République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A.MIRA-BEJAIA
Faculté des Sciences Exactes
Département de Recherche Opérationnelle
Unité de recherche LaMOS (Modélisation et Optimisation des Systèmes)

THÈSE

Présentée par

HASSAINI Katia

Pour l'obtention du grade de

DOCTEUR EN SCIENCES

Filière : Mathématiques Appliquées

Option : Modélisation Mathématique et Techniques de Décision

Thème

Etude des M-matrices et leurs applications en optimisation

Soutenue le : 09/07/2025

Devant le Jury composé de :

Nom et Prénom	Grade		
Mr RADJEF Mohammed Said	Professeur	Univ. de Béjaïa	Président
Mr BIBI Mohand Ouamer	Professeur	Univ. de Béjaïa	Rapporteur
Mr AIDENE Mohamed	Professeur	Univ. de Tizi Ouzou	Examinateur
Mr OUKACHA Brahim	Professeur	Univ. de Tizi Ouzou	Examinateur
Mr BRAHMI Belkacem	MCA	Univ. de Béjaïa	Examinateur
Mr RAHAL Mohamed	MCA	Univ. de Sétif 1	Examinateur

Année Universitaire : **2024/2025**

Remerciements

Soyons reconnaissants aux personnes qui nous donnent
du bonheur ; elles sont les charmants jardiniers
par qui nos âmes sont fleuries.
Marcel Proust

Avant tout, je remercie Dieu, le Tout-Puissant, de m'avoir accordé la volonté, la patience et le courage nécessaires à l'aboutissement de cette thèse.

J'exprime ma sincère gratitude et mes profonds remerciements à mon directeur de thèse, Monsieur **Mohand Ouamer BIBI**, Professeur à l'Université A. Mira de Béjaïa. Son encadrement bienveillant, sa précieuse disponibilité et son soutien inestimable ont été déterminants dans l'avancement de mes recherches. Grâce à lui, j'ai pu nourrir ma curiosité scientifique et approfondir mon sujet avec rigueur et passion.

Je tiens également à exprimer ma profonde reconnaissance au Professeur **Mohammed Said RADJEF**, de l'Université de Béjaïa, pour l'honneur qu'il m'a fait en acceptant de présider ma soutenance.

Mes sincères remerciements vont également au Professeur **Mohamed AIDENE**, de l'Université de Tizi Ouzou, pour avoir accepté de faire partie du jury de cette thèse.

J'exprime aussi toute ma reconnaissance au Professeur **Brahim OUKACHA**, de l'Université de Tizi Ouzou, pour avoir accepté d'examiner et d'évaluer mon travail.

Je remercie chaleureusement Monsieur **Belkacem BRAHMI**, Maître de Conférences à l'Université de Béjaïa, pour son engagement et sa contribution à l'évaluation de mon travail.

Enfin, j'adresse mes vifs remerciements à Monsieur **Mohamed RAHAL**, Maître de Conférence à l'Université de Sétif 1, pour son intérêt envers mon travail et pour avoir accepté de l'examiner.

Ma gratitude va également à tous les membres de l'unité de recherche **LaMOS** (Laboratoire de Modélisation et d'Optimisation des Systèmes), avec qui les échanges et discussions ont été une source précieuse d'enrichissement tout au long de mon parcours.

Que toutes celles et ceux qui, de près ou de loin, m'ont soutenue, encouragée ou inspirée, ainsi que ceux qui ont contribué, même involontairement, à cette aventure scientifique, trouvent ici l'expression de ma profonde reconnaissance.

Dédicaces

À la mémoire de mon père,
Qui reste ma source d'inspiration et de force. Ton amour, tes valeurs et ton exemple continuent d'éclairer mon chemin, même en ton absence. Chaque accomplissement est une manière de te rendre hommage, et c'est à toi que je dédie ce travail. Tu es toujours avec moi, dans mon cœur et mes pensées.

À ma mère, qui a tant souffert dans sa vie,
Pour sa force inébranlable, son courage et son amour incommensurable. Malgré toutes les épreuves, tu as toujours su m'offrir le meilleur de toi-même, et c'est grâce à toi que je suis celle que je suis aujourd'hui. Cette thèse est aussi dédiée à toi, en reconnaissance de ta lutte, de ta résilience et de l'amour que tu m'as donné.

À mes frères et sœurs,
Pour leur soutien constant, leur affection et leur présence précieuse dans ma vie.

À mon mari,
Pour son amour, sa patience et son soutien indéfectible. Ton soutien moral et émotionnel m'a permis de traverser chaque étape de ce travail avec détermination. Tu es ma source de réconfort et d'inspiration.

À mes petites filles Massilia-Yassmin et Mailis-Léa, à ma nièce Sara,
Pour leur joie, leur innocence et leur lumière. Vous êtes mon rayon de soleil et ma motivation à aller de l'avant. Cette thèse est dédiée à vous toutes, avec tout mon amour.

À ma belle-famille,
Pour leur accueil chaleureux et leur soutien précieux. Merci de m'avoir intégrée avec générosité et compréhension.

À la mémoire de mon père.

Valorisation des travaux de thèse

Les résultats des travaux présentés dans cette thèse ont fait l'objet de deux conférences nationales, trois conférences internationales et un article publié dans une revue internationale.

- **Conférences nationales** :

  — K. HASSAINI and M. O. BIBI. Résolution d'un problème de programmation quadratique avec une M-matrice. Séminaire Mathématique de Bejaia, Algérie, Juillet 2014.

  — K. HASSAINI and M. O. BIBI. Résolution d'un problème de programmation quadratique à variables bornées avec une M-matrice. Séminaire Mathématique de Bejaia, Algérie, Février 2015.

- **Conférences internationales** :

  — K. HASSAINI and M. O. BIBI. Résolution d'un problème de programmation quadratique avec une M-matrice. *COSI'2010, Actes du Colloque International sur l'Optimisation et les Systèmes d'Information COSI'2010, pages 123-134, Ouargla, Algérie, Avril 18 - 20, 2010*.

  — K. HASSAINI and M. O. BIBI. Quadratic Programming Problems with a Diagonally Dominant M-matrix. *MOAD'2022, Actes du Colloque International sur les Méthodes et Outils d'Aide à la Décision, pages 128–134, Bejaia, Algeria, November 15 - 17, 2022*.

  — K. HASSAINI and M. O. BIBI. Quadratic Programming Problems with Preprocessing and a Diagonally Dominant M-matrix. *Research in Computer Science : 17th African Conference on Research in Computer Science and Applied Mathematics, CARI 2024, Bejaia, Algeria, November 24-26, 2024, Proceedings.*

- **Article publié** :

  — K. HASSAINI and M. O. BIBI. Quadratic programming method with an $M-$matrix. *Statistics, Optimization and Information Computing*, Vol 12(2) : 405–417, 2024.

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Glossaire

$B \geq 0$**:** Tous les coefficients de la matrice B sont non négatifs . . . . . p. 19

$D \preceq 0$**:** La matrice D est semi-définie négative . . . . . p. 18

$D \succ 0$**:** La matrice D est définie positive . . . . . p. 16–18

$D \succeq 0$**:** La matrice D est semi-définie positive . . . . . p. 16–18

$Z^{n\times n}$**:** L'ensemble des matrices carrées réelles d'ordre n de la forme $A = (a_{ij})$, où $a_{ij} \leq 0,\ \forall i \neq j, 1 \leq i, j \leq n$ . . . . . p. 2

Box-QP: Problème de Programmation Quadratique avec une M-matrice à diagonale dominante et des contraintes de Bornes . . . . . p. 72–74, 76, 77, 79, 80, 82–85, 88, VIII

DN: Définie Négative . . . . . p. 16

DP: Définie Positive . . . . . p. 16, 18

EDP: Équations aux Dérivées Partielles . . . . . p. 51, 71

ND: Non Définie . . . . . p. 16

PCL: Problème de Complémentarité Linéaire . . . . . p. 2, 39, 45, 50–52

PPQ: Problème de Programmation Quadratique . . . . p. 39, 51, 52, 54, 62, 63, 87, 88, VIII

PPQs: Problèmes de Programmation Quadratique . . . . . p. 1, 3, 4, 45, 52, 71, 87, 88

PQS: Programmation Quadratique Séquentielle . . . . . p. 1

SDN: Semi-Définie Négative . . . . . p. 16

SDP: Semi-Définie Positive . . . . . p. 16, 18

SRS: Solution Réalisable de Support . . . . . p. 78, 82, 83, VIII

Introduction générale

L'optimisation est une discipline fondamentale des mathématiques appliquées et de la recherche opérationnelle, jouant un rôle central dans de nombreux domaines tels que l'économie, l'ingénierie et les sciences physiques. Parmi ses branches, la Programmation Quadratique (PQ) se distingue par son importance théorique et ses nombreuses applications pratiques. En effet, un grand nombre de problèmes concrets, qu'ils relèvent de l'économie, de la commande optimale, de l'ingénierie ou de la recherche opérationnelle, se modélisent naturellement sous forme de modèles quadratiques.

De manière générale, un problème de programmation quadratique consiste à minimiser une fonction objectif quadratique soumise à des contraintes linéaires ou non linéaires. Ce cadre analytique permet de représenter une vaste gamme de scénarios pratiques, allant de la gestion de portefeuilles financiers à l'optimisation des réseaux, en passant par l'apprentissage automatique et la mécanique structurelle. Grâce à ses propriétés analytiques et numériques, la programmation quadratique constitue également une base solide pour le développement de méthodes avancées, notamment en optimisation non linéaire.

Sur le plan théorique, vu que l'optimisation non linéaire ne dispose pas d'un cadre universel pour la construction d'algorithmes, la programmation quadratique joue un rôle central. Par exemple, les *méthodes de Programmation Quadratique Séquentielle (PQS)* exploitent des approximations quadratiques des fonctions non linéaires, permettant ainsi de résoudre des problèmes complexes avec une plus grande robustesse. Ces caractéristiques font de la programmation quadratique un outil indispensable, non seulement pour traiter des problèmes pratiques, mais aussi pour concevoir des algorithmes performants dans des contextes variés [13, 80].

Un élément clé dans la résolution des Problèmes de Programmation Quadratique (PPQs) réside dans la matrice associée à la forme quadratique. Ses qualités structurelles nous renseignent sur de nombreuses propriétés, telles que la convexité du problème, la stabilité des solutions et l'efficacité des algorithmes de résolution. Plusieurs approches ont été proposées pour résoudre un problème de programmation quadratique convexe, c'est à dire lorsque la matrice associée est semi-définie positive [9, 10, 11, 12, 14, 15, 23, 24, 44, 82, 87, 92, 111]. Cependant, une

classe spécifique de matrices, appelées M-matrices, joue encore un rôle fondamental en mathématiques appliquées et en optimisation. Ces matrices se distinguent par des caractéristiques uniques, notamment leur positivité, leur dominance diagonale et leur stabilité.

Les M-matrices sont largement reconnues pour la diversité de leurs applications dans divers domaines, tels que la modélisation des systèmes dynamiques, des sciences économiques et de l'écologie [8, 107, 100]. Ces matrices interviennent dans la résolution de divers problèmes, notamment les problèmes de Dirichlet avec obstacles [52, 55, 90] et des modèles d'application de torsion à une barre [16, 89]. Elles apparaissent souvent dans l'analyse des chaînes de Markov [59]. Elles possèdent plusieurs propriétés fondamentales, couramment employées pour démontrer des résultats de stabilité dans les systèmes évolutifs (dynamiques) [90, 94, 99, 100]. La minimisation quadratique impliquant une M-matrice joue également un rôle central dans de nombreuses applications, telles que l'optimisation de portefeuille avec des coûts de transaction [71], la valorisation des options américaines [25, 88, 108] et la segmentation d'images [49].

Historiquement, le terme de M-matrice a été utilisé pour la première fois par *Ostrowski en 1937* [81], ce dernier inspiré des travaux de *Minkowski* [75, 76, 77] qui a prouvé que si $A \in Z^{n\times n}$ a toutes ses sommes de lignes positives, alors le déterminant de A est positif. Depuis, leur étude a évolué pour englober des propriétés structurelles et des applications spécifiques dans plusieurs disciplines. Le premier effort systématique pour caractériser les M-matrices a été réalisé par *Fiedler et Pták en 1962* [27], qui ont également identifié de nombreuses propriétés fondamentales de ces matrices, contribuant ainsi à établir leur rôle dans les systèmes linéaires. Une première étude de la théorie des M-matrices a été réalisée par *Poole et Bouillon en 1974* [86]. En *1976, Varga* [103, 104] a étudié le rôle des matrices à diagonale dominante dans la théorie des M-matrices soulignant leur importance dans l'analyse numérique et les problèmes de stabilité. En outre, *Schröder* en 1976 [93] a étudié certaines des propriétés des M-matrices non singulières à l'aide de la théorie des opérateurs et des espaces linéaires partiellement ordonnés. Entre-temps, *Kaneko* [56] a rédigé une liste exhaustive de caractéristiques et d'applications des M-matrices non singulières, particulièrement dans le contexte du Problème de Complémentarité Linéaire (PCL) en recherche opérationnelle.

En *1979*, *Berman et Plemmons* [8] ont proposé une définition plus rigoureuse des M-matrices, en mettant en avant leurs propriétés de stabilité ainsi que leur lien avec les matrices à éléments non positifs. Ce travail a constitué une avancée majeure dans la compréhension de ces matrices, en formalisant leur classification et en élargissant leurs applications, notamment dans les problèmes de complémentarité. En *1980*, *Luk et Pagano* [69] ont publié un article intitulé "*Quadratic Programming with M-Matrices*", prolongeant les travaux de *Chandrasekaran* en 1970 [17], qui avait proposé un algorithme pour résoudre le problème de complémentarité

linéaire à une seule variable, ainsi que ceux de *Pang* [83], qui avait développé un algorithme analogue pour le problème de complémentarité linéaire avec des variables bornées. En *1989*, *Stachurski* [95, 96, 97, 98] a proposé un algorithme général pour résoudre les problèmes de programmation quadratique (PPQs) impliquant une M-matrice et des contraintes de bornes. Cet algorithme, dont les propriétés de convergence ont été minutieusement étudiées, construit une séquence de points dont les composantes augmentent de manière monotone. Par ailleurs, il a été démontré que plusieurs algorithmes déjà existants, notamment ceux de *Pang* [83] et de *Scarpini* [90], peuvent être interprétés comme des cas particuliers de ce cadre algorithmique général.

Des travaux plus récents [6, 53, 55, 57, 67] sur la programmation quadratique convexe avec une M-matrice ont été menés et approfondis au fil des années, en raison du fait qu'une grande partie des problèmes d'obstacles, présents dans des domaines variés tels que l'optimisation, la mécanique des milieux continus, la dynamique des structures et les problèmes de contrôle, sont fréquemment modélisés sous la forme de problèmes de programmation quadratique. *En 1999*, Un algorithme basé sur la méthode des points intérieurs a été développé pour traiter le problème d'obstacle bilatéral affine avec une M-matrice [52]. *En 2003*, une méthode d'activation de contraintes a également été utilisée pour résoudre le même problème [57]. La même année, une extension de la méthode des points externes a été proposée dans [63]. *En 2011*, un algorithme direct a été présenté dans [55]. Enfin, en *2017*, une méthode destinée à traiter les problèmes quadratiques strictement convexes, reposant sur l'extension de la méthode des points externes mentionnée précédemment, a été proposée dans [53].

Notre contribution principale dans cette thèse consiste en l'élaboration d'une nouvelle méthode spécifiquement conçue pour traiter des problèmes de programmation quadratique impliquant une M-matrice et des contraintes simples. En nous s'inspirant des travaux de R. Chandrasekaran [17], de F.T. Luk et M. Pagano [69], et en introduisant le concept de support pour la fonction objectif proposé par Gabasov et al. [33, 34, 35, 37], nous avons conçu une méthode qui se distingue des approches précédentes par l'introduction d'une condition plus générale. Cette dernière facilite l'obtention d'une solution réalisable initiale qui est significativement plus rapprochée de l'optimum, ce qui améliore considérablement la rapidité de convergence du processus. Nous étendons également notre étude aux problèmes de programmation quadratique impliquant une M-matrice à diagonale dominante et des contraintes de bornes. Dans ce cadre, une procédure de prétraitement est introduite pour exploiter la structure particulière de la M-matrice, réduisant ainsi la taille du problème et améliorant l'efficacité computationnelle. Un algorithme d'optimisation, inspiré de la méthode des points extérieurs [106] et intégrant le concept de support, est proposé pour résoudre ce problème réduit. Cette approche permet non seulement d'améliorer la convergence, mais également de réduire significativement le temps de

calcul, rendant ainsi la résolution à la fois plus rapide et plus efficace.

Cette thèse est structurée comme suit : une introduction générale, suivie de quatre chapitres détaillant les différentes étapes de la recherche, et enfin une conclusion générale qui résume les principales contributions ainsi que les perspectives futures.

Le premier chapitre est consacré à la présentation des bases théoriques nécessaires à la compréhension des enjeux liés à la programmation quadratique et aux M-matrices. Nous y introduisons les concepts fondamentaux de la convexité et de l'analyse convexe, les propriétés des ensembles et des fonctions convexes, ainsi que les caractéristiques des formes quadratiques, telles que leur gradient et leur Hessien. Nous détaillons également les M-matrices, en exposant leur structure et leur rôle crucial dans la résolution des systèmes linéaires et des problèmes d'optimisation. Enfin, ce chapitre aborde les principes de l'optimisation convexe et de la programmation non linéaire, en mettant en lumière les conditions d'optimalité pour les problèmes avec ou sans contraintes. Ces rappels théoriques fournissent une base solide pour le développement des méthodes de résolution présentées dans les chapitres suivants.

Dans le deuxième chapitre, nous approfondissons l'approche théorique en présentant deux exemples concrets de problèmes pratiques pouvant être modélisés sous la forme d'un programme quadratique, en particulier ceux impliquant des M-matrices. Le premier exemple, le **"Circus Tent Problem"** qui appartient à la catégorie des problèmes d'obstacles et qui illustre comment les M-matrices émergent dans la modélisation de contraintes physiques et géométriques complexes. Le second exemple concerne la **"Valorisation des options américaines"**, un problème financier où les M-matrices jouent un rôle clé dans la formulation et la résolution du modèle sous-jacent. Ces deux cas mettent en évidence l'importance des propriétés structurelles des M-matrices dans des situations réelles et ouvrent la voie à une réflexion approfondie sur les méthodes numériques les plus appropriées pour résoudre de tels problèmes.

Le troisième chapitre, cœur de cette thèse et représentant notre contribution principale, est consacré à la proposition d'une nouvelle méthode de résolution des Problèmes de Programmation Quadratique (PPQs), lorsque la matrice associée est une M-matrice et les contraintes sont simples. En exploitant les propriétés spécifiques des M-matrices, notre approche génère une suite monotone de solutions réalisables, garantissant ainsi une convergence rapide vers la solution optimale. Nous y avons intégré le concept de support, développé par Gabasov et al. [33, 34, 35, 36, 37], afin d'obtenir une solution réalisable initiale plus proche de l'optimum, ce qui permet d'améliorer considérablement la rapidité de convergence par rapport aux méthodes existantes. Une implémentation en MATLAB a été réalisée pour comparer nos résultats à ceux d'autres approches, mettant ainsi en évidence l'efficacité de notre méthode.

Dans le quatrième chapitre, nous étendons cette méthode à des cas spécifiques, notamment la résolution de problèmes de programmation quadratique avec une M-matrice à diagonale dominante et des contraintes de bornes. Une procédure de prétraitement est introduite pour exploiter la structure particulière de la M-matrice, réduisant ainsi la taille du problème et améliorant l'efficacité computationnelle. Un algorithme d'optimisation, inspiré de la méthode des points extérieurs, est proposé pour résoudre ce problème réduit. Cette approche permet d'améliorer la convergence et de réduire significativement le temps de calcul, rendant ainsi la résolution plus rapide et plus efficace.

Enfin, cette thèse se conclut par un récapitulatif des contributions principales, une analyse des résultats obtenus et des perspectives pour de futures recherches dans ce domaine. Les travaux futurs visent à améliorer les méthodes développées et à les étendre à des problèmes encore plus complexes, ouvrant ainsi de nouveaux axes pour la recherche en optimisation.

Chapitre 1

Rappels Mathématiques pour la Programmation Non-linéaire

1.1 Introduction

Ce chapitre est essentiel pour établir les bases mathématiques nécessaires à cette thèse. Il commence par un rappel sur les valeurs propres et vecteurs propres d'une matrice et sur les concepts fondamentaux de la convexité et de l'analyse convexe, englobant les ensembles convexes ainsi que les fonctions convexes et non convexes. Ensuite, il est important de revoir les notions relatives aux formes quadratiques définies (ou semi-définies) positives, en mettant l'accent sur leurs propriétés, qui sont fondamentales dans le contexte de l'optimisation non linéaire.

Par ailleurs, nous présentons les M-matrices en soulignant leurs propriétés structurelles, telles que la dominance diagonale et la positivité de l'inverse. Ces caractéristiques confèrent aux M-matrices un rôle essentiel en modélisation mathématique, notamment pour l'analyse de la stabilité et la résolution de systèmes linéaires. Elles occupent une place centrale dans notre recherche en raison de leur importance dans l'optimisation et la modélisation de divers phénomènes physiques réels.

La dernière partie du chapitre est dédiée aux résultats fondamentaux en optimisation non linéaire. Nous y abordons les conditions nécessaires et suffisantes d'optimalité, tant pour des problèmes sans contraintes que pour des problèmes avec contraintes. Ces résultats sont cruciaux pour comprendre comment aborder et résoudre efficacement des problèmes de programmation non linéaire.

1.2 Introduction aux vecteurs et matrices

1.2.1 Espace vectoriel

Définition 1.1. *Un ensemble ordonné de n nombres réels* $x = (x_1, x_2, \cdots, x_n)^T$ *s'appelle point ou vecteur d'un espace de dimension n et les nombres* $x_1, x_2, \cdots, x_n$ *sont dits composantes ou coordonnées du vecteur x.*

Définition 1.2. *On définit un espace vectoriel en munissant un ensemble E de deux lois : une addition interne et une multiplication externe, qui vérifient les propriétés suivantes :*

- *L'ensemble E muni de la loi + est un groupe commutatif ,*
- *De plus, on a*
  - $(\alpha + \beta)x = \alpha x + \beta x$,
  - $\alpha(x + y) = \alpha x + \alpha y$,
  - $\alpha(\beta x) = (\alpha\beta)x$,
  - $1x = x$,

  où α *et* β *sont des scalaires quelconques, x et y sont des vecteurs, des éléments quelconques de E. Si E est de dimension n,* $x = (x_1, x_2, \cdots, x_n)^T$, $y = (y_1, y_2, \cdots, y_n)^T$ *et* $\alpha \in \mathbb{R}$*, on a alors :*

$$x + y = (x_1 + y_1, x_2 + y_2, \cdots, x_n + y_n)^T, \tag{1.1}$$

$$\alpha x = (\alpha x_1, \alpha x_2, \cdots, \alpha x_n)^T. \tag{1.2}$$

  L'ensemble $\mathbb{R}^n$ *est muni des propriétés classiques d'un espace vectoriel, c'est donc un espace vectoriel sur* $\mathbb{R}$*.*

1.2.2 Sous-espace linéaire

Un ensemble E de vecteurs de l'espace $\mathbb{R}^n$ de dimension n s'appelle un *sous-espace linéaire* de $\mathbb{R}^n$ s'il vérifie les conditions suivantes :

1. $x + y \in E$, $x \in E$ et $y \in E$,
2. $\alpha x \in E$, $x \in E$ et α un scalaire quelconque.

1.2.3 Produit scalaire

Définition 1.3. *On appelle* ***produit scalaire*** *sur E une application de* $E \times E$ *sur* $\mathbb{R}$*, notée* $\langle x, y \rangle$*, et possédant les propriétés suivantes :*

- $\langle x, x \rangle \geq 0$,
- $\langle x, x \rangle = 0 \iff x = 0$,
- $\langle x, y \rangle = \langle y, x \rangle$,

— $\langle x, y + z\rangle = \langle x, y\rangle + \langle x, z\rangle$,
— $\langle x, \lambda y\rangle = \lambda\langle x, y\rangle, \quad \forall\lambda \in \mathbb{R}$.

1.2.4 Normes

Définition 1.4. *On appelle **norme** sur E une application de E dans $\mathbb{R}^+$, notée $x \mapsto ||x||$, qui possède les propriétés suivantes :*

1. $||x|| = 0 \iff x = 0$,
2. $||\lambda x|| = |\lambda|||x||, \quad \forall\lambda \in \mathbb{R}$,
3. $||x + y|| \leq ||x|| + ||y||$.

Définition 1.5. *(Norme euclidienne sur $\mathbb{R}^n$)*
Dans l'espace vectoriel $\mathbb{R}^n$, on définit le produit scalaire entre deux vecteurs x et y par :

$$\langle x, y\rangle = \sum_{i=1}^{n} x_i y_i,$$

*et la **norme euclidienne** d'un vecteur x associée est :*

$$||x||_2 = \sqrt{\langle x, x\rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Sur l'espace vectoriel $\mathbb{R}^n$, d'autres normes en plus de la norme euclidienne peuvent être définies. Par exemple, l'application suivante définit une norme :

$$||x||_p = \left(\sum_{i=1}^{n} x_i^p\right)^{\frac{1}{p}}, \quad \forall p \geq 1.$$

Les autres normes les plus utilisées sont :

$$||x||_1 = \sum_{i=1}^{n} |x_i|, \quad ||x||_\infty = \max_{i=1,\ldots,n} |x_i|,$$

*appelées respectivement **norme 1** et **norme infinie**.*

1.2.5 Systèmes orthogonaux de vecteurs

Définition 1.6. *Un ensemble quelconque de n vecteurs linéairement indépendants d'un espace de dimension n s'appelle **base** de cet espace.*

Définition 1.7. *On appelle **base canonique** la base composée des vecteurs $\{e_i\}_{i=1,\ldots,n}$, où le*

vecteur e_i possède des composantes nulles, sauf la i-ème qui vaut 1 *:*

$$e_i = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \longleftarrow i^{ème} \text{élément}.$$

Ainsi, tout vecteur x de $\mathbb{R}^n$ se décompose sur la base canonique de la manière suivante :

$$\forall x \in \mathbb{R}^n, \quad x = \sum_{i=1}^{n} \langle x, e_i \rangle e_i = \sum_{i=1}^{n} x_i e_i.$$

Définition 1.8. *Deux vecteurs x et y de $E^n = \mathbb{R}^n$ sont dits **orthogonaux** si leur produit scalaire est nul :*

$$\langle x, y \rangle = 0.$$

Le vecteur nul est évidemment orthogonal à tout vecteur de l'espace.

Définition 1.9. *Un système de vecteurs $x^1, x^2, \ldots, x^m$ s'appelle **orthogonal** si tous ses vecteurs sont orthogonaux deux à deux :*

$$\langle x^j, x^k \rangle = 0 \quad avec\ j \neq k.$$

Définition 1.10. *La base $e_1, e_2, \ldots, e_n$ de $\mathbb{R}^n$ est dite **orthogonale** si les vecteurs de la base sont orthogonaux deux à deux :*

$$\langle e_j, e_k \rangle = 0 \quad si\ j \neq k, \quad j, k = 1, 2, \ldots, n.$$

*Si, de plus, on a $\|e_k\| = 1$ pour $k = 1, \ldots, n$, alors la base orthogonale s'appelle **orthonormée**. Dans ce cas, on a :*

$$\langle e_j, e_k \rangle = \delta_{jk},$$

*où δ_{jk} est le **symbole de Kronecker** défini par :*

$$\delta_{jk} = \begin{cases} 1 & si\ j = k, \\ 0 & si\ j \neq k. \end{cases}$$

1.2.6 Les Matrices

Définition 1.11. *Soient $n, m \in \mathbb{N}^*$. Une matrice de dimension $m \times n$ à coefficients réels est un tableau constitué de m lignes et n colonnes, représenté comme suit :*

$$A = A(I, J) = (a_{ij},\ i \in I,\ j \in J) = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix},$$

où les ensembles $I = \{1, 2, \ldots, m\}$ et $J = \{1, 2, \ldots, n\}$ désignent respectivement les indices des lignes et des colonnes de A.

Pour des raisons pratiques, la matrice A peut aussi être notée sous la forme suivante :

$$A = (a_1, a_2, \ldots, a_j, \ldots, a_n) = \begin{pmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_i^T \\ \vdots \\ A_m^T \end{pmatrix},$$

où

$$a_j = A(I, j) = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix}$$

est un vecteur-colonne de dimension m, et

$$A_i^T = A(i, J) = (a_{i1}, a_{i2}, \ldots, a_{in})$$

est un vecteur-ligne de dimension n. Le symbole $(^T)$ représente la transposition. Ainsi, un vecteur $x = x(J) = (x_j,\ j \in J)$ est considéré comme un vecteur-colonne, tandis que son transposé, noté x^T, est un vecteur-ligne.

