing*, 39(3) :529–545.

Idier, J., editor (2008). *Bayesian Approach to Inverse Problems*. ISTE Ltd and John Wiley & Sons Inc.

Iordache, M.-D., Bioucas-Dias, J. M., and Plaza, A. (2011). Sparse Unmixing of Hyperspectral Data. *IEEE Transactions on Geoscience and Remote Sensing*, 49(6) :2014–2039.

Karahanoglu, N. B. and Erdogan, H. (2012). A* orthogonal matching pursuit : Best-first search for compressed sensing signal recovery. *Digital Signal Processing*, 22(4) :555 – 568.

Keshava, N. and Mustard, J. F. (2002). Spectral unmixing. *IEEE Signal Processing Magazine*, 19(1) :44–57.

Kormylo, J. and Mendel, J. (1982). Maximum likelihood detection and estimation of bernoulli-gaussian processes. *IEEE Transactions on Information Theory*, 28(3) :482–488.

Lai, M. and Wang, J. (2011). An unconstrained ℓ_q minimization with $q \leq 1$ for sparse solution of underdetermined linear systems. *SIAM Journal on Optimization*, 21(1) :82–101.

Land, A. H. and Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica : Journal of the Econometric Society*, 28(3) :497–520.

Laughhunn, D. J. (1970). Quadratic Binary Programming with Application to Capital-Budgeting Problems. *Operations Research*, 18(3) :454–461.

Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2007). Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808.

Li, D., Sun, X., and Wang, J. (2006). Optimal Lot Solution to Cardinality Constrained Mean–Variance Formulation for Portfolio Selection. *Mathematical Finance*, 16(1) :83–101.

Liang, X. and Wang, Y. (2017). Homotopy algorithm for box-constrained LASSO and its convergence. *International Journal of Pure and Applied Mathematics*, 112(2).

Linderoth, J. T. and Savelsbergh, M. W. P. (1999). A Computational Study of Search Strategies for Mixed Integer Programming. *INFORMS Journal on Computing*, 11(2) :173–187.

Liu, Y., Canu, S., Honeine, P., and Ruan, S. (2019). Mixed Integer Programming for Sparse Coding : Application to Image Denoising. *IEEE Transactions on Computational Imaging*, pages 1–1.

Loth, M. (2011). *Active Set Algorithms for the LASSO.* Thèses, Université des Sciences et Technologies de Lille - Lille I.

Malioutov, D. M., Cetin, M., and Willsky, A. S. (2005). Homotopy continuation for sparse signal representation. In *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 5, pages 733–736.

Mallat, S. and Zhang, Z. (1993). Matching Pursuits with Time-Frequency Dictionaries. *IEEE Trans. on Signal Processing*, 41(12) :3397–3415.

Marler, R. and Arora, J. (2004). Survey of Multi-Objective Optimization Methods for Engineering. *Structural and Multidisciplinary Optimization*, 26 :369–395.

Mendel, J. (1986). Some modeling problems in reflection seismology. *IEEE ASSP Magazine*, 3(2) :4–17.

Michelon, P. and Maculan, N. (1991). Lagrangean decomposition for integer nonlinear programming with linear constraints. *Mathematical Programming*, 52(1) :303–313.

Miller, A. (2002). *Subset selection in regression.* Chapman and Hall/CRC.

Moulin, P. and Liu, J. (1999). Analysis of multiresolution image denoising schemes using generalized Gaussian and complexity priors. *IEEE Transactions on Information Theory*, 45(3) :909—919.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2) :227–234.

Nocedal, J. and Wright, S. J. (2006). *Numerical optimization*. Springer series in operations research. Springer, New York, 2nd ed edition. OCLC : ocm68629100.

O'Brien, M. S., Sinclair, A. N., and Kramer, S. M. (1994). Recovery of a sparse spike time series by L1 norm deconvolution. *IEEE Transactions on Signal Processing*, 42 :3353–3365.

Osborne, M. R., Presnell, B., and Turlach, B. (2000). A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*.

Pareto, V. (1906). *Manuale di economia politica*, volume 13. Societa Editrice.

Parra, L. C., Spence, C., Sajda, P., Ziehe, A., and Müller, K.-R. (2000). Unmixing Hyperspectral Data. In Solla, S. A., Leen, T. K., and Müller, K., editors, *Advances in Neural Information Processing Systems 12*, pages 942–948. MIT Press.

