REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE Ferhat ABBAS SETIF 1

THESE

Présentée

Pour l'obtention du diplôme de DOCTORAT LMD 3ème cycle

Filière : Mathématiques

Spécialité : Optimisation et Contrôle

Par

BENNANI Ahlem

THEME

Sur les performances des méthodes projectives pour les problèmes d'optimisation fractionnaires

Soutenue le : 09/12/2021

Devant le jury :

M. Rachid ZITOUNI	**Professeur**	**Univ. Ferhat Abbas Sétif 1**	**Président**
M. Djamel BENTERKI	**Professeur**	**Univ. Ferhat Abbas Sétif 1**	**Rapporteur**
M. Abdelkrim MERZOUGUI	**Professeur**	**Univ. Med Boudiaf M'sila**	**Examinateur**
M. Mousaab BOUAFIA	**MCA**	**Univ. 8 mai 1945 Guelma**	**Examinateur**
Mme. Zahira KEBAILI	**MCA**	**Univ. Ferhat Abbas Sétif 1**	**Examinateur**
Mme. Hassina GRAR	**MCA**	**Univ. Ferhat Abbas Sétif 1**	**Invitée**

REMERCIEMENTS

Je remercie Dieu de m'avoir donné la force et le courage pour accomplir ce modeste travail.

Mes vifs remerciements vont à toutes les personnes ayant contribués de près ou de loin au bon déroulement et à l'aboutissement de cette thèse, tant sur le plan professionnel que sur le plan personnel.

En premier lieu, je tiens à exprimer ma profonde gratitude à mon directeur de thèse :

- ***Mr. Djamel BENTERKI**, Professeur à l'Université Ferhat Abbas Sétif-1 pour la qualité de son encadrement exceptionnel, sa rigueur, sa disponibilité, sa confiance, sa motivation, sa compréhension, son soutien continu, et surtout sa patience et ses conseils tout au long de ces années de recherche.*

*Je remercie **Mr. Rachid ZITOUNI**, Professeur à l'Université Ferhat Abbas Sétif-1, de m'avoir fait l'honneur de faire partie de ce jury et d'en être le président.*

Je désire remercier également :

- ***Mr. Abdelkrim MERZOUGUI**, Professeur à l'Université Mohammed Boudiaf M'sila,*
- ***Mr. Mousaab BOUAFIA**, Maître de Conférence « A » à l'Université 8 mai 1945 Guelma,*
- ***Mme. Zahira KEBAILI**, Maître de Conférence « A » à l'Université Ferhat Abbas Sétif-1,*

d'avoir accepté de juger ce travail et d'en être les examinateurs.

Je remercie aussi :

- ***Mme. Hassina GRAR**, Maître de Conférence « A » à l'Université Ferhat Abbas Sétif-1, pour son aide, sa disponibilité, ses remarques et ses précieux conseils .*

Je voudrais adresser mes remerciements à tous les membres du Laboratoire de Mathématiques Fondamentales et Numériques, à toute l'équipe administrative du département de mathématiques à Université Ferhat Abbas sétif-1, et mes chères

amies pour leurs soutien moral et leurs encouragements, particulièrement : ***Amina ZERARI, Sara KENTACHE*** *et* ***Hadjer KHOMS.***

Je remercie chaleureusement ma mère, celle qui m'a donné la vie, le symbole de tendresse, qui a tout sacrifié pour mon bonheur et ma réussite.

Je remercie mon père, pour sa motivation et encouragement tout au long de ma vie.

Je remercie mes sœurs et mon frère, ainsi que mon oncle « ***Kamel »****, qui a été toujours présent pour m'aider et me soutenir.*

Je remercie ma belle famille pour le soutien et pour l'encouragement et surtout mon mari pour sa compréhension et son sacrifice durant l'élaboration de cette thèse.

DÉDICACE

*Je dédie cette thèse à la mémoire de mes grands parents « **Beba Taher** et **Ma Zebida** » qui m'ont toujours soutenu tout au long de mon parcours.*

*Ils nous ont quitté trop tôt « **08/07/2020-15/07/2020** » et n'ont pas pu être là pour me voir accomplir ce travail. Que dieu, le tout puissant, les bénisse de sa grande miséricorde.*

Je dédie ce travail aussi à mes filles, la force qui me pousse vers l'avant, j'essayerai chaque jour de ma vie de devenir meilleure juste pour elles.

Table des matières

Introduction

La programmation mathématique est une branche des mathématiques qui a connu une diversification spectaculaire au cours de ces dernières décennies.

Le concept de la programmation mathématique impose le recours à l'optimisation qui consiste à trouver parmi un ensemble de données un meilleur élément minimisant ou maximisant une fonction donnée. Les problèmes d'optimisation servent à modéliser de nombreux problèmes de la vie réelle.

La diversité de ces problèmes exige la variété des méthodes de résolution, il n'est donc pas surprenant de partager l'optimisation en plusieurs catégories : linéaire, non linéaire, continue ou en nombres entiers... etc.

Dans notre travail, on s'intéresse à une classe particulière de la programmation mathématique qui s'appelle la programmation fractionnaire (PF) où l'objectif s'exprime comme le rapport de deux fonctions. Ce domaine a été développé en grande partie par le mathématicien Hongrois B. Martos et ses associés dans les années 60.

L'intérêt de ce sujet a été généré par le fait que divers problèmes d'optimisation qui proviennent de la pratique considèrent à optimiser un rapport de deux fonctions ; coût/temps, coût/volume, coût/profit ou autres données spécifiques pour mesurer l'efficacité d'un système. Par exemple en économie, la productivité des systèmes industriels définie comme le rapport entre la quantité des unités produites au cours d'une période donnée et les ressources utilisées, est considérée comme l'un des meilleurs indicateurs de la qualité de leur fonctionnement. De tels problèmes, où la fonction objectif apparaît comme un rapport de deux fonctions, constituent les problèmes de programmation fractionnaire générale. En raison de son importance dans la modélisation de nombreux processus décisionnels en science de gestion, l'ingénierie, la recherche opérationnelle et l'économie, et aussi en raison de son apparition fréquente dans d'autres problèmes pas nécessairement économiques comme la théorie de l'information, l'analyse numérique,

la programmation stochastique, ce domaine a reçu une attention particulière au cours des trois dernières décennies dans le but de développer une théorie plus riche et d'élaborer des algorithmes de résolution très efficaces.

Alors dans ce contexte, le but de cette thèse est de participer à ces développements en introduisant de nouvelles techniques de résolution. Notre travail consiste à étudier et résoudre numériquement (PF) en le transformant en un programme équivalent.

Dans un premier lieu, on traite les problèmes fractionnaires linéaires (PFL), c'est à dire on optimise un objectif qui s'écrit comme le rapport de deux fonctions linéaires ou affines sous contraintes linéaires.

On s'engage premièrement à établir une formulation efficace qui permet de ramener (PFL) à un programme linéaire (PL) équivalent tout en gardant la taille du problème inchangée. La formulation proposée rend la résolution de (PFL) très facile, vu qu'elle transforme ce dernier en un (PL) sans agrandir la taille du problème. Donc, on peut résoudre n'importe quel (PFL) en utilisant cette formulation. La résolution numérique se réalise en appliquant une approche adéquate, celle de Ye-Lustig, qui représente une variante efficace de points intérieurs.

L'algorithme de Ye-Lustig [41] est une forme améliorée de l'algorithme de Karmarkar [30], publié par son auteur en 1984. Ces derniers ont permis de résoudre des problèmes avec une complexité algorithmique de moindre coût de type polynômial, et leur efficacité se manifeste beaucoup plus pour les problèmes à grande taille. Contrairement à la méthode du simplexe introduite par G. Dantzig en 1947, qui est d'une complexité algorithmique exponentielle. Cette dernière a été la seule méthode utilisée pour résoudre les différents problèmes avant l'apparition de l'algorithme de Karmarkar et ses variantes.