La matrice transposée de A est définie par :

$$A^T = A^T(J, I) = (a_{ji} = a_{ij},\ j \in J,\ i \in I).$$

À noter qu'un vecteur-colonne de dimension n peut être vu comme une matrice de taille $(n \times 1)$, tandis qu'un vecteur-ligne de dimension n correspond à une matrice de taille $(1 \times n)$.

*Une matrice A est dite **carrée** lorsque $n = m$. De plus, si $A = A^T$, elle est qualifiée de*

symétrique. *Enfin, la matrice identité d'ordre n est notée* I_n.

1.2.7 Valeurs propres et vecteurs propres de matrices

Définition 1.12. *Soit A une matrice carrée d'ordre n à coefficients dans* $\mathbb{R}$. *Un scalaire* $\lambda \in \mathbb{C}$ *est appelé valeur propre de A s'il existe un vecteur* $x \in \mathbb{C}^n$ *non nul tel que :*

$$Ax = \lambda x.$$

Le vecteur x satisfaisant cette dernière relation est appelé vecteur propre de A correspondant à la valeur propre λ.

Définition 1.13. *On appelle spectre de la matrice A l'ensemble des valeurs propres de A, que l'on note : spec*(A).

Remarque 1.1. *L'ensemble* E_λ *de tous les vecteurs propres associés à la valeur propre* λ, *où l'on ajoute le vecteur nul, est un sous-espace de* $\mathbb{C}^n$, *appelé sous-espace propre associé à la valeur propre* λ.

Définition 1.14. *Le rayon spectral de A est le nombre*

$$\rho(A) = \max_{i=1,\dots,n} |\lambda_i|.$$

Remarque 1.2. *Un vecteur propre n'est défini qu'à un facteur multiplicatif près :*

$$Ax = \lambda x \quad \Rightarrow \quad A(\alpha x) = \lambda(\alpha x), \quad \forall \alpha \neq 0.$$

On peut donc choisir α *de telle sorte que* $\|x\| = 1$.

Le théorème suivant permet d'avoir une règle générale pour calculer les valeurs et les vecteurs propres d'une matrice.

Théorème 1.1. *Soit A une matrice carrée d'ordre n et à éléments dans* $\mathbb{R}$. *Alors les propriétés suivantes sont équivalentes :*

1. *Le scalaire* λ *est une valeur propre de A,*
2. *La matrice* $M = A - \lambda I_n$ *est singulière,*
3. *Le scalaire* λ *est une racine du polynôme de degré n, appelé polynôme caractéristique de A :*

$$P_A(\lambda) = \det(A - \lambda I_n) = 0.$$

Proposition 1.1. *Soit A une matrice carrée d'ordre n telle que* $A = A^T$. *Les propriétés suivantes s'appliquent à A :*

— *Les valeurs propres de A sont réelles,*

— *Les vecteurs propres associés à des valeurs propres de A distinctes sont orthogonaux entre eux.*

Ces résultats découlent du théorème spectral pour les matrices symétriques, qui stipule qu'une matrice symétrique A peut être diagonalisée par une matrice orthogonale Q. En d'autres termes, il existe une matrice orthogonale Q et une matrice diagonale Λ telles que :

$$A = Q\Lambda Q^T,$$

où Λ est une matrice diagonale contenant les valeurs propres réelles de A, et les colonnes de Q sont les vecteurs propres orthonormés de A, avec $QQ^T = I_n$.

1.3 Convexité et Analyse Convexe

1.3.1 Ensembles convexes

Définition 1.15. *Un ensemble C de* $\mathbb{R}^n$ *est dit convexe si, pour tout* $x_1, x_2 \in C$ *et tout* $\lambda \in [0, 1]$*, le vecteur*

$$x = \lambda x_1 + (1 - \lambda)x_2$$

appartient également à C.

La figure suivante illustre cette notion :

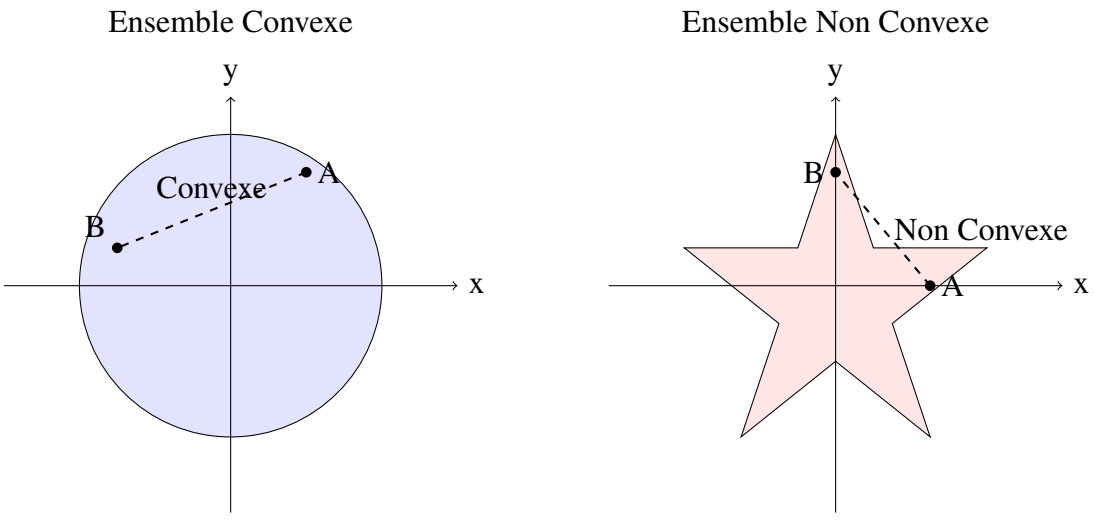

FIGURE 1.1 – Ensembles convexe et non convexe dans $\mathbb{R}^2$.

1.3.2 Propriétés des ensembles convexes

Propriété 1.1. *Si C est convexe, et $\alpha \in \mathbb{R}$, alors l'ensemble $E = \{y : \ y = \alpha x, \ x \in C\}$ est également convexe.*

Propriété 1.2. *Si C_1 et C_2 sont deux ensembles convexes de $\mathbb{R}^n$, alors l'ensemble $C = C_1 \cap C_2$ est convexe et l'ensemble $C = C_1 + C_2 = \{x : \ x = x_1 + x_2, \ x_1 \in C_1 \text{ et } x_2 \in C_2\}$ est convexe.*

Propriété 1.3. *L'intersection de n'importe quelle collection-finie ou infinie-d'ensembles convexes est convexe.*

Définition 1.16. *Un point x d'un ensemble convexe C est dit un point extrême s'il n'y a pas de points distincts $x_1, \ x_2 \in C$ tels que :*

$$x = \lambda x_1 + (1 - \lambda)x_2, \ \ 0 < \lambda < 1.$$

1.3.3 Fonctions convexes

La convexité est un concept clé en optimisation traditionnelle, jouant un rôle essentiel dans l'établissement des critères d'optimalité simultanément nécessaires et suffisants. En effet, la majorité des algorithmes de minimisation convexe s'avèrent particulièrement efficaces. Dans cette section, nous présenterons un bref aperçu des propriétés des fonctions convexes.

Définition 1.17. *Une fonction $f : \mathbb{R}^n \to \mathbb{R}$ est dite convexe si, pour tout $x, y \in \mathbb{R}^n$ et pour tout $\lambda \in [0, 1]$, l'inégalité suivante est satisfaite :*

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y). \tag{1.1}$$

Cette propriété signifie que le segment de droite reliant les points $(x, f(x))$ et $(y, f(y))$ est toujours situé au-dessus du graphe de la fonction.

Définition 1.18. *Une fonction $f : \mathbb{R}^n \to \mathbb{R}$ est dite strictement convexe, si l'inégalité (1.1) est stricte pour tous les points $x, \ y \in \mathbb{R}^n$, tels que $x \neq y$, avec $0 < \ \lambda \ < 1$.*

Remarque 1.3.

1- *Une fonction $f : \mathbb{R}^n \to \mathbb{R}$ est dite concave si la fonction $-f$ est convexe.*
2- *Une fonction $f : \mathbb{R}^n \to \mathbb{R}$ est dite strictement concave si la fonction $-f$ est strictement convexe.*

La figure ci-dessous illustre deux exemples : une fonction quadratique présentant une convexité et une autre ne possédant pas cette propriété.

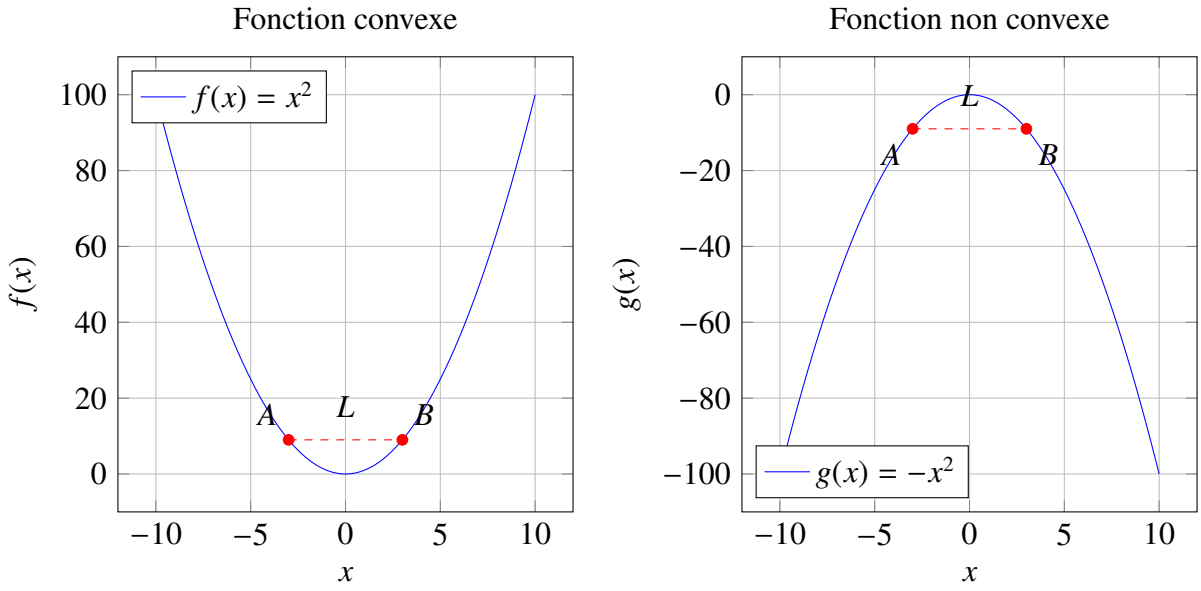

FIGURE 1.2 – Exemple d'une fonction quadratique convexe et non convexe

1.3.4 Propriétés des fonctions convexes

La notion de la convexité peut se généraliser de la façon suivante :

Définition 1.19. ***Combinaison convexe***

On dit qu'un vecteur X est une combinaison convexe des points $x_1, x_2, ..., x_p \in \mathbb{R}^n$ *si on a*

$$X = \sum_{i=1}^{p} \lambda_i x_i, \quad \sum_{i=1}^{p} \lambda_i = 1, \quad \lambda_i \geq 0, \ i = \overline{1, p}. \tag{1.2}$$

Propriété 1.4. ***Inégalité de Jensen***

Pour une fonction convexe f *et des points* $x_1, x_2, \ldots, x_p \in \mathbb{R}^n$ *et des poids* $\lambda_1, \lambda_2, \ldots, \lambda_p$ *tels que* $\lambda_i \geq 0$ *et* $\sum_{i=1}^{p} \lambda_i = 1$*, l'inégalité de Jensen s'énonce comme suit :*

$$f\left(\sum_{i=1}^{p} \lambda_i x_i\right) \leq \sum_{i=1}^{p} \lambda_i f(x_i). \tag{1.3}$$

Propriété 1.5. ***Convexité et différentiabilité***

— *Soit* f *une fonction réelle, définie sur un ensemble convexe* $C \subset \mathbb{R}^n$*, et de classe* C^1*. Alors* f *est convexe si seulement si :*

$$f(y) - f(x) \geq \nabla f^T(x)(y - x), \quad \forall \ y, \ x \in C. \tag{1.4}$$

— *Soit* f *une fonction réelle deux fois continûment différentiable. Alors* f *est convexe sur* C *si seulement si le hessien* $H(x) = \nabla^2 f(x) \geq 0, \ \forall x \in C$ *(i.e., le hessien de* f *est semi-défini positif).*

1.4 Formes quadratiques

Définition 1.20. *Une fonction réelle $F(x)$, définie sur $\mathbb{R}^n$ et exprimée sous la forme :*

$$F(x) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_i x_j, \tag{1.5}$$

*est appelée **forme quadratique** à n variables $x_1, x_2, \ldots, x_n$, où $x = (x_1, x_2, \ldots, x_n)^T = x(J) = (x_i,\ i \in J)$ est un vecteur de $\mathbb{R}^n$, et $J = \{1, 2, \ldots, n\}$ désigne l'ensemble des indices des variables. L'expression (1.5) peut également être écrite sous forme matricielle :*

$$F(x) = \frac{1}{2}x^T Dx, \tag{1.6}$$

où $D = (d_{ij}, 1 \leq i, j \leq n)$ peut être choisie comme une matrice symétrique d'ordre n, i.e., $D^T = D$.

1.4.1 Gradient et Hessien d'une forme quadratique

Définition 1.21. *Soit $F : \mathbb{R}^n \to \mathbb{R}$ une fonction réelle continûment différentiable, représentée sous la forme (1.6). Le gradient de la forme quadratique F au point x est défini par :*

$$g(x) = \nabla F(x) = \begin{pmatrix} \frac{\partial F}{\partial x_1} \\ \frac{\partial F}{\partial x_2} \\ \vdots \\ \frac{\partial F}{\partial x_n} \end{pmatrix} = Dx, \tag{1.7}$$

où $\frac{\partial F}{\partial x_i},\ i \in J$, est la dérivée partielle de $F(x)$ par rapport à la variable x_i.

Définition 1.22. *Soit $F : \mathbb{R}^n \to \mathbb{R}$ la forme quadratique (1.6). Le hessien de F est défini par :*

$$H(x) = \nabla^2 F(x) = \begin{pmatrix} \frac{\partial^2 F}{\partial x_1^2} & \frac{\partial^2 F}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 F}{\partial x_1 \partial x_n} \\ \frac{\partial^2 F}{\partial x_2 \partial x_1} & \frac{\partial^2 F}{\partial x_2^2} & \cdots & \frac{\partial^2 F}{\partial x_2 \partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 F}{\partial x_n \partial x_1} & \frac{\partial^2 F}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 F}{\partial x_n^2} \end{pmatrix} = D. \tag{1.8}$$

Définition 1.23. *Soit $f : \mathbb{R}^n \to \mathbb{R}$ une fonction réelle de classe C^1. La dérivée directionnelle*

de f dans la direction d au point x est :

$$\begin{aligned} f'(x,d) = \frac{\partial f(x)}{\partial d} &= \lim_{t\to 0^+} \frac{f(x+td)-f(x)}{t} \\ &= \frac{\partial f}{\partial x_1}(x+td)\mid_{t=0^+} .d_1 + \cdots + \frac{\partial f}{\partial x_n}(x+td)\mid_{t=0^+} .d_n \\ &= \nabla f^T(x)d. \end{aligned}$$

1.4.2 Formes quadratiques définies et non définies

Soit la fonction quadratique $F(x) = \frac{1}{2}x^T Dx$, où D est une matrice symétrique d'ordre n.

Définition 1.24. *La fonction quadratique $F(x)$ est dite Définie Positive (DP) si $x^T Dx > 0,\ \forall x \in \mathbb{R}^n$ et $x \neq 0$. Elle est dite Semi-Définie Positive (SDP) si $x^T Dx \geq 0,\ \forall x \in \mathbb{R}^n$ et $x \neq 0$.*

Définition 1.25. *La fonction quadratique $F(x)$ est dite Définie Négative (DN) si $x^T Dx < 0,\ \forall x \in \mathbb{R}^n$ et $x \neq 0$. Elle est dite Semi-Définie Négative (SDN) si $x^T Dx \leq 0,\ \forall x \in \mathbb{R}^n$ et $x \neq 0$.*

Définition 1.26. *Une matrice symétrique D est dite matrice Définie Positive (DP) (Semi-Définie Positive (SDP)) et on note $D \succ 0$ $(D \succeq 0)$, si elle est associée à une forme quadratique Définie Positive (DP) (Semi-Définie Positive (SDP)).*

Définition 1.27. *Une forme quadratique $F(x)$ est dite Non Définie (ND) si $F(x)$ est positive pour certaines valeurs de x et négative pour d'autres.*

1.4.3 Caractérisation des formes quadratiques définies

Pour déterminer si une matrice symétrique est définie ou semi-définie positive, le critère de Sylvester est particulièrement utile. Ce critère offre une condition nécessaire et suffisante, en se basant sur les déterminants des sous-matrices principales, également appelés mineurs principaux de la matrice.

Définition 1.28. *Considérons la matrice symétrique suivante :*

$$D = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{pmatrix}.$$

Le mineur d'ordre p de la matrice D, constitué des lignes $i_1, i_2, \ldots, i_p$ et des colonnes $j_1, j_2, \ldots, j_p$, est noté comme suit :

$$D\begin{pmatrix} i_1, i_2, \ldots, i_p \\ j_1, j_2, \ldots, j_p \end{pmatrix} = \begin{vmatrix} d_{i_1 j_1} & d_{i_1 j_2} & \cdots & d_{i_1 j_p} \\ d_{i_2 j_1} & d_{i_2 j_2} & \cdots & d_{i_2 j_p} \\ \vdots & \vdots & \ddots & \vdots \\ d_{i_p j_1} & d_{i_p j_2} & \cdots & d_{i_p j_p} \end{vmatrix}.$$

Ce mineur est dit principal si $i_1 = j_1, i_2 = j_2, \ldots, i_p = j_p$; c-à-d s'il est formé de lignes et de colonnes portant les mêmes numéros. Les mineurs suivants

$$D_1 = d_{11}, \quad D_2 = \begin{vmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{vmatrix}, \ldots, \quad D_n = \begin{vmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{vmatrix} = |D|,$$

sont appelés mineurs principaux successifs.

Alors, le critère de Sylvester se formule comme suit :

Théorème 1.2. ***(Critère de Sylvester)***

1. *Pour qu'une matrice symétrique D soit définie positive ($D \succ 0$), il est nécessaire et suffisant que tous ses mineurs principaux successifs soient positifs :*

$$D_1 > 0, \quad D_2 > 0, \ldots, \quad D_n > 0. \tag{1.9}$$

2. *Pour que la matrice D soit semi-définie positive ($D \succeq 0$), il est nécessaire et suffisant que tous ses mineurs principaux soient non négatifs :*

$$D\begin{pmatrix} i_1, i_2, \ldots, i_p \\ i_1, i_2, \ldots, i_p \end{pmatrix} \geq 0, \qquad 1 \leq i_1 < i_2 < \cdots < i_p \leq n, \quad p = 1, 2, \cdots, n. \tag{1.10}$$

Remarque 1.4. *La condition*

$$D_1 \geq 0, \ D_2 \geq 0, \ldots, D_n \geq 0,$$

n'est pas suffisante pour garantir que la matrice D soit semi-définie positive.
En effet, pour la matrice

$$D = \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{pmatrix}$$

on obtient

$$D_1 = 0, \ D_2 = 0.$$

La forme quadratique associée à D s'écrit alors :

$$F(x) = \frac{1}{2}(d_{11}x_1^2 + 2d_{12}x_1x_2 + d_{22}x_2^2) = -\frac{1}{2}x_2^2.$$

Cette forme n'est pas comme on voudrait semi-définie positive ($D \succeq 0$) (elle est semi-définie négative $D \preceq 0$). La raison est que le mineur principal $D\begin{pmatrix} 2 \\ 2 \end{pmatrix} = -1 < 0.$

Les formes quadratiques définies et semi-définies positives peuvent également être liées aux valeurs propres.

Théorème 1.3. *Soit D une matrice symétrique d'ordre n, et soient $\{\lambda_i\}_{i=1,\dots,n}$ ses valeurs propres réelles. Les équivalences suivantes sont vérifiées :*

1. $D \succeq 0 \Leftrightarrow \lambda_i \geq 0$, *pour tout* $i = 1, \dots, n$.
2. $D \succ 0 \Leftrightarrow \lambda_i > 0$, *pour tout* $i = 1, \dots, n$.
3. *D est indéfinie si et seulement si certaines valeurs λ_i sont positives et d'autres négatives.*

1.4.4 Propriétés des formes quadratiques définies et semi-définies positives

Les matrices symétriques définies (semi-définies) positives ont des propriétés très intéressantes, dont nous citons quelques unes des plus usuelles :

Propriété 1.6. *Soit la matrice D partitionnée de la manière suivante :*

$$D = \begin{array}{c} \\ m \\ k \end{array} \begin{array}{c} \begin{array}{cc} m & k \end{array} \\ \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \end{array}, \quad m + k = n.$$

Si $D \succ 0$ ($D \succeq 0$), alors les sous-matrices principales D_{11} et D_{22} sont aussi définies positives (DP) (semi-définies positives (SDP)) . D'une façon générale, toute sous-matrice principale d'une matrice DP (SDP) est aussi DP (SDP).

Proposition 1.2. *Un élément de la diagonale d'une matrice symétrique D semi-définie positive ne peut s'annuler que si tous les autres éléments de la même ligne et colonne s'annulent aussi.*

Proposition 1.3. *Soient D une matrice symétrique semi-définie positive, et $x \in \mathbb{R}^n$ un point quelconque tel que $x \neq 0$. Alors $x^T Dx = 0 \Leftrightarrow Dx = 0$.*

1.5 Les M-Matrices

La nécessité d'étudier les M-matrices est apparue suite aux nombreuses applications des matrices ayant une telle structure en analyse numérique. Il s'avère que ce type de matrices possède de nombreuses propriétés utiles qui en font un outil précieux en biomathématique, en robotique et en finance [18, 60, 73]. Par exemple, la non-négativité de l'inverse d'une M-matrice, est primordiale, car certains des modèles décrivant la vie réelle et les marchés financiers utilisent des coefficients et des variables non négatifs.

Un autre avantage des M-matrices est leur application pour résoudre des équations aux dérivées partielles paraboliques telles que celle de Black-Scholes [54]. De nouveaux schémas de différences finies ou des schémas modifiés pourraient être réalisés en utilisant des M-matrices, car la stabilité des schémas traditionnels de différences finies échoue dans les cas d'équations aux dérivées partielles avec des conditions aux limites non lisses [51]. Cette classe de M-matrices est souvent une condition suffisante pour obtenir une solution numérique qui ne présente pas d'oscillations [74]. Tous ces arguments nous encouragent à présenter une étude consacrée à la structure et aux principales propriétés des M-matrices. Beaucoup d'informations sur les M-matrices sont proposées par Windisch dans [109] et par Varga dans [102]. Une brève revue des M-matrices est présentée dans [84, 86, 105].

1.5.1 Définitions et structure des M-matrices

Définition 1.29. *Une matrice réelle, carrée D est appelée une M-matrice si elle est carrée, avec $d_{ij} \leq 0$ pour tout $i \neq j$ et tous ses mineurs principaux sont positifs.*

Théorème 1.4. *Une matrice D avec des éléments hors-diagonaux non positifs est une M-matrice si et seulement si D est non singulière et que D^{-1} est non négative (tous les éléments de D^{-1} sont positifs ou nuls).*

Définition 1.30. *Une M- matrice D peut s'écrire sous la forme $D = sI - B$, avec $s > 0$, $B \geq 0$ et $s > \rho(B)$, où $\rho(B)$ est le rayon spectral de la matrice non négative B. Pour $s = \rho(B)$, D est une M-matrice singulière.*

Définition 1.31. *Une matrice de Stieltjes est une matrice réelle, carrée D avec $d_{ij} \leq 0$ pour tout $i \neq j$, qui est symétrique et définie positive. Toutes les matrices de Stieltjes sont des M-matrices, mais l'inverse n'est pas toujours vrai.*

1.5.2 Propriétés des M-matrices

Lemme 1.1. *[102] Une matrice réelle, symétrique, dont les éléments non diagonaux sont négatifs ou nuls, est définie positive si et seulement si elle est une M-matrice.*

Propriété 1.7. *Les éléments diagonaux d'une M-matrice sont tous positifs.*

Propriété 1.8. *[109] Pour toute M-matrice non singulière D, il existe un vecteur $x > 0$ tel que $Dx \geq 0$. Notons que le vecteur x n'est pas unique.*

Théorème 1.5. *[102] Soit D une M-matrice, et C n'importe quelle matrice obtenue à partir de D en posant certaines éléments non-diagonaux égaux à zéro. Alors C est aussi une M-matrice.*

Preuve. (Voir [102]).

Lemme 1.2. *[8] Toute sous-matrice principale d'une M-matrice est une M-matrice.*

Lemme 1.3. *Soit A une M-matrice et B une matrice telle que $a_{ij} \leq b_{ij} \leq 0, \forall i \neq j$ et $b_{ii} \geq a_{ii} > 0$ pour $1 \leq i \leq n$. Alors B est une M-matrice.*

Preuve. En notant par D_A (resp. D_B) la matrice diagonale, constituée des éléments diagonaux de A (resp. B), on aura alors

$$D_A - A \geq D_B - B \quad \text{et} \quad D_A^{-1} \geq D_B^{-1} > 0.$$

D'où

$$J_A = I - D_A^{-1}A = D_A^{-1}(D_A - A) \geq D_B^{-1}(D_B - B) = I - D_B^{-1}B = J_B \geq 0$$

Comme A est une M-matrice et d'après [102] on a $\rho(J_A) < 1$ et donc aussi $\rho(J_B) < 1$, ce qui fait que B est aussi une M-matrice.

Théorème 1.6. *[102] Soit D une M-matrice et soit un sous-ensemble non vide I de $J = \{1, 2, \ldots, n\}$. La matrice*

$$V = D_{II} - D_{I\bar{I}}D_{\bar{I}\bar{I}}^{-1}D_{\bar{I}I}, \quad \bar{I} = J\backslash I, \tag{1.11}$$

est une matrice de Stieltjes.

Preuve.
Montrons que V est une matrice symétrique.
En effet, D est une matrice de Stieltjes, donc

$$D_{I\bar{I}} = D_{\bar{I}I}^T,$$

et d'après la relation (1.11), on déduit que V est symétrique.

Montrons que V est définie positive.
Posons

$$W = D^{-1}.$$

Comme D est définie positive, alors W l'est aussi. Il en est de même pour les sous-matrices W_{II} et W_{II}^{-1}. D'après [30], on a $V = W_{II}^{-1}$, par conséquent V est définie positive.

Enfin, comme $D_{\bar{I}\bar{I}}$ est une M-matrice (d'après le lemme 1.2), alors $D_{\bar{I}\bar{I}}^{-1} \geq 0$. Et comme $D_{\bar{I}I} \leq 0$, $D_{I\bar{I}} \leq 0$, les éléments non diagonaux de V sont donc non positifs puisque les éléments non diagonaux de D_{II} sont non positifs. ■

Définition 1.32. *On dit qu'une matrice réelle carrée et symétrique D d'ordre n est à diagonale dominante par lignes si :*

$$| d_{ii} | \geq \sum_{\substack{j=1 \\ j \neq i}}^{n} | d_{ij} |, \quad pour \;\; i = 1, 2, \ldots, n. \tag{1.12}$$

On dit qu'elle est une matrice à diagonale dominante par colonnes si

$$| d_{jj} | \geq \sum_{\substack{i=1 \\ i \neq j}}^{n} | d_{ij} |, \quad pour \; j = 1, 2, \ldots, n.$$

Si l'inégalité (1.12) est stricte, alors la matrice D est qualifiée de matrice à diagonale dominante stricte par lignes (ou, respectivement, par colonnes).