Pati, Y., Rezaiifar, R., and Krishnaprasad, P. S. (1993). Orthogonal matching pursuit : recursive function approximation with applications to wavelet decomposition. In *Asilomar Conference on Signals, Systems and Computers*, pages 40–44 vol.1.

Rockafellar, R. T. (1970). *Convex Analysis*. Princeton University Press.

Schmidt, F., Legendre, M., and Mouëlic, S. L. (2014). Minerals detection for hyperspectral images using adapted linear unmixing : LinMin. *Icarus*, 237.

Shaw, D. X., Liu, S., and Kopman, L. (2008). Lagrangian relaxation procedure for cardinality-constrained portfolio optimization. *Optimization Methods and Software*, 23(3) :411–420.

Singer, R. B. and McCord, T. B. (1979). Mars-large scale mixing of bright and dark surface materials and implications for analysis of spectral reflectance. In *Lunar and Planetary Science Conference Proceedings*, volume 10, pages 1835–1848.

Soussen, C., Idier, J., Brie, D., and Duan, J. (2011). From Bernoulli Gaussian Deconvolution to Sparse Signal Restoration. *IEEE Transactions on Signal Processing*, 59(10) :4572–4584.

Starck, J.-L., Pantin, E., and Murtagh, F. (2002). Deconvolution in astronomy : A review. *Publications of the Astronomical Society of the Pacific*, 114(800) :1051.

Taylor, H. L., Banks, S. C., and McCoy, J. F. (1979). Deconvolution with the ℓ_1 norm. *Geophysics*, 44(1) :39–52.

Tibshirani, R. (1996). Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58 :267–288.

Tropp, J. A. (2004). Greed is good : Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10) :2231–2242.

Tropp, J. A. and Wright, S. J. (2009). Computational methods for sparse solution of linear inverse problems.

Tropp, J. A. and Wright, S. J. (2010). Computational Methods for Sparse Solution of Linear Inverse Problems. *Proceedings of the IEEE*, 98(6).

Tuia, D., Flamary, R., and Barlaud, M. (2016). Nonconvex regularization in remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, 54(11) :6470–6480.

Wolsey, L. A. and Nemhauser, G. L. (1999). *Integer and Combinatorial Optimization*. John Wiley & Sons.

Xu, Z., Zhang, H., Wang, Y., Chang, X., and Liang, Y. (2010). L1/2 regularization. *Science China Information Sciences*, 53(6) :1159–1169.

Zala, C. (1992). High-resolution inversion of ultrasonic traces. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 39(4) :458–463.

Zare, A. and Ho, K. C. (2014). Endmember variability in hyperspectral analysis : Addressing spectral variability during spectral unmixing. *IEEE Signal Processing Magazine*, 31(1) :95–104.

Zheng, X., Sun, X., Li, D., and Sun, J. (2014). Successive convex approximations to cardinality-constrained convex programs : a piecewise-linear DC approach. *Computational Optimization and Applications*, 59(1) :379–397.

Annexe A

Démonstrations du chapitre 3

A.1 Conditions d'optimalité du problème pénalisé $\hat{\mathcal{Q}}_{2+1}$ en Section 3.2

Nous détaillons ici le calcul permettant d'avoir les solutions (3.14b) et (3.14a) des équations (3.13f) et (3.13g) dans le chapitre 3 :

$$\begin{cases} \boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} &= \left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\mathbf{H}_{\bar{\mathbb{S}}_{in}}\right)^{-1}\left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\underline{\boldsymbol{r}} - \lambda\ \mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})\right) \\ \boldsymbol{x}^*_{\mathbb{S}_{in}} &= (\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}(\mathbf{H}^T_{\mathbb{S}_{in}}\underline{\boldsymbol{r}} - \mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) \end{cases}$$

où $\underline{\boldsymbol{r}} := \boldsymbol{y} - \mathbf{H}_{\bar{\mathbb{S}}_0}\boldsymbol{x}^*_{\bar{\mathbb{S}}_0} - \mathbf{H}_{\bar{\mathbb{S}}_\square}\boldsymbol{x}^*_{\bar{\mathbb{S}}_\square} - \mathbf{H}_{\mathbb{S}_\square}\boldsymbol{x}^*_{\mathbb{S}_\square}$ est un vecteur constant. Nous rappelons que les équations (3.13f) et (3.13g) sont des systèmes linéaires couplés en $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$ et $\boldsymbol{x}^*_{\mathbb{S}_{in}}$:

$$\begin{cases} \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}\boldsymbol{x}^*_{\mathbb{S}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = \lambda\mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) & \text{(A.1)} \\ \mathbf{H}^T_{\mathbb{S}_{in}}(\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}\boldsymbol{x}^*_{\mathbb{S}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0 & \text{(A.2)} \end{cases}$$