Deuxièment, on sait que certains problèmes d'optimisation non linéaires avec contraintes, notamment un programme fractionnaire linéaire peut être considéré comme l'un des cas particuliers les plus intéressants du problème d'inégalités variationnelles (VIP) moyennant les conditions d'optimalité. Ce problème très connu en analyse mathématique a fait l'objet de recherches théoriques et numériques intensives de grande qualité. Pour une bonne littérature concernant ce problème, le lecteur peut se référer aux références suivantes [18, 25]. En se basant sur ce concept, notre objectif est d'appliquer une des méthodes de projection les plus performantes de résolution de (VIP), pour résoudre cette fois les programmes fractionnaires linéaires. En effet, il s'agit de la méthode de projection

proposée par Grar et Benterki [24] qui a fait preuve de son efficacité numérique pour la résolution des problèmes d'inégalités variationnelles (VIP).

En second lieu, on traite les programmes fractionnaires quadratiques (PFQ), où ici l'objectif s'écrit comme le rapport de deux fonctions quadratiques sous contraintes linéaires. En suivant les mêmes volets précédents, on ramène (PFQ) à deux programmes équivalents : La première formulation revient à Sivri et al. [55] qui ramène (PFQ) à un (PL), en se basant sur le développement de Taylor de l'objectif. Pour la deuxième formulation, on propose une technique qui permet de ramener (PFQ) à un programme quadratique (PQ) équivalent qu'on résout via un algorithme dit DCA (Algorithme de Différence de deux fonctions Convexes), proposé pour la première fois par T. Pham Dinh [47] en 1985. La popularité de la programmation DC (Différence de deux fonctions Convexes) et les algorithmes DCA réside dans leur simplicité, flexibilité, robustesse, rapidité et performance comparées à des méthodes existantes, et leur adaptation aux structures des problèmes traités avec une capacité à résoudre des problèmes industriels de grande dimension.

La thèse est structurée en quatre chapitres comme suit :

- **Chapitre 1 :** On rappelle quelques notions de base d'analyse convexe, les principales notions de convexité généralisée, la programmation mathématique et en particulier la programmation linéaire.
- **Chapitre 2 :** On présente une étude détaillée sur les problèmes de programmation fractionnaire linéaire sous contraintes linéaires en donnant des résultats d'existence et unicité d'une solution optimale et les méthodes de résolution de (PFL). Par la suite, on présente notre première contribution où on donne en détails une nouvelle formulation qui permet de convertir (PFL) en un (PL) équivalent. L'avantage de cette formulation est la préservation de la taille du problème (aucune augmentation ni des variables ni des contraintes). Ce chapitre sera achevé par des tests numériques d'ordre comparatif réalisés sur des exemples de différentes tailles à savoir : Taille fixe et taille variable. Nous considérons dans ces expérimentations l'algorithme de Ye-Lustig [41] pour résoudre (PFL) moyennant notre nouvelle formulation et celle de Charnes et Cooper [15].

  L'ensemble des résultats obtenus dans cette partie a fait l'objet d'une publication internationale qui apparaîtera dans la revue "International Journal of Computing Science and Mathematics" sous le thème "Efficient projec-

tive algorithm for linear fractional programming problem based on a linear programming formulation".

- **Chapitre 3 :** Dans ce chapitre, on présente notre deuxième contribution basée sur les problèmes d'inégalités variationnelles. On débute par une étude approfondie sur les problèmes d'inégalités variationnelles, où on donne les résultats d'existence et d'unicité d'une solution de (VIP), ainsi que les méthodes de résolution de projection les plus populaires. Plus précisément, on donne en détails l'algorithme de Solodov et Svaiter [56] et celui de Grar et Benterki [24]. Par la suite, on établit le lien entre (PFL) et (VIP) moyennant les conditions d'optimalité sur (PFL). Enfin, des tests numériques comparatifs sont réalisés sur des différents exemples moyennant l'algorithme obtenu via la technique de (VIP) et celui obtenu par la nouvelle formulation donnée dans le chapitre 2.

  Le contenu de ce chapitre a fait l'objet d'une deuxième publication dans la revue internationale "RAIRO-Operations Research" sous le thème "Adaptive projection methods for linear fractional programming" (2021).
- **Chapitre 4 :** L'objet de ce chapitre est l'extension des travaux réalisés dans les chapitres 2 et 3, à la programmation fractionnaire quadratique. Dans un premier temps, on présente une formulation introduite par Sivri et al. [55] qui permet de ramener (PFQ) à un problème linéaire (PL) en utilisant le développement de Taylor de l'objectif. Tandis que, on propose une nouvelle formulation permettant de ramener (PFQ) à un programme quadratique (PQ). Ensuite, on envisage de résoudre (PFQ) par deux variantes numériques. La première concerne la méthode projective de Ye-Lustig moyennant la formulation de Sivri et al. [55]. Tandis que, la deuxième concerne l'algorithme de différence de deux fonctions convexes appelé DCA moyennant notre nouvelle formulation. Les tests numériques de ce travail sont en cours de réalisation en vue de son éventuel publication.

Cette thèse est achevée par une conclusion générale et une liste de références.

Chapitre 1

Notions fondamentales

Résumé : Dans ce chapitre, nous rappelons quelques notions de base sur l'analyse convexe et ses applications, la programmation mathématique en particulier le cas linéaire. Nous rappelons également, les résultats de dualité en programmation linéaire.

Contenu :

1.1 Symboles et notations

On se place dans l'espace des vecteurs réels $\mathbb{R}^n$ muni de la norme euclidienne et le produit scalaire associé

$$\|x\| = \sqrt{\langle x, x\rangle} \text{ où } \langle x, x\rangle = x^t x = \sum_{i=1}^{n} x_i^2, \forall x \in \mathbb{R}^n,$$

x^t le vecteur transposé de x.

Etant donné $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction réelle deux fois différentiable, le gradient de f noté ∇f est le vecteur colonne défini par les éléments

$$(\nabla f(x))_i = \frac{\partial f(x)}{\partial x_i}, \quad i = 1, \ldots, n.$$

Le Hessien de f noté $\nabla^2 f$ est la matrice de $\mathbb{R}^{n\times n}$ définie par les éléments

$$\left(\nabla^2 f(x)\right)_{i,j} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}, \quad i, j = 1, \ldots, n.$$

Etant donné $F : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ un opérateur

$$F(x) = F(x_1, x_2, ..., x_n) = \begin{pmatrix} F_1(x_1, x_2, \ldots, x_n) \\ F_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ F_m(x_1, x_2, \ldots, x_n) \end{pmatrix}.$$

Le Jacobien de F noté JF est défini par la matrice de $\mathbb{R}^{m\times n}$ dont les cœfficients

$$(JF(x))_{i,j} = \frac{\partial F_i(x)}{\partial x_j}, \quad i = 1, \ldots, m \text{ et } \ j = 1, \ldots, n.$$

1.2 Eléments d'analyse convexe et programmation mathématique

1.2.1 Analyse convexe

Dans cette partie, on présente un rappel des notions fondamentales de l'analyse convexe et la programmation mathématique qui serviront d'appuis pour la suite. Pour plus de détails voir [33, 52].

a) Ensembles convexes

1. Un sous ensemble C de $\mathbb{R}^n$ est dit affine si :

$$\lambda x + (1-\lambda)\, y \in C, \forall x, y \in C, \forall \lambda \in \mathbb{R}.$$

2. Un sous ensemble C de $\mathbb{R}^n$ est dit convexe si

$$\lambda x + (1-\lambda)\, y \in C,\ \forall x, y \in C,\ \forall \lambda \in [0,1]\,.$$

D'un point de vue géométrique, un convexe est donc un ensemble qui, lorsqu'il contient deux points, contient nécessairement le segment reliant ces deux points.

Définition 1.2.1 *Combinaison linéaire convexe*

Un vecteur $y \in C$ est une combinaison linéaire convexe des points $\{x_1, ..., x_p\}$ s'il existe des coefficients réels $\lambda_i,\ i \in \{1, ..., p\}$, tels que :

$$y = \sum_{i=1}^{p} \lambda_i x_i \text{ avec } \lambda_i \geq 0,\ \forall i \in \{1, ..., p\} \text{ et } \sum_{i=1}^{p} \lambda_i = 1.$$

Définition 1.2.2 *Enveloppe convexe*

L'enveloppe convexe d'un ensemble $C \subset \mathbb{R}^n$, est l'ensemble des points de $\mathbb{R}^n$ qui s'écrivent comme combinaison convexe des points de C. Elle est notée :

$$conv(C) = \left\{ x \in \mathbb{R}^n / \ x = \sum_{i=1}^{p} \lambda_i x_i,\ x_i \in C,\ \lambda_i \geq 0,\ \forall i = \{1, ..., p\} \text{ et } \sum_{i=1}^{p} \lambda_i = 1 \right\}.$$

$conv(C)$ est le plus petit convexe contenant C.