Remarque 1.5. *Si on est en présence d'une matrice D vérifiant les conditions (i) et (ii) suivantes :*

(i) $d_{ii} > 0$*, pour tout* $i = 1, 2, \ldots, n$*,*

(ii) $d_{ij} \leq 0$*, pour tout* $i \neq j$*,*

alors la condition de diagonale dominante (1.12) s'écrit plus simplement pour tout $i = 1, 2, \ldots, n$ *:*

$$\sum_{j=1}^{n} d_{ij} \geq 0.$$

Lemme 1.4. *Soit D une matrice symétrique d'ordre n, définie par* $D = \left(d_{ij},\ 1 \leq i, j \leq n\right)$*. Si D est strictement à diagonale dominante, alors* $\det D \neq 0$*.*

Preuve.
Soit $D = \left(d_{ij},\ 1 \leq i, j \leq n\right)$ une matrice à diagonale dominante stricte, et supposons que $\det D = 0$. Cela implique que :

$$\exists x \neq 0 : \;\; Dx = 0, \quad x \in \mathbb{R}^n.$$

Posons

$$\|x\|_\infty = \max_{i=1,n} |x_i| .$$

Alors il existe un indice k tel que

$$|x_k| = \|x\|_\infty > 0.$$

Pour la $k^{\text{ième}}$ ligne du système $Dx = 0$, on a

$$d_{kk}x_k = -\sum_{j\neq k} d_{kj}x_j.$$

En prenant les valeurs absolues des deux côtés et en utilisant le fait que $\left|x_j\right| \leq |x_k|$, on peut alors écrire :

$$|d_{kk}|\,|x_k| = \left|\sum_{j\neq k} d_{kj}x_j\right| \leq \sum_{j\neq k} \left|d_{kj}\right| \left|x_j\right| \leq |x_k| \left(\sum_{j\neq k} \left|d_{kj}\right|\right).$$

Donc en simplifiant on aura

$$|d_{kk}| \leq \sum_{j\neq k} \left|d_{kj}\right|,$$

ce qui contredit l'hypothèse que D est une matrice à diagonale dominante stricte. Par conséquent, $\det D \neq 0$. ■

La relation entre les M-matrices et les matrices à diagonale dominante est donnée par la propriété suivante :

Proposition 1.4. *Une matrice carrée D qui est à diagonale dominante stricte par lignes et dont les termes satisfont les inégalités $d_{ij} \leq 0$ pour $i \neq j$ et $d_{ii} > 0$, est une M-matrice.*

Preuve.

Considérons une matrice carrée $D = (d_{ij})$ qui satisfait les conditions :

— $d_{ij} \leq 0$ pour $i \neq j$ (les termes hors diagonale sont négatifs ou nuls)

— $d_{ii} > 0$ pour tout i (les termes diagonaux sont strictement positifs)

— D est à diagonale dominante stricte, c'est-à-dire :

$$d_{ii} > \sum_{j\neq i} |d_{ij}| \quad pour \quad j = 1, 2, \cdots, n.$$

Montrons que D est une M-matrice. Par définition, une matrice D est une M-matrice si elle peut s'écrire sous la forme :

$$D = sI - B$$

avec $s > 0$, I la matrice identité, et B une matrice à coefficients non négatifs telle que $s \geq \rho(B)$, où $\rho(B)$ est le rayon spectral de B.

Posons :

$$B = \alpha I - D$$

avec :

$$\alpha = \max_i \left(\sum_{j \neq i} |d_{ij}| \right)$$

Ainsi, $\alpha > 0$.

Les éléments de B sont :

— $b_{ii} = \alpha - d_{ii}$, et grâce à la diagonale dominante stricte, on a $\alpha - d_{ii} \geq 0$.

— $b_{ij} = -d_{ij} \geq 0$ pour $i \neq j$, puisque $d_{ij} \leq 0$.

Donc B est une matrice à coefficients non négatifs.

Enfin, le rayon spectral de B vérifie :

$$\rho(B) \leq \max_i \left(\sum_{j \neq i} |d_{ij}| \right) < d_{ii}$$

En posant $s = \alpha$, on obtient :

$$D = sI - B, \text{ avec } s > \rho(B)$$

Cela prouve que D est une M-matrice. ■

Théorème 1.7. *Une matrice D symétrique et à diagonale dominante stricte est définie positive.*

Preuve.

Soit $x \in \mathbb{R}^n$, nous cherchons à montrer que :

$$x^T Dx > 0 \quad \text{pour tout } x \neq 0.$$

Par définition, nous avons :

$$x^T Dx = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_i x_j.$$

Nous séparons les termes diagonaux et hors diagonaux :

$$x^T Dx = \sum_{i=1}^{n} d_{ii} x_i^2 + 2 \sum_{1 \leq i < j \leq n} d_{ij} x_i x_j.$$

Nous voulons prouver que cette somme est strictement positive. En utilisant l'inégalité classique de Cauchy-Schwarz :

$$|\langle x, y \rangle|^2 \leq \|x\|^2 \|y\|^2,$$

avec $x = (x_i, x_j)$ et $y = (1, 1)$, nous avons :

$$|x_i + x_j|^2 \leq 2(x_i^2 + x_j^2).$$

En développant le carré :

$$(x_i + x_j)^2 = x_i^2 + 2x_i x_j + x_j^2,$$

on déduit que :

$$2|x_i x_j| \leq x_i^2 + x_j^2.$$

En encadrant les termes hors diagonaux :

$$-\sum_{i \neq j} d_{ij} x_i x_j \leq \sum_{i<j} |d_{ij}|(x_i^2 + x_j^2),$$

et en réorganisant les termes, on obtient :

$$-\sum_{i \neq j} d_{ij} x_i x_j \leq \sum_i \left(\sum_{j \neq i} |d_{ij}| \right) x_i^2. \tag{1.13}$$

Or, par hypothèse, D est à diagonale dominante stricte, ce qui signifie que :

$$d_{ii} > \sum_{j \neq i} |d_{ij}|.$$

Ainsi, pour tout i, on a :

$$d_{ii} - \sum_{j \neq i} |d_{ij}| > 0.$$

En vertu de l'inégalité (1.13), on déduit que :

$$x^T D x = \sum_i d_{ii} x_i^2 + \sum_{i \neq j} d_{ij} x_i x_j > \sum_i \left(d_{ii} - \sum_{j \neq i} |d_{ij}| \right) x_i^2 > 0, \quad \textit{tant que} \quad x \neq 0.$$

Par conséquent, $x^T D x > 0$ pour tout $x \neq 0$, ce qui prouve que D est définie positive. ■

Théorème 1.8. *Soit $D = \left(d_{ij},\ 1 \leq i, j \leq n\right)$, une matrice réelle symétrique et soit*

$$\Lambda_i = \sum_{j=1, j \neq i}^{n} \left| d_{ij} \right|, \quad 1 \leq i \leq n.$$

Alors toutes les valeurs propres λ de D sont contenues dans la réunion des disques :

$$|z - d_{ii}| \leq \Lambda_i.$$

Preuve.

Soit λ une valeur propre de la matrice D et soit x un vecteur propre correspondant à λ. On

normalise le vecteur propre x tel que sa plus grande composante en module soit égale à 1. Par définition, on a

$$(\lambda - d_{ii})\, x_i = \sum_{j=1, j\neq i}^{n} d_{ij} x_j, \quad i = \overline{1, n}.$$

En particulier, si $|x_r| = 1$, on obtient

$$|\lambda - d_{rr}| \leq \sum_{j=1, j\neq r}^{n} \left|d_{rj}\right| \cdot \left|x_j\right| \leq \sum_{j=1, j\neq r}^{n} \left|d_{rj}\right| = \Lambda_r.$$

Ainsi, la valeur propre λ est comprise dans la réunion des disques

$$|z - d_{ii}| \leq \Lambda_i, \quad 1 \leq i \leq n.$$

Comme λ est une valeur propre arbitraire de D, il s'ensuit que toutes les valeurs propres de D sont comprises dans la réunion de ces disques. ■

Théorème 1.9. *Toute valeur propre d'une matrice D symétrique à éléments réels est réelle.*

Preuve.
Soit λ une valeur propre de la matrice D et x un vecteur propre correspondant :

$$Dx = \lambda x, \quad x \neq 0. \tag{1.14}$$

Comme $D^T = D$, on a

$$\langle Dx, x\rangle = \langle x, Dx\rangle,$$

et, en vertu de l'égalité (1.14), on obtient

$$\langle \lambda x, x\rangle = \langle x, \lambda x\rangle.$$

D'où

$$\lambda\langle x, x\rangle = \bar{\lambda}\langle x, x\rangle,$$

où $\bar{\lambda}$ est le conjugué du nombre complexe λ.
Un vecteur propre est non nul par définition ; donc $\langle x, x\rangle = \| x \|^2 \neq 0$. D'où $\lambda = \bar{\lambda}$, c'est-à-dire λ est un nombre réel. ■

Théorème 1.10. *Soit $D = \left(d_{ij},\ 1 \leq i, j \leq n\right)$ une matrice réelle symétrique. Alors toute valeur propre λ de D se trouve dans l'intervalle :*

$$\min_i \left(d_{ii} - \sum_{j=1, j\neq i}^{n} \left| d_{ij} \right| \right) \leq \lambda \leq \max_i \left(d_{ii} + \sum_{j=1, j\neq i}^{n} \left| d_{ij} \right| \right).$$

Preuve.
D'après le théorème 1.8, toutes les valeurs propres de D sont contenues dans la réunion des disques :

$$|z - d_{ii}| \leq \sum_{j\neq i} \left| d_{ij} \right|, \quad i = \overline{1, n}.$$

Soit λ une valeur propre quelconque de D. Il existe alors un disque C_r tel que $\lambda \in C_r$. Comme λ est une valeur réelle, on a alors

$$\lambda \in C_r \Leftrightarrow |\lambda - d_{rr}| \leq \sum_{j\neq r} \left| d_{rj} \right| \Leftrightarrow d_{rr} - \sum_{j\neq r} \left| d_{rj} \right| \leq \lambda \leq d_{rr} + \sum_{j\neq r} \left| d_{rj} \right|$$

Donc

$$\min_i \left(d_{ii} - \sum_{j\neq i} \left| d_{ij} \right| \right) \leq \lambda \leq \max_i \left(d_{ii} + \sum_{j\neq i} \left| d_{ij} \right| \right). \quad \blacksquare$$

Ainsi, pour toute matrice carrée symétrique D d'ordre n, il existe une base orthonormée de vecteurs propres de D telle que D possède une forme matricielle diagonale dans cette base :

$$\exists P \text{ telle que} \quad P^{-1}DP = P^T DP = \operatorname{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n),$$

où la matrice P, formée des vecteurs de cette base, est appelée matrice orthogonale, avec la propriété $P^{-1} = P^T$.

Théorème 1.11. *Soit D une matrice symétrique réelle et*

$$\alpha = \min \lambda_i, \quad i = \overline{1, n},$$
$$\beta = \max \lambda_i, \quad i = \overline{1, n},$$

où les scalaires $\lambda_i, i = \overline{1, n}$, sont les valeurs propres de D. Alors l'inégalité

$$\alpha \|x\|^2 \leq x^T Dx \leq \beta \|x\|^2 \tag{1.15}$$

est vérifiée pour tout vecteur x.

Preuve. La matrice D possède un système de vecteurs propres $e_1, e_2, \cdots, e_n$ tels que

$$De_j = \lambda_j e_j \quad j = 1, 2, \cdots, n.$$

Ces vecteurs forment une base orthonormée de l'espace $\mathbb{R}^n$. Alors tout vecteur x peut être mis sous la forme

$$x = x_1 e_1 + x_2 e_2 + \ldots + x_n e_n,$$

$x_1, x_2, \cdots, x_n$ étant les coordonnées du vecteur x par rapport à cette base. Par suite, on a

$$Dx = x_1 De_1 + x_2 De_2 + \ldots + x_n De_n = \lambda_1 x_1 e_1 + \lambda_2 x_2 e_2 + \ldots + \lambda_n x_n e_n.$$

En tenant compte du fait que les vecteurs de la base sont orthogonaux, on aura :

$$x^T Dx = \sum_{j=1}^{n} \sum_{k=1}^{n} \lambda_j x_j x_k \left\langle e_j, e_k \right\rangle = \sum_{j=1}^{n} \sum_{k=1}^{n} \lambda_j x_j x_k \delta_{jk} = \sum_{j=1}^{n} \lambda_j x_j^2,$$

c'est-à-dire

$$x^T Dx = \sum_{j=1}^{n} \lambda_j x_j^2. \tag{1.16}$$

En remplaçant dans l'égalité (1.16) λ_j par la plus petite valeur α, on obtient :

$$x^T Dx \geq \alpha \sum_{j=1}^{n} x_j^2 = \alpha \left\| x^2 \right\|.$$

De façon analogue, en substituant à λ_j dans l'égalité (1.16) la valeur maximale β, on obtient :

$$x^T Dx \leq \beta \sum_{j=1}^{n} x_j^2 = \beta \left\| x^2 \right\|.$$

Ainsi, l'inégalité (1.15) est démontrée. ■

Corollaire 1.1. *Les valeurs propres minimale α et maximale β d'une matrice symétrique réelle D sont respectivement les valeurs minimale et maximale de la forme quadratique $x^T Dx$ sur la sphère $\| x \| = 1$.*

Preuve. Soit D une matrice symétrique réelle. Par le théorème spectral, D admet une base orthonormale de vecteurs propres $\{v_1, \ldots, v_n\}$ associés aux valeurs propres $\lambda_1 \leq \cdots \leq \lambda_n$.

Tout vecteur $x \in \mathbb{R}^n$ tel que $\|x\| = 1$ peut s'écrire comme une combinaison linéaire de ces vecteurs propres :

$$x = \sum_{i=1}^{n} a_i v_i, \quad \text{où} \ \sum_{i=1}^{n} a_i^2 = \|x\|^2 = 1.$$

La forme quadratique devient :

$$x^T Dx = \left(\sum_{i=1}^{n} a_i v_i \right)^T D \left(\sum_{j=1}^{n} a_j v_j \right).$$

En utilisant le fait que $Dv_i = \lambda_i v_i$, on obtient :

$$x^T Dx = \sum_{i=1}^{n} \lambda_i a_i^2 \implies \alpha \sum_{i=1}^{n} a_i^2 \leq \sum_{i=1}^{n} \lambda_i a_i^2 \leq \beta \sum_{i=1}^{n} a_i^2.$$

Puisque les a_i^2 sont des coefficients positifs ou nuls vérifiant $\sum_{i=1}^{n} a_i^2 = 1$, alors on déduit

$$\alpha \leq x^T Dx \leq \beta$$

Donc

$$\alpha = \min_{\|x\|=1} x^T Dx \ \ avec \ \ x^T Dx = x^T(\lambda_1 x) = \lambda_1 \|x\|^2 = \lambda_1,$$

$$\beta = \max_{\|x\|=1} x^T Dx = \ \ avec \ \ x^T Dx = x^T(\lambda_n x) = \lambda_n \|x\|^2 = \lambda_n.$$

D'où le résultat. ■

Théorème 1.12. *Une matrice symétrique réelle D est définie positive si et seulement si toutes ses valeurs propres est strictement positives.*

Preuve. Si D est une matrice symétrique réelle et ses valeurs propres λ_j sont telles que $\lambda_j > 0, \quad j = 1, \cdots, n$, alors :

$$x^T Dx = \sum_{j=1}^{n} \lambda_j x_j^2,$$

où le vecteur $x = (x_1, x_2, \cdots, x_n)^T$. D'où, pour $x \neq 0$:

$$x^T Dx > 0,$$

ce qui prouve que D est définie positive.

Inversement, soit D une matrice symétrique réelle définie positive. En vertu du corollaire 1.1, toutes ses valeurs propres $\lambda_1, \lambda_2 \cdots, \lambda_n$ sont réelles et :

$$\alpha = \min_{\|x\|=1} x^T Dx = \lambda_1 > 0.$$

La valeur α étant positive, on a donc :

$$\alpha > 0 \implies \lambda_j > 0 \quad \text{pour} \quad j = 1, 2, \cdots, n. \quad \blacksquare$$

Proposition 1.5. *Une matrice symétrique à diagonale dominante stricte, avec des éléments diagonaux positifs est définie positive.*

Preuve. D'après le théorème 1.10, pour toute valeur propre λ, on a :

$$\lambda \geq \min_i \left(d_{ii} - \sum_{j \neq i} \left| d_{ij} \right| \right).$$

Comme D est une matrice à diagonale dominante stricte et que $d_{ii} > 0$, il en résulte que $\lambda > 0$. Et d'après le théorème 1.12, il en résulte que D est définie positive. ■

1.6 Optimisation convexe

La convexité est un concept clé dans divers domaines de la programmation mathématique. Par exemple, lorsque la fonction est convexe, les conditions nécessaires d'optimalité deviennent également suffisantes pour un problème d'un minimum. De plus, un minimum local est toujours un minimum global, ce qui permet d'appliquer tous les résultats d'optimisation locale à l'optimisation globale. Ci-dessous, nous présentons quelques définitions et résultats associés à la minimisation convexe.

Définition 1.33. *Un problème de programmation mathématique est qualifié de convexe (respectivement strictement convexe) lorsqu'il consiste à minimiser une fonction convexe (respectivement strictement convexe) sur un ensemble convexe.*

L'hypothèse de convexité joue un rôle essentiel en optimisation. Il est important de souligner que cette hypothèse est fondamentale. Notons que :

- Les problèmes convexes sont synonymes de minimisation, et les problèmes concaves sont synonymes de maximisation ;
- Les problèmes convexes sont beaucoup plus faciles à résoudre, ceux pour lesquels il existe des algorithmes de résolution efficaces ;
- L'hypothèse de convexité ne suffit pas à assurer l'existence ou l'unicité d'une éventuelle solution.

Pour tout problème de minimisation convexe, nous avons les propriétés suivantes :

Propriété 1.9. *Soit f une fonction convexe définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors l'ensemble des points où f atteint son minimum sur C est convexe.*

Propriété 1.10. *Tout minimum local dans C est un minimum global.*

Propriété 1.11. *Si la fonction f est strictement convexe, alors son minimum global sur C est unique.*

Plus particulièrement, considérons un problème de minimisation convexe sans contraintes, donné sous la forme suivante :

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1.17}$$

où f est une fonction réelle convexe de classe C^1 sur $\mathbb{R}^n$.

Théorème 1.13. *Soit f une fonction convexe et continûment différentiable sur $\mathbb{R}^n$. Les affirmations suivantes sont équivalentes :*

(i) *x^0 est un minimiseur global de f sur $\mathbb{R}^n$;*

(ii) *x^0 est un minimiseur local de f ;*

(iii) *x^0 est un point stationnaire de f, c'est-à-dire que $\nabla f(x^0) = 0$.*

1.7 Programmation non linéaire

On parle d'un problème d'optimisation non linéaire lorsque la fonction objectif à minimiser est non linéaire et/ou lorsque l'ensemble S des contraintes est non linéaire. Un tel modèle peut être formulé sans contraintes ($S = \mathbb{R}^n$) ou avec des contraintes linéaires ou non, sous forme d'équations ou d'inéquations. Dans cette section, nous présentons d'abord un problème de minimisation non linéaire sans contraintes, puis avec contraintes.

Un problème d'optimisation non linéaire sans contraintes consiste à minimiser une fonction non linéaire f sans imposer de restrictions sur le vecteur x. Il se formule généralement comme suit :

$$\min_{x \in \mathbb{R}^n} f(x). \tag{1.18}$$

Ce type d'optimisation, également appelé optimisation non restreinte, vise à minimiser une fonction sans aucune contrainte imposée sur les variables de décision. De tels problèmes apparaissent fréquemment dans diverses applications, notamment en apprentissage automatique, en économie et en physique, où l'on cherche à optimiser des modèles complexes sans restriction explicite.

— **Ingénierie** : Minimisation de coûts ou optimisation des performances sans limitations explicites, comme la réduction de la consommation d'énergie d'un moteur.

— **Apprentissage automatique** : Ajustement des paramètres de modèles (comme les réseaux de neurones) en minimisant une fonction de perte, sans contraintes sur les variables.

— **Finance** : Optimisation de portefeuilles pour maximiser les rendements ou minimiser les risques sans restrictions sur les actifs.

— **Traitement d'images** : Affiner les modèles de détection et de classification en minimisant les erreurs, sans contraintes sur les paramètres.

Nous nous concentrons sur ces problèmes sans contraintes, car les conditions d'optimalité des problèmes avec contraintes sont une extension logique de celles sans contraintes [7]. D'ailleurs, une stratégie courante pour résoudre un problème avec contraintes est de traiter une série de problèmes sans contraintes, avec des fonctions de pénalité.

Nous rappelons ci-dessous quelques définitions d'un minimum local et global pour un problème sans contraintes, ainsi que les conditions nécessaires et suffisantes d'optimalité locale.

Définition 1.34.

1. *Un vecteur $x^* \in \mathbb{R}^n$ est dit solution optimale du problème (1.18) si :*

$$f(x^*) \leq f(x), \quad \forall x \in \mathbb{R}^n.$$

On note alors :

$$f(x^*) = \min_{x \in \mathbb{R}^n} f(x).$$

*Le vecteur x^* est également appelé **minimum global** de f sur $\mathbb{R}^n$.*

2. *Un vecteur $x^0 \in \mathbb{R}^n$ est dit **minimum local** du problème (1.18) s'il existe un réel $r > 0$ tel que :*

$$f(x^0) \leq f(x), \quad \forall x \in B(x^0, r),$$

où la boule $B(x^0, r)$ de centre x^0 et de rayon r est définie par :

$$B(x^0, r) = \{x \in \mathbb{R}^n : \ \|x - x^0\| \leq r\}.$$

3. *Un vecteur $x^0 \in \mathbb{R}^n$ est appelé minimum local strict du problème (1.18) s'il existe un réel $r > 0$, tel que :*

$$f(x^0) < f(x), \quad \forall x \in B(x^0, r), \ x \neq x^0.$$

Définition 1.35. *Soit $f : \mathbb{R}^n \to \mathbb{R}$ une fonction de classe C^1. La dérivée directionnelle de f au point x dans la direction d est :*

$$f'(x, d) = \frac{\partial f}{\partial d}(x) = \lim_{h \to 0^+} \frac{f(x + hd) - f(x)}{h}$$

$$= \frac{\partial f}{\partial x_1}(x + hd) \mid_{h=0^+} .d_1 + ... + \frac{\partial f}{\partial x_n}(x + hd) \mid_{h=0^+} .d_n.$$

$$= \nabla f^T(x) d.$$

Si $\|d\| = 1$, la dérivée directionnelle exprime le taux de variation de f en x selon la direction d. Il convient de noter que ce taux est maximal dans la direction du gradient de f.

1.7.1 Conditions d'optimalité pour un problème non linéaire sans contraintes

Soit le problème non linéaire et sans contraintes (1.18) donné ci-dessus, où la fonction f est supposée au moins deux fois continûment différentiable.

Théorème 1.14. *[7] Si x^* est un minimum local (ou global) pour le problème (1.18), alors*

$$\nabla f(x^*) = 0. \tag{1.19}$$

Remarque 1.6. *Un point vérifiant la condition* (1.19) *est appelé un point stationnaire.*

La condition (1.19) utilise le vecteur gradient dont les composantes sont les premières dérivées partielles de f. Par conséquent, elle s'appelle la condition nécessaire du premier ordre. Les conditions nécessaires peuvent également être énoncées en termes de la matrice hessienne H, dont les coefficients sont les deuxièmes dérivées partielles de f ; elles sont alors appelées les conditions nécessaires de second ordre, présentées comme suit :

Conditions nécessaires d'optimalité de second ordre

Théorème 1.15. *[7] Soit la fonction $f : \mathbb{R}^n \to \mathbb{R}$, où $f \in C^2$. Si x^0 est un minimum local (ou global) pour le problème (1.18), alors*

(i) *$\nabla f(x^0) = 0$ (stationnarité),*

(ii) *$H(x^0)$ est semi-définie positive,*

où $H(x^0) = \nabla^2 f(x^0)$ est la matrice hessienne de f au point x^0.

Conditions suffisantes d'optimalité du second ordre

Les conditions nécessaires énoncées ci-dessus sont insuffisantes pour déterminer si un point est un minimum local (ou global). Par conséquent, un point qui satisfait ces conditions n'est pas forcément un minimum local (ou global). Le théorème qui suit donne une condition suffisante pour un minimum local strict.

Théorème 1.16. *[7] Soit la fonction $f : \mathbb{R}^n \to \mathbb{R}$, où f est supposée deux fois différentiable au point x^0. Si les conditions suivantes*

(i) *$\nabla f(x^0) = 0$ (stationnarité),*

(ii) *$H(x^0)$ est définie positive,*

sont vérifiées, alors x^0 est un minimum local strict pour le problème (1.18).

1.7.2 Conditions d'optimalité pour un problème non linéaire avec contraintes

Un problème d'optimisation non linéaire avec contraintes générales se formule de la manière suivante :

$$\min_{x \in \mathcal{S}} f(x), \tag{1.20}$$

où $\mathcal{S} = \{x \in \mathbb{R}^n : \ g_i(x) = 0, \ i = 1, 2, \ldots, k, \ g_i(x) \leq 0, \ i = k+1, \ldots, m\}$ désigne l'ensemble des solutions réalisables. On suppose que les fonctions f et g_i $(i = 1, \ldots, m)$ sont de classe C^1, i.e, continûment différentiables sur $\mathbb{R}^n$.

Définition 1.36.

1. *Un vecteur $x \in \mathbb{R}^n$ est appelé solution réalisable du problème* (1.20) *s'il vérifie toutes les contraintes du problème, c'est-à-dire, que $x \in \mathcal{S}$;*
2. *Une solution réalisable x^* est appelée solution optimale du problème* (1.20) *si*

$$f(x^*) \leq f(x), \quad \forall x \in \mathcal{S},$$

et on note $f(x^*) = \min\limits_{x \in \mathcal{S}} f(x)$;

3. *Un vecteur $x^0 \in \mathcal{S}$ est appelé minimum local du problème* (1.20) *s'il existe un réel $r > 0$, tel que :*

$$f(x^0) \leq f(x), \quad \forall x \in \mathcal{S} \cap B(x^0, r),$$

où $B(x^0, r) = \{x \in \mathbb{R}^n : \ \|x - x^0\| \leq r\}$ est la boule de centre x^0 et de rayon r.

Définition 1.37. *Un vecteur $d \in \mathbb{R}^n$, $d \neq 0$, est appelé direction admissible en un point $x \in \mathcal{S}$ s'il existe un réel $\alpha > 0$ tel que $x + \theta d \in \mathcal{S}$, $\forall \theta \in [0, \ \alpha]$. Si x est un point intérieur, alors toutes les directions sont admissibles.*

Théorème 1.17. *[7] Soit la fonction $f : \mathbb{R}^n \to \mathbb{R}$ de classe C^1. Si x^0 est un point de minimum local (ou global) du problème* (1.20)*, alors pour toute direction admissible $d \in \mathbb{R}^n$ en x^0, on a*

$$d^T \nabla f(x^0) \geq 0. \tag{1.21}$$

1.7.2.1 Conditions d'optimalité pour le cas des contraintes linéaires de type égalités

Le problème se formule sous la forme suivante :

$$\begin{cases} \min f(x), \\ s.c \ \ Ax = b, \end{cases} \tag{1.22}$$

où $x \in \mathcal{S} = \{x \in \mathbb{R}^n : \ g_i(x) = A_i^T x - b_i = 0, \ i \in I = \{1, 2, \ldots, m\}\}$, $f : \mathbb{R}^n \to \mathbb{R}$ est une fonction continûment différentiable, $A = A(I, J) = (a_{ij}, \ i \in I, j \in J)$ est une matrice d'ordre $m \times n$, formée des vecteurs-colonnes et vecteurs-lignes suivants :

$$A = (a_1, a_2, \ldots, a_j, \ldots, a_n), \ \ A = \begin{pmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_i^T \\ \vdots \\ A_m^T \end{pmatrix},$$

où $a_j = (a_{1j}, ..., a_{ij}, \ldots, a_{mj})^T$ est la $j^{ème}$ colonne de A, $j \in J = \{1, 2, \ldots, n\}$; $A_i^T = (a_{i1}, ..., a_{ij}, \ldots, a_{in})$ est la $i^{ème}$ ligne de la matrice A. Le vecteur $b = (b_1, ..., b_i, ..., b_m)^T$ est un vecteur-colonne de $\mathbb{R}^m$.

Proposition 1.6. *Un vecteur $d \in \mathbb{R}^n$ est une direction admissible au point $x \in \mathcal{S}$ si et seulement si*

$$Ad = 0. \tag{1.23}$$

De plus, on a

$$x(\theta) = x + \theta d \in \mathcal{S}, \quad \forall \theta \in \mathbb{R}.$$

Remarque 1.7. *Pour que l'ensemble des solutions réalisables $\mathcal{S}$ ne soit pas vide ou ne soit pas réduit à un point isolé, on considère que $rangA = m < n$.*

Définition 1.38. *La fonction $L(x, \lambda) = f(x) + \lambda^T(Ax - b) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x)$ est appelée fonction de **Lagrange** associée au problème* (1.22)*, où $\lambda = (\lambda_1, ..., \lambda_i, ..., \lambda_m)^T \in \mathbb{R}^m$ est le vecteur des multiplicateurs de Lagrange.*

Condition nécessaire d'optimalité du premier ordre

Théorème 1.18. *Soit x^0 un minimum local (ou global) du problème* (1.22)*. Alors il existe nécessairement un vecteur $\lambda \in \mathbb{R}^m$ vérifiant*

$$\nabla f(x^0) + A^T \lambda = 0. \tag{1.24}$$

La condition (1.24) peut être donnée autrement, en utilisant la fonction de Lagrange :

$$\nabla_x L(x, \lambda) = \nabla f(x) + A^T \lambda = 0. \tag{1.25}$$

De plus, un minimum local est tout d'abord un point réalisable, qui vérifie

$$Ax = b \Rightarrow \nabla_\lambda L(x, \lambda) = Ax - b = 0. \tag{1.26}$$

En combinant les relations (1.25) et (1.26), on obtient alors la condition nécessaire d'optimalité de premier ordre pour le problème (1.22). On a alors le théorème suivant :

Théorème 1.19. *(Théorème de Lagrange)*
Soit f une fonction continûment différentiable sur $\mathbb{R}^n$. Si x^0 est un minimum local (ou global) du problème (1.22)*, alors il existe un vecteur multiplicateur de Lagrange $\lambda^0 \in \mathbb{R}^m$, tel que :*

$$\nabla L(x^0, \lambda^0) = 0 \Leftrightarrow \begin{cases} \nabla_x L(x^0, \lambda^0) = 0, \\ \nabla_\lambda L(x^0, \lambda^0) = 0. \end{cases} \tag{1.27}$$

Le couple (x^0, λ^0) est appelé point stationnaire de la fonction de Lagrange.