À partir de (A.2), la solution de $\boldsymbol{x}^*_{\mathbb{S}_{in}}$ peut être calculée en fonction de $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$:

$$(A.2) \Leftrightarrow \boldsymbol{x}^*_{\mathbb{S}_{in}} = (\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}(\mathbf{H}^T_{\mathbb{S}_{in}}\underline{\boldsymbol{r}} - \mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}). \tag{A.3}$$

En récrivant (A.1) sous la forme

$$-\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\underline{\boldsymbol{r}} - \mathbf{H}_{\mathbb{S}_{in}}\boldsymbol{x}^*_{\mathbb{S}_{in}} - \mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) + \lambda\mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0$$

et en y injectant l'expression de $\boldsymbol{x}^*_{\mathbb{S}_{in}}$, on a :

$$-\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\underline{\boldsymbol{r}} + \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\mathbf{H}_{\mathbb{S}_{in}}[(\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}(\mathbf{H}^T_{\mathbb{S}_{in}}\underline{\boldsymbol{r}} - \mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})] + \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} + \lambda\mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0$$

Soit $\mathbf{P}^{\mathbb{S}_{in}} = \mathbf{H}_{\mathbb{S}_{in}}(\mathbf{H}^T_{\mathbb{S}_{in}}\mathbf{H}_{\mathbb{S}_{in}})^{-1}\mathbf{H}^T_{\mathbb{S}_{in}}$ la matrice de projection sur le sous-espace engendré par les colonnes de $\mathbf{H}_{\mathbb{S}_{in}}$ et $\mathbf{I}$ la matrice identité de taille appropriée. L'équation précédente devient :

$$-\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\underline{\boldsymbol{r}} + \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\mathbf{P}^{\mathbb{S}_{in}}\underline{\boldsymbol{r}} - \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\mathbf{P}^{\mathbb{S}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} + \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} + \lambda\mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0$$

$$\Leftrightarrow -\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\underline{\boldsymbol{r}} + \mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\mathbf{H}_{\bar{\mathbb{S}}_{in}}\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} + \lambda\mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}) = 0$$

Ainsi, la solution $\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}}$ est donnée par :

$$\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}} = \left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\mathbf{H}_{\bar{\mathbb{S}}_{in}}\right)^{-1}\left(\mathbf{H}^T_{\bar{\mathbb{S}}_{in}}(\mathbf{I}-\mathbf{P}^{\mathbb{S}_{in}})\underline{\boldsymbol{r}} - \lambda\ \mathrm{sgn}(\boldsymbol{x}^*_{\bar{\mathbb{S}}_{in}})\right). \tag{A.4}$$

Les équations (A.4) et (A.3) forment la solution recherchée.

A.2 Solutions pour les problèmes contraints $\hat{\mathcal{Q}}_{2/1}$ et $\hat{\mathcal{Q}}_{1/2}$ en Section 3.3.5

Nous établissons ici les expressions du paramètre $\gamma^\star$ permettant de calculer exactement, par l'algorithme homotopique, les solutions respectives du critère $\hat{\mathcal{Q}}_{2/1}$ contraint en norme ℓ_1 (3.1) et du critère $\hat{\mathcal{Q}}_{1/2}$ contraint par l'erreur quadratique (3.2) (voir le § 3.3.5).

— Pour $\hat{\mathcal{Q}}_{2/1}$, comme indiqué dans la section 3.3.5, l'algorithme s'arrête au premier point de rupture tel que la norme ℓ_1 des variables pénalisées $\|\boldsymbol{x}^{(k)}_{\bar{\mathbb{S}}}\|_1$ dépasse la valeur $\tau_c := M(K-n_1)$. Ensuite, dans l'intervalle correspondant $[\lambda^{(k)}, \lambda^{(k-1)}]$, la solution est donnée par l'équation (3.21)

$$\boldsymbol{x}^R = \boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}.$$