- *L'ensemble $\{x \in \mathbb{R}^n / \ a^t x = b\}$ où $a \in \mathbb{R}^n$ et $b \in \mathbb{R}$, représente un hyperplan de $\mathbb{R}^n$.*
- *L'ensemble $\{x \in \mathbb{R}^n / a^t x \leq b\}$ représente un demi-espace fermé de $\mathbb{R}^n$ dont l'hyperplan correspondant constitue la frontière.*

Définition 1.2.3 *Polyèdre*

1) Un polyèdre S est l'intersection d'un nombre fini de demi-espaces fermés et/ou d'hyperplans.

$$S = \{x \in \mathbb{R}^n : A_i^t x \leq b_i, i = 1, ..., m\} \text{ où } A_i \in \mathbb{R}^n \text{ et } b_i \in \mathbb{R}$$

S peut s'écrire sous la forme matricielle suivante :

$$S = \{x \in \mathbb{R}^n / \ Ax \leq b\} \ \text{où} \ A \in \mathbb{R}^{m \times n} \ \text{et} \ b \in \mathbb{R}^m$$

S est convexe et fermé.

2) Un polyèdre S est borné s'il existe une valeur finie et positive β telle que :

$$|x_j| \leq \beta \quad \forall j \in \{1, ..., n\} \quad \text{et} \ \forall x \in S.$$

Définition 1.2.4 *S_n est un (n-simplexe) s'il est de la forme :*

$$S_n = \left\{ x \in \mathbb{R}^n_+ : \sum_{i=1}^{n} x_i = 1 \right\}$$

Définition 1.2.5 *Point extrême*

Soit S un convexe non vide de $\mathbb{R}^n$. x est dit point extrême ou sommet de S si :

$$x = \lambda x_1 + (1 - \lambda) x_2 \quad \text{alors} \quad x = x_1 = x_2 \quad \forall x_1, x_2 \in S \quad \text{et} \quad \lambda \in]0, 1[.$$

Rappellons en bref la notion de séparation, cette dernière est reconnue comme l'une des notions les plus fertiles de l'analyse convexe et d'un usage courant dans la programmation mathématique.

On signale qu'il existe plusieurs types de séparations de deux ensembles C_1 et C_2, et l'intérêt de la notion de séparation apparaît surtout pour les ensembles convexes.

Théorème 1.2.1 *Séparation large*

Soit $H = \{x \in \mathbb{R}^n / \ a^t x = \alpha\}$ avec $a \in \mathbb{R}^n, a \neq 0$ et $\alpha \in \mathbb{R}$. C_1, C_2 deux sous-ensembles non vides de $\mathbb{R}^n$ sont séparés au sens large par H si :

$$a^t x \geq \alpha \quad \forall x \in C_1 \quad \text{et} \ a^t x \leq \alpha \quad \forall x \in C_2, \quad \text{avec} \ C_1 \cup C_2 \subseteq H.$$

Théorème 1.2.2 *Séparation propre*

Soit $H = \{x \in \mathbb{R}^n / \ a^t x = \alpha\}$ avec $a \in \mathbb{R}^n, a \neq 0$ et $\alpha \in \mathbb{R}$. C_1, C_2 deux sous-ensembles non vides de $\mathbb{R}^n$ sont proprement séparés par H si :

$$a^T x \geq \alpha, \forall x \in C_1 \quad \text{et} \quad a^T x \leq \alpha, \forall x \in C_2, \quad \text{avec} \ C_1 \cup C_2 \subsetneq H.$$

Théorème 1.2.3 *Séparation stricte*

Soit $H = \{x \in \mathbb{R}^n / \ a^t x = \alpha\}$ *avec* $a \in \mathbb{R}^n, a \neq 0$ *et* $\alpha \in \mathbb{R}$. C_1, C_2 *deux sous-ensembles non vides de* $\mathbb{R}^n$ *sont strictement séparés par* H *si :*

$$a^t x > \alpha \ \ \forall x \in C_1 \text{ et } a^t x < \alpha \ \ \forall x \in C_2.$$

Théorème 1.2.4 *Séparation forte*

Soit $H = \{x \in \mathbb{R}^n / \ a^t x = \alpha\}$ *avec* $a \in \mathbb{R}^n, a \neq 0$ *et* $\alpha \in \mathbb{R}$. C_1, C_2 *deux sous-ensembles non vides de* $\mathbb{R}^n$ *sont fortement séparés par* H *si :*

$$\begin{aligned} \exists \varepsilon \quad &> \quad 0 \ / \ a^T x \geq \alpha + \varepsilon, \forall x \in C_1 \ \ \text{et} \ \ a^T x \leq \alpha - \varepsilon, \forall x \in C_2 \\ \Longleftrightarrow \quad & C_1 + \varepsilon B \subseteq int(H^+) \text{ et } C_2 + \varepsilon B \subseteq int(H^-). \end{aligned}$$

Où

$H^+ = \left\{x \in \mathbb{R}^n / \ a^t x \geq \alpha\right\}$, $H^- = \left\{x \in \mathbb{R}^n / \ a^t x \leq \alpha\right\}$ *et* $B = \{x \in \mathbb{R}^n / \ \|x\| \leq 1\}$.

Remarque 1.2.1 *On peut toujours séparer strictement un point* y *d'un ensemble convexe fermé non vide* C *ne contenant pas* y. *Ce résultat est donné dans le lemme suivant.*

Lemme 1.2.1 *Soit* C *un convexe fermé non vide de* $\mathbb{R}^n$ *et* $y \in (\mathbb{R}^n / C)$. *Alors, il existe* $a \in \mathbb{R}^n, a \neq 0$ *et un scalaire* $\alpha \in \mathbb{R}$ *tels que :*

$$a^t y > \alpha \qquad et \qquad a^t x \leq \alpha \ \ \forall x \in C.$$

Comme on constate également que les théorèmes de séparation sont la conséquence fondamentale du théorème de projection dans $\mathbb{R}^n$ que nous rappelons par la suite.

b) Fonctions convexes

Définition 1.2.6 *Soit* $f : C \longrightarrow \mathbb{R}$ *une fonction et* C *un sous ensemble convexe de* $\mathbb{R}^n$.

– f *est dite convexe sur* C *si l'inégalité suivante est satisfaite :*

$$f(\lambda x + (1-\lambda) y) \leq \lambda f(x) + (1-\lambda) f(y), \ \forall \lambda \in [0,1], \ \forall x, y \in C.$$

– *f est dite strictement convexe sur C si :*

$$f(\lambda x + (1-\lambda)y) < \lambda f(x) + (1-\lambda) f(y),\ \forall \lambda \in]0,1[,\ \forall x, y \in C \text{ et } x \neq y.$$

– *f est dite fortement convexe sur C s'il existe $\alpha > 0$, tel que :*

$$\begin{aligned} f(\lambda x + (1-\lambda)y) &\leq \lambda f(x) + (1-\lambda) f(y) - \frac{1}{2}\alpha\lambda(1-\lambda)\|x-y\|^2, \\ \forall \lambda &\in]0,1[,\ \forall x, y \in C \text{ et } x \neq y. \end{aligned}$$

(On dit aussi que f est α-convexe).

– *f est concave $\Longleftrightarrow -f$ est convexe.*

– *f fortement convexe $\Longrightarrow f$ strictement convexe $\Longrightarrow f$ convexe.*

Définition 1.2.7 *Épigraphe*

On appelle épigraphe de f, noté épi(f) le sous ensemble de $\mathbb{R}^{n+1}$ défini par :

$$épi(f) = \{(x, y) \in C \times \mathbb{R} \ / \ f(x) \leq y\}.$$

Définition 1.2.8 *Hypographe*

On appelle hypographe de f, noté hyp(f) le sous ensemble de $\mathbb{R}^{n+1}$ défini par :

$$hyp(f) = \{(x, y) \in C \times \mathbb{R} \ / \ f(x) \geq y\}.$$

Théorème 1.2.5 *[6] [11]*

1) f est convexe si et seulement si épi(f) est convexe.

2) f est concave si et seulement si hyp(f) est convexe.