Remarque 1.8. *Le vecteur $\lambda^0 = (\lambda_1^0, \ldots, \lambda_i^0, \ldots, \lambda_m^0)^T \in \mathbb{R}^m$, constitué des multiplicateurs de Lagrange λ_i^0, est unique grâce à la **remarque 1.7**.*

Cette dernière garantit en outre que la condition de qualification des contraintes est satisfaite, à savoir que les vecteurs gradients $\nabla g_i(x^0) = A_i$, pour $i = 1, \ldots, m$, sont linéairement indépendants.

Condition nécessaire d'optimalité du second ordre

La condition nécessaire de second ordre est donnée dans le théorème suivant :

Théorème 1.20. *Soit x^0 est un minimum local (ou global) du problème* (1.22)*, où f est de classe C^2 et λ^0 un vecteur multiplicateur de Lagrange, tel que le couple (x^0, λ^0) vérifie la condition* (1.27)*. On a alors*

$$d^T \frac{\partial^2 L}{\partial x^2}(x^0, \lambda^0) d \geq 0, \quad \forall d \in V_0, \ d \neq 0, \tag{1.28}$$

où $V_0 = \{d \in \mathbb{R}^n : \ Ad = 0\}$ et $\frac{\partial^2 L}{\partial x^2}(x^0, \lambda^0) = \frac{\partial^2 f}{\partial x^2}(x^0) = \nabla^2 f(x^0)$.

Condition suffisante d'optimalité du second ordre

La condition suffisante de second ordre est donnée dans le théorème suivant :

Théorème 1.21. *Soit f une fonction deux fois continûment différentiable sur $\mathbb{R}^n$ et soit (x^0, λ^0) un couple de vecteurs vérifiant la condition nécessaire d'optimalité de premier ordre du problème* (1.22)*, c-à-d :*

$$\nabla L(x^0, \lambda^0) = 0.$$

Pour que x^0 soit un minimum local du problème (1.22)*, il est suffisant que la matrice $\nabla^2 f(x^0)$ soit définie positive sur le sous-espace vectoriel $V_0 = \{d \in \mathbb{R}^n : \ Ad = 0\}$.*

Cas d'une fonction convexe

Considérons maintenant le problème de minimisation (1.22) donné ci-dessus avec contraintes linéaires de type égalités, où f est supposée convexe de classe C^1.

Théorème 1.22. *Considérons $(x^\star, \lambda^\star)$ un couple de vecteurs vérifiant les conditions d'optimalité de KKT du premier ordre :*

$$\nabla L_x(x^\star, \lambda^\star) = \nabla f(x^\star) + A^T \lambda^\star = 0, \quad \nabla L_\lambda(x^\star, \lambda^\star) = Ax^\star - b = 0.$$

Alors le vecteur $x^\star$ est un minimum global du problème (1.22)*.*

Preuve. Soit $(x^\star, \lambda^\star)$ un couple de vecteurs vérifiant les conditions d'optimalité de KKT du premier ordre et soit x une solution réalisable quelconque du problème (1.22). Comme f est convexe, alors on a

$$f(x) - f(x^\star) \geq \nabla f^T(x^\star)(x - x^\star), \ \forall x \in \mathcal{S},$$

i.e.,

$$f(x) - f(x^\star) \geq [-A^T\lambda^\star]^T(x - x^\star),\ \forall x \in \mathcal{S},$$

d'où

$$f(x) - f(x^\star) \geq -\sum_{i=1}^{m} \lambda_i^\star A_i^T(x - x^\star),\ \forall x \in \mathcal{S},$$

et donc

$$f(x) - f(x^\star) \geq -\sum_{i=1}^{m} \lambda_i^\star(b_i - b_i) = 0,\ \forall x \in \mathcal{S}.$$

Par conséquent, $x^\star$ est un minimum global du problème (1.22). ■

Les conditions d'optimalité de KKT sont donc à la fois nécessaires et suffisantes dans le cas où f est convexe.

1.7.2.2 Conditions d'optimalité pour le cas des contraintes linéaires de type inégalités

Considérons maintenant un problème de minimisation non linéaire avec contraintes linéaires de type inégalités :

$$\begin{cases} \min f(x), \\ s.c\ \ Ax \leq b, \end{cases} \tag{1.29}$$

où $x \in \mathcal{S} = \{x \in \mathbb{R}^n :\ g_i(x) = A_i^T x - b_i \leq 0,\ i \in I = \{1, 2, \ldots, m\}\}$, $f : \mathbb{R}^n \to \mathbb{R}$ est une fonction continûment différentiable, A est une matrice d'ordre $m \times n$, formée des vecteurs colonnes et lignes définis précédemment et b est un vecteur colonne de $\mathbb{R}^m$.

Définition 1.39. *Soit x une solution réalisable du problème* (1.29)*. L'ensemble des contraintes actives (saturées) au point x est l'ensemble d'indices suivants :*

$$I_a = I_a(x) = \{i \in I :\ \ g_i(x) = A_i^T x - b_i = 0\}.$$

Lemme 1.5. *Pour les contraintes d'inégalités du problème* (1.29)*, un vecteur $d \in \mathbb{R}^n$ est une direction admissible au point x si et seulement si*

$$A_i^T d \leq 0,\ \ \forall i \in I_a(x). \tag{1.30}$$

Lemme 1.6. *(Lemme de **Farkas** [7])*
Soit $(m + 1)$ vecteurs de $\mathbb{R}^n :\ c,\ A_i,\ i = 1, 2, \ldots, m$, avec $m < n$.
Si l'on a l'implication suivante

$$A_i^T x \leq 0,\ i = 1, \ldots, m,\ \Rightarrow\ c^T x \leq 0, \tag{1.31}$$

alors il existe des multiplicateurs $\lambda_i \geq 0,\ i = 1, \ldots, m$, tels que $c = \sum_{i=1}^{m} \lambda_i A_i$.

Condition nécessaire d'optimalité du premier ordre

Le lemme de **Farkas** est souvent utilisé pour démontrer le théorème suivant :

Théorème 1.23. *(Théorème de Karush-Kuhn-Tucker [1])*
Soit x^0 un minimum local (ou global) du problème (1.29)*. Alors il existe un vecteur $\lambda^0 = (\lambda_1^0, ..., \lambda_i^0, ..., \lambda_m^0)^T \geq 0$ tel que :*

(i) *Pour la fonction de Lagrange $L(x^0, \lambda^0) = f(x^0) + \sum_{i=1}^{m} \lambda_i^0 (A_i^T x^0 - b_i)$, la condition de stationnarité suivante est vérifiée :*

$$\nabla L_x(x^0, \lambda^0) = \nabla f(x^0) + \sum_{i=1}^{m} \lambda_i^0 A_i = 0; \tag{1.32}$$

(ii) *La condition de complémentarité (écarts complémentaires) suivante est satisfaite :*

$$\lambda_i^0 (A_i^T x^0 - b_i) = 0, \quad \forall i \in I. \tag{1.33}$$

Condition nécessaire d'optimalité du second ordre

Théorème 1.24. *[7] Soit f une fonction continûment différentiable sur $\mathbb{R}^n$ et soit x^0 un minimum local (ou global) du problème* (1.29)*. Alors il existe un vecteur multiplicateur de Lagrange $\lambda^0 \geq 0$ tel que le couple de vecteurs (x^0, λ^0) vérifie les conditions* (1.32) *et* (1.33) *et de plus on a :*

$$d^T \nabla^2 f(x^0) d \geq 0, \quad \forall d \in H_0, \ d \neq 0, \tag{1.34}$$

où $H_0 = \{d \in \mathbb{R}^n : A_i^T d = 0, \ \forall i \in I_a(x^0)\}$, c'est à dire que la matrice $\nabla^2 f(x^0)$ est semi-définie positive sur le sous-espace vectoriel H_0.

Cas d'une fonction convexe

Considérons maintenant le problème de minimisation (1.29) donné précédemment avec des contraintes linéaires de type inégalités, où f est supposée convexe de classe C^1.

Théorème 1.25. *[7] Soient $x^\star$ une solution réalisable du problème* (1.29) *et $(x^\star, \lambda^\star)$ un couple de vecteurs vérifiant les conditions d'optimalité de KKT du premier ordre :*

(i) $\nabla L_x(x^\star, \lambda^\star) = \nabla f(x^\star) + A^T \lambda^\star = 0$ *;*

(ii) $\lambda_i^\star \geq 0, \quad \forall i \in I$ *;*

(iii) $\lambda_i^\star g_i(x^\star) = 0, \quad \forall i \in I.$

Alors le vecteur $x^\star$ est un minimum global du problème (1.29)*.*

1.8 Conclusion

Dans ce chapitre, nous avons rappelé les concepts mathématiques essentiels à cette thèse, en commençant par les notions de convexité et d'analyse convexe, notamment les ensembles convexes et les fonctions convexes et non convexes. Nous avons également approfondi les notions de formes quadratiques définies ou semi-définies positives, en mettant l'accent sur leurs propriétés fondamentales pour l'optimisation non linéaire. En outre, nous avons présenté les M-matrices, en soulignant leurs caractéristiques structurelles telles que la dominance diagonale et la positivité de l'inverse, qui jouent un rôle clé dans la modélisation mathématique et la résolution de systèmes linéaires. Enfin, nous avons abordé les résultats principaux en optimisation non linéaire, en discutant des conditions d'optimalité pour les problèmes sans et avec contraintes, fournissant ainsi un cadre solide pour traiter les problèmes de programmation non linéaire.

Chapitre 2

Étude des M-matrices et leurs applications en optimisation

2.1 Introduction

Dans ce chapitre, nous explorons deux exemples de problèmes pratiques pouvant être modélisés sous la forme de Problème de Programmation Quadratique (PPQ), en mettant particulièrement l'accent sur ceux impliquant des **M-matrices**. Ces dernières jouent un rôle essentiel dans la résolution de problèmes complexes grâce à leurs propriétés structurelles uniques, qui en facilitent la manipulation et la résolution. En particulier, leur inverse non négative les rend particulièrement adaptées pour modéliser des systèmes d'équations linéaires, justifiant ainsi leur utilisation fréquente dans des domaines tels que l'optimisation, l'ingénierie et la finance.

Nous commençons par le problème bien connu de l'optimisation de la toile d'une tente de cirque, également appelé en anglais **Circus Tent Problem**, qui appartient à la catégorie des problèmes d'obstacle. Ensuite, nous examinons comment des problèmes financiers complexes, tels que la valorisation des options américaines, peuvent être modélisés sous la forme de problème de complémentarité linéaire (PCL) ou de problème de PPQ intégrant des **M-matrices**.

2.2 Circus Tent Problem

La forme naturelle de la toile d'une tente de cirque, constituée d'un matériau lourd et élastique, peut être déterminée en modélisant le problème comme une minimisation de l'énergie de déformation, formulée sous forme de problème de programmation quadratique PPQ. L'énergie potentielle totale à minimiser se compose de deux termes principaux : l'énergie élastique, qui représente la déformation du matériau, et l'énergie gravitationnelle, liée au poids de la toile. La solution finale doit respecter des contraintes physiques imposées par les points d'ancrage, tels que les poteaux de soutien et les limites du domaine. En tenant compte de ces interactions physiques et géométriques, des outils d'optimisation quadratique permettent de calculer une

solution réaliste et physiquement cohérente.

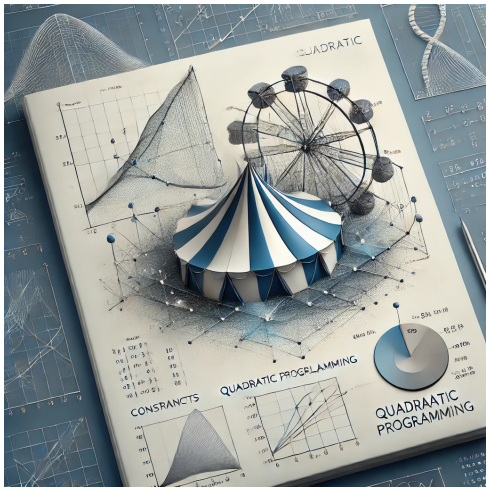

FIGURE 2.1 – Optimisation de la Toile d'une Tente de Cirque.

2.2.1 Formulation du problème

— **Définition des variables et des paramètres :** Soit $h(p)$ la hauteur de la toile au point $p = (i, j)$ de la grille, où i, j sont les indices de la position dans la grille discrétisée. La grille discrétisée a $N \times N$ points, et chaque point de la grille est connecté à ses voisins immédiats dans les directions x et y.

* $h(i, j)$: hauteur de la toile en chaque point $p = (i, j)$ de la grille.
* $N \times N$: dimensions de la grille discrétisée représentant la toile.
* $c = \frac{1}{3000}$: facteur constant représentant l'effet du poids du matériau de la toile.

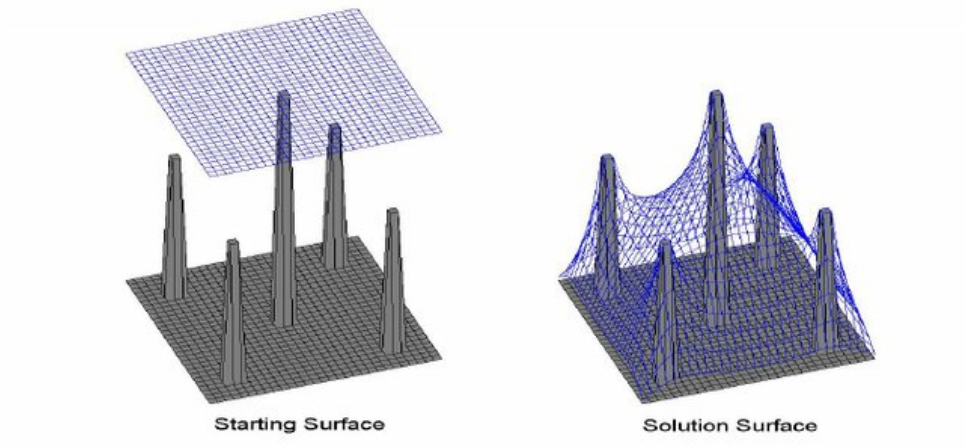

FIGURE 2.2 – Schéma des pôles et de la région à couvrir pour l'optimisation de la forme de la toile

— **Expression de l'énergie totale :** L'énergie totale à minimiser est la somme de deux énergies principales :

* ***L'énergie gravitationnelle :*** donnée par $\frac{1}{3000}\sum_p h(p)$.
* ***L'énergie élastique (énergie de Dirichlet) :***liée à la déformation de la toile sous l'effet de la tension et qui dépend des dérivées secondes de la hauteur $h(p)$ approximées à l'aide de l'opérateur de Laplace discrétisé.

2.2.1.1 Formulation de l'Énergie de Dirichlet

L'énergie de Dirichlet est une mesure mathématique qui quantifie la régularité d'une fonction. Elle est liée à la variation de la fonction de hauteur $h(p)$, où p représente un point dans le domaine. Cette énergie est définie par :

$$E(h) = \int \|\nabla h(p)\|^2 \, dp$$

Elle est également modélisée par l'application de l'opérateur de Laplace à la fonction $h(p)$, ce qui permet de mesurer la diffusion des valeurs autour de chaque point. Intuitivement, l'énergie de Dirichlet peut être interprétée comme l'énergie élastique d'une toile tendue : elle cherche à minimiser les variations de hauteur entre les points voisins, conduisant ainsi à une surface aussi lisse et équilibrée que possible.

Opérateur de Laplace

L'opérateur de Laplace en 2D appliqué à la fonction $h(p)$ (représentant la hauteur de la toile) est donné par la somme des dérivées secondes de $h(p)$ dans les directions x et y :

$$\Delta h(p) = \frac{\partial^2 h}{\partial x^2}(p) + \frac{\partial^2 h}{\partial y^2}(p),$$

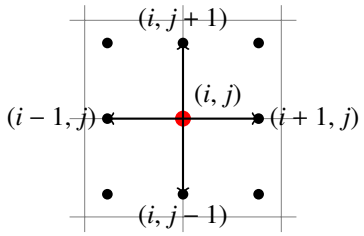

FIGURE 2.3 – Schéma illustrant l'opérateur de Laplace discrétisé appliqué à un point $p = (i, j)$ de la grille.

Les dérivées secondes de $h(p)$ dans les directions x et y sont approximées à l'aide de différences finies centrée sur une grille discrète avec un pas uniforme $\Delta x = \Delta y$:

— **Dérivées secondes par rapport à** x

L'approximation de la dérivée seconde de $h(p)$ par rapport à x au point $p = (i, j)$ sur une grille discrétisée est donnée par :

$$\frac{\partial^2 h}{\partial x^2}(p) \approx \frac{h(i+1, j) - 2h(i, j) + h(i-1, j)}{(\Delta x)^2}.$$

— **Dérivées secondes par rapport à** y

De même, la dérivée seconde de $h(p)$ par rapport à y au point $p = (i, j)$ est donnée par :

$$\frac{\partial^2 h}{\partial y^2}(p) \approx \frac{h(i, j+1) - 2h(i, j) + h(i, j-1)}{(\Delta y)^2}.$$

En combinant ces deux expressions et en utilisant $\Delta x = \Delta y$, on peut écrire l'opérateur de Laplace discret Δ appliqué à $h(p)$ en $2D$ comme suit :

$$\Delta h(p) = \frac{\partial^2 h}{\partial x^2}(i, j) + \frac{\partial^2 h}{\partial y^2}(i, j),$$

$$\Delta h(p) \approx \frac{h(i+1, j) + h(i-1, j) + h(i, j+1) + h(i, j-1) - 4h(i, j)}{(\Delta x)^2}.$$

— **Discrétisation de l'espace**

La grille de la toile est discrétisée en une matrice $N \times N$, où chaque élément de la matrice correspond à un point $p = (i, j)$ de la toile. La hauteur de la toile à chaque point est représentée par un vecteur h de taille N^2, et les dérivées secondes sont calculées à l'aide de la matrice de Laplace discrétisée.

Énergie de Dirichlet

L'énergie de Dirichlet, qui mesure la variation de la hauteur de la toile entre les points voisins, peut être formulée discrètement comme :

$$E_{\text{Dirichlet}}(h) = \sum_p \left(\frac{\partial^2 h}{\partial x^2}(p) + \frac{\partial^2 h}{\partial y^2}(p) \right) h(p).$$

En écrivant l'énergie de Dirichlet en fonction des indices i et j, nous notons que $h(p)$ représente la hauteur de la toile en un point $p = (i, j)$. En appliquant l'approximation par différences finies pour l'opérateur de Laplace sur une grille régulière de pas $\Delta x = \Delta y = 1$, on obtient :

$$E_{\text{Dirichlet}}(h) = \sum_{i,j} \Big(-1 \cdot (h(i+1,j) + h(i-1,j) + h(i,j+1) + h(i,j-1)) + 4h(i,j) \Big) \cdot h(i,j).$$

Pour mieux comprendre les termes de l'énergie de Dirichlet dans le modèle de la toile de tente, décomposons la somme des forces élastiques entre un point p et ses voisins immédiats.

Soit $p = (i, j)$ un point de la grille discrétisée, et soit Δx et Δy les espacements dans les directions x et y, respectivement. Les termes de la somme représentant les forces élastiques entre le point p et ses voisins sont :

- $h(i+1, j)$: La hauteur de la toile au voisin immédiat de p dans la direction x, c'est-à-dire le point situé à droite de p.
- $h(i-1, j)$: La hauteur de la toile au voisin immédiat de p dans la direction x, c'est-à-dire le point situé à gauche de p.
- $h(i, j+1)$: La hauteur de la toile au voisin immédiat de p dans la direction y, c'est-à-dire le point situé au-dessus de p.
- $h(i, j-1)$: La hauteur de la toile au voisin immédiat de p dans la direction y, c'est-à-dire le point situé en dessous de p.

Ces termes sont utilisés pour représenter les forces élastiques exercées sur le point p par ses voisins immédiats. Ils sont associés à l'opérateur de Laplace qui, en calcul différentiel, mesure la variation de la hauteur entre les points voisins dans les directions x et y.

2.2.1.2 Formulation de l'Énergie Totale

L'énergie totale $E_{\text{total}}(x)$ prend en compte l'énergie de Dirichlet, représentant l'énergie élastique de la toile, et l'énergie gravitationnelle, qui est proportionnelle à la hauteur $h(p)$ de la toile. L'énergie gravitationnelle est modélisée comme suit :

$$E_{\text{grav}}(x) = \frac{1}{3000} \sum_{i,j} h(i,j),$$

où $\frac{1}{3000}$ est un facteur de proportionnalité.

Ainsi, l'énergie totale $E_{\text{total}}(h)$ devient :

$$E_{\text{total}}(h) = \sum_{i,j} \left[\Big(-1 \cdot (h(i+1,j) + h(i-1,j) + h(i,j+1) + h(i,j-1)) + 4h(i,j) \Big) \cdot h(i,j) + \frac{1}{3000} h(i,j) \right].$$

Cette expression combine l'énergie de Dirichlet et l'énergie gravitationnelle pour donner la fonction à minimiser.

2.2.2 Matrice de Discrétisation

L'énergie élastique s'écrit en termes matriciels grâce à la matrice Q qui est une matrice creuse de taille $N^2 \times N^2$, définie par :

$$Q(i,j) = \begin{cases} 4, & \text{si } i = j, \\ -1, & \text{si } i \text{ et } j \text{ sont voisins immédiats,} \\ 0, & \text{sinon.} \end{cases}$$

Dans cet exemple, la matrice de discrétisation de Laplace est confondue avec une M-matrice symétrique.

2.2.3 Formulation de la fonction objectif

La fonction objectif E_{total} à minimiser est de la forme quadratique suivante :

$$E_{\text{total}}(h) = \frac{1}{2} h^T Q h + c \sum_{i,j} h(i,j),$$

sous la contrainte :

$$h \geq z.$$

Dans ce problème, nous utilisons les notations suivantes :

— h est un vecteur de taille N^2 représentant les hauteurs des points de la grille ;

— Q est une matrice carrée de taille $N^2 \times N^2$, où N est le nombre de points dans une direction de la grille ;

— Le terme $\frac{1}{2} h^T Q h$ représente l'énergie élastique, tandis que le terme $c \sum_{i,j} h(i,j)$ représente l'énergie gravitationnelle, qui est proportionnelle à la somme des hauteurs des points de la grille ;

— $h \geq z$ représente la contrainte sur la hauteur minimale ou maximale du sol ou des poteaux. Par exemple, la hauteur au point p ne peut pas être inférieure à la hauteur du sol, ou les mâts doivent avoir une hauteur positive ;

— Les contraintes de bord imposent que la hauteur de la toile soit nulle aux bords de la grille. Cela signifie que les points situés sur les limites de la grille ne contribuent pas à la déformation de la toile. Cela peut être formulé comme :

$$h_{\text{bords}} = 0.$$

Les éléments de h correspondant aux points de la grille situés aux bords sont fixés à zéro pour assurer une structure stable.

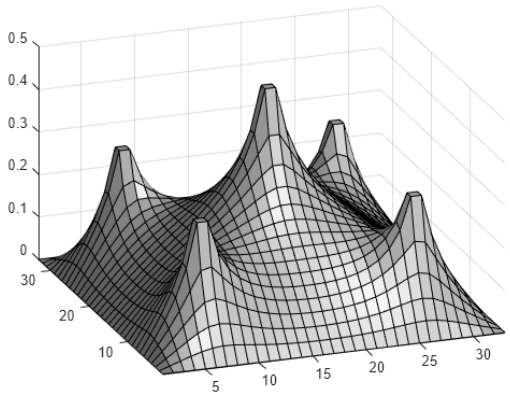

FIGURE 2.4 – Optimisation de l'énergie totale pour déterminer la forme de la toile.

Conclusion

Dans cette section, nous avons présenté un modèle mathématique permettant de déterminer la forme naturelle de la toile d'une tente de cirque sous l'effet des forces physiques qui lui sont appliquées. Ce modèle repose sur l'optimisation quadratique et s'appuie sur la discrétisation de l'opérateur de Laplace pour modéliser la répartition des forces élastiques dans la toile.

L'énergie de Dirichlet, qui mesure la variation de la hauteur entre les points voisins, a été discrétisée sur une grille régulière. Cette formulation nous a permis d'exprimer l'énergie élastique sous forme matricielle à l'aide d'une M-matrice creuse Q, représentant le laplacien discret. L'ajout de l'énergie gravitationnelle, qui dépend directement des hauteurs des points de la grille, complète l'expression de l'énergie totale de la structure.

L'objectif est alors de minimiser cette énergie totale sous des contraintes physiques imposées par la structure de la tente. En particulier, nous avons introduit des conditions aux limites garantissant que la toile reste attachée aux bords et que certaines hauteurs minimales sont respectées pour assurer la stabilité de l'ensemble.

Cette modélisation permet ainsi d'obtenir numériquement une configuration optimale de la toile, en trouvant l'équilibre entre la tension élastique et la gravité. Dans la suite de cette étude, nous pourrons explorer différentes méthodes numériques pour résoudre efficacement ce problème d'optimisation et analyser l'impact de divers paramètres, tels que la densité du maillage, la rigidité du matériau ou encore les forces extérieures appliquées à la toile.

2.3 Valorisation des options américaines

Cette section illustre comment des problèmes financiers complexes, tels que la valorisation des options américaines, peuvent être modélisés sous forme de **Problèmes de Complémentarité Linéaire (PCL)** ou de **Problèmes de Programmation Quadratique (PPQs)** avec des M-matrices [25, 88, 108]. Nous présentons les formulations mathématiques, les approches nu-

mériques et les propriétés des M-matrices qui garantissent la stabilité et la convergence des solutions.

Les options américaines permettent un exercice à tout moment avant leur échéance, ce qui ajoute une contrainte supplémentaire par rapport aux options européennes. La valorisation de ces options peut être modélisée par l'équation de Black-Scholes, complétée par une inégalité liée à la contrainte d'exercice anticipé.

2.3.1 Formulation du problème

L'équation de Black-Scholes pour la valeur de l'option $V(S,t)$ est une équation aux dérivées partielles utilisée pour modéliser la dynamique des prix d'instruments financiers. Elle est donnée par :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0, \quad S > 0,\ 0 \le t \le T, \tag{2.1}$$

où :

— S : prix de l'actif sous-jacent,
— t : temps restant avant l'échéance,
— σ : volatilité du prix de l'actif,
— r : taux d'intérêt sans risque.

Pour une option américaine, la contrainte d'exercice anticipé impose :

$$V(S,t) \ge \max(S - K, 0),$$

$$\left(\frac{\partial V}{\partial t} + \mathcal{L}[V]\right) \cdot (V(S,t) - \max(S - K, 0)) = 0,$$

avec :

— K : le prix d'exercice de l'option ;
— $\max(S - K, 0)$: la valeur intrinsèque de l'option ;
— $\mathcal{L}[V]$: l'opérateur différentiel de Black-Scholes, défini comme suit :

$$\mathcal{L}[V] = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV.$$

L'équation de Black-Scholes peut alors être réécrite sous la forme :

$$\frac{\partial V}{\partial t} + \mathcal{L}[V] = 0.$$

Dans le cadre de la résolution numérique, on peut reformuler l'équation de Black-Scholes

comme suit :

$$\mathcal{L}[V] = -\frac{\partial V}{\partial t}.$$

Ce terme $\mathcal{L}[V]$ capture les effets dynamiques de l'évolution de l'option en tenant compte de la volatilité σ, du taux d'intérêt r, et du prix S de l'actif.

2.3.1.1 Approximations Numériques

Pour résoudre l'équation de Black-Scholes numériquement, nous utilisons souvent les méthodes des différences finies. L'idée principale est de discrétiser le domaine en temps et en espace, puis d'approximer les dérivées partielles par des expressions en différences finies.