Par construction, il n'y a pas de changement de signe entre $\boldsymbol{x}^{(k-1)}$ et la solution optimale $\boldsymbol{x}^R$ telle que $\|\boldsymbol{x}^R\|_1 = \tau_c$. Ainsi, la valeur de $\gamma^\star$ telle que $\|\boldsymbol{x}^R\|_1 = \tau_c$ satisfait

$$\|\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}\|_1 = \tau_c \Leftrightarrow \mathrm{sgn}\left(\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}\right)^T (\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}) = \tau_c \tag{A.5}$$

avec $\mathrm{sgn}\left(\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}\right) = \mathrm{sgn}(\boldsymbol{x}^{(k-1)})$ puisque, par construction de la procédure homotopique, $\boldsymbol{x}^{(k-1)} + \gamma \boldsymbol{d}^{(k)}$ ne s'annule pas $\forall \gamma \in\ [0, \gamma^*]$. L'équation (A.5) s'écrit alors :

$$\mathrm{sgn}(\boldsymbol{x}^{(k-1)})^T(\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)}) = \tau_c \Leftrightarrow \|\boldsymbol{x}^{(k-1)}\|_1 + \gamma^\star \mathrm{sgn}(\boldsymbol{x}^{(k-1)})^T \boldsymbol{d}^{(k)} = \tau_c,$$

d'où l'expression finale :

$$\gamma^\star := \frac{\tau_c - \|\boldsymbol{x}^{(k-1)}\|_1}{\mathrm{sgn}(\boldsymbol{x}^{(k-1)})^T \boldsymbol{d}^{(k)}}.$$

— De même, pour $\hat{\mathcal{Q}}_{1/2}$, l'algorithme s'arrête au premier point de rupture tel que $\frac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^{(k)}\|_2^2 \leq \epsilon_c$. En remplaçant l'équation (3.21) dans l'expression des moindres carrés, la valeur de $\gamma^\star$ telle que

$$\tfrac{1}{2}\|\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^R\|_2^2 = \epsilon_c \quad \Leftrightarrow \|\boldsymbol{y} - \mathbf{H}(\boldsymbol{x}^{(k-1)} + \gamma^\star \boldsymbol{d}^{(k)})\|_2^2 = 2\epsilon_c$$

peut être trouvée en résolvant une équation quadratique scalaire. Soient $\boldsymbol{t}^{(k-1)} := \boldsymbol{y} - \mathbf{H}\boldsymbol{x}^{(k-1)}$ et $\boldsymbol{u}^{(k)} := \mathbf{H}\boldsymbol{d}^{(k)}$. Cette équation s'écrit :

$$\begin{aligned}\|\boldsymbol{t}^{(k-1)} - \gamma^\star \boldsymbol{u}^{(k)}\|_2^2 = 2\epsilon_c \quad &\Leftrightarrow \quad \gamma^{\star 2}(\boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}) - \gamma^\star(2\boldsymbol{t}^{(k-1)^T}\boldsymbol{u}^{(k)}) + (\boldsymbol{t}^{(k-1)^T}\boldsymbol{t}^{(k-1)} - 2\epsilon_c) = 0 \\ &\Leftrightarrow \quad \phi(\gamma) = 0,\end{aligned}$$

$$\text{avec } \phi(\gamma) := \underbrace{\boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}}_{:=a}\gamma^{\star 2} + \underbrace{(-2\boldsymbol{t}^{(k-1)T}\boldsymbol{u}^{(k)})}_{:=b}\gamma^\star + \underbrace{(\boldsymbol{t}^{(k-1)T}\boldsymbol{t}^{(k-1)} - 2\epsilon_c}_{:=c}.$$

Par construction de $\boldsymbol{x}^{(k-1)}$, $\phi(0) = \|\boldsymbol{y} - \boldsymbol{x}^{(k-1)}\|^2 - 2\epsilon_c \geq 0$ (l'erreur quadratique décroît lorsque λ décroît dans la construction homotopique). La plus petite racine de ϕ est alors en $\gamma^* = \frac{-\sqrt{\Delta}-b}{2a}$ avec $\Delta := b^2 - 4ac$, soit :

$$\gamma^* := \frac{\boldsymbol{t}^{(k-1)^T}\boldsymbol{u}^{(k)} - \sqrt{\left(\boldsymbol{t}^{(k-1)^T}\boldsymbol{u}^{(k)}\right)^2 - \boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}(\boldsymbol{t}^{(k-1)^T}\boldsymbol{t}^{(k-1)} - 2\epsilon_c)}}{\boldsymbol{u}^{(k)^T}\boldsymbol{u}^{(k)}},$$

qui est l'expression donnée par l'équation (3.23).