Lemme 1.2.2 *Si f est une fonction convexe, alors l'ensemble de niveau*

$$L(f, \alpha) = \{x \in C / \ f(x) \leq \alpha\},$$

est convexe pour tout $\alpha \in \mathbb{R}$.

Preuve. Soient $x_1,\ x_2 \in L(f, \alpha) \subset C$, donc $f(x_1) \leq \alpha$ et $f(x_2) \leq \alpha$.

Soit $\lambda \in [0, 1]$ et $x = \lambda x_1 + (1-\lambda)x_2$, comme C est convexe donc $x \in C$ et par la convexité de f on a

$$f(x) = f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2) \leq \lambda\alpha + (1-\lambda)\alpha = \alpha.$$

D'où $x \in L(f, \alpha)$ est par conséquent $L(f, \alpha)$ est convexe. ■

Définition 1.2.9 – *Une fonction $f : \mathbb{R}^n \longrightarrow \overline{\mathbb{R}}$ est dite semi-continue inférieurement (s.c.i) en $x^0 \in \mathbb{R}^n$ si :*

$$\forall \varepsilon > 0,\ \exists \delta > 0 /\quad f(x) \geq f(x^0) - \varepsilon,\ \text{dès que}\ \left\|x - x^0\right\| \leq \delta.$$

- *f est dite semi-continue supérieurement (s.c.s) en x^0 si $-f$ est semi-continue inférieurement (s.c.i) en x^0.*
- *f est continue en x^0 si elle est à la fois semi-continue inférieurement et semi-continue supérieurement en x^0.*
- *f est dite s.c.i si elle est s.c.i en tout point $x^0 \in \mathbb{R}^n$.*

Théorème 1.2.6 *Pour toute fonction $f : \mathbb{R}^n \longrightarrow \overline{\mathbb{R}}$; les propriétés suivantes sont équivalentes :*

1. *f est s.c.i.*
2. *$épi(f)$ est fermé.*
3. *Tous les ensembles de niveau inférieurs $L(f, \alpha) = \{x :\ f(x) \leq \alpha\}$ où $\alpha \in \mathbb{R}$ sont fermés.*

c) Fonctions convexes généralisées

Définition 1.2.10 *Soit $f : C \longrightarrow \mathbb{R}$ une fonction et C un sous ensemble convexe de $\mathbb{R}^n$.*

- *f est dite quasiconvexe sur C si :*

$$f(\lambda x + (1-\lambda)y) \leq \max\{f(x), f(y)\},\ \forall x, y \in C,\ \forall \lambda \in [0,1]$$

Ou aussi si et seulement si :

$$f(x) \leq f(y) \Longrightarrow f(\lambda x + (1-\lambda)y) \leq f(y),\ \forall x, y \in C, \forall \lambda \in [0,1]$$

- *f est dite quasiconcave sur C si :*

$$f(\lambda x + (1-\lambda)y) \geq \min\{f(x), f(y)\},\ \forall x, y \in C, \forall \lambda \in [0,1]$$

- *La fonction f est quasiconcave $\Longleftrightarrow$ $(-f)$ est quasiconvexe.*
- *Une fonction qui est à la fois quasiconvexe et quasiconcave est dite quasi-monotone ou quasilinéaire.*

– *Les fonctions quasiconvexes, quasiconcaves et quasilinéaires sont caractérisées par la convexité de leurs ensembles de niveaux.*

Théorème 1.2.7 *1) f est quasiconvexe si et seulement si l'ensemble*

$$L(f,\alpha) = \{x \in C \ : \ f(x) \leq \alpha\}$$

est convexe pour tout $\alpha \in \mathbb{R}$.

2) f est quasiconcave si et seulement si l'ensemble

$$U(f,\alpha) = \{x \in C \ : \ f(x) \geq \alpha\}$$

est convexe pour tout $\alpha \in \mathbb{R}$.

Preuve. 1) $\Longrightarrow$) Supposons que f est quasiconvexe et soient x_1, $x_2 \in L(f,\alpha)$ donc x_1, $x_2 \in C$ car $L(f,\alpha) \subset C$ et

$$\max\{f(x_1), f(x_2)\} \leq \alpha.$$

Soit $\lambda \in [0,1]$ et $x = \lambda x_1 + (1-\lambda)x_2$, comme C est convexe donc $x \in C$ et par la quasiconvexité de f on a

$$f(x) \leq \max\{f(x_1), f(x_2)\} \leq \alpha,$$

d'où $x \in L(f,\alpha)$, ce qui donne $L(f,\alpha)$ est convexe.

$\Longleftarrow$) Supposons que $L(f,\alpha)$ est convexe pour tout réel α. Soient x_1 et x_2 dans C, $\lambda \in [0,1]$ et $x = \lambda x_1 + (1-\lambda)x_2$. Notons que x_1 et $x_2 \in L(f,\alpha)$ pour $\alpha = \max\{f(x_1), f(x_2)\}$.

$L(f,\alpha)$ étant convexe par hypothèse donc

$$x \in L(f,\alpha) \text{ ce qui donne } f(x) \leq \alpha = \max\{f(x_1), f(x_2)\},$$

c'est à dire

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \max\{f(x_1), f(x_2)\}.$$

Par conséquent, f est quasiconvexe.

On montre d'une manière similaire 2). ■

Il est clair qu'une fonction convexe sur un ensemble C convexe est quasiconvexe.

Définition 1.2.11 *La fonction f est dite strictement quasiconvexe sur C si :*

$f(\lambda x+(1-\lambda)y) < \max\{f(x), f(y)\}$, $\forall x, y \in C$ *tel que,* $f(x) \neq f(y)$, $\forall \lambda \in]0,1[$.

Proposition 1.2.1 *Soit f une fonction strictement quasiconvexe définie sur un ensemble convexe non vide C de $\mathbb{R}^n$. Si x^* est un minimum local du problème qui minimise f sur C, alors il est l'unique minimum global.*

Proposition 1.2.2 *Soit f une fonction différentiable définie sur un ensemble convexe non vide C de $\mathbb{R}^n$*

- *f est quasiconvexe sur C si et seulement si :*

$$f(x_2) \leq f(x_1) \Longrightarrow \langle \nabla f(x_1), x_2 - x_1 \rangle \leq 0, \ \forall x_1, x_2 \in C,$$

- *ou*

$$\langle \nabla f(x_1), x_2 - x_1 \rangle > 0 \Longrightarrow f(x_2) > f(x_1), \ \forall x_1, x_2 \in C.$$

Proposition 1.2.3

1. *f convexe $\Longrightarrow$ f quasiconvexe.*
2. *f strictement convexe $\Longrightarrow$ f strictement quasiconvexe.*
3. *f strictement quasiconvexe $\Longrightarrow$ f quasiconvexe.*
4. *f quasiconcave $\Longleftrightarrow$ $(-f)$ quasiconvexe.*
5. *f est strictement quasiconcave $\Longleftrightarrow$ $(-f)$ est strictement quasiconvexe.*

Définition 1.2.12 *Soit f une fonction différentiable sur un ouvert non vide C de $\mathbb{R}^n$.*

- *f est dite pseudoconvexe sur C si :*

$$(y - x)^t \nabla f(x) \geq 0 \Longrightarrow f(y) \geq f(x), \ \forall x, y \in C.$$

Définition 1.2.13 *f est dite pseudoconcave sur C si :*

$$(y - x)^t \nabla f(x) \leq 0 \Longrightarrow f(y) \leq f(x), \ \forall x, y \in C.$$

De plus, f est dite strictement pseudoconvexe (strictement pseudoconcave) si les inégalités précédentes sont strictes respectivement pour $x \neq y$.

Théorème 1.2.8 *[6][11] Soit $f : C \longrightarrow \mathbb{R}$ différentiable et pseudoconvexe sur un convexe ouvert non vide C de $\mathbb{R}^n$. Alors, f est à la fois quasiconvexe et strictement quasiconvexe.*

Théorème 1.2.9 *[1] Soit f une fonction différentiable et pseudoconvexe sur C un convexe ouvert non vide de $\mathbb{R}^n$. Soit x^* un point de C tel que $\nabla f(x^*) = 0$, alors le point x^* est un minimum global de f sur C.*

Corollaire 1.2.1 *[1] Si f est une fonction différentiable et strictement pseudoconvexe sur un convexe ouvert C de $\mathbb{R}^n$, alors elle admet au plus un minimum global.*

Proposition 1.2.4 *[63]*

1) Si f est convexe, alors f est quasiconvexe. Si de plus, f est différentiable alors elle est pseudoconvexe.