— **Discrétisation temporelle :**
Le domaine temporel $[0, T]$ est divisé en N_t intervalles de longueur $\Delta t = T/N_t$. Les instants discrétisés sont notés :

$$t_n = n\Delta t, \quad n = 0, 1, \ldots, N_t.$$

— **Discrétisation spatiale :**
Le domaine des prix de l'actif sous-jacent $[0, S_{\max}]$ est divisé en N_S intervalles de longueur $\Delta S = S_{\max}/N_S$. Les points discrétisés sont notés :

$$S_i = i\Delta S, \quad i = 0, 1, \ldots, N_S.$$

Schémas Numériques

La valeur $V(S, t)$ est discrétisée en V_i^n, représentant $V(S_i, t_n)$. Les dérivées partielles de l'équation de Black-Scholes sont approximées comme suit :

— **Dérivée temporelle :**

$$\frac{\partial V}{\partial t} \approx \frac{V_i^{n+1} - V_i^n}{\Delta t};$$

— **Dérivée spatiale première :**

$$\frac{\partial V}{\partial S} \approx \frac{V_{i+1}^n - V_{i-1}^n}{2\Delta S};$$

— **Dérivée spatiale seconde :**

$$\frac{\partial^2 V}{\partial S^2} \approx \frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta S^2}.$$

2.3.1.2 Assemblage de la Discrétisation

Ces approximations transforment l'équation continue en un système d'équations linéaires discret. Les valeurs des V_i^{n+1} sont calculées directement à partir des valeurs de V_i^n obtenues à l'instant précédent :

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} + \frac{1}{2}\sigma^2 S_i^2 \frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta S^2} + rS_i \frac{V_{i+1}^n - V_{i-1}^n}{2\Delta S} - rV_i^n = 0. \tag{2.2}$$

En substituant ces approximations dans l'équation de Black-Scholes, on obtient un système linéaire permettant de calculer les valeurs V_i^{n+1} à partir de V_i^n. Dans le cas d'un schéma implicite, ce système peut être écrit sous forme matricielle :

$$AV^{n+1} = b, \tag{2.3}$$

où :

- A est une matrice tridiagonale dérivée de la discrétisation implicite de l'opérateur différentiel $\mathcal{L}$;
- V^{n+1} est le vecteur des valeurs de l'option à l'instant t_{n+1} ;
- b est un vecteur dépendant de V^n, auquel s'ajoutent les éventuelles contributions des conditions aux limites, c'est-à-dire :

  $$b = V^n + \Delta t \cdot f,$$

  avec f représentant les termes correctifs aux bords.

2.3.2 Matrice de Discrétisation

La matrice A dans un schéma implicite est construite pour représenter l'opérateur différentiel de l'équation de Black-Scholes. Elle est tridiagonale, et généralement c'est une M-matrice en raison des dépendances spatiales aux points voisins $i-1, i$, et $i+1$. Voici la structure de la matrice A pour N points de discrétisation spatiale :

$$A = \begin{bmatrix} a_1 & \beta_1 & 0 & \cdots & 0 \\ \gamma_2 & a_2 & \beta_2 & \cdots & 0 \\ 0 & \gamma_3 & a_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \beta_{N-1} \\ 0 & 0 & 0 & \gamma_N & a_N \end{bmatrix},$$

où les coefficients a_i, β_i, et γ_i sont définis comme suit :

$$\alpha = \frac{\Delta t}{\Delta S^2},$$

$$a_i = 1 + r\Delta t + \sigma^2 S_i^2 \alpha,$$

$$\beta_i = -\frac{1}{2}\left(\sigma^2 S_i^2 \alpha + \frac{rS_i \Delta t}{\Delta S}\right),$$

$$\gamma_i = -\frac{1}{2}\left(\sigma^2 S_i^2 \alpha - \frac{rS_i \Delta t}{\Delta S}\right).$$

Ainsi, A est une M-matrice à diagonale dominante stricte, utilisée pour résoudre le système discret dans le cadre du schéma implicite de l'équation de Black-Scholes.

Remarques :

— $\alpha = \frac{\Delta t}{\Delta S^2}$ est un paramètre qui regroupe les dépendances en Δt et ΔS^2 ;

— Les termes diagonaux a_i combinent les contributions de la dérivée temporelle et des termes quadratiques $\sigma^2 S^2$, ainsi que des termes constants ;

— Les termes hors diagonaux (interactions avec les points voisins) : β_i et γ_i représentent les interactions avec les points voisins $(i + 1)$ et $(i - 1)$;

— Les conditions aux limites affectent les premières et dernières lignes de la matrice A pour imposer des comportements spécifiques (par exemple, conditions de Dirichlet ou Neumann) ;

  — **Condition en $S = 0$ (frontière inférieure) :** on impose que $V(0, t) = 0$, ce qui reflète l'idée que la valeur de l'option devient nulle si le prix de l'actif est nul ;

  — **Condition en $S \to \infty$ (frontière supérieure) :** on impose que $V(S, t) \to S - Ke^{-r(T-t)}$ à mesure que S devient très grand, ce qui correspond à la valeur d'une option européenne.

Le vecteur b contient les termes issus des conditions aux limites et initiales. Il est constitué des valeurs associées à chaque point S_i et à chaque instant de temps t_n. Par exemple, à l'échéance (au temps final T), b prend les valeurs suivantes :

$$b = \begin{bmatrix} \max(S_1 - K, 0) \\ \max(S_2 - K, 0) \\ \vdots \\ \max(S_N - K, 0) \end{bmatrix}.$$

En outre, les conditions aux limites doivent être appliquées à b, ce qui peut être formulé comme suit :

— À $S = 0$ on impose que $V(0, t) = 0$, ce qui implique que la première ligne de A et le premier élément de b sont ajustés en conséquence ;

— À $S \to \infty$, on impose que $V(S, t) \to S - Ke^{-r(T-t)}$, ce qui permet d'ajuster la dernière ligne de A et le dernier élément de b.

2.3.3 Formulation en Problème de Complémentarité Linéaire (PCL)

Pour une option américaine, la contrainte $V(S,t) \geq \max(S-K,0)$ impose un comportement non linéaire. Cela conduit à un problème de complémentarité linéaire PCL, exprimé comme suit :

Détail de la Transition vers le Problème de Complémentarité Linéaire (PCL)

— **Ajout de la contrainte intrinsèque** : Les options américaines doivent toujours respecter la contrainte suivante à tout instant t :

$$V(S,t) \geq \max(S-K,0),$$

ce qui implique que les valeurs calculées pour V^{n+1} (à chaque pas de temps) doivent être ajustées pour respecter cette borne inférieure.

— **Couplage entre la contrainte et l'équation de Black-Scholes** : Lorsque $V(S,t) > \max(S-K,0)$, l'option suit l'évolution dictée par l'équation de Black-Scholes discrétisée. En revanche, si $V(S,t) = \max(S-K,0)$, cela signifie que l'option est exercée à ce moment précis.

— **Formulation en complémentarité** : Ces deux comportements (continuité de l'évolution et respect de la contrainte) sont exprimés ensemble à travers le problème de complémentarité linéaire :

$$\begin{aligned} &AV^{n+1} \geq b, \\ &V^{n+1} \geq \max(S-K,0), \\ &\left(V^{n+1} - \max(S-K,0)\right)^{\top} \cdot \left(AV^{n+1} - b\right) = 0. \end{aligned}$$

Cette équation indique que :
- Si $V^{n+1} > \max(S-K,0)$, alors $AV^{n+1} = b$ (évolution classique de Black-Scholes),
- Si $V^{n+1} = \max(S-K,0)$, alors l'option est exercée et la contrainte est activée.

Une fois la matrice A et le vecteur b construits, le système $AV^{n+1} = b$ (pour les options européennes) ou le système d'inéquations $AV^{n+1} \geq b$ (pour les options américaines) peut être résolu numériquement à chaque pas de temps t_n.

2.3.4 Formulation en Problème de Programmation Quadratique (PPQ)

La valorisation des options américaines peut également être vue comme un problème de minimisation sous contraintes. Cela donne lieu à une formulation en problème de programma-

tion quadratique PPQ, définie comme suit :

$$\text{Minimiser} \quad \frac{1}{2}V^T QV - c^T V, \tag{2.4}$$

$$\text{sous les contraintes} \quad V_i \geq \max(S_i - K, 0), \quad \forall i. \tag{2.5}$$

avec :

— V : le vecteur des valeurs discrètes de l'option ;
— Q : une matrice symétrique semi-définie positive, souvent dérivée de la matrice A du PCL ;
— c : un vecteur correspondant aux termes constants du problème.

La matrice Q est généralement obtenue à partir de la discrétisation des Équations aux Dérivées Partielles (EDP) qui régissent le prix des options. Par exemple, en utilisant des schémas de différences finies implicites, Q est construite comme une matrice de discrétisation, où :

$$Q = A + \lambda I,$$

avec :

— A représentant la matrice tridiagonale associée à l'opérateur différentiel discret ;
— λ un terme de régularisation, souvent lié au taux d'actualisation ;
— I la matrice identité.

La structure tridiagonale de A et les propriétés de λI garantissent que Q satisfait les propriétés de la M-matrice.

Dans le cadre financier, l'utilisation d'une M-matrice dans la formulation QP permet de garantir que :

— Les prix calculés de l'option sont économiquement cohérents (non négatifs) ;
— Les schémas de résolution numérique sont stables et robustes face aux perturbations ;
— Les solutions discrètes convergent correctement vers la solution continue.

La M-matrice constitue donc une base mathématique essentielle pour assurer la fiabilité des calculs dans la valorisation numérique des options américaines.

Ainsi, la transition de la discrétisation de l'équation de Black-Scholes vers un problème de complémentarité linéaire (PCL) ou de programmation quadratique (PPQ) est motivée par la nécessité de modéliser avec précision les contraintes spécifiques aux options américaines. Contrairement aux options européennes, qui ne peuvent être exercées qu'à l'échéance, les options américaines peuvent être exercées à tout moment avant la maturité. Cette flexibilité introduit une contrainte supplémentaire sur la valeur de l'option : elle ne doit jamais être inférieure

à sa valeur intrinsèque.

L'approche basée sur un PCL consiste à formuler le problème sous la forme d'un système d'inégalités et de complémentarité, garantissant ainsi, dans les régions où l'exercice anticipé est optimal, l'égalité entre la valeur de l'option et sa valeur intrinsèque. En revanche, lorsque l'option n'est pas exercée immédiatement, sa dynamique est régie par l'équation de Black-Scholes.

La formulation en PPQ repose sur la minimisation d'une fonction quadratique sous contraintes, permettant d'incorporer efficacement les conditions de non-négativité et d'exercice anticipé. Ces méthodes, associées à des algorithmes numériques avancés comme les méthodes actives ou les solveurs spécialisés en optimisation convexe, assurent une résolution robuste et efficace du problème.

2.4 Conclusion

En conclusion, ce chapitre a montré comment des problèmes pratiques complexes, tels que l'optimisation de la toile d'une tente de cirque et la valorisation des options américaines, peuvent être modélisés et résolus à l'aide des M-matrices dans le cadre des Problèmes de Programmation Quadratique (PPQs) ou des Problèmes de Complémentarité Linéaire (PCLs). Ces exemples illustrent la puissance et la polyvalence des M-matrices pour traiter efficacement des problèmes en optimisation, en ingénierie et en finance.

Chapitre 3

Problème de Programmation Quadratique (PPQ) avec une M-matrice

3.1 Introduction

Dans ce chapitre, nous introduisons une méthode permettant de résoudre un problème de programmation quadratique avec une M-matrice sous des contraintes simples. Cette méthode exploite le fait qu'une M-matrice possède une inverse non négative (tous les éléments de la matrice inverse D^{-1} sont non négatifs), ce qui permet de générer une suite monotone de solutions réalisables [95, 96, 98]. En intégrant le concept de support, formulé par Gabasov et all. [36, 37], afin de considérer la non-linéarité de la fonction objectif, notre méthode [47] diffère de la méthode présentée par Luk et Pagano [69] par une condition plus générale qui nous permet d'avoir une solution réalisable initiale, plus proche de la solution optimale. Cette caractéristique facilite une convergence plus rapide vers la solution optimale, réduisant ainsi le nombre d'itérations nécessaires par rapport aux approches de Chandrasekaran et de Luk et Pagano.

Pour approfondir notre approche, ce chapitre est organisé comme suit. Dans la prochaine section, nous présenterons le problème quadratique traité et fournirons quelques définitions relatives à l'approche proposée. La section 3 aborde le critère d'optimalité qui nous permet d'évaluer l'optimalité d'une solution réalisable dans le cadre du problème formulé. Dans la section 4, nous discuterons du problème dual ainsi que de ses propriétés. La section 5 rappelle les théorèmes sur lesquels notre approche est fondée. La section 6 décrit le schéma de l'algorithme de la méthode proposée. La section 7 traite des expérimentations numériques concernant la résolution des problèmes de programmation quadratique avec une M-matrice générée aléatoirement, où nous comparons notre approche à celles de Chandrasekaran et de Luk et Pagano, implémentées sous Matlab. Ce chapitre se termine par une conclusion, qui constitue la dernière section.

3.2 Formulation du problème et définitions

La formulation du problème de programmation quadratique PPQ avec des contraintes simples s'écrit comme suit :

$$\begin{cases} \min\ F(x) = \frac{1}{2}x^T Dx + c^T x, \\ \quad x \geq 0, \qquad x \in \mathbb{R}^n, \end{cases} \tag{3.1}$$

où $c = c(J) = (c_j, \quad j \in J)$ et $x = x(J) = (x_j, \quad j \in J)$ sont des n-vecteurs réels, avec $J = \{1, 2, \cdots, n\}$. La matrice $D = D(J, J)$ est une M-matrice carrée d'ordre n, symétrique et définie positive $\left(x^T Dx > 0, \quad \forall x \neq 0\right)$.

Rappelons qu'une matrice $D = (d_{ij},\ 1 \leq i, j \leq n)$ est dite M-matrice si elle satisfait les propriétés suivantes :

$$d_{ii} > 0 \quad \text{et} \quad d_{ij} \leq 0 \quad \text{pour} \quad i \neq j, \quad \text{avec} \quad D^{-1} \geq 0,$$

où l'expression $D^{-1} \geq 0$ signifie que tous les coefficients de la matrice D^{-1} sont non négatifs.

Avant d'aller plus loin, rappelons quelques définitions de base afin de faciliter la compréhension des éléments qui suivront.

— Un vecteur $x \geq 0$ est appelé *solution réalisable* ou *plan* du problème (3.1). Une solution réalisable x^0 est dite *optimale* si

$$F(x^0) = \frac{1}{2}x^{0^T} Dx^0 + c^T x^0 = \min_x F(x),$$

où x est pris parmi toutes les solutions réalisables du problème.

— Le vecteur

$$g(x) = g(J) = (g_j,\ j \in J) = Dx + c,$$

désigne le gradient de la fonction objectif F du problème (3.1) au point x.

3.3 Critère d'optimalité

Définissons maintenant le support de la fonction objectif :

— Le sous-ensemble $J_S \subset J$ tel que $\det D_S = \det D(J_S, J_S) \neq 0$ est appelé *support* de la fonction objectif F.

— Tout sous-ensemble $J_S \subset J$ est un support de la fonction objectif du problème (3.1), car il vérifie toujours la condition :

$$\det D_S = \det D(J_S, J_S) \neq 0.$$

La paire $J_p = \{J_S, J_N\}$, avec $J_N = J \setminus J_S$, est alors dite *support du problème (3.1).*

— Le couple $\{x, J_p\}$ est appelé *solution réalisable de support*.

— Si la solution réalisable x et le support J_S sont tels que

$$g_j(x) = 0, \quad j \in J_S,$$

alors la solution réalisable de support $\{x, J_P\}$ est dite *accordée*.

Pour évaluer l'optimalité d'une solution réalisable dans le cadre du problème formulé, il est indispensable d'examiner les conditions nécessaires et suffisantes avec le critère d'optimalité suivant :

Théorème 3.1. *[37] Pour qu'une solution réalisable x^0 du problème (3.1) soit optimale, il faut et il suffit que les conditions suivantes soient satisfaites pour tout $j \in J$:*

$$\begin{cases} x_j^0 = 0 & \Rightarrow \quad g_j(x^0) \geq 0, \\ x_j^0 > 0 & \Rightarrow \quad g_j(x^0) = 0, \quad j \in J, \end{cases} \tag{3.2}$$

où $g(x) = g(J) = Dx + c$ est le gradient de la fonction objectif F au point x.

Soit J_S et J_N une partition de J : $J_S \cup J_N = J$, $J_S \cap J_N = \emptyset$. Alors le gradient de la fonction objectif F au point x peut s'écrire sous la forme suivante :

$$g = \begin{pmatrix} g_S \\ g_N \end{pmatrix}, \quad g_S = g(J_S) = D_S x_S + D_{SN} x_N + c_S, \quad g_N = g(J_N) = D_{NS} x_S + D_N x_N + c_N,$$

où

$$x = \begin{pmatrix} x_S \\ x_N \end{pmatrix}, \; c = \begin{pmatrix} c_S \\ c_N \end{pmatrix}, \; D_S = D(J_S, J_S), \; D_N = D(J_N, J_N), \; D_{SN} = D(J_S, J_N), \; D_{NS} = D(J_N, J_S).$$

3.4 Formulation du problème dual et définitions

Pour écrire le dual du problème (3.1), on utilise la fonction de Lagrange définie par :

$$L(\lambda) = F(\kappa) - v^T \kappa,$$

où $\lambda = (\kappa, v) \in \mathbb{R}^n \times \mathbb{R}^n$, $v \geq 0$.
Notons que pour κ vérifiant la contrainte $\kappa \geq 0$ du problème primal (3.1), on a

$$L(\lambda) \leq F(\kappa).$$

En utilisant le théorème de Karush-Kuhn-Tucker [58, 62], le vecteur κ doit vérifier la condition de stationnarité suivante :

$$\frac{\partial L}{\partial \kappa}(\lambda) = 0,$$

c'est-à-dire :

$$D\kappa + c - v = 0.$$

En tenant compte de cette dernière relation, la fonction de Lagrange devient :

$$\begin{aligned} L(\lambda) = L(\kappa, v) &= \frac{1}{2}\kappa^T D\kappa + c^T\kappa - v^T\kappa, \\ &= \frac{1}{2}\kappa^T D\kappa + (v - D\kappa)^T\kappa - v^T\kappa, \\ &= -\frac{1}{2}\kappa^T D\kappa. \end{aligned}$$

Puisque $L(\lambda)$ ne dépend pas de v, et $v \geq 0$, alors la condition de stationnarité peut s'écrire $D\kappa + c \geq 0$.
Par conséquent, le dual du problème primal (3.1) se formule de la manière suivante :

$$\begin{cases} L(\kappa) = -\frac{1}{2}\kappa^T D\kappa \longrightarrow \max, \\ D\kappa + c \geq 0, \\ \kappa \in \mathbb{R}^n. \end{cases} \tag{3.3}$$

Notons que le problème dual (3.3) est un programme quadratique concave. La différence par rapport à son problème primal (3.1) réside dans le fait que la variable κ est sans restriction de signe.

Définition 3.1.

— ***Pseudo-solution.*** *Un vecteur* $\kappa = \kappa(J) = (\kappa(J_S), \kappa(J_N))$ *vérifiant*

$$\begin{cases} \kappa_N = 0, \\ \kappa_S = -D_S^{-1} c_S, \end{cases}$$

est appelée pseudo-solution *du problème (3.1).*
Toute pseudo-solution κ *satisfait toujours* $g_S(\kappa) = 0$.

— ***Support coordinateur.*** *On appelle* support coordinateur *un support* $J_p = \{J_S, J_N\}$ *tel qu'il existe une pseudo-solution* κ *vérifiant :*

$$g_j(\kappa) \geq 0, \quad \forall j \in J_N. \tag{3.4}$$

Dans ce cas, on dit que la pseudo-solution κ est associée au support coordinateur J_p.

— ***Vecteur coplan.*** *Le n-vecteur $\delta(\kappa) = D\kappa + c$ est appelé* vecteur coplan*, associé à la pseudo-solution κ.*

Théorème 3.2. *Une pseudo-solution κ liée à un support coordinateur J_p est optimale si et seulement si les conditions suivantes sont satisfaites :*

$$\kappa_j \geq 0, \quad j \in J_S. \tag{3.5}$$

Remarque 3.1. *Toute pseudo-solution κ, liée à un support coordinateur J_p, constitue une solution réalisable du problème dual associé au problème primal (3.1) :*

$$\begin{cases} F(\kappa) = -\frac{1}{2}\kappa^T D\kappa \longrightarrow max, \\ D\kappa + c \geq 0. \end{cases} \tag{3.6}$$

3.5 Méthode de résolution

3.5.1 Construction d'une solution réalisable initiale

Considérons le problème de programmation quadratique sans contraintes suivant :

$$\begin{cases} \min F(x) = \frac{1}{2}x^T Dx + c^T x, \\ \quad x \in \mathbb{R}^n, \end{cases} \tag{3.7}$$

où $c = c(J) = (c_j,\ j \in J)$ et $x = x(J) = (x_j,\ j \in J)$ sont des n-vecteurs réels, avec $J = \{1, 2, \cdots, n\}$. La matrice $D = D(J, J)$ est une M-matrice carrée d'ordre n, symétrique et définie positive.

Une solution optimale $\hat{x}$ du problème (3.7) satisfait l'équation suivante :

$$g(\hat{x}) = D\hat{x} + c = 0.$$

Ainsi, en résolvant pour $\hat{x}$, on obtient :

$$\hat{x} = -D^{-1}c.$$

Remarque 3.2. *Étant donné que $g(\hat{x}) = 0$, la solution optimale $\hat{x}$ du problème sans contraintes (3.7) constitue une pseudo-solution du problème (3.1), associée au support coordinateur $J_p = \{J_S, J_N\}$, où :*

$$J_S = J \quad et \quad J_N = \emptyset.$$

D'après le théorème 3.2, si $\hat{x} \geq 0$, alors $x^0 = \hat{x}$ est une solution optimale du problème (3.1).

On formule le lemme suivant :

Lemme 3.1. *[69]*

(a) *Si le vecteur $c \geq 0$, alors $x^0 = 0$ résout le problème (3.1),*

(b) *Si le vecteur $c \leq 0$, alors $x^0 = -D^{-1}c$ résout le problème (3.1).*

Preuve.

(a) On a

$$x^0 = 0, \quad g(x^0) = Dx^0 + c = c \geq 0,$$

et d'après le critère d'optimalité (3.2), x^0 est solution optimale du problème (3.1).

(b) Puisque $D^{-1} \geq 0$ et $c \leq 0$, on aura

$$x^0 = -D^{-1}c \geq 0 \quad \text{et} \quad g(x^0) = Dx^0 + c = 0.$$

Par conséquent, x^0 résout le problème (3.1). ■

Afin d'exclure les deux cas triviaux précédemment évoqués, considérons maintenant le cas général où le vecteur c possède à la fois des composantes positives et négatives. Dans ce cas, introduisons deux ensembles distincts définis comme suit :

$$J_S = \{j \in J : \hat{x}_j \geq 0\}, \quad J_N = \{j \in J : \hat{x}_j < 0\}, \quad J_S \cup J_N = J.$$

— Si $J_S = J$, alors $x^0 = \hat{x} = -D^{-1}c$ est une solution optimale du problème (3.1).
— Sinon, considérons y comme la projection de $\hat{x}$ sur le domaine admissible du problème(3.1)

$$y = (y_j,\ j \in J), \quad \text{avec} \quad y_j = \max(0, \hat{x}_j).$$

D'où

$$\begin{cases} y_S = \hat{x}_S, \\ y_N = 0 > \hat{x}_N. \end{cases}$$

Lemme 3.2. *L'inégalité suivante est vérifiée*

$$g_S(y) \leq g_S(\hat{x}).$$

Preuve.

$$\begin{aligned} g_S(\hat{x}) &= D_S\hat{x}_S + D(J_S, J_N)\hat{x}_N + c_S \\ &= D_S y_S + c_S + D(J_S, J_N)\hat{x}_N \\ &= g_S(y) + D(J_S, J_N)\hat{x}_N \\ &\geq g_S(y), \end{aligned}$$

car $\quad D(J_S, J_N) \leq 0 \quad$ et $\quad \hat{x}_N < 0.$ ■

Étant donné que $g_S(\hat{x}) = 0$, le lemme 3.2 implique que $g_S(y) \leq 0$. Nous construisons alors un vecteur x tel que :

$$\begin{cases} x_N = y_N = 0, \\ x_S = -D_S^{-1} c_S. \end{cases} \tag{3.8}$$

On a donc

$$g_S(y) \leq 0 \quad \text{et} \quad g_S(x) = 0.$$

Lemme 3.3. *Les deux vecteurs x et y satisfont l'inégalité suivante :*

$$x_S \geq y_S \geq 0.$$

Preuve.

$$D_S(x_S - y_S) = D_S x_S + c_S - [D_S y_S + c_S],$$

comme $x_N = y_N = 0, \quad g_S(y) \leq 0 \quad$ et $\quad g_S(x) = 0$, alors on déduit

$$D_S(x_S - y_S) = g_S(x) - g_S(y) \geq 0.$$

La sous-matrice D_S étant une M-matrice [69], il s'ensuit que $D_S^{-1} \geq 0$. Ainsi, en multipliant l'inégalité précédente à gauche par D_S^{-1}, on obtient :

$$x_S \geq y_S \geq 0. \quad ■$$

D'après le lemme 3.3, le vecteur x construit selon (3.8) constitue une solution réalisable du problème (3.1), satisfaisant $x_S \geq 0$. Par ailleurs, selon [98], il vérifie également l'inégalité $x \leq x^0$, où x^0 désigne la solution optimale du problème (3.1).

3.5.2 Accroissement de la fonctionnelle

Calculons alors l'accroissement de la fonctionnelle

$$F(x) - F(y) = \frac{1}{2} x^T D x + c^T x - \frac{1}{2} y^T D y - c^T y,$$

où $\quad x_S \geq y_S \geq 0, \quad x_N = y_N = 0, \quad x_S = -D_S^{-1} c_S.$

Lemme 3.4. *On a*

$$F(x) \leq F(y).$$

Preuve. Soit

$$2F(x) \;=\; x_S^T\, D_S x_S + 2\, c_S^T\, x_S.$$

Puisque $x_S = -D_S^{-1} c_S$, nous avons alors

$$2F(x) \;=\; c_S^T\, D_S^{-1} c_S - 2\, c_S^T\, D_S^{-1} c_S = -c_S^T\, D_S^{-1} c_S.$$

Étant donné que la sous-matrice D_S est définie positive, on peut alors écrire :

$$\begin{aligned}
2F(x) &\leq (y_S - x_S)^T\, D_S (y_S - x_S) - c_S^T\, D_S^{-1} c_S \\
&\leq y_S^T\, D_S\, y_S + x_S^T\, D_S\, x_S - 2 y_S^T\, D_S\, x_S - c_S^T\, D_S^{-1}\, c_S \\
&\leq y_S^T\, D_S\, y_S + c_S^T\, D_S^{-1}\, D_S\, D_S^{-1}\, c_S + 2\, y_S^T\, D_S\, D_S^{-1}\, c_S - c_S^T\, D_S^{-1}\, c_S \\
&\leq y_S^T\, D_S\, y_S + 2\, c_S^T\, y_S \\
&\leq 2\, F(y).
\end{aligned}$$

D'où

$$F(x) \leq F(y). \quad \blacksquare$$

Remarque 3.3. *Le vecteur x construit selon (3.8) satisfait ainsi l'inégalité suivante :*

$$F(\hat{x}) < F(x_0) \leq F(x) \leq F(y), \tag{3.9}$$

où y représente la projection de $\hat{x}$ sur le domaine admissible du problème (3.1).

Nous rappelons ainsi le théorème suivant

Théorème 3.3. *[69] Supposons que l'inégalité $D_S^{-1} c_S \leq 0$ soit satisfaite pour un sous-ensemble non vide $J_S \subseteq J$. Définissons alors un vecteur x avec*

$$x_S = -D_S^{-1} c_S \;\; et \;\; x_N = 0.$$

Soient J_N^- et J_N^+ deux ensembles formant une partition de J_N, définis par :

$$J_N^- = \{j \in J_N : \;\; g_j(x) < 0\}, \;\; J_N^+ = \{j \in J_N : \;\; g_j(x) \geq 0\}.$$

Dans le cas où l'ensemble J_N^- est vide, il en résulte que :

1. *Le vecteur x résout le problème (3.1), sinon*
2. *Soit $\overline{J}_S := J_S \cup J_N^-$. Construire $\overline{x}$ avec $\overline{x}(\overline{J}_S) = -D^{-1}(\overline{J}_S, \overline{J}_S)\, c(\overline{J}_S)$ et $\overline{x}(J_N^+) = 0$. On obtient :*

(a) *Le vecteur* $\overline{x}(J_S) \geq x(J_S) \geq 0, \quad \overline{x}(J_N^-) \geq 0, \quad \overline{x}(J_N^+) = 0,$

(b) $g_j(\overline{x}) \leq g_j(x), \quad j \in J_N^+,$

(c) $F(\overline{x}) < F(x).$

3.6 Schéma de l'algorithme

En nous appuyant sur les résultats du théorème précédent, nous proposons l'algorithme suivant. Cet algorithme exploite les propriétés établies par le théorème précédent afin de garantir une solution à la fois efficace et précise au problème considéré.