A.3 Calcul récursif de la matrice inverse des équations (3.18a)–(3.18b) en Section 3.3.6

La résolution des systèmes linéaires d'équations (3.18a)–(3.18b), dont la taille correspond respectivement à celle de $\bar{\mathbb{S}}_{in}$ et $\mathbb{S}_{in}$, est la partie majeure du temps de calcul à chaque itération. Puisque la configuration de support ne change que par une composante entre deux points de rupture, la matrice inverse des équations (3.18a)–(3.18b) peut être calculée de manière récursive en effectuant des mises à jour de rang 1. Pour cela, nous utilisons une stratégie rapide basée sur le lemme d'inversion des matrices partitionnées [Hager, 1989].

Quand il s'agit d'un ajout d'une colonne $\boldsymbol{h}_q$ à $\mathbf{H}_{\mathbb{S}}$ avec $s = (\#\mathbb{S})$, le calcul de $\hat{\mathbf{F}} := (\mathbf{H}^T_{\mathbb{S}\cup\{q\}}\mathbf{H}_{\mathbb{S}\cup\{q\}})^{-1}$ est donné récursivement à partir de $\mathbf{F} := (\mathbf{H}^T_{\mathbb{S}}\mathbf{H}_{\mathbb{S}})^{-1}$ par le lemme d'inversion des matrices partitionnées comme suit :

$$\hat{\mathbf{F}} = \left(\begin{array}{c|c} \boldsymbol{\beta} & \boldsymbol{\nu} \\ \hline \boldsymbol{\nu}^T & d \end{array}\right) \quad \text{avec} \left\{\begin{array}{rcl} \boldsymbol{\beta} & = & \mathbf{F} + d\mathbf{F}\mathbf{H}^T_{\mathbb{S}}\boldsymbol{h}_q\boldsymbol{h}_q^T\mathbf{H}_{\mathbb{S}}\mathbf{F}^T \\ \boldsymbol{\nu} & = & -d\mathbf{F}\mathbf{H}^T_{\mathbb{S}}\boldsymbol{h}_q \\ d & = & 1/(\boldsymbol{h}_q^T\boldsymbol{h}_q - \boldsymbol{h}_q^T\mathbf{H}_{\mathbb{S}}\mathbf{F}^T\mathbf{H}^T_{\mathbb{S}}\boldsymbol{h}_q) \end{array}\right.$$

L'algorithme est résumé dans Algorithme 6.

Algorithme 6 : Algorithme utilisé pour l'ajout d'une colonne ; Entrée : $(\mathbf{F}, \mathbf{H}_{\mathbb{S}}, \boldsymbol{h}_q)$ Sortie : $\hat{\mathbf{F}}$

$u_1 \leftarrow \mathbf{H}^T_{\mathbb{S}}\boldsymbol{h}_q$;	$\triangleright\ s \times N$
$u_2 \leftarrow \mathbf{F}u_1$;	$\triangleright\ s^2$
$d \leftarrow 1/(\boldsymbol{h}_q^T\boldsymbol{h}_q - u_1^T u_2)$;	$\triangleright\ s + N$
$\boldsymbol{\nu} \leftarrow -du_2$;	$\triangleright\ s$
$\boldsymbol{\beta} \leftarrow \mathbf{F} + du_2^T u_2$;	$\triangleright\ 2s^2 + s$
$\hat{\mathbf{F}} \leftarrow \begin{bmatrix} \boldsymbol{\beta} & \boldsymbol{\nu} \\ \boldsymbol{\nu}^T & d \end{bmatrix}$;	$\triangleright\ s^2$
	Complexité : $4s^2 + 4s$

À l'inverse de l'ajout, quand il s'agit d'un retrait d'une colonne $\boldsymbol{h}_q$ de $\mathbf{H}_{\mathbb{S}}$ avec $s = (\#\mathbb{S})$, le calcul de $\hat{\mathbf{F}} := (\mathbf{H}^T_{\mathbb{S}\setminus\{q\}}\mathbf{H}_{\mathbb{S}\setminus\{q\}})^{-1}$ est donné aussi récursivement à partir de $\mathbf{F} := (\mathbf{H}^T_{\mathbb{S}}\mathbf{H}_{\mathbb{S}})^{-1}$ par le lemme d'inversion des matrices partitionnées. L'algorithme est résumé dans Algorithme 7.