2) Si f est concave alors f est quasiconcave. Si de plus, f est différentiable alors elle est pseudoconcave.

Définition 1.2.14 *[1] Soit f une fonction définie sur C un ensemble convexe de $\mathbb{R}^n$.*

- *f est dite semi-strictement quasiconvexe sur C si et seulement si $\forall x, y \in C$ tels que $f(x) \neq f(y)$ et $\forall\ z \in]x, y[$, on a :*

$$f(z) < max\{f(x), f(y)\}.$$

Théorème 1.2.10 *[1] Soit f une fonction semi-strictement quasiconvexe et semi continue inférieurement sur C, alors f est quasiconvexe sur C.*

Théorème 1.2.11 *[1] Soit f une fonction semi-strictement quasiconvexe sur C. Alors, tout minimum local de f sur C est global.*

Proposition 1.2.5 *[1] Soit f une fonction différentiable sur un convexe ouvert non vide C de $\mathbb{R}^n$. On a :*

- *Si f est pseudoconvexe sur C, alors elle est semi strictement quasiconvexe et donc quasiconvexe sur C.*
- *Si f est quasiconvexe sur C et si $\nabla f(x) \neq 0\ \forall\ x \in C$, alors f est pseudoconvexe sur C.*

d) Caractérisation d'une fonction convexe différentiable

Si $f \in \mathrm{C}^1(C)$, où C est un ensemble convexe, alors on a les équivalences suivantes :

– f est convexe si et seulement si

$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle, \ \forall x, y \in C,$$

– ou encore si et seulement si

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq 0, \ \forall x, y \in C.$$

De plus, f est dite strictement convexe si l'une ou l'autre des inégalités précédentes sont strictes pour $x \neq y$.

– f est fortement convexe si et seulement s'il existe $\alpha > 0$, tel que :

$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle + \frac{\alpha}{2} \|y - x\|^2, \ \forall x, y \in C.$$

Si $f \in \mathrm{C}^2(C)$, alors :

– f est convexe si et seulement si $\nabla^2 f(x)$ (le Hessien de f) est semi-défini positif sur C (c'est à dire que $y^t \ \nabla^2 f(x) y \geq 0, \ \forall x, y \in C$).

– f est strictement convexe si et seulement si $\nabla^2 f(x)$ est défini positif sur C (c'est à dire que $y^t \ \nabla^2 f(x) y > 0, \ \forall x, y \in C$ et $y \neq 0$).

Remarque 1.2.2 *f est fortement convexe sur C si et seulement si $f - \frac{1}{2\alpha} \| \ . \|^2$ est convexe sur C avec $\alpha > 0$.*

La stricte convexité est une hypothèse un peu restrictive, mais elle constitue un cadre théorique agréable pour de nombreux problèmes d'optimisation en particulier le cas des fonctions quadratiques, très important en pratique.

Proposition 1.2.6 *[1] Soient f, g deux fonctions définies sur un ensemble convexe non vide C de $\mathbb{R}^n$. La fonction $h(x) = \frac{f(x)}{g(x)}$ est semi strictement quasi-convexe si :*

– *la fonction f est positive et convexe sur C, et la fonction g est strictement positive et concave sur C.*

– *la fonction f est négative et convexe sur C, et g est strictement positive et convexe sur C.*

– *la fonction f est convexe sur C et la fonction g est strictement positive et affine sur C.*

e) Opérateurs monotones

Soient F un opérateur et C un sous ensemble de $\mathbb{R}^n$ tel que : $F : C \longrightarrow \mathbb{R}^n$.

Définition 1.2.15 – *F est dit monotone sur C si :*

$$\langle F(x) - F(y), x - y\rangle \geq 0, \qquad \forall x, y \in C.$$

– *F est dit strictement monotone sur C si :*

$$\langle F(x) - F(y), x - y\rangle > 0, \qquad \forall x, y \in C et \;\; x \neq y.$$

– *F est dit fortement monotone sur C, s'il existe $\alpha > 0$, tel que :*

$$\langle F(x) - F(y), x - y\rangle \geq \alpha \left\|x - y\right\|^2 \;, \qquad \forall x, y \in C \text{ et } x \neq y.$$

Définition 1.2.16 – *F est dit pseudomonotone sur C si l'une des deux conditions suivantes est satisfaite :*

1. $\langle F(x), y - x\rangle \geq 0 \implies \langle F(y), y - x\rangle \geq 0, \forall\; x, y \in C.$
2. $\langle F(x), y - x\rangle > 0 \implies \langle F(y), y - x\rangle > 0, \forall\; x, y \in C.$

– *F est strictement quasimonotone sur C si :*

$$\langle F(x), y - x\rangle > 0 \Longrightarrow \langle F(y), y - x\rangle \geq 0, \;\forall x, y \in C.$$

Remarque 1.2.3 *La relation qui lie ces différents types de monotonie de f est comme suit :*

Ffortement monotone $\implies$ Fstrictement monotone $\implies$ F monotone $\implies$ F pseudomonotone $\implies$ F quasimonotone.

De plus, la convexité peut être aussi exprimée à partir de la notion de monotonie dans le cas où la fonction f est différentiable sur C, alors on a les équivalences suivantes :

1. *f est convexe si et seulement si ∇f est monotone.*
2. *f est strictment convexe si et seulement si ∇f est strictement monotone.*
3. *f est fortement convexe si et seulement si ∇f est fortement monotone.*
4. *f est pseudoconvexe si et seulement si ∇f est pseudomonotone.*

1.2.2 Programmation mathématique

La programmation mathématique constitue un domaine vaste et riche dans l'analyse numérique et traduit plusieurs problèmes pratiques importants.

D'une façon générale, un programme mathématique est un problème d'optimisation avec contraintes de type :

$$(P_m) \begin{cases} \min\ f(x) \\ x \in C \end{cases}$$

$$\text{où} \quad C = \left\{ \begin{array}{r} x \in \mathbb{R}^n / g_i(x) \leq 0,\ i = 1, \ldots, m \\ h_j(x) = 0,\ j = 1, \ldots, p \end{array} \right\}$$

et f, g_i, h_j sont des fonctions données de $\mathbb{R}^n$ vers $\mathbb{R}$.

On appelle f la fonction objectif et C l'ensemble des solutions réalisables ou admissibles.

Définition 1.2.17 *Minimum global*

Une solution admissible $\overline{x}$ est dite minimum global de (P_m) si et seulement si

$$f(\overline{x}) \leq f(x), \forall x \in C.$$

Définition 1.2.18 *Minimum local*

Une solution admissible $\overline{x}$ est dite minimum local de (P_m) si et seulement s'il existe un voisinage $V_\varepsilon(\overline{x})$ de $\overline{x}$ tel que

$$f(\overline{x}) \leq f(x),\ \forall x \in C \cap V_\varepsilon(\overline{x}).$$

Théorème 1.2.12 *Soit $f : C \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction convexe sur un ensemble convexe non vide C de $\mathbb{R}^n$.*

Si $\overline{x}$ est un minimum local alors, $\overline{x}$ est un minimum global pour le problème (P_m).

Remarque 1.2.4 *Tout minimum global est évidemment un minimum local. En général, on est beaucoup plus intéressé par la recherche de minima globaux (car ils sont les seuls à garantir que la valeur de leur fonction objectif ne peut être réduite), mais ceux-ci sont malheureusement également beaucoup plus diffciles à calculer (intuitivement, la raison en est qu'il suffit pour prouver qu'un minimum est local de vérifier qu'il n'existe pas de meilleure solution dans un voisinage restreint autour de ce minimum, tandis que prouver qu'un minimum est global requiert l'analyse de la fonction objectif sur l'entièreté du domaine admissible). Cependant, à l'aide de la notion de convexité, de quasiconvexité et de peudoconvexité, nous allons décrire une classe de problèmes pour laquelle la situation est bien plus favorable.*

Théorème 1.2.13 *Existence*

Soit $f : C \longrightarrow \mathbb{R}$ une fonction quasiconvexe (ou quasiconcave cas de maximisation) et continue sur C compact, alors au moins une solution optimale $\overline{x}$ de (P_m), est un point extrême de C.