Algorithm 1 Algorithme de résolution du PPQ

Entrée : Matrice D, vecteur c

Sortie : Solution optimale x^0

Début

Étape 1 : Calculer la solution optimale $\hat{x}$ du problème (3.7) :

$$g(\hat{x}) = D\hat{x} + c = 0 \quad \Longrightarrow \quad \hat{x} = -D^{-1}c.$$

Étape 2 : Vérifier si $\hat{x} \geq 0$

if $\hat{x} \geq 0$ **then**

Arrêter. Le vecteur $x^0 = \hat{x}$ est la solution optimale du problème (3.1).

else

Définissons les deux sous-ensembles J_S *et* J_N de sorte que :

$$J_S = \{j \in J : \hat{x}_j \geq 0\}, \quad J_N = \{j \in J : \hat{x}_j < 0\}.$$

Étape 3 : Construire x comme suit :

$$x_N = 0, \quad x_S = -D_S^{-1}c_S.$$

Étape 4 : Partitionner J_N en deux sous-ensembles :

$$J_N^- = \{j \in J_N : g_j(x) < 0\}, \quad J_N^+ = \{j \in J_N : g_j(x) \geq 0\}.$$

Étape 5 : Répéter jusqu'à ce que $J_N^- = \emptyset$

1. Calculer $g_N(x) = D(J_N, J_S)x_S + c_N$

2. Mettre à jour l'ensemble J_N^- : $J_N^- = \{j \in J_N : g_j(x) < 0.\}$

if J_N^- est non vide **then**

a. Mettre à jour les ensembles :

$$J_S := J_S \cup J_N^-, \quad J_N := J_N \setminus J_N^-$$

b. Reconstruire x comme suit :

$$x_N = 0, \quad x_S = -D_S^{-1}c_S.$$

Fin de la répétition

Fin.

3.7 Résultats numériques et comparaisons

Dans cette section, nous avons sélectionné deux problèmes représentatifs. L'objectif est de démontrer l'efficacité de l'algorithme proposé en effectuant une comparaison numérique avec l'algorithme de Luk et Pagano [69]. Toutes les expérimentations ont été réalisées sur un ordinateur équipé d'un processeur Intel(R) Core(TM) i3-2350 @ 2,30 GHz, disposant de 4,00 Go de RAM, et fonctionnant sous le système d'exploitation Windows 7, avec le langage de programmation MATLAB R2015a.

Le critère de comparaison entre les deux méthodes est le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) nécessaires pour obtenir la solution optimale du problème. Tous ces tests ont été réalisés sur le même ordinateur. Les valeurs présentées dans les tableaux représentent la moyenne de 5 problèmes tests pour chaque dimension n de la matrice . Nous définissons par

— **Algorithm1** : méthode de Luk and Pagano [69],

— **Algorithm2** : l'algorithme proposé,

— NJ_S : le nombre d'éléments dans le support J_S de la fonction objectif, juste après avoir calculé $\hat{x} = -D^{-1}c$,

— $\overline{NJ_S}$: le nombre d'éléments dans le support J_S de la fonction objectif à l'optimum,

— NP : le nombre d'éléments dans l'ensemble P, lors de l'initialisation de x, avec $P = \{j \in J : c_j \leq 0\}$ et $\overline{P} = J \setminus P$,

— $\overline{NP}$: le nombre d'éléments de l'ensemble P à l'optimum,

— **Avr-Iter** : le nombre moyen d'itérations effectuées par chaque algorithme,

— **Avr-CPU** : le temps machine (CPU time) moyen en secondes nécessaire pour obtenir la solution optimale du problème.

3.7.1 Exemple 1.

Considérons le problème de programmation quadratique PPQ (3.1) :

$$\begin{cases} \min\ F(x) = \frac{1}{2}x^T Dx + c^T x, \\ \quad x \geq 0, \qquad x \in \mathbb{R}^n, \end{cases}$$

Dans cet exemple, nous considérons le PPQ avec des contraintes simples. La matrice D est la matrice correspondant à la discrétisation par différences finies du problème de Dirichlet

unidimensionnel [109] :

$$D = \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix}_{n\times n} . \tag{3.10}$$

Nous choisissons de générer le vecteur c de manière à obtenir trois cas possibles pour le sous ensemble J_S. Soit r_i un nombre réel aléatoire issu d'une distribution uniforme $r_i \in U[0, 1]$. L'évaluation de la performance des deux méthodes repose sur deux critères : le temps moyen d'exécution (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) nécessaires pour atteindre la solution optimale du problème. Les résultats obtenus sont synthétisés dans les tableaux suivants :

Cas 1 : $NJ_S = n$

Le vecteur c est généré de telle sorte que $NJ_S = n \Rightarrow \hat{x} \geq 0$. Cela se traduit par la relation suivante pour chaque composante $i \in J$ du vecteur c :

$$c_i = 11 - 20\, r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{3.11}$$

Les résultats correspondants sont présentés dans la Table 3.1 ci-dessous.

Dimension	Algorithm1				Algorithm2			
n	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
500	07	280	500	0.0798	0	500	500	0.0048
1000	06	559	1000	0.1172	0	1000	1000	0.0067
1500	07	846	1500	0.1642	0	1500	1500	0.0091
2000	08	1077	2000	0.1867	0	2000	2000	0.0098
2500	08	1356	2500	0.1910	0	2500	2500	0.0111
3000	08	1625	3000	0.1999	0	3000	3000	0.0130
4000	09	2187	4000	0.2022	0	4000	4000	0.0193
5000	11	2742	5000	0.2153	0	5000	5000	0.0269

Table 3.1 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 1, avec $NJ_S = n$.

Les temps moyens de calcul (Avr-CPU) pour l'Algorithm1 augmentent progressivement avec la dimension n de la matrice, passant de 0.0798 secondes à 0.2153 secondes, tandis que l'Algorithm2 est beaucoup plus rapide, avec des temps variant de 0.0048 à 0.0269 secondes. Dans l'ensemble, l'Algorithm2 est beaucoup plus efficace en termes de temps et d'itérations.

Cas 2 : $NJ_S < n$

Pour ce cas, le vecteur c est généré selon la relation suivante :

$$c_i = 11 - 22\, r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{3.12}$$

Dimension	Algorithm1				Algorithm2			
n	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
500	14	249	498	0.1099	06	420	498	0.0146
1000	21	499	999	0.1182	07	982	999	0.0165
1500	23	727	1487	0.1226	21	142	1487	0.0378
2000	26	1010	1960	0.1396	30	851	1960	0.0569
2500	22	1242	2454	0.1406	32	330	2454	0.0598
3000	26	1482	2923	0.1473	35	1074	2923	0.0621
4000	26	2001	3958	0.1914	40	1172	3958	0.1101
5000	26	2500	4969	0.1974	40	2237	4969	0.1230

TABLE 3.2 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 1, avec $NJ_S < n$.

Même si les temps de calcul augmentent pour l'Algorithm2, ils restent généralement plus bas que ceux de l'Algorithm1 pour des dimensions n équivalentes. Ainsi, l'Algorithm2 se révèle plus efficace dans les applications, où la rapidité d'exécution est essentielle.

Cas 3 : $NJ_S = 0$

Dans ce cas, le vecteur c est généré de la manière suivante :

$$c_i = 11 - 25\, r_i \quad pour \;\; i = 1, 2, \ldots, n. \tag{3.13}$$

Dimension	Algorithm1				Algorithm2			
n	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
500	07	199	335	0.1053	11	0	335	0.0167
1000	09	444	752	0.1099	12	0	752	0.0232
1500	09	648	1051	0.1106	13	0	1051	0.0287
2000	10	853	1444	0.1218	16	0	1444	0.0431
2500	11	1149	1894	0.1240	16	0	1894	0.0487
3000	08	1360	2272	0.1249	11	0	2272	0.0529
4000	09	1758	2862	0.1341	14	0	2862	0.0681
5000	08	2206	3660	0.1456	17	0	3660	0.0936

TABLE 3.3 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 1, avec $NJ_S = 0$.

Une comparaison du temps moyen entre l'algorithme proposé et l'algorithme de Luk et Pagano [69] est présentée dans la FIGURE 3.1 ci-dessous . Cette figure illustre les performances des deux algorithmes sur un ensemble de tests, montrant une amélioration significative du temps de traitement avec l'algorithme proposé.

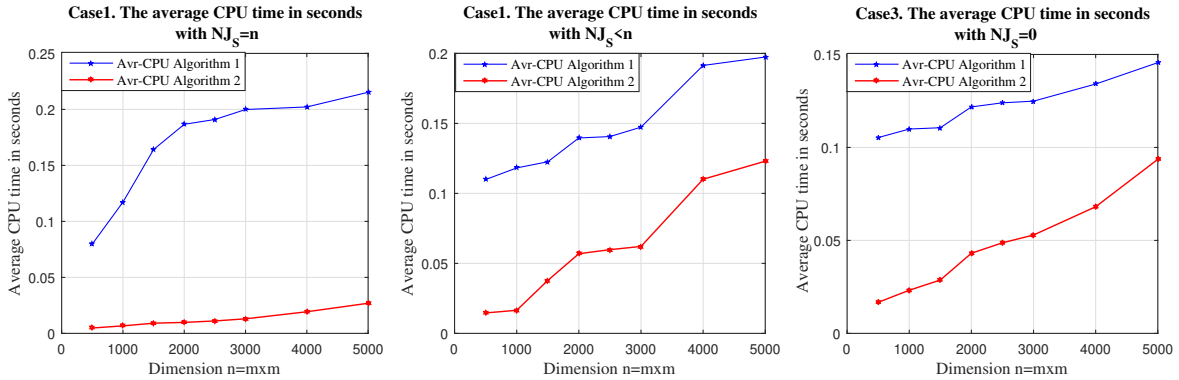

FIGURE 3.1 – Le temps moyen (Avr-CPU) en secondes, effectué par chaque algorithme pour l'exemple 1.

La FIGURE 3.1 ci-dessus présente une comparaison des temps moyens de traitement (Avr-CPU) entre l'algorithme proposé et l'algorithme de Luk et Pagano [69]. L'algorithme proposé démontre des temps de traitement nettement inférieurs par rapport à celui de Luk et Pagano, et ce, quel que soit le nombre d'éléments dans l'ensemble de support J_S de la fonction objectif à l'étape initiale de l'algorithme. Cette amélioration suggère une efficacité accrue dans le traitement des données et l'optimisation des calculs.

3.7.2 Exemple 2.

Dans ce deuxième exemple, la matrice D est choisie comme une matrice issue de l'approximation du Laplacien par la méthode des différences finies en 5-points [109] :

$$D = \begin{pmatrix} B & -I & 0 & \cdots & 0 \\ -I & B & -I & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -I & B & -I \\ 0 & \cdots & 0 & -I & B \end{pmatrix}_{m^2\times m^2}, \quad B = \begin{pmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 4 & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{pmatrix}_{m\times m}, \tag{3.14}$$

avec $n = m^2$ et I désigne la matrice identité de dimension m.

Nous générons le vecteur c de manière à constituer trois cas distincts de l'ensemble J_S, comme dans l'exemple précédent. Le critère de comparaison entre les deux méthodes repose toujours sur le temps moyen de traitement (Avr-CPU) en secondes et le nombre moyen d'ité-

rations (Avr-Iter) nécessaires pour atteindre la solution optimale du problème. Les tableaux ci-dessus résument les résultats obtenus.

Cas 1 : $NJ_S = n$.

Le vecteur c est généré de telle sorte que $NJ_S = m \times m = n$. Cela se traduit par la relation suivante pour chaque composante $i \in J$ de vecteur c :

$$c_i = 8 - 10\, r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{3.15}$$

Dimension	Algorithm1				Algorithm2			
$n = m \times m$	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
20×20	01	393	400	0.0768	0	400	400	0.0053
30×30	01	813	900	0.0791	0	900	900	0.0070
40×40	01	1222	1600	0.0801	0	1600	1600	0.0081
50×50	01	1603	2500	0.0891	0	2500	2500	0.0100
60×60	01	2387	3600	0.0960	0	3600	3600	0.0146
70×70	01	3207	4900	0.1320	0	4900	4900	0.0225

TABLE 3.4 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 2, avec $NJ_S = n$.

Pour toutes les dimensions n, Algorithm2 surpasse largement Algorithm1 en termes de temps moyen (Avr-CPU). Par exemple, pour une dimension de 20×20, Algorithm2 prend seulement 0.0053 secondes, contre 0.0768 secondes pour Algorithm1. Cette différence s'accroît avec la taille du problème. En termes de nombre d'itérations, Algorithm2 ne nécessite aucune itération (Avr-Iter = 0), car il commence avec une solution réalisable optimale dans ce cas, alors que l'Algorithm1 nécessite au moins une itération.

Cas 2 : $NJ_S < n$

Pour ce cas, le vecteur c est généré selon la relation suivante :

$$c_i = 8 - 16\, r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{3.16}$$

Dimension	Algorithm1				Algorithm2			
$n \times n$	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
20×20	12	198	380	0.1093	15	46	380	0.0179
30×30	12	441	882	0.1143	15	453	882	0.0233
40×40	15	849	1590	0.1199	17	895	1590	0.0283
50×50	16	1253	2498	0.1323	18	2464	2498	0.0353
60×60	19	1792	3522	0.1492	24	590	3522	0.0653
70×70	24	2360	4860	0.1734	25	2948	4890	0.0679

TABLE 3.5 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 2, avec $NJ_S < n$.

Ici aussi, l'Algorithm2 est plus rapide, même si l'écart avec l'Algorithm1 est réduit par rapport au cas 1. Pour une dimension de 20×20, l'Algorithm2 prend 0.0179 secondes, contre 0.1093 secondes pour Algorithm1.
Algorithm1 reste plus efficace en termes d'itérations, bien que l'avantage soit cette fois-ci modeste, nécessitant légèrement moins d'itérations que Algorithm2.

Cas 3 : $NJ_S = 0$

Dans ce cas, le vecteur c est généré de la manière suivante :

$$c_i = 8 - 20\, r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{3.17}$$

Dimension	Algorithm1				Algorithm2			
$n \times n$	Avr-Iter	NP	$\overline{NP}$	Avr-CPU	Avr-Iter	NJ_S	$\overline{NJ_S}$	Avr-CPU
20×20	04	154	219	0.1037	06	0	219	0.0141
30×30	05	339	506	0.1079	07	0	506	0.0204
40×40	05	64 7	937	0.1108	07	0	937	0.0259
50×50	04	899	1221	0.1139	05	0	1221	0.0306
60×60	04	1277	1740	0.1199	05	0	1740	0.0458
70×70	06	1891	2592	0.1401	07	0	2592	0.0628

TABLE 3.6 – Le temps moyen (Avr-CPU) en secondes et le nombre moyen d'itérations (Avr-Iter) obtenus pour la résolution de l'exemple 2, avec $NJ_S = 0$.

Dans ce cas, Algorithm2 conserve un avantage, même s'il devient plus modeste. Pour 20×20, Algorithm2 prend 0.0141 secondes, contre 0.1037secondes pour Algorithm1.
En ce qui concerne les itérations, Algorithm1 nécessite moins d'itérations (Avr-Iter), mais cela n'a pas de répercussion significative sur le temps moyen (Avr-CPU).

La figure ci-dessous illustre les performances des deux algorithmes sur un ensemble de tests, montrant une amélioration significative du temps de traitement avec l'algorithme proposé.

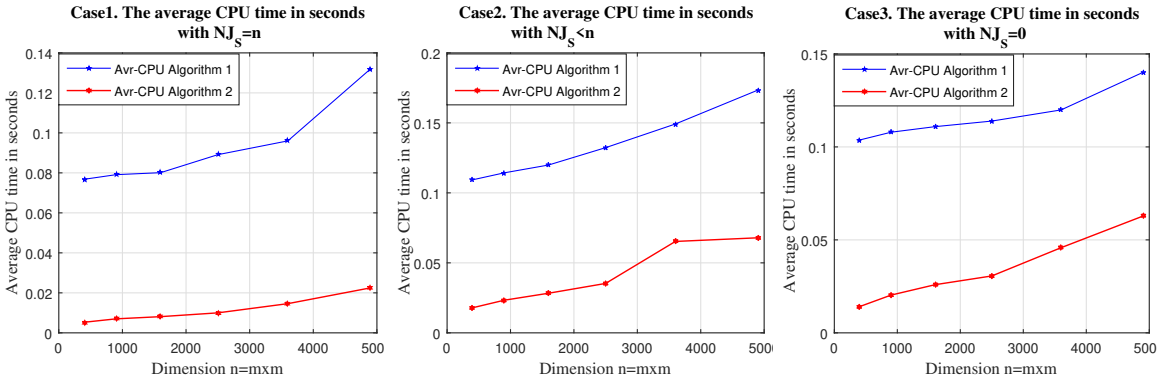

FIGURE 3.2 – Le temps moyen (Avr-CPU) en secondes, effectué par chaque algorithme dans l'exemple 2.

Globalement, l'Algorithm2 est clairement plus performant que l'Algorithm1 dans tous les cas testés, et cela est principalement dû à la stratégie d'initialisation plus efficace de l'Algorithm2. Cette initialisation permet de commencer avec une solution réalisable qui est souvent déjà optimale ou proche de l'optimum, ce qui réduit considérablement le nombre d'itérations et le temps moyen (Avr-CPU). L'efficacité de l'Algorithm2 est particulièrement remarquable dans les cas où $NJ_S = n$, où il ne nécessite aucune itération supplémentaire et converge directement vers une solution optimale. Il est également à noter que l'efficacité de l'Algorithm2 diminue légèrement dans les cas où NJ_S est strictement inférieur à n, mais il reste néanmoins plus performant que l'Algorithm1.

3.8 Conclusion

Les lemmes 3.2, 3.3 et 3.4, ainsi que l'inégalité (3.9), ont rendu possible le démarrage de l'algorithme avec une solution réalisable x satisfaisant les conditions du théorème 3.3, ainsi que l'inégalité (3.9). Lorsque la matrice associée à la fonction objectif est $D = I_n$, où I_n désigne la matrice identité, on obtient une partition de l'ensemble des indices J en fonction du support de la fonction objectif :

$$J_S = \{j \in J : c_j \leq 0\}, \quad J_N = \{j \in J : c_j > 0\}.$$

Ainsi, on retrouve les conditions d'initialisation de l'algorithme de Luk and Pagano [17, 69], avec le vecteur nul $x = 0$. Notons également que ces algorithmes se terminent avec J_N ou J_N^- vide, tandis que le nôtre aboutit systématiquement à $J_N^- = \emptyset$ et $J_N \neq \emptyset$, en raison de notre choix d'initialisation. En effet, le cas $J_N = \emptyset$ correspond à la solution optimale $x^0 = \hat{x}$.

Par ailleurs, les deux exemples numériques illustrent que notre approche est plus perfor-

mante en termes de temps de calcul que celle de Luk et Pagano. Ce constat demeure valable pour les deux cas étudiés, indépendamment du cardinal du support J_S de la fonction objectif à l'étape initiale de l'algorithme.

Chapitre 4

Problème de Programmation Quadratique avec une M-matrice à diagonale dominante et des contraintes de Bornes (Box-QP)

4.1 Introduction

Les Problèmes de Programmation Quadratique (PPQs) avec contraintes de bornes apparaissent très souvent dans la résolution des Équations aux Dérivées Partielles (EDP), en contrôle optimal, en recherche opérationnelle, et surtout comme sous-problèmes lors de la résolution de problèmes généraux d'optimisation non linéaire par des méthodes de programmation quadratique séquentielle [5, 72, 78]. Ces modèles sont également une partie importante des problèmes de gestion de portefeuille en finance [61, 88], ainsi qu'en statistiques avec la méthode des moindres carrés linéaires [70].

Notre principal objectif dans ce chapitre est de proposer une nouvelle méthode efficace pour résoudre les problèmes de programmation quadratique avec une M-matrice à diagonale dominante et des contraintes de bornes. Cette dernière est basée sur la technique de prétraitement, ainsi que sur un algorithme inspiré d'une méthode de Voglis et Lagaris [106]. Les techniques de prétraitement (ou de presolving) ont une longue histoire dans le domaine de la programmation linéaire et quadratique [2, 79, 91], où elles sont considérées comme faisant partie intégrante des logiciels de haute qualité pour ce type de problèmes. L'idée du presolving est d'utiliser des relations logiques simples entre les contraintes et de fixer certaines variables à l'optimum pour simplifier le problème autant que possible à faible coût, avant de transmettre le problème transformé résultant à un solveur efficace. Une fois que le problème transformé a été résolu, l'opération inverse est effectuée pour restaurer sa solution en termes de la formulation origi-

nale, une opération souvent appelée postsolving (ou post-traitement).

Le principe de notre approche est d'appliquer une technique de prétraitement, et ce, en utilisant les propriétés de la M-matrice diagonalement dominante afin de réduire la taille du problème original. Il s'agit d'identifier des indices actifs à l'optimum, afin de fixer à l'avance les valeurs correspondantes de certaines variables, et de placer certaines autres dans l'ensemble du support J_S. Ensuite, le problème réduit résultant sera résolu avec une méthode inspirée de Voglis et Lagaris [106]. En s'appuyant sur le concept de support pour une fonction objectif, tel que développé dans [14, 24, 36], notre approche diffère par une condition plus générale qui permet de déterminer une pseudo-solution initiale, associée au support coordinateur de la fonction objectif et proche de la solution optimale.

Ce chapitre est structuré comme suit : la deuxième section est consacrée à la position du problème et à la présentation de certaines notions clés relatives à notre approche. Dans la section 3, nous abordons les techniques de prétraitement que nous avons appliquées aux problèmes quadratiques comportant une M-matrice à diagonale dominante. La méthode de résolution avec support est détaillée dans la section 4. Quant à la section 5, elle propose un algorithme pour résoudre le Problème de Programmation Quadratique avec une M-matrice à diagonale dominante et des contraintes de Bornes (Box-QP). Enfin, nous concluons ce chapitre par une synthèse des résultats obtenus.

4.2 Position du problème et préliminaires

Considérons le problème de programmation quadratique avec une M-matrice à diagonale dominante et des contraintes de Borne (Box-QP) :

$$\min \quad F(x) \quad = \quad \frac{1}{2}x^T Dx - c^T x, \tag{4.1}$$

$$0 \leq x \leq ue, \tag{4.2}$$

où $u \in R,\ u > 0$; $c = c(J) = (c_j,\ j \in J)$, $e = (1, 1, \ldots, 1)^T \in R^n$ sont des n-vecteurs réels, avec $j \in J = \{1, 2, \cdots, n\}$. La matrice $D = D(J, J) = (d_{ij},\ 1 \leq\ i,\ j \leq n)$ est une M-matrice d'ordre n symétrique et diagonalement dominante [109], c'est-à-dire

$$d_{ii} > 0, \quad d_{ij} \leq 0,\ \ i \neq j, \quad D^{-1} \geq 0,\ \ et\ \ d_{ii} \geq \sum_{j \neq i} \mid d_{ij} \mid,\ \ i = \overline{1, n},$$

où le signe $D^{-1} \geq 0$ signifie que tous les coefficients de la matrice inverse D^{-1} sont non négatifs.

On note qu'une M-matrice symétrique est toujours définie positive ($x^T Dx > 0,\ \forall\ x \neq 0$). De plus, toute sous-matrice principale d'une M-matrice est elle-même une M-matrice.

Avant d'aller plus loin, on va d'abord rappeler quelques définitions de base pour mieux comprendre ce qui va suivre.

4.3 Définitions et propriétés

Définition 4.1.

— *Tout vecteur x satisfaisant la contrainte* $0 \leq x \leq ue$ *est appelé* solution réalisable *ou* plan *du problème Box-QP. Une solution réalisable* x^0 *est dite* optimale *si*

$$F(x^0) = \frac{1}{2} x^{0^T} Dx^0 - c^T x^0 = \min_x F(x),$$

où x est pris parmi toutes les solutions réalisables du problème.

— *Pour un nombre* $\epsilon > 0$ *donné, une solution réalisable* x^ϵ *est dite* $\epsilon-$*optimale ou suboptimale si*

$$F(x^\epsilon) - F(x^0) \leq \epsilon,$$

où x^0 *est une solution optimale du problème Box-QP.*

— *Si x est une solution réalisable du problème Box-QP, alors le vecteur*

$$E = E(x) = Dx - c,$$

est le gradient de F au point x, également appelé vecteur des estimations.

Théorème 4.1. *[37]* ***Critère d'optimalité.***
Pour qu'une solution réalisable x^0 *du problème Box-QP soit optimale, il faut et il suffit que les conditions suivantes soient satisfaites pour tout* $j \in J$ *:*

$$\begin{cases} E_j(x^0) \geq 0, & pour \quad x_j^0 = 0, \\ E_j(x^0) \leq 0, & pour \quad x_j^0 = u, \\ E_j(x^0) = 0, & pour \quad 0 < x_j^0 < u, \quad j \in J. \end{cases} \tag{4.3}$$

Notre nouvelle approche initialement introduite en [46], développée et prouvée rigoureusement dans [48], pour résoudre un problème de programmation quadratique avec une M-matrice diagonalement dominante Box-QP, s'articule en deux étapes. La première consiste à appliquer une technique de prétraitement (presolving) afin de réduire la taille du problème initial. Ensuite, le problème réduit ainsi obtenu est résolu à l'aide d'une méthode inspirée du travail de Voglis

et Lagaris [106]. En introduisant le concept de support d'une fonction objectif développé par Gabasov et al. [36, 37], notre approche se distingue par une condition plus générale qui permet de générer une pseudo-solution initiale, liée au support coordinateur, et proche de la solution optimale finale, ce qui améliore la convergence de l'algorithme et réduit le nombre d'itérations nécessaires pour atteindre une solution optimale.

4.4 Technique de prétraitement

Avant de présenter un algorithme de résolution du problème de programmation quadratique Box-QP, nous utiliserons une technique de prétraitement permettant d'identifier certains indices actifs à l'optimum, afin de fixer à l'avance les variables correspondantes, réduisant ainsi la taille du problème. Nous recherchons également d'autres indices de variables pour lesquels les composantes correspondantes du gradient seront nulles à l'optimum. Ce processus est connu sous le nom de procédure de prétraitement ou de presolving. Dans cette optique, nous allons définir les matrices et vecteurs nécessaires pour mettre en œuvre cette procédure.

En effet, à partir de la matrice D, définissons les deux matrices suivantes :

$$D^+ = diag(d_{ii}, \quad i = \overline{1,n}), \qquad D^- = D - D^+ = \begin{cases} 0 & Si \ \ i = j, \\ d_{ij} & Si \ \ i \neq j, \end{cases}$$

et calculons également les vecteurs :

$$y^+ = (y_i^+, i \in J) = uD^+e \quad \Leftrightarrow \quad y_i^+ = ud_{ii} > 0, \ \ i = \overline{1,n}; \tag{4.4}$$

$$y^- = (y_i^-, i \in J) = uD^-e \quad \Leftrightarrow \quad y_i^- = -u \sum_{j \neq i} \mid d_{ij} \mid, \ \ i = \overline{1,n}. \tag{4.5}$$

Pour toute solution réalisable x du problème Box-QP, nous avons la double inégalité :

$$y^- - c \leq E \leq y^+ - c. \tag{4.6}$$

En particulier, nous déduisons pour la solution optimale x^* :

$$y^- - c \leq E^* \leq y^+ - c, \tag{4.7}$$

où $E^* = g^* = Dx^* - c$.

Le prétraitement vise à réduire la taille du problème de programmation quadratique convexe Box-QP en suivant les étapes suivantes :

- **La première étape :** Le lemme suivant nous permet de fixer certaines composantes de la solution optimale x^* à leurs bornes.

  Lemme 4.1. *Soit $i \in J = \{1, 2, \cdots, n\}$. On a alors :*
  (1) Si $c_i < y_i^- \Rightarrow x_i^* = 0$;
  (2) Si $c_i > y_i^+ \Rightarrow x_i^* = u$.

  Preuve. Ceci découle des relations (4.7) et des conditions d'optimalité (4.3). ■

- **La deuxième étape :** Ce second lemme nous permet d'exclure la possibilité qu'une composante x_i^* puisse être à une certaine borne.

  Lemme 4.2. *Soit $i \in J = \{1, 2, \cdots, n\}$. On a alors :*
  (1) Si $c_i > 0 \implies x_i^* > 0$;
  (2) Si $c_i < y_i^+ + y_i^- = u(d_{ii} - \sum_{i \neq j} \mid d_{ij} \mid) \implies x_i^* < u$.