Algorithme 7 : Algorithme utilisé pour la suppression d'une colonne ; Entrée $(\mathbf{F}, \mathbb{S}, q)$ Sortie : $\hat{\mathbf{F}}$

% Trouvez la position de q dans $\mathbb{S}$

$p \leftarrow \text{pos}(q \in \mathbb{S})$; ▷ s

%Permutez la colonne p et la ligne p de $\mathbf{F}$ à la dernière position.

$\boldsymbol{\beta} \leftarrow \mathbf{F}(1 : p-1, 1 : p-1)$; ▷ s^2

$d \leftarrow \mathbf{F}(p, p)$;

$\boldsymbol{\nu} \leftarrow -\mathbf{F}(1 : p, p)$; ▷ s

$u_2 \leftarrow \boldsymbol{\nu}/d$; ▷ s

$\hat{\mathbf{F}} \leftarrow \boldsymbol{\beta} - du_2u_2^T$; ▷ $2s^2 + s$

Complexité : $3s^2 + 5s$

Titre : Méthodes de programmation en nombres mixtes pour l'optimisation parcimonieuse en traitement du signal

Mot clés : traitement du signal, recherche opérationnelle, parcimonie, optimisation en norme ℓ_0, programmation en nombres mixtes, algorithmes branch-and-bound, démélange spectral.

Résumé : L'approximation parcimonieuse consiste à ajuster un modèle de données linéaire au sens des moindres carrés avec un faible nombre de composantes non nulles (la "norme" ℓ_0). En raison de sa complexité combinatoire, ce problème d'optimisation est souvent abordé par des méthodes sous-optimales. Il a cependant récemment été montré que sa résolution exacte était envisageable au moyen d'une reformulation en programme en nombres mixtes (MIP), couplée à un solveur MIP générique, mettant en œuvre des stratégies de type $branch\text{-}and\text{-}bound$.

Cette thèse aborde le problème d'approximation parcimonieuse en norme ℓ_0 par la construction d'algorithmes $branch\text{-}and\text{-}bound$ dédiés, exploitant les structures mathématiques du problème. D'une part, nous interprétons l'évaluation de chaque nœud comme l'optimisation d'un critère en norme ℓ_1, pour lequel nous proposons des méthodes dédiées. D'autre part, nous construisons une stratégie d'exploration efficace exploitant la parcimonie de la solution, privilégiant l'activation de variables non nulles dans le parcours de l'arbre de décision. La méthode proposée dépasse largement les performances du solveur CPLEX, réduisant le temps de calcul et permettant d'aborder des problèmes de plus grande taille. Dans un deuxième volet de la thèse, nous proposons et étudions des reformulations MIP du problème de démélange spectral sous contrainte de parcimonie en norme ℓ_0 et sous des contraintes plus complexes de parcimonie structurée, généralement abordées de manière relâchée dans la littérature. Nous montrons que, pour des problèmes de complexité limitée, la prise en compte de manière exacte de ces contraintes est possible et permet d'améliorer l'estimation par rapport aux approches existantes.

Title: Mixed-Integer Programming Methods for sparse optimization in signal processing

Keywords: signal processing, operations research, sparsity, ℓ_0-norm optimization, mixed integer programming, branch-and-bound algorithms, spectral unmixing.

Abstract: Sparse approximation aims to fit a linear model in a least-squares sense, with a small number of non-zero components (the ℓ_0 "norm"). Due to its combinatorial nature, it is often addressed by suboptimal methods. It was recently shown, however, that exact resolution could be performed through a mixed integer program (MIP) reformulation solved by a generic solver, implementing branch-and-bound techniques.

This thesis addresses the ℓ_0-norm sparse approximation problem with tailored branch-and-bound resolution methods, exploiting the mathematical structures of the problem. First, we show that each node evaluation amounts to solving an ℓ_1-norm problem, for which we propose dedicated methods. Then, we build an efficient exploration strategy exploiting the sparsity of the solution, by activating first the non-zero variables in the tree search. The proposed method outperforms the CPLEX solver, reducing the computation time and making it possible to address larger problems. In a second part of the thesis, we propose and study the MIP reformulations of the spectral unmixing problem with ℓ_0-norm sparsity more advanced structured sparsity constraints, which are usually addressed through relaxations in the literature. We show that, for problems with limited complexity (highly sparse solutions, good signal-to-noise ratio), such constraints can be accounted for exactly and improve the estimation quality over standard approaches.