Remarque 1.2.5 *[63] Les fonctions strictement quasiconvexes et strictement quasiconcaves, sont particulièrement importantes dans la programmation non linéaire, car elles assurent qu'un minimum local et un maximum local sur un ensemble convexe soit, respectivement, un minimum global et un maximum global.*

Théorème 1.2.14 *[63] Soit $f : C \longrightarrow \mathbb{R}$ strictement quasiconvexe sur un convexe non vide C de $\mathbb{R}^n$. Si $\overline{x}$ est un minimum local, alors $\overline{x}$ est un minimum global.*

Preuve. Supposons, au contraire, qu'il existe $\widehat{x} \in C$ avec $f(\widehat{x}) < f(\overline{x})$.
C étant convexe donc $\lambda\widehat{x} + (1-\lambda)\overline{x} \in C$, $\forall\lambda \in [0,1]$.
Comme $\overline{x}$ est un minimum local par hypothèse donc

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

f est strictement quasiconvexe, alors

$$si\ f(\widehat{x}) < f(\overline{x})\ on\ aura\ f(\lambda\widehat{x} + (1-\lambda)\overline{x}) < f(\overline{x}),\ \forall\lambda \in [0,1],$$

ceci contredit l'optimalité locale de $\overline{x}$ (1.1). ■

Théorème 1.2.15 *Soit $f : C \longrightarrow \mathbb{R}$ une fonction pseudoconvexe et différentiable sur un convexe non vide C. Si $\overline{x}$ est un minimum local alors, $\overline{x}$ est un minimum global de f sur C.*

La classification de (P_m) et son traitement numérique sont établis à partir des propriétés fondamentales des fonctions f, g_i, h_j à savoir la convexité, la différentiabilité et la linéarité.

Parmi les cas particuliers les plus étudiés on note :

- La programmation linéaire (f linéaire, g_i, h_j affines, C ortant positif).
- La programmation convexe (f, g_i convexes, h_j affine, C convexe).
- La programmation en nombres entiers (C discret).

On peut classer (P_m) aussi en fonction des caractéristiques mathématiques de la fonction objectif, des contraintes et des variables d'optimisation comme suit :

Caractéristiques	Propriétés	Classification
Nombre de variables	Une seule variable	Monovariable
	Plus d'une variable	Multivariable
Type de variables	Réelles	Continue
	Entières	Discrète
	Réelles et entières	Mixte
	Entières avec permutations	Combinatoire
Type de la fonction objectif	Linéaire en fonction des variables	Linéaire
	Quadratique en fonction des variables	Quadratique
	Non linéaire en fonction des variables	Non linéaire
Formulation du problème	Soumis à des limitations	Avec contraintes
	Pas de limitations	Sans contraintes

1.2.3 Conditions d'optimalité de Karuch-Khun-Tucker

Avant de donner les conditions d'optimalité de Karuch-Khun-Tucker (K.K.T) de (P_m), on exige que les contraintes doivent satisfaire certains critères dits de qualification.

a) Qualification des contraintes

Une contrainte g_i est dite active (ou saturée) en $\overline{x} \in C$ si $g_i(\overline{x}) = 0$.

Un point $\overline{x} \in C$ est dit régulier (on dit également que les contraintes sont qualifiées en $\overline{x}$) si les gradients des contraintes saturées en $\overline{x}$ sont linéairement indépendants.

Il existe aussi deux critères usuels de qualification en tout point de C, à savoir :

- Si toutes les contraintes sont affines.
- Si C est défini uniquement par des inégalités, on a le critère de Slater suivant : $g_i(x)$ est convexe pour tout $i = 1, \ldots, m$ et qu'il existe un point x^0 tel que $g_i(x^0) < 0$, ($int(C) \neq \varnothing$).

b) Conditions nécessaires d'optimalité

Théorème 1.2.16 *(Karuch-Khun-Tuker)*

Soit $\overline{x} \in C$ satisfaisant l'une des conditions de qualification précédentes et supposons que f, g_i, h_j sont $C^1(\mathbb{R}^n)$, on a :

Si $\overline{x}$ est un optimum local, alors il existe des multiplicateurs : $\mu_i \in \mathbb{R}^+$, $i = 1, \ldots, m$ et $\lambda_j \in \mathbb{R}$, $j = 1, \ldots, p$ tels que :

$$\begin{cases} \nabla f(\overline{x}) + \sum_{i=1}^{m} \mu_i \nabla g_i(\overline{x}) + \sum_{j=1}^{p} \lambda_j \nabla h_j(\overline{x}) = 0, & \textit{(conditions d'optimalité)} \\ \mu_i g_i(\overline{x}) = 0 \quad , i = 1, \ldots, m. & \textit{(conditions de complémentarité)} \\ h_j(\overline{x}) = 0 \quad , j = 1, \ldots, p. \end{cases}$$

Si de plus, f, g_i, h_j sont convexes, les conditions précédentes sont à la fois nécessaires et suffisantes pour que $\overline{x}$ soit un optimum global pour (P_m).

Les problèmes avec contraintes de type (P_m) peuvent souvent être transformés en problèmes sans contraintes à l'aide des multiplicateurs de Karuch-Khun-Tuker (KKT). Cette méthode qui nous permet de trouver les points stationnaires $\overline{x}$ de la fonction f, revient en effet à associer à chaque contrainte un scalaire inconnu μ_i , λ_j. On forme par la suite une combinaison linéaire des gradients de la fonction objectif et des contraintes comme le montre le système ci-dessus.

Dans le cas particulier où $m = 0$, les conditions précédentes sont appelées les conditions de Lagrange et s'écrivent comme suit :

Si $\overline{x}$ est un optimum local, alors il existe $\lambda_j \in \mathbb{R}$, $j = 1, \ldots, p$ tel que :

$$\begin{cases} \nabla f(\overline{x}) + \sum_{j=1}^{p} \lambda_j \nabla h_j(\overline{x}) = 0 \\ h_j(\overline{x}) = 0 \qquad j = 1, ..., p. \end{cases}$$

Application (Projection sur un hyperplan)

Soit $H = \left\{ x \in \mathbb{R}^n / \; a^T x = \alpha \right\}$ (où $0 \neq a \in \mathbb{R}^n$ et $\alpha \in \mathbb{R}$) un hyperplan de $\mathbb{R}^n$ et $y \in \mathbb{R}^n \backslash H$.

H étant convexe fermé et non vide, l'existence et l'unicité du projeté orthogonal de y sur H sont assurées par le théorème de séparation.

En termes de programmation mathématique, le problème convexe différentiable suivant admet une solution unique

$$(P_H) \quad \begin{cases} \min \; \left(\frac{1}{2} \|x - y\|^2 \right) \\ x \in H. \end{cases}$$

Laquelle est parfaitement caractérisée par les conditions de (KKT).

$$\overline{x} = Proj_H(y) \Longleftrightarrow \overline{x} = y - \left(\frac{a^T y - \alpha}{\|a\|^2}\right) a.$$

c) Fonction de Lagrange et dualité lagrangienne classique

On considère le programme de type suivant :

$$(P_g) \quad \begin{cases} \min \ f(x) \\ g_i(x) \leq 0, \ i = 1, \ldots, m \\ x \in C \subset \mathbb{R}^n \end{cases}$$

Assoçions à chaque contrainte un nombre réel $\lambda_i \geq 0$ appelé multiplicateur de Lagrange.

La fonction de Lagrange (ou lagrangien) associée à ce problème est définie par :

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x), \qquad x \in C, \quad \lambda_i \geq 0.$$

Définition 1.2.19 *On appelle $(\overline{x}, \overline{\lambda})$ un point selle (ou point col) pour la fonction de Lagrange sur $C \times \mathbb{R}^m_+$, si pour tout point $(x, \lambda) \in C \times \mathbb{R}^m_+$ on a :*

$$\min_{x \in C} L(x, \lambda) = L(\overline{x}, \lambda) \leq \min_{x \in C} \max_{\lambda \in \mathbb{R}^m_+} L(x, \lambda) = L(\overline{x}, \overline{\lambda}) \leq \max_{\lambda \in \mathbb{R}^m_+} L(x, \lambda) = L(x, \overline{\lambda})$$

Rappelons une propriété caractéristique des points selles :
$(\overline{x}, \overline{\lambda})$ est un point selle si et seulement si :

- $L(\overline{x}, \overline{\lambda}) = \min_{x \in C} L(x, \overline{\lambda})$
- $g_i(\overline{x}) \leq 0, \ i = 1, \ldots, m.$
- $\overline{\lambda}_i g_i(\overline{x}) = 0, \ i = 1, \ldots, m.$

De plus, on a le résultat suivant : si $(\overline{x}, \overline{\lambda})$ est un point selle pour la fonction de Lagrange, alors $\overline{x}$ est un minimum global pour (P_g).