  Preuve.
  (1) Supposons que pour un certain indice $i \in J$ nous avons $x_i^* = 0$. D'après les conditions d'optimalité (4.3), nous en déduisons que $E_i^* \geq 0$. Cependant, d'un autre côté, nous avons :

  $$E_i^* = d_{ii}x_i^* + \sum_{j \neq i} d_{ij}x_j^* - c_i = \sum_{j \neq i} d_{ij}x_j^* - c_i \leq -c_i < 0,$$

  ce qui contredit les conditions d'optimalité (4.3). Par conséquent, nous en déduisons que $x_i^* > 0$.
  (2) Supposons que pour un certain indice $i \in J$, nous avons $x_i^* = u$. Alors, nous avons $E_i^* \leq 0$. Cependant, d'un autre côté, nous avons :

  $$\begin{aligned} E_i^* &= d_{ii}x_i^* + \sum_{j \neq i} d_{ij}x_j^* - c_i, \\ E_i^* &\geq d_{ii}u - u \sum_{j \neq i} \mid d_{ij} \mid -c_i, \\ E_i^* &\geq y_i^+ + y_i^- - c_i > 0. \end{aligned}$$

  La dernière inégalité stricte contredit les conditions d'optimalité (4.3). Par conséquent, nous en concluons que $x_i^* < u$. ■

- **La troisième étape :** ce troisième lemme nous permet de prédire que pour un certain indice $i \in J$, l'estimation E_i^* du vecteur optimal x^* doit être égal à zéro.

  Lemme 4.3. *Soit $i \in J = \{1, 2, \cdots, n\}$. Si*

  $$0 < c_i < u(d_{ii} - \sum_{j \neq i} \mid d_{ij} \mid) = y_i^+ + y_i^-,$$

il en résulte alors que

$$E_i^* = 0.$$

Preuve. L'application du lemme 4.2 implique que $0 < x_i^* < u$, et les conditions d'optimalité (4.3) donnent $E_i^* = 0$. ■

4.4.1 Procédure de prétraitement (presolving)

Dans cette section, nous présentons l'algorithme de prétraitement (presolving). En se basant sur les lemmes précédents 4.1-4.3, qui fournissent des résultats importants, cet algorithme décrit la procédure de prétraitement à appliquer au Box-QP. Son objectif est de réduire la taille du problème initial et de commencer par un bon support , et ce, afin de simplifier sa résolution. La procédure de prétraitement est présentée par l'algorithme suivant :

Algorithm 2 Procédure de prétraitement

1: **Entrée :** Matrice D, vecteur c et le scalaire u.

2: **Sortie :** Le problème réduit.

3: **Début**

(1) Calculons les vecteurs y^+ et y^- comme suit :

$$\begin{aligned} y^+ &= (y_i^+, i \in J) = uD^+e \quad \Leftrightarrow \quad y_i^+ = ud_{ii} > 0, \quad i = \overline{1,n}; \\ y^- &= (y_i^-, i \in J) = uD^-e \quad \Leftrightarrow \quad y_i^- = -u \sum_{j \neq i} |d_{ij}|, \quad i = \overline{1,n}. \end{aligned}$$

(2) Définissons les sous-ensembles suivants J_0 et J_u de J :

$$\begin{aligned} J_0 &= \{i \in J : c_i \leq y_i^-\}; \\ J_u &= \{i \in J : c_i \geq y_i^+\}; \\ \overline{J} &= J \setminus (J_0 \cup J_u). \end{aligned}$$

(3) Selon cette partition, fixons

$$x_i^* = \begin{cases} x_0 = x(J_0) = 0, \\ x_u = x(J_u) = ue(J_u), \end{cases}$$

et le vecteur c sera égal au vecteur

$$\overline{c}(\overline{J}) = c(\overline{J}) + D(\overline{J}, (J_0 \cup J_u))x^*(J_0 \cup J_u).$$

(4) Définissons les données du problème réduit en posant les relations suivantes :

$$J := \overline{J}, \quad n := |\overline{J}|, \quad c(J) := \overline{c}(\overline{J}), \quad D(J,J) := D(\overline{J}, \overline{J}), \quad x_r = x(J) := x(\overline{J}).$$

4: **Fin.**

Cette procédure de presolving sera intégrée dans notre algorithme proposé ci-dessous afin d'obtenir une bonne initialisation de la solution réalisable initiale. Par conséquent, cela réduira la taille du problème, tout en fournissant un bon support initial J_S, ce qui contribuera à diminuer sa complexité, à accélérer sa convergence et à améliorer ses performances.

4.5 Méthode de support

Afin de développer un algorithme de résolution pour le problème Box-QP, nous introduisons la notion de support de Gabasov et al. [36, 37], liée à la non-linéarité de la fonction objectif F du problème Box-QP. Notre algorithme se distingue par une condition plus générale,

permettant d'avoir une pseudo-solution initiale associée à un support coordinateur et proche de la solution optimale.

Définition 4.2.

— *Un sous-ensemble $J_S \subset J$ est dit* support de la fonction objectif *F, puisque la condition suivante est toujours vérifie :* $\det D_S = \det D(J_S, J_S) \neq 0$.

— *La paire $\{x, J_S\}$, constituée d'une solution réalisable x et d'un support J_S est dite Solution Réalisable de Support (SRS).*

— *La SRS $\{x, J_S\}$ est dite accordée si $E(J_S) = 0$.*

À partir d'une solution réalisable accordée $\{x, J_S\}$, une itération de l'algorithme consiste à construire une nouvelle solution réalisable de support accordée $\{\overline{x}, \overline{J_S}\}$ telle que $F(\overline{x}) \leq F(x)$. Cette construction se fait en modifiant la solution x ainsi que le support J_S de manière à améliorer la valeur de la fonction objectif F. Le but est de garantir que chaque itération produit une nouvelle solution qui est réalisable, et aussi bonne, sinon meilleure, que la précédente en termes de la valeur de la fonction objectif F. Ce processus itératif se poursuit jusqu'à ce que l'algorithme converge vers une solution optimale.

Selon la partition de $J = J_S \cup J_N$, avec $J_N = J \setminus J_S$, nous pouvons diviser la matrice et les vecteurs en blocs comme suit :

$$D = \begin{pmatrix} D_S & D_{SN} \\ D_{NS} & D_N \end{pmatrix}, \text{ avec } \quad D_S = D(J_S, J_S), \quad D_{SN} = D(J_S, J_N), \quad D_N = D(J_N, J_N);$$

$$c = c(J) = \begin{pmatrix} c_S \\ \hline c_N \end{pmatrix}, \quad c_S = c(J_S) = (c_j,\ j \in J_S), \quad c_N = c(J_N) = (c_j,\ j \in J_N);$$

$$x = x(J) = \begin{pmatrix} x_S \\ \hline x_N \end{pmatrix}, \quad x_S = x(J_S) = (x_j,\ \ j \in J_S), \quad x_N = x(J_N) = (x_j,\ \ j \in J_N);$$

$$g(J) = E(J) = \begin{pmatrix} E_S \\ \hline E_N \end{pmatrix}, \quad E_S = E(J_S) = (E_j,\ j \in J_S), \quad E_N = E(J_N) = (E_j,\ j \in J_N).$$

Définition 4.3.

— *Soit $J_N = J_{N^+} \cup J_{N^-}$, avec $J_{N^+} \cap J_{N^-} = \emptyset$ et $\kappa = \kappa(J) = \big(\kappa(J_{N^+}), \kappa(J_{N^-}), \kappa(J_S)\big)$ un vecteur*

satisfaisant

$$\begin{cases} \kappa_{N^+} = \kappa(J_{N^+}) = 0, \\ \kappa_{N^-} = \kappa(J_{N^-}) = ue(J_{N^-}), \\ \kappa_S = \kappa(J_S) = -D_S^{-1}[D_{SN}\kappa_N - c_S]. \end{cases} \tag{4.8}$$

Un vecteur κ satisfaisant (4.8) est appelé pseudo-solution *du problème Box-QP.*

— *D'après (4.8) une pseudo-solution κ vérifie $E_S(\kappa) = 0$. Si en plus, κ vérifie l'inégalité $0 \leq \kappa(J_S) \leq ue(J_S)$, alors κ est une solution réalisable.*

— *Le triplet $J_p = \{J_S, J_{N^+}, J_{N^-}\}$ est alors appelé* support *du problème Box-QP.*

— *Le triplet $J_p = \{J_S, J_{N^+}, J_{N^-}\}$ est appelé* support coordinateur *du problème Box-QP s'il existe une pseudo-solution κ telle que*

$$E_j(\kappa) \geq 0, \qquad j \in J_{N^+}, \tag{4.9}$$

$$E_j(\kappa) \leq 0, \qquad j \in J_{N^-}. \tag{4.10}$$

On dit dans ce cas, que la pseudo-solution κ est associée au support coordinateur J_P.

Théorème 4.2. *Une pseudo-solution κ liée à un support coordinateur J_p est optimale si et seulement si*

$$0 \leq \kappa_j \leq u, \qquad \forall \ j \in J_S. \tag{4.11}$$

Preuve. Le vecteur κ est une solution réalisable et vérifie le critère d'optimalité (4.3). ■

4.5.1 Le problème dual et ses propriétés

Le dual du problème primaire Box-QP est formulé comme un problème quadratique concave de la façon suivante :

$$\begin{cases} \max \varphi(\lambda) = -\frac{1}{2}\kappa^T D\kappa - uw^T e, \\ D\kappa + c - v + w = 0, \\ \lambda = (\kappa, v, w), \ \ \kappa \in \mathbb{R}^n, \ v \geq 0, \ w \geq 0. \end{cases} \tag{4.12}$$

Pour le problème dual, introduisons les définitions suivantes.

Définition 4.4.

— *Le triplet $\lambda = (\kappa, v, w)$ vérifiant toutes les contraintes du problème (4.12) est appelé une* solution réalisable duale.

— *Soit $J_P = \{J_S, J_{N^+}, J_{N^-}\}$ un support coordinateur du problème (4.12), associé à une*

pseudo-solution κ. *Alors le vecteur* $\lambda = (\kappa, v, w)$ *défini par*

$$\begin{cases} v_j = E_j(\kappa), & w_j = 0 & j \in J_{N+}, \\ v_j = 0, & w_j = -E_j(\kappa) & j \in J_{N-}, \\ v_j = w_j = 0, & & j \in J_S, \end{cases}$$

est appelée solution réalisable accordée *du problème dual (4.12).*

Une solution réalisable duale accordée possède des propriétés très intéressantes, qui sont présentées dans le lemme suivant.

Lemme 4.4. *[14] Pour toute solution duale réalisable* $\lambda = (\kappa, v, w)$*, il existe une solution duale réalisable accordée* $\overline{\lambda} = (\overline{\kappa}, \overline{v}, \overline{w})$*, vérifiant :*

$$\varphi(\lambda) \leq \varphi(\overline{\lambda}). \tag{4.13}$$

D'après ce lemme et le principe de la dualité en optimisation quadratique convexe, la double inégalité est satisfaite :

$$\varphi(\lambda) \leq F(x^*) \leq F(x),$$

où λ est n'importe quelle solution réalisable duale accordée dans le problème (4.12) et x est une solution réalisable pour le problème primal Box-QP. Par conséquent, nous nous intéressons dorénavant qu'aux solutions réalisables duales accordées.

Lemme 4.5. *[36] Si une solution duale réalisable accordée* $\lambda = (\kappa, v, w)$ *est associée à un support coordinateur* J_P*, alors la relation suivante est vérifiée :*

$$\varphi(\lambda) = F(\kappa). \tag{4.14}$$

Dans ce cas, le nombre $\beta(x, J_P) = F(x) - F(\kappa)$ *est appelé estimation de suboptimalité, puisque nous avons toujours* $F(x) - F(x^*) \leq \beta(x, J_P)$*, où* x^* *est une solution optimale du problème Box-QP. Par conséquent, si* $\beta(x, J_P) \leq \epsilon$*, alors* x *est une solution* $\epsilon-$*optimale,* $\epsilon \geq 0$ *étant une précision donnée.*

En exploitant les propriétés précédemment établies, nous élaborons une méthode numérique efficace permettant de résoudre le problème d'optimisation quadratique avec contraintes de bornes (désigné par Box-QP). Cette approche repose essentiellement sur la structure particulière du problème.

4.5.2 Construction d'une Solution Réalisable de Support (SRS) initiale

Considérons le problème réduit obtenu par l'application de la procédure de presolving. Construisons la partition suivante $J = \{J_S, J_{N^+}, J_{N^-}\}$, avec

$$\begin{aligned} J_{N^+} &= \{i \in J : c_i \leq 0\}, & J_{N^-} &= \{i \in J : c_i \geq y_i^+ + y_i^-\}, \\ J_S &= \{i \in J : 0 < c_i < y_i^+ + y_i^-\}, & J_N &= J \setminus J_S = J_{N^+} \cup J_{N^-}. \end{aligned}$$

En utilisant les lemmes 4.2 et 4.3, la solution réalisable de support initiale $\{x, J_S\}$ à prendre est le vecteur $x = x(J) = (x(J_{N^+}), x(J_{N^-}), x(J_S))$, tel que

$$\begin{cases} x_{N^+} = x(J_{N^+}) = 0, \\ x_{N^-} = x(J_{N^-}) = ue(J_{N^-}) = ue_{N^-}, \\ x_S = x(J_S) = -D_S^{-1}[D_{SN}x_N - c_S], \qquad \text{tels que} \quad E_S(x) = 0, \end{cases} \tag{4.15}$$

où x_S est déduit de l'équation :

$$E_S(x) = D(J_S, J)x - c_S = D(J_S, J_S)x_S + D(J_S, J_N)x_N - c_S = 0,$$

c'est à dire

$$D_S x_S = -D_{SN}x_N + c_S \Longrightarrow x_S = D_S^{-1}(-D_{SN}x_N + c_S). \tag{4.16}$$

Puisque $-D_{SN}x_N \geq 0$, $c_S > 0$, $D_S^{-1} \geq 0$ et aucune ligne de D_S^{-1} n'est nulle, on en déduit que $x_i > 0$, $\forall\ i \in J_S$, et donc $x_S > 0$.

Nous montrons également que $x_S < ue_S$. En effet, d'après (4.16) on peut écrire

$$\alpha = D_S(x_S - ue_S) = D_S x_S - uD_S e_S = -D_{SN}x_N + c_S - uD_S e_S,$$

où la i-ième composante de α est égale à

$$\begin{aligned} \alpha_i &= -\sum_{j \in J_N} d_{ij}x_j + c_i - u\sum_{j \in J_S} d_{ij}, \quad i \in J_S, \\ \alpha_i &= -\sum_{j \in J_N} d_{ij}x_j + c_i - ud_{ii} - u\sum_{j \neq i, j \in J_S} d_{ij}, \quad i \in J_S. \end{aligned}$$

Puisque $0 < c_i < u(d_{ii} - \sum_{j \neq i} \mid d_{ij})$, cela implique que pour tout $i \in J_S$, on a :

$$\alpha_i < -\sum_{j \in J_N} d_{ij}x_j + ud_{ii} - u\sum_{j \neq i} \mid d_{ij} \mid -ud_{ii} - u\sum_{j \neq i, j \in J_S} d_{ij} \leq u\sum_{j \neq i, j \in J_N \cup J_S} \mid d_{ij} \mid -u\sum_{j \neq i, j \in J} \mid d_{ij} \mid = 0.$$

D'où $\alpha_i < 0$, $\forall i \in J_S$ *et* $x_S - ue_S = D_S^{-1}\alpha$. Puisque $D_S^{-1} \geq 0$ et qu'aucune ligne de D_S^{-1} n'est

nulle, il s'ensuit que

$$x_S - ue_S < 0 \Longrightarrow x_S < ue_S.$$

Par conséquent, le vecteur $x = (x_S, x_{N^+}, x_{N^-})$ construit selon (4.15) est une solution réalisable et J_S est un *support* de la fonction objectif F du problème Box-QP.

L'algorithme suivant nous permet alors de construire une solution réalisable de support accordée initiale, qui sert de point de départ pour les itérations ultérieures, et qui garantit que les contraintes du problème sont préalablement respectées. En établissant un point de départ fiable, nous facilitons la convergence vers la solution optimale, rendant ainsi le processus d'optimisation plus efficace.

Algorithm 3 Construction d'une SRS accordée $\{x_r, J_S\}$

1: **Entrées :** Le problème réduit $D,\ c,\ u$.

2: **Sorties :** Une SRS accordée $\{x_r, J_S\}$.

3: **Début**

4: **Étape 1 :** Définir les ensembles J_{N^+}, J_{N^-} et J_S comme suit

$$\begin{aligned} J_{N^+} &= \{i \in J : c_i \leq 0\}, & J_{N^-} &= \{i \in J : c_i \geq y_i^+ + y_i^-\}, \\ J_S &= \{i \in J : 0 < c_i < y_i^+ + y_i^-\}, & J_N &= J \setminus J_S = J_{N^+} \cup J_{N^-}. \end{aligned}$$

5: **Étape 2 :** Construire une solution réalisable de support $\{x_r, J_S\}$ comme suit :

$$\begin{cases} x_r(J_{N^+}) = 0, \\ x_r(J_{N^-}) = ue(J_{N^-}) = ue_{N^-}, \\ x_r(J_S) = -D_S^{-1}[D_{SN}x_N - c_S], \quad \text{tel que} \quad E_S(x_r) = 0. \end{cases}$$

6: **Fin.**

4.6 Méthode de résolution d'un problème Box-QP

Le principe de l'algorithme est d'appliquer la procédure de prétraitement pour réduire le problème original. Ensuite, le problème réduit résultant sera résolu par la méthode Box-QP. Cette dernière s'inspire d'une approche basée sur la méthode de Voglis et Lagaris [106], qui entre dans la catégorie des méthodes de points extérieurs ; de plus, elle utilise la notion de support pour la fonction objectif [36]. Après la résolution du problème réduit, une étape de post-traitement sera nécessaire pour restituer la solution du problème original. Cette étape est cruciale, car elle consiste à analyser et interpréter les résultats obtenus et de les adapter au contexte du problème de départ.

4.6.1 Schéma de l'algorithme

Algorithm 4 Algorithme du Box-QP

Entrée : Le problème réduit D, c, u et une SRS accordée initiale $\{x_r, J_S\}$.

Sortie : La solution optimale du problème réduit $\{x_r^*, J_S^*\}$.

Début

Étape 0 : Initialiser $k := 0$ et poser $\{x_r, J_S\} = \{\kappa^{(0)}, J^{(0)}\}$.

if le critère d'optimalité (4.3) est vérifié **then**

  Arrêter. La solution $x_r(J_S) = \kappa^{(0)}(J^{(0)}) = x_r^*(J_S^*)$, obtenue par l'algorithme 3, est la solution optimale du problème réduit.

else

  Étape 1 : À l'itération k, redéfinir les ensembles $J_{N^+}^{(k)}$, $J_{N^-}^{(k)}$ et $J_S^{(k)}$ de la manière suivante :

$$
\begin{aligned}
J_{N^+}^{(k)} &= \{j \in J^{(k)} : \kappa_j^{(k)} < 0 \text{ ou } \kappa_j^{(k)} = 0 \text{ et } E_j^{(k)} \geq 0\},\\
J_{N^-}^{(k)} &= \{j \in J^{(k)} : \kappa_j^{(k)} > u_j \text{ ou } \kappa_j^{(k)} = u_j \text{ et } E_j^{(k)} \leq 0\},\\
J_S^{(k)} &= \{j \in J^{(k)} : 0 < \kappa_j^{(k)} < u_j \text{ ou } \kappa_j^{(k)} = 0 \text{ et } E_j^{(k)} < 0,\\
&\qquad \text{ou } \kappa_j^{(k)} = u_j \text{ et } E_j^{(k)} > 0\},\\
J_N^{(k)} &= J_{N^+}^{(k)} \cup J_{N^-}^{(k)}, \quad J^{(k)} = J_N^{(k)} \cup J_S^{(k)}.
\end{aligned}
$$

  Étape 2 : Construire la nouvelle pseudo-solution $\kappa^{(k+1)}$ comme suit

$$
\begin{cases}
\kappa_j^{(k+1)} &= 0, \quad j \in J_{N^+}^{(k)},\\
\kappa_j^{(k+1)} &= u_j, \quad j \in J_{N^-}^{(k)},\\
\kappa_S^{(k+1)} &= D^{-1}(J_S^{(k)}, J_S^{(k)}) \left[c(J_S^{(k)}) - D(J_S^{(k)}, J_N^{(k)})\kappa^{(k)}(J_N^{(k)})\right].
\end{cases}
$$

  Étape 3 : Calculer le gradient ou le vecteur des estimations

$$
E_N^{(k+1)} = D(J_N^{(k)}, J_S^{(k)})\kappa_S^{(k+1)} + D(J_N^{(k)}, J_N^{(k)})\kappa_N^{(k+1)} - c(J_N^{(k)}).
$$

  Étape 4 : Vérifier si la nouvelle pseudo-solution est optimale.

  if $0 \leq \kappa_j^{(k+1)} \leq u_j,\ \forall j \in J_S^{(k)},\ E_j^{(k+1)} \geq 0,\ \forall j \in J_{N^+}^{(k)},$ et $E_j^{(k+1)} \leq 0,\ \forall j \in J_{N^-}^{(k)}$ **then**

    Arrêter. La solution optimale est $x_r^* := \kappa^{(k+1)}$ et $J_S^* := J^{(k)}$.

  else

    Poser $k := k + 1$ et aller à l'Étape 1.

Fin.

Enfin, une étape de post-traitement est nécessaire pour interpréter correctement la solution du problème réduit et en déduire la solution du problème originel. Cette étape consiste à réintroduire les variables ou contraintes qui ont été éliminées lors du prétraitement, permettant

ainsi de reconstruire la solution complète du problème initial.

Algorithm 5 Procédure de post-traitement

Entrées : Le problème initial D, c, u.

Sorties : La solution optimale x^* du problème Box-QP et $F(x^*)$.

Début

Étape 1 : Utiliser la procédure de prétraitement pour déduire le problème réduit.

Étape 2 : Après avoir déterminé les ensembles J_{N^+}, J_{N^-}, J_S, construire une SRS initiale accordée $\{x_r, J_S\}$ par l'algorithme (3).

Étape 3 : Résoudre le problème réduit par l'algorithme Box-QP, avec x_r^* comme solution optimale obtenue.

Étape 4 : Déduction de la solution optimale du problème originel (post-traitement) :

$x^* = x^*(J) = (x_j^*, j \in J) = x^*(J_0 \cup J_u \cup \overline{J})$, avec :

$$x_i^* = \begin{cases} x_0 = x(J_0) = 0, \\ x_u = x(J_u) = ue(J_u), \\ x^*(\overline{J}) = x_r^*. \end{cases}$$

Fin.

4.6.2 Exemple illustratif

Pour illustrer la méthode développée précédemment, nous considérons le problème de programmation quadratique (4.1) avec :

$$D = \begin{pmatrix} 4 & -1 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & -1 & 4 \end{pmatrix}, \; c = \begin{pmatrix} -5 \\ 4 \\ -11 \\ 10 \\ 1 \\ 2 \end{pmatrix}, \; l = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \; ue = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{pmatrix}.$$

Les itérations de l'algorithme sont les suivantes :

1. **Prétraitement :** on a $y^- = -3(1,2,2,2,2,1)^T$, $y^+ = 12(1,1,1,1,1,1)^T$, alors $J_0 = \{1,3\}$, $J_u = \emptyset$, $\overline{J} = J \setminus J_0 \cup J_u = \{2,4,5,6\}$. Donc $x_0^* = (0,0)^T$. Le vecteur $\overline{c}(\overline{J}) = (4,10,1,2)^T$. Le problème réduit est le suivant $J := \overline{J} = \{2,4,5,6\}$, $n :=\mid \overline{J} \mid = 4$, avec

$$D = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix},\ c = \begin{pmatrix} 4 \\ 10 \\ 1 \\ 2 \end{pmatrix},\ l = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},\ ue = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix}, x_r(J) = \begin{pmatrix} x_2 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix};$$

2. Construire une SRS initiale accordée $\{x, J_S\}$ par l'Algorithme 3.
$J_{N^+} = \emptyset,\ J_{N^-} = \{4\},\ J_S = \{2,5,6\},\ J_N = \{4\},\ donc\ x_{N^-} = x_4 = 3,\ x_S = (x_2, x_5, x_6)^T = (1, 1.2, 0.8)^T$.

3. Algorithme Box-QP : Résoudre le problème réduit.

Étape 0. Poser $k = 0$ et $\{x, J_S\} = \{\kappa^{(0)}, J_S^{(0)}\}$, $J_S^{(0)} = \{2,5,6\}$, $J_{N^-}^{(0)} = \{4\}$ avec $\kappa_S^{(0)} = (1, 1.2, 0.8)^T$,
$\kappa_{N^-}^{(0)} = \kappa_4^{(0)} = 3$. $E_S^{(0)} = (0,0,0)^T$ et $E_{N^-}^{(0)} = 0.8$. Le critère d'optimalité n'est pas satisfait pour $j \in J_{N^-}^{(0)}$.

Étape 1. Poser $k = 1$. Redéfinir les ensembles $J_{N^+}^{(1)}$, $J_{N^-}^{(1)}$, et $J_S^{(1)}$.
$J_{N^+}^{(1)} = \emptyset,\ J_{N^-}^{(1)} = \emptyset,\ J_S^{(1)} = \{2,4,5,6\}$.

Étape 2. La nouvelle pseudo-solution est donc la suivante :
$\kappa_S^{(1)} = (1, 2.7857, 1.1429, 0.7857)^T = (1, \frac{13909}{4993}, \frac{3807}{3331}, \frac{3923}{4993})^T$.

Étape 3. L'ensemble $J_N^{(0)} = \emptyset$. Donc on ne peut pas calculer $E_N^{(0)}$.

Étape 4. La nouvelle pseudo-solution $\kappa_S^{(1)}$ satisfait le critère d'optimalité, elle est donc optimale pour le problème réduit $\kappa^{(1)} = x^* = x_r^*(\overline{J})$.

4. Post-traitement : Déduction de la solution optimale du problème originel. Poser $J := J_0 \cup J_u \cup \overline{J} = \{1,3\} \cup \{2,4,5,6\}$ et la solution optimale vaut :
$x^* = (x_0^*, x_u^*, x_r^*) = (0, 1, 0, 2.7857, 1.1429, 0.7857)^T = (0, 1, 0, \frac{13909}{4993}, \frac{3807}{3331}, \frac{3923}{4993})^T$, avec $F(x^*) = -17.2857 = \frac{-121}{7}$.

4.7 Conclusion

Dans ce chapitre, nous avons développé une méthode pour résoudre un problème PQ avec une M-matrice à diagonale dominante et des contraintes de bornes. Cette méthode inclut une procédure de prétraitement qui exploite de manière spécifique la structure de la matrice D. Grâce à cette propriété, la procédure de prétraitement simplifie le problème initial en réduisant la taille de la matrice à traiter, ce qui améliore l'efficacité computationnelle. Cette étape préliminaire permet également de diminuer la complexité globale du problème, rendant ainsi la résolution plus rapide et plus pratique.

L'algorithme d'optimisation pour le problème réduit qui en résulte est conçu en s'inspirant d'une méthode des points extérieurs [106] et en intégrant le concept de support pour la fonction objectif [14, 33, 36, 37]. Cette approche permet d'optimiser efficacement le problème en

exploitant la structure de la solution, ce qui peut améliorer la convergence et réduire le temps de calcul.

Dans le cadre de nos travaux futurs, nous mettrons en œuvre l'algorithme proposé afin d'en évaluer les performances de calcul et aussi en le comparant à d'autres méthodes existantes. Cette comparaison se concentrera sur des critères tels que le temps d'exécution, l'utilisation de la mémoire et la robustesse. Nous évaluerons également la capacité de l'algorithme à résoudre des problèmes de plus grande taille. L'objectif est d'identifier les forces et les limites de notre approche par rapport aux méthodes existantes, dans le but d'améliorer ses performances pour des applications pratiques.

Conclusion et perspectives

Les travaux réalisés dans cette thèse se sont concentrés sur l'élaboration d'une méthode de résolution efficace pour un Problème de Programmation Quadratique (PPQ) avec des contraintes simples ou de bornes, en mettant en avant les cas impliquant une M-matrice. Ces matrices, grâce à leurs propriétés spécifiques, jouent un rôle clé dans des domaines tels que l'optimisation et l'analyse numérique.