Définissons maintenant pour $\lambda \geq 0$ la fonction q donnée par :

$$q(\lambda) = \min_{x \in C} L(x, \lambda).$$

Et le programme :

$$(D_g) \quad \begin{cases} \max \ q(\lambda) \ = \max \ \min L(x, \lambda) \\ \qquad\qquad\qquad \lambda \in \mathbb{R}^m_+ \end{cases}$$

La fonction q est appelée la fonction duale et (D_g) le programme dual associé au programme primal (P_g).

Dualité lagrangienne

Soit :

$$C = \{g_i(x) \leq 0,\ i = 1, ..., p,\ h_j(x) = 0,\quad j = 1, ..., m\},$$

et on considère le problème d'optimisation :

$$m^* = \inf_x [f(x),\quad x \in C].$$

Le lagrangien associé est la fonction $L : C \times [0, +\infty[^p \times \mathbb{R}^m \longrightarrow \mathbb{R}$ définie par :

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^{p} \lambda_i g_i(x) + \sum_{j=1}^{m} \mu_j h_j(x).$$

On pose :

$$\alpha(x) = \sup_{\lambda,\mu} [L(x, \lambda, \mu) : \lambda \geq 0] = \begin{cases} f(x) \text{ si } g_i(x) \leq 0 \text{ et } h_j(x) = 0 \\ +\infty \text{ sinon.} \end{cases}$$

Donc :

$$\overline{\alpha} = \inf_{x \in C} \alpha(x) = \inf_x [f(x) : x \in C],$$

le problème dual est :

$$\overline{\beta} = \sup_{(\lambda,\mu)} \inf^{x \in C} [L(x, \lambda, \mu)].$$

On a l'inégalité de dualité

$$-\infty \leq \overline{\beta} \leq \overline{\alpha}.$$

Revenons à la programmation mathématique linéaire qui sera un outil important pour démontrer un résultat fondamental ultérieurement.

1.2.4 Programmation linéaire

La programmation linéaire dans $\mathbb{R}^n$ traite la résolution des problèmes d'optimisation pour lesquels la fonction objectif et les contraintes sont linéaires. Cependant les spécialistes la considèrent comme étant la technique la plus utilisée dans la recherche opérationnelle.

Un programme linéaire peut être écrit sous différentes formes équivalentes, la plus utilisée en pratique est la forme standard :

$$(PL) \begin{cases} \min\ c^t x \\ Ax = b \\ x \geq 0 \end{cases},$$

avec $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ et A une matrice de $\mathbb{R}^{m \times n}$.

Le programme linéaire (PL) caractérisé par :

$$(PL) \begin{cases} \min\ c^t x \\ Ax \leq b\ (\geq b) \\ x \geq 0 \end{cases},$$

représente sa forme canonique.

Dans toute la suite, on s'intéresse au programme linéaire sous sa forme standard.

On peut toujours passer d'une forme à une autre en introduisant des variables d'écart comme suit :

a- Ramener (PL) de la forme standard à la forme canonique par l'équivalence suivante :

$$Ax = b \Longleftrightarrow Ax \leq b \ et\ Ax \geq b$$

b- Ramener (PL) de la forme canonique à la forme standard comme suit :

$Ax \leq b \Longrightarrow Ax + y = b$ avec $y \geq 0$ ou $Ax \geq b \Longrightarrow Ax - y = b$ avec $y \geq 0$

Définitions et propriétés

- Un vecteur x appartenant à P, $P = \{x \in \mathbb{R}^n \ / \ Ax = b \text{ et } x \geq 0\}$ est appelé solution réalisable de (PL).
- Un vecteur x appartenant à $int(P)$, $int(P) = \{x \in \mathbb{R}^n \ / \ Ax = b \text{ et } x > 0\}$ est appelé solution strictement réalisable de (PL).
- Un vecteur $x \in P$ vérifiant (PL) est appelé solution optimale de (PL).
- Un programme linéaire (PL) réalisable est borné si l'objectif est borné sur P.
- Un point x est un point extrême de P si et seulement si les colonnes $\{A^j : x_j > 0\}$ de A sont linéairement indépendantes. On en déduit que si P est non vide, alors il existe au moins un point extrême.

1.2.5 Dualité en programmation linéaire

Soit un programme linéaire (PL) mis sous sa forme standard, son programme dual (DL) est défini comme suit :

$$(DL) \begin{cases} \max\ b^t y \\ A^t y \leq c, \quad y \in \mathbb{R}^m \end{cases}$$

En effet, la fonction de Lagrange de (PL) est :

$$L(x,y) = c^t x + y^t (b - Ax) \ , \ y \in \mathbb{R}^m$$

qui peut s'écrire aussi sons la forme

$$L(x,y) = \left(c - A^t y\right)^t x \ + b^t y.$$

Donc

$$\min_{x \in \mathbb{R}^n_+} L(x,y) = \begin{cases} b^t y & \text{si} \quad (c - A^t y) \geq 0 \\ -\infty & \text{sinon} \end{cases}$$

D'où le dual de (PL) est $\max\limits_{y \in \mathbb{R}^m} \min\limits_{x \in \mathbb{R}^n_+} L(x,y)$ défini par :

$$(DL) \begin{cases} \max b^t y \\ A^t y \leq c, \ y \in \mathbb{R}^m. \end{cases}$$

Remarque 1.2.6 *On remarque que :*

1. *La matrice des contraintes de (DL) est la transposée de la matrice des contraintes de (PL).*
2. *Le vecteur coût de (PL) est le vecteur du second membre des contraintes de (DL).*
3. *Le vecteur coût de (DL) est le vecteur du second membre des contraintes de (PL).*

Définition 1.2.20 *(Définition du dual dans le cas général)*

Evidemment, on peut définir le dual d'un problème linéaire quelconque (pas nécessairement sous sa forme standard), le tableau suivant résume les correspondances entre le problème primal et son dual et permet d'écrire directement le dual d'un problème linéaire quelconque.

Primal *(min)*	***Dual*** *(max)*
Vecteur coût	*Second membre*
Second membre	*Vecteur coût*
A matrice des contraintes	*A^t matrice des contraintes*
Contrainte $i \geq$	*Variable $y_i \geq 0$*
Contrainte $i =$	*Variable y_i non restreinte*
Variable $x_j \geq 0$	*Contrainte $j \leq$*
Variable x_j non restreinte	*Contrainte $j =$*

Donnons quelques résultats fondamentaux de dualité en programmation linéaire :

Théorème 1.2.17 *(Dualité faible) Si x et y sont respectivement des solutions réalisables pour (PL) et (DL), alors*

$$c^t x \geq b^t y.$$

Théorème 1.2.18 *(Dualité forte) Si $\overline{x}$ et $\overline{y}$ sont respectivement des solutions réalisables pour (PL) et (DL) telles que*

$$c^t \overline{x} = b^t \overline{y},$$

alors $\overline{x}$ est une solution optimale primale de (PL) et $\overline{y}$ est une solution optimale duale de (DL).

Propriétés

- Si l'un des problèmes (PL) et (DL) admet une solution optimale, il en est de même pour l'autre, et leurs valeurs optimales correspondantes sont égales.
- Si l'un des problèmes (PL) et (DL) a une valeur optimale non bornée, l'autre n'a pas de solution optimale.

Remarque 1.2.7

- *Le dual du problème dual (DL) est le problème primal (PL)*
- *On peut transformer un problème de maximisation à un problème de minimisation en écrivant* $\max f = -\min(-f)$.

Chapitre 2

Résolution d'un problème fractionnaire linéaire via la programmation linéaire

Résumé : Dans ce chapitre, on présente une étude détaillée sur le problème de programmation fractionnaire linéaire sous contraintes linéaires. On présente tout d'abord quelques résultats d'existence d'une solution optimale d'un problème d'optimisation soumis à des fonctions généralisées et on cite quelques résultats théoriques sur l'optimisation fractionnaire linéaire, puis on passe aux méthodes de résolution d'un problème de programmation fractionnaire linéaire. On présente deux formulations de (PFL) en (PL). Nous achevons ce chapitre par la résolution numérique du (PFL) par une méthode projective de type de points intérieurs. Des tests numériques seront présentés à la fin de ce chapitre.