Dans le premier chapitre, nous avons posé les bases théoriques nécessaires en abordant les notions de convexité et d'analyse convexe, les propriétés des ensembles et fonctions convexes, ainsi que les caractéristiques des formes quadratiques, notamment leur gradient et leur Hessien. Nous avons également étudié les M-matrices, en détaillant leur structure et leur importance dans la résolution des systèmes linéaires et des problèmes d'optimisation. Enfin, nous avons exploré les principes de l'optimisation convexe et de la programmation non linéaire, en analysant les conditions d'optimalité pour des problèmes avec ou sans contraintes. Ces rappels théoriques forment une base solide pour développer notre méthode de résolution dans les chapitres suivants.

Concernant le deuxième chapitre, nous avons présenté deux exemples concrets où la formulation conduit à des Problèmes de programmation quadratique (PPQs) impliquant des M-matrices. Le premier est le "Circus Tent Problem", un problème bien connu lié à l'optimisation de la toile d'une tente de cirque, et le second concerne la valorisation des options américaines, un problème financier complexe. Ces exemples illustrent comment les M-matrices émergent dans des situations réelles et montrent leur importance pour modéliser et résoudre des problèmes pratiques.

L'une des contributions majeures de ce travail, présentée dans le chapitre 3, réside dans la proposition d'une nouvelle méthode de résolution d'un PPQ, avec une M-matrice et des contraintes simples. En tirant profit de la propriété des M-matrices, dont l'inverse est non négative, notre approche génère une suite monotone de solutions réalisables [95, 98, 96]. En intégrant le concept de support, développé par Gabasov et al. [36, 37], notre méthode [47] se distingue par une condition plus générale qui fournit une solution initiale réalisable plus proche

de l'optimum. Cette caractéristique améliore la vitesse de convergence et réduit le nombre d'itérations par rapport aux méthodes de Chandrasekaran et de Luk et Pagano [17, 69]. Pour valider ces approches, une implémentation des deux méthodes a été réalisée en langage MATLAB. Les résultats numériques obtenus ont permis d'effectuer une analyse comparative des performances, mettant en évidence les avantages et inconvénients de chaque méthode. Ces expérimentations ont confirmé l'efficacité de notre méthode proposée, notamment en termes de rapidité et de proximité de la solution optimale, tout en soulignant certains aspects spécifiques à améliorer en fonction des configurations des problèmes étudiés.

Dans une perspective d'approfondissement, une autre méthode de résolution, appelée Box-QP, a été proposée dans le chapitre 4, où nous avons étendu notre étude à des cas plus spécifiques. Cette méthode vise à résoudre un PPQ avec une M-matrice à diagonale dominante et des contraintes de bornes. L'approche inclut une procédure de prétraitement (Algorithm 2) qui exploite la structure particulière de la matrice D, permettant de simplifier le problème initial en réduisant la taille de la matrice à traiter, ce qui améliore significativement l'efficacité computationnelle. Cette étape préliminaire diminue la complexité globale du problème, rendant la résolution plus rapide et plus pratique. L'algorithme d'optimisation (Algorithm 4) pour le problème réduit s'inspire de la méthode des points extérieurs et intègre le concept de support pour la fonction objectif. Cette intégration permet d'optimiser efficacement le problème en exploitant la structure de la solution, ce qui améliore la convergence et réduit le temps de calcul.

Perspectives

Les travaux que nous avons élaborés au cours de cette thèse nous ont permis de mettre en évidence plusieurs perspectives intéressantes pour approfondir et élargir l'étude des problèmes de programmation quadratique (PPQs), en particulier ceux impliquant des M-matrices. Plusieurs pistes peuvent être envisagées :

— Valider expérimentalement la méthode développée dans le chapitre 4 à l'aide d'une implémentation MATLAB. Les résultats numériques permettront de comparer les performances de cette méthode avec d'autres approches, en évaluant sa rapidité de convergence et sa proximité à la solution optimale, tout en identifiant les aspects à améliorer selon les configurations des problèmes étudiés ;
— Extension de ces méthodes pour la résolution de problèmes de programmation quadratique avec des M-matrices et des contraintes généralisées, tout en conservant les propriétés avantageuses des M-matrices ;
— Appliquer les techniques développées à des problèmes quadratiques multicritères, en tenant compte des spécificités des M-matrices pour équilibrer plusieurs objectifs.

Ces perspectives offrent des axes prometteurs, tant sur le plan théorique qu'appliqué, renforçant ainsi l'intérêt des M-matrices, indispensables à la formulation et à la résolution de problèmes complexes en optimisation et en analyse numérique.

Bibliographie

[1] J. Abadie. *On the Karush-Kuhn-Tucker Theorem*. North-Holland Publishing, Amsterdam, 1967.

[2] E. D. Andersen and K. D. Andersen. Presolving in linear programming. *Mathematical Programming*, 71(1) :221–245, 1995.

[3] A. Andjouh and M. O. Bibi. Adaptive global algorithm for solving box-constrained non-convex quadratic minimization problems. *Journal of Optimization Theory and Applications*, 192(1) :360–378, 2022.

[4] A. Andjouh and M. O. Bibi. Support solving method for box-constrained indefinite quadratic minimisation problems. *International Journal of Mathematics in Modeling and Numerical Optimisation*, 12(4) :390–415, 2022.

[5] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt. Augmented lagrangian methods under the constant positive linear dependence constraint qualification. *Mathematical Programming*, 111 :5–32, 2008.

[6] A. Atamtürk and A. Gómez. Strong formulations for quadratic optimization with $M-$matrices and indicator variables. *Mathematical Programming*, 170 :141–176, 2018.

[7] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming : Theory and Algorithms*. John Wiley & Sons, 3rd edition, 2006.

[8] A. Berman and R. J. Plemmons. *Nonnegative Matrices in Mathematical Sciences*. Academic Press, New York, 1979.

[9] D. P. Bertsekas. Projected newton method for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2) :221–246, 1982.

[10] M. O. Bibi, N. Ikheneche, and M. Bentobache. A hybrid direction algorithm for solving a convex quadratic problem. *International Journal of Mathematics in Operations Research*, 16(2) :159–178, 2020.

[11] J. C. G. Boot. *Quadratic Programming : Algorithms, Anomalies and Applications*. North-Holland Publishing Company, Amsterdam, The Netherlands, 1964.

[12] A. Bounceur, S. Djemai, B. Brahmi, M. O. Bibi, and R. Euler. A classification approach for an accurate analog/rf bist evaluation based on the process parameters. *Journal of Electronic Testing*, 34(3) :321–335, 2018.

[13] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[14] B. Brahmi and M. O. Bibi. Dual support method for solving convex quadratic programs. *Optimization*, 59(6) :851–872, 2010.

[15] R. Cambini and C. Sodini. A sequential method for a class of box constrained quadratic programming problems. *Mathematical Methods of Operations Research*, 67(2) :223–243, 2008.

[16] J. Céa and R. Glowinski. Sur des méthodes d'optimisation par relaxation. *RAIRO*, (R-3) :5–32, 1973.

[17] R. Chandrasekaran. A special case of the complementary pivot problem. *Opsearch*, 7 :263–268, 1970.

[18] B. M. Chen-Charpentier and H. V. Kojouharov. An unconditionally positivity preserving scheme for advection-diffusion reaction equations. *Mathematical and Computer Modelling*, 57 :2177–2185, 2013.

[19] G. Cimatti. On a problem of the theory of lubrication governed by a variational inequality. *Applied Mathematics and Optimization*, 3(2-3) :227–242, 1976.

[20] T. F. Coleman and L. A. Hulbert. A direct active set algorithm for large sparse quadratic programs with simple bounds. *Mathematical Programming*, 45(1) :373–406, 1989.

[21] R. W. Cottle and M. S. Goheen. A special class of large quadratic programs. In O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, editors, *Nonlinear Programming 3*, pages 361–390. Academic Press, 1978.

[22] Y. H. Dai and R. Fletcher. Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. *Numerical Mathematics*, 100 :21–47, 2005.

[23] A. N. Daryina and A. F. Izmailov. On active-set methods for the quadratic programming problem. *Computational Mathematics and Mathematical Physics*, 52(4) :512–523, 2012.

[24] S. Djemai, B. Brahmi, and M. O. Bibi. A primal–dual method for SVM training. *Neurocomputing*, 211 :34–40, 2016.

[25] M. Fiedler. *Special Matrices and Their Applications in Numerical Mathematics*. Nijhoff, Lancaster, 1986.

[26] M. Fiedler and S. Hellwig. Successive relaxation and $M-$matrices. *Numerische Mathematik*, 57 :357–366, 1990.

[27] M. Fiedler and V. Pták. On matrices with non-positive off-diagonal elements and positive principal minors. *Czechoslovak Mathematical Journal*, 12 :382–400, 1962.

[28] R. Fletcher and M.P. Jackson. Minimization of a quadratic function of many variables subject only to lower and upper bounds. *IMA Journal of Applied Mathematics*, 14(2) :159–174, 1974.

[29] A. Friedlander, J. Martinez, and M. Raydan. A new method for large-scale box constrained convex quadratic minimization problems. *Optimization Methods and Software*, 5(1) :57–74, 1995.

[30] C. E. Fröberg. Introduction to numerical analysis. *Addison-Wesley, Reading Mass*, 1969.

[31] R. Funderlic and R. J. Plemmons. *LU* decomposition of $M-$matrices by elimination without pivoting. *Linear Algebra and Its Applications*, 41 :99–110, 1981.

[32] R. Funderlic and R. J. Plemmons. A combined direct-iterative method for certain $M-$matrix linear systems. *SIAM Journal on Algebraic and Discrete Methods*, 5 :33–42, 1984.

[33] R. Gabasov. Adaptive method of linear programming. Preprint, University of Karlsruhe, Institute of Statistics and Mathematics, Germany, 1993.

[34] R. Gabasov, F. M. Kirillova, and O. I. Kostyukova. A method of solving general linear programming problems. *Doklady AN BSSR*, 23(3) :197–200, 1979.

[35] R. Gabasov, F. M. Kirillova, and O. I. Kostyukova. Solution of linear quadratic extremal problems. *Soviet Math. Dokl*, 31 :99–103, 1985.

[36] R. Gabasov, F. M. Kirillova, O. I. Kostyukova, and V. M. Raketsky. *Constructive methods of optimization, volume 4 : Convex Problems*. University Press, Minsk, 1987.

[37] R. Gabasov, F. M. Kirillova, and V. M. Raketsky. On methods for solving the general problem of convex quadratic programming. *Soviet Math. Dokl*, 23 :653–657, 1981.

[38] J. H. Gallier. *The Schur Complement and Symmetric Positive Semidefinite (and Definite) Matrices*. Computer Sciences Commons, 2010.

[39] R. Glowinski. *Lectures on Numerical Methods for Non-Linear Variational Problems*. Springer Science & Business Media, Berlin, New York, 2008.

[40] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

[41] J. Gondzio. Presolve analysis of linear programs prior to applying an interior point method. *INFORMS Journal on Computing*, 9(1) :73–91, 1997.

[42] N. Gould and P. L. Toint. Preprocessing for quadratic programming. *Mathematical Programming*, 100(1) :95–132, 2004.

[43] N. I. M. Gould. An algorithm for large-scale quadratic programming. *SIAM Journal on Numerical Analysis*, 11 :299–324, 1991.

[44] N. I. M. Gould and P. L. Toint. A quadratic programming bibliography. Technical Report 2000-1, RAL. Numerical Analysis Group, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2000.

[45] K. Hassaini and M. O. Bibi. Résolution d'un problème de programmation quadratique avec une M-matrice. In *COSI'2010, Actes du Colloque International sur l'Optimisation et les Systèmes d'Information*, pages 123–134, Ouargla, Algérie, Avril 2010.

[46] K. Hassaini and M. O. Bibi. Quadratic programming problems with a diagonally dominant M-matrix. In *MOAD'2022, Actes du Colloque International sur les Méthodes et Outils d'Aide à la Décision*, pages 128–134, Bejaia, Algeria, November 2022.

[47] K. Hassaini and M. O Bibi. Quadratic programming method with an $M-$matrix. *Statistics, Optimization and Information Computing*, 12(2) :405–417, March 2024.

[48] K. Hassaini and M. O. Bibi. Quadratic programming problems with preprocessing and a diagonally dominant M-matrix. In *CARI'2024 : African Conference on Research in Computer Science and Applied Mathematics*, Bejaia, Algérie, November 2024.

[49] D. S. Hochbaum. Multi-label markov random fields as an efficient and effective tool for image segmentation, total variations and regularization. *Numerical Mathematics : Theory, Methods and Applications*, 6(1) :169–198, 2013.

[50] L. Hogben. The symmetric $M-$matrix and symmetric inverse $M-$matrix completion problems. *Linear Algebra and Its Applications*, 353(1-3) :159–168, 2002.

[51] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press, England, 1986.

[52] Y. C. Hsieh and D. L. Bricker. Solving obstacle problems by using a new infeasible interior point algorithm. *Journal of the Chinese Institute of Industrial Engineers*, 16(6) :771–780, 1999.

[53] P. Hungerländer and F. Rendl. A feasible active set method for strictly convex quadratic problems with simple bounds. *SIAM Journal on Optimization*, 25(3) :1633–1659, 2017.

[54] S. Ikonen. *Efficient Numerical Solution of the Black-Scholes Equation by Finite Difference Method*. Phd thesis, University of Jyvaskyla, Department of Mathematical Information Technology, Jyvaskyla, Finland, 2003.

[55] Y. J. Jiang and J. P. Zeng. Direct algorithm for the solution of two-sided obstacle problems with $M-$matrix. *Numerical Linear Algebra with Applications*, 18(1) :167–173, 2011.

[56] I. Kaneko. Linear complementarity problems and characterizations of Minkowski matrices. *Linear Algebra and its Applications*, 20(2) :111–129, 1978.

[57] T. Kärkkäinen, K. Kunisch, and P. Tarvainen. Augmented lagrangian active set methods for obstacle problems. *Journal of Optimization Theory and Applications*, 119(3) :499–533, 2003.

[58] W. Karush. Minima of functions of several variables with inequalities as side conditions. Technical report, Dept. of Mathematics, University of Chicago, Chicago, Illinois, 1939.

[59] J. Keilson and G.P.H. Styan. Markov chains and $M-$matrices : inequalities and equalities. *Journal of Mathematical Analysis and Applications*, 41 :439–459, 1973.

[60] N. Kochev, A. Terziski, and M. Milev. Numerical modelling of three-phase mass transition with an application in atmospheric chemistry. *Applied Mathematics*, 4(8A) :100–106, 2013.

[61] H. Konno and A. Wijayanayake. Portfolio optimization problem under concave transaction costs and minimal transaction unit constraints. *Mathematical Programming*, Ser. B 89 :233–250, 2001.

[62] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, 1951. University of California Press.

[63] K. Kunisch and F. Rendl. An infeasible active set method for convex problems with simple bounds. *SIAM Journal on Optimization*, 14(1) :35–52, 2003.

[64] A. Laouar and M. O. Bibi. A new presolving technique for convex box-constrained quadratic programming problems. *Advances in Nonlinear Variational Inequalities*, 28(6s) :106–118, 2005.

[65] A. Laouar and M. O. Bibi. An adaptive hybrid direction algorithm for convex box-qp problems with enhanced pre-solving. *International Journal of Mathematics in Operations Research*, 2023.

[66] G. Levati, F. Scarpini, and G. Volpi. *Sul trattamento numerico di alcuni problemi variazionali di tipo unilaterale*. L.A.N. Pub.82, University of Pavia, 1974.

[67] L. Li and Y. Kobayashi. A block recursive algorithm for the linear complementarity problem with an $M-$matrix. *International Journal of Innovative Computing, Information and Control*, 2(6) :1327–1335, 2006.

[68] Y. Lin and C. W. Ciyer. An alternating direction implicit algorithm for the solution of linear complementarity problems arising from free boundary problems. *Applied Mathematics and Optimization*, 13(1) :1–17, 1985.

[69] F. T. Luk and M. Pagano. Quadratic programming with $M-$matrices. *Linear Algebra and Its Applications*, 33 :15–40, 1980.

[70] P. Lötstedt. Solving the minimal least square problems subject to bounds on the variables. *BIT. Numerical mathematics*, 24(2) :206–224, 1984.

[71] F. M. Mazel M. S. Lobo and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152 :341–365, 2007.

[72] J. M. Martínez. BOX-QUACAN and the implementation of augmented lagrangian algorithms for minimization with inequality constraints. *Computational and Applied Mathematics*, 19 :31–56, 2000.

[73] M. Milev and A. Tagliani. Nonstandard finite difference schemes with application to finance : Option pricing. *Serdica Mathematical Journal*, 36 :75–88, 2010.

[74] M. Milev and A. Tagliani. Efficient implicit scheme with positivity preserving and smoothing properties. *Journal of Computational and Applied Mathematics*, 243 :1–9, 2013.

[75] H. Minkowski. Ueber positive quadratische formen. *Journal für die reine und angewandte Mathematik*, 99 :1–9, 1886.

[76] H. Minkowski. Zur theorie der einheiten in den algebraischen zahlkörpern. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, 2 :90–93, 1900.

[77] H. Minkowski. *Diophantische Approximationen*. Teubner, Leipzig, 1907.

[78] J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1) :93–113, 1991.

[79] C. Mészáros and U. H. Suhl. Advanced preprocessing techniques for linear and quadratic programming. *OR Spectrum*, 25(4) :575–595, 2003.

[80] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Science & Business Media, New York, 2nd edition, 2006.

[81] A. Ostrowski. Über die determinanten mit überwiegender hauptdiagonale. *Commentarii Mathematici Helvetici*, 10 :69–96, 1937.

[82] M. Pakdaman and S. Effati. Bounds for convex quadratic programming problems and some important applications. *International Journal of Operational Research*, 30(2) :277–287, 2017.

[83] J. S. Pang. On a class of least-element complementarity problems. *Mathematical programming*, 16 :111–126, 1979.

[84] R. J. Plemmons. A survey of $M-$matrix characterizations i : Nonsingular $M-$matrices. Technical Report Technical Summary Report 1651, University of Wisconsin - Madison, Mathematics Research Center, 1976.

[85] E. Polak. *Computational Methods in Optimization : A Unified Approach*. Academic Press, London, 1971.

[86] G. Poole and T. Boullion. A survey on $M-$matrices. *SIAM Review*, 16(4) :419–427, 1974.

[87] S. Radjef and M. O. Bibi. An effective generalization of the direct support method in quadratic convex programming. *Applied Mathematical Sciences*, 6(31) :1525–1540, 2012.

[88] C. Reisinger and J. H. Witte. On the use of policy iteration as an easy way of pricing american options. *SIAM Journal on Financial Mathematics*, 3(1) :459–478, 2012.

[89] J. F. Rodrigues. *Obstacle Problems in Mathematical Physics*. North-Holland, New York, 1987.

[90] F. Scarpini. Some algorithms solving the unilateral dirichlet problems with two constraints. *Calcolo*, 12(2) :113–149, 1975.

[91] C. Schmid and L. T. Biegler. Quadratic programming methods for reduced hessian SQP. *Computers Chemical Engineering*, 18(9) :817–832, 1994.

[92] B. Schofield. On active set algorithms for solving bound-constrained least squares control allocation problems. In *Proceedings of the American Control Conference*, pages 2597–2602, Seattle, Washington, USA, June 2008.

[93] J. Schröder. $M-$matrices and generalizations using an operator theory approach. *SIAM Review*, 20(2) :213–244, 1978.

[94] Q. Song, F. Liu, J. Cao, and W. Yu. $M-$matrix strategies for pinning-controlled leader-following consensus in multiagent systems with nonlinear dynamics. *IEEE Transactions on Cybernetics*, 43(6) :1688–1697, 2013.

[95] A. Stachurski. Monotone sequences of feasible solutions for quadratic programming problems with $M-$matrices and box constraints. In *System Modeling and Optimization*, volume 84 of *Lecture Notes in Control and Information Sciences*, pages 896–902. 1986.

[96] A. Stachurski. Analysis of the newton projection method in application to quadratic programming problems with $M-$matrices and box constraints. *Archiwum Automatyki i Telemechanika*, 34(1-2) :155–164, 1989.

[97] A. Stachurski. Quadratic programming problems with $M-$matrices and box constraints. *Bulletin of the Polish Academy of Sciences : Technical Sciences*, 37(3-4) :247–262, 1989.

[98] A. Stachurski. An equivalence between two algorithms for a class of quadratic programming problems with $M-$matrices. *Optimization*, 21(6) :871–878, 1990.

[99] D. M. Stipanovic, S. Shankaran, and C. J. Tomlin. Multi-agent avoidance control using an $M-$matrix property. *The Electronic Journal of Linear Algebra*, 12 :64–72, 2005.

[100] D. M. Stipanovic and D. D. Šiljak. Stability of polytopic systems via convex $M-$matrices and parameter-dependent Liapunov functions. *Nonlinear Analysis : Theory, Methods and Applications*, 40(1) :589–609, 2000.

[101] W. Y. Sun and Y. X. Yuan. Optimization theory and methods : Nonlinear programming. *Springer Series in Operations Research and Financial Engineering*, pages 1–721, 2006.

[102] R. S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, N.J, 1962.

[103] R. S. Varga. $M-$matrix theory and recent results in numerical linear algebra. In *Sparse Matrix Computations*, pages 375–387. Academic Press, Inc., New York, 1976.

[104] R. S. Varga. On recurring theorems on diagonal dominance. *Linear Algebra and its Applications*, 13(1) :1–9, 1976.

[105] V. V. Voevodin and J. A. Kuznetzov. *Matrices and Computations*. Russia, Moscow, 1984.

[106] C. Voglis and I. E. Lagaris. BOXCQP : An algorithm for bound constrained convex quadratic problems. In *1st International Conference "From Scientific Computing To Computational Engineering"*, volume 1, pages 261–268, Athens, Greece, September 2004.

[107] D. D. Šiljak. *Large-scale Dynamic Systems : Stability and Structure*. North-Holland, New York, 1978.

[108] P. Wilmott, J. Dewynne, and S. Howison. *Option Pricing : Mathematical Models and Computation*. Oxford Financial Press, Oxford, 1993.

[109] G. Windisch. *M−matrices in Numerical Analysis*. Springer-Verlag, Germany, 2013.

[110] P. Wolfe. The simplex method for quadratic programming. *Econometrica*, 27 :382–398, 1959.

[111] X. S. Zhang and H. C. Zhu. A neural network model for quadratic programming with simple upper and lower bounds and its application to linear programming. In *International Symposium on Algorithms and Computation*, pages 119–127. Springer, Berlin, Heidelberg, 1994.

RÉSUMÉ

Dans cette thèse, nous proposons une nouvelle approche pour résoudre un problème de programmation quadratique avec une M-matrice et des contraintes simples. Cette approche est basée sur les algorithmes de Luk et Pagano. Ces méthodes utilisent le fait qu'une M-matrice possède une inverse non négative qui permet d'obtenir une suite de points réalisables croissante et monotone. En introduisant le concept de support d'une fonction objective, notre approche conduit à une condition plus générale permettant d'obtenir une solution initiale réalisable, proche de la solution optimale. L'implémentation de notre méthode et de celle de Luk et Pagano a permis une analyse numérique, révélant que notre approche est plus efficace que celle de Luk et Pagano. En outre, une autre méthode de résolution (Box-QP), a été proposée, où nous nous sommes intéressés à résoudre un problème de programmation quadratique avec une M-matrice à diagonale dominante et des contraintes de bornes. Cette nouvelle approche inclut une procédure de prétraitement qui exploite la structure particulière de la M-matrice, permettant de simplifier le problème initial en réduisant la taille de la M-matrice. L'algorithme d'optimisation pour le problème réduit s'inspire de la méthode des points extérieurs et intègre le concept de support pour la fonction objectif. Cette intégration permet d'optimiser efficacement le problème.

Mots Clés : Programmation quadratique convexe, M-Matrices, M-Matrice diagonalement dominante, procédure de prétraitement, Méthode de Support, Solution Réalisable du Support (SRS).

ABSTRACT

In this thesis, we propose a new approach for solving a quadratic programming problem with an M-matrix and simple constraints. This approach is based on the algorithms of Luk, and Pagano. These methods use the fact that an M-matrix possesses a nonnegative inverse, which allows a sequence of feasible points to increase monotonically. By introducing the concept of support for an objective function, our approach leads to a more general condition allowing an initial feasible solution, close to the optimal solution. The implementation of our method and that of Luk and Pagano allowed a numerical analysis, revealing that our approach is more efficient than that of Luk and Pagano. Furthermore, another solution method (Box-QP), has been proposed, where we have been interested to solve a quadratic programming problem with a dominant-diagonal M-matrix and bound constraints. This new approach includes a pre-processing procedure that exploits the particular structure of the M-matrix, allowing simplifying thus the initial problem by reducing the size of the M-matrix. The optimization algorithm for the reduced problem is inspired by the method of exterior points and incorporates the concept of support for the objective function. This integration enables the problem to be optimized efficiently.

Keywords: Convex Quadratic Programming, M-Matrices, Diagonally dominant M-matrix, Preprocessing procedure, Support Method, Support Feasible Solution (SFS).

ملخص

في هذه الأطروحة, نقترح نهجا جديدا لحل مشكلة برمجة تربيعية ذات $M-$مصفوفة وقيود بسيطة. يعتمد هذا النهج على خوارزميات لوك وباغانو. تستخدم هذه الأساليب حقيقة أن $M-$مصفوفة تمتلك معكوسا غير سالب يسمح بالحصول على تسلسل من النقاط الممكنة التي تتزايد بشكل رتيب. من خلال تقديم مفهوم الدعم لدالة الهدف, يؤدي نهجنا إلى شرط أكثر عمومية يسمح بالحصول على حل ممكن مبدئيا قريبا من الحل الأمثل. لقد أتاحت لنا برمجة طريقتنا وطريقة لوك وباغانو إجراء مقارنة بينهما، عبر توضيح مثالين عمليين. تشير النتائج العددية إلى أن نهجنا أكثر كفاءة من النهج الذي اقترحه لوك وباغانو. بالإضافة إلى ذلك، تم اقتراح طريقة حل أخرى تسمى $(Box-QP)$، حيث اهتممنا بحل مشكلة برمجة تربيعية مع $M-$مصفوفة ذات قيود قطرية ومحدودة مهيمنة. يتضمن هذا النهج إجراء معالجة مسبقة يستغل البنية الخاصة للمصفوفة، مما يجعل من الممكن تبسيط المشكلة الأولية عن طريق تقليل حجم المصفوفة المراد معالجتها. خوارزمية التحسين للمشكلة المختصرة مستوحاة من طريقة النقاط الخارجية وتتضمن مفهوم الدعم لدالة الهدف. يسمح هذا التكامل بتحسين المشكلة بكفاءة من خلال استغلال بنية الحل.

الكلمات المفتاحية: البرمجة التربيعية المحدبة, $M-$المصفوفات, $M-$مصفوفة المهيمنة قطرياً, إجراء المعالجة المسبقة, طريقة الدعم, حل الدعم القابل للتحقيق (SRS).

AGZUL

Deg ugmmir ayi, nsumer-ed yiwen uxwarzim amaynut i wakken ad' nefru yiwen wegnu n usmihel asnuzmir es yiwen M-isirew aked tmariwin tushilin. Tayaẓt ayi tres γef ixwarzimen n Luk aked Pagano. Tarrayin agi semrasent tamsalt d-iqqaren belli M-isirew yesεa imiṯṯi awer uzdir, i yaγ ye$\check{g}\check{g}$an ad' neṣk asartu yeṯnernin n tneqqiḍin yeṯwaqbalen. Mi'ara d-nessidef amγal n usalel i tesγent tamesγarut, tayaẓt nneγ teṯṯawi γer tewtilt d-tamatut ugar, i-yzemren ad' d-tḥelli yiwet n tifrat tamezwart yeṯṯwaqbalen, yuẓan γer tifrat takkayt. I wakken ad' nessemyif ger tarrayt nneγ aked tin n Luk ed Pagano, muggent termiyin tumḍinin i d-isseknen tamellilt n tayaẓt nneγ γef tin n Luk aked Pagano. Diγen, yiwet n tarrayt-nniḍen n tifrat (Box-QP) teṯṯwasumered, anda i nefra yiwen wegnu n usmihel asnuzmir es yiwen M-isirew bu tubdist tamagart aked tmariwin n yegmiren. Tayaẓt ayi tamaynutt tewwi yiwet n tifrat n tezwart yeṯṯamden tamṣukt tamaẓlayt n M-isirew, i yaγ-ye$\check{g}\check{g}$an ad'nessifses agnu amenzu es usemẓi n tiddi n M-isirew. Axwarzim n takkayit i wegnu yeṯṯwasemẓin yewwi-d sγur tarrayt n tneqqiḍin yeffγen u yeγred amγal n usalel i tesγent tamesγarut. Aseγred ayi yetreg i wegnu akken ad itwasekkey es tmellilt.

Awalen n tsura : Asmihel asnuzmir afensu; M-isirwen; M-isirew bu tubdist tamagart; Tifrat n tezwart; Tarrayt n usalel; Tifrat yeṯṯwaqbalen n usalel (SRS).