Contenu :

2.1 Introduction

Les programmes mathématiques fractionnaires [45], c'est-à-dire les programmes dont l'objectif s'exprime comme le rapport de deux fonctions, linéaires ou non linéaires, en variables réelles, binaires ou mixtes, apparaissent dans plusieurs domaines de la recherche opérationnelle tels que les bases de données, l'ingénierie, l'optimisation combinatoire, la programmation stochastique, l'informatique et l'économie.

Comme domaines d'applications des problèmes fractionnaires soumis à un ensemble de contraintes, nous pouvons citer par exemple :

- Le domaine des finances où il est souvent question d'optimiser un rapport de deux fonctions telles que [dette/capitaux propres], [rendement/employé].
- Le domaine de l'économie offre un large éventail d'applications. En effet, la mesure de l'efficacité des systèmes étudiés s'exprime sous forme de rapports entre les critères techniques et/ou économiques. Par exemple, [rendement/risque], [rendement/investissement], [coût/temps].
- Le domaine de la santé comme la planification dans un hôpital [coût/patient], [infirmière/patient], [docteur/patient]...etc.
- Les processus de décision de Markov peuvent également mener à la maximisation du rapport [moyenne/écart type].

2.2 Formulation du programme fractionnaire

Les programmes fractionnaires consistent à optimiser un objectif mis sous la forme d'un rapport de deux fonctions linéaires ou non linéaires, soumis à un ensemble de contraintes linéaires ou non linéaires.

Soient f, h et $g_i, i = 1, ..., m$, des fonctions réelles définies sur $\mathbb{R}^n$ dans $\mathbb{R}$, avec h ne s'annule pas sur un sous-ensemble X de $\mathbb{R}^n$.

Le problème de programmation fractionnaire (PF) consiste à déterminer un élément $x^* \in X$ optimisant la fonction $\frac{f(x)}{h(x)}$ sur un domaine réalisable $S = \{x \in X \subset \mathbb{R}^n \ / \ g_i(x) = 0, i = 1, ..., m\}$,

c.à.d :

$$(PF) \begin{cases} \min G(x) = \frac{f(x)}{h(x)} \\ x \in S = \{x \in X \subset \mathbb{R}^n / g_i(x) = 0, i = 1, ..., m\} \end{cases},$$

vérifiant les hypothèses suivantes :

- Les fonctions f, h et g_i sont continues sur $\mathbb{R}^n$.
- S est un domaine non vide et borné de $\mathbb{R}^n$.
- $h(x) > 0,\ \forall x \in S$.

Définition 2.2.1 *Le problème de minimisation (PF) est appelé programme fractionnaire convexe-concave si f et g_i pour $i = 1, ..., m$, sont convexes et h est concave.*

Lemme 2.2.1 *Si f est convexe et h linéaire telle que $h(x) > 0$, pour tout $x \in S$, alors $G = \frac{f}{h}$ est quasiconvexe.*

Preuve. $\forall\ \lambda \in [0,1]$, $\forall\ x, y \in S$, on a :

$$\begin{aligned} G((1-\lambda)x + \lambda y) &\leq \frac{(1-\lambda)f(x) + \lambda f(y)}{(1-\lambda)h(x) + \lambda h(y)} \\ &= \frac{(1-\lambda)G(x)h(x) + \lambda G(y)h(y)}{(1-\lambda)h(x) + \lambda h(y)} \\ &\leq \max\{G(x), G(y)\} \frac{(1-\lambda)h(x) + \lambda h(y)}{(1-\lambda)h(x) + \lambda h(y)} \\ &= \max\{G(x), G(y)\}. \end{aligned}$$

D'où le résultat. ■

Lemme 2.2.2 *Si f est convexe et h est linéaire telle que $h(x) > 0,\ \forall x \in S$, et S est compact (fermé et borné), alors le problème (PF) admet au moins une solution optimale.*

Preuve. Il suffit d'appliquer le théorème 1.2.13 ■

2.3 Programmation fractionnaire linéaire

(PF) est dit problème de programmation fractionnaire linéaire si les fonctions f, h et g_i sont linéaires ou affines en x. Dans ce cas (PF) s'écrit ainsi :

$$(PFL) \begin{cases} \min G(x) = \frac{c^t x + \alpha}{d^t x + \beta} \\ x \in P = \{x \in \mathbb{R}^n / Ax = b,\ x \geq 0\}, \end{cases}$$

où $c, d \in \mathbb{R}^n$, $\alpha, \beta \in \mathbb{R}$; $A \in \mathbb{R}^{m \times n}$ de plein rang ; $b \in \mathbb{R}^m$ avec $d^t x + \beta > 0$.

2.3.1 Résultats théoriques sur l'optimisation fractionnaire linéaire

Lemme 2.3.1 *[63] La fonction* $G(x) = \frac{c^t x+\alpha}{d^t x+\beta}$ *est à la fois pseudoconvexe et pseudoconcave.*

Preuve. Prouvons que G est pseudoconvexe. Soient x, y deux points réalisables de P tels que :

$$(y-x)^t \nabla G(x) \geq 0,$$

et montrons que $G(y) \geq G(x)$.

On a

$$\nabla G(x) = \frac{(d^t x+\beta)\, c - (c^t x+\alpha)\, d}{(d^t x+\beta)^{\,2}}.$$

Comme

$$(y-x)^t \nabla G(x) \geq 0 \text{ et } d^t x+\beta > 0,$$

donc

$$(y-x)^t \left(\left(d^t x+\beta\right) c - \left(c^t x+\alpha\right) d\right) \geq 0.$$

On a aussi :

$$(y-x)^t \left(\left(d^t x+\beta\right) c - \left(c^t x+\alpha\right) d\right) = \left(c^t y+\alpha\right)\left(d^t x+\beta\right) - \left(d^t y+\beta\right)\left(c^t x+\alpha\right) \geq 0.$$

D'où

$$\left(c^t y+\alpha\right)\left(d^t x+\beta\right) \geq \left(d^t y+\beta\right)\left(c^t x+\alpha\right).$$

En divisant les deux termes de l'inégalité par l'expression

$$\left(d^t x+\beta\right)\left(d^t y+\beta\right),$$

on obtient :

$$\frac{(c^t y+\alpha)}{(d^t y+\beta)} \geq \frac{(c^t x+\alpha)}{(d^t x+\beta)},$$

ce qui signifie que

$$G(y) \geq G(x),$$

et donc G est pseudoconvexe sur P.

De manière similaire, on montre que G est pseudoconcave sur P. ■

Lemme 2.3.2 *Si P est borné, alors (PFL) admet au moins une solution optimale.*

Preuve. Il suffit d'utiliser le théorème 1.2.13 et le lemme 2.2.2 ■

Quelques conséquences des lemmes et des sous sections précédentes, peuvent être citées.

Conséquences

1. Puisque la fonction objectif G est à la fois pseudoconvexe et pseudoconcave donc, d'après le théorème 1.2.8, elle est aussi quasiconvexe, strictement quasiconvexe, quasiconcave et strictement quasiconcave.
2. Puisque la fonction objectif G est à la fois quasiconvexe et quasiconcave donc elle est quasilinéaire.
3. Puisque la fonction objectif G est strictement quasiconvexe et strictement quasiconcave alors, d'après le théorème 1.2.14, un minimum (maximum) local est aussi un minimum (maximum) global respectivement.
4. Puisque la fonction objectif G est quasiconvexe et quasiconcave et si elle est continue alors, d'après le théorème 1.2.13, au moins une solution optimale du (PFL) est atteinte en un point extrême de P.
5. Puisque la fonction objectif G est pseudoconvexe et pseudoconcave alors, d'après le théorème 1.2.15, un minimum (maximum) local est aussi un minimum (maximum) global respectivement.

2.4 Méthodes de résolution d'un (PFL)

La forme particulière des programmes fractionnaires a fait que de nombreux auteurs ont entrepris d'élaborer des méthodes de résolution spécifiques qui se sont avérées plus efficaces que l'application directe des méthodes générales de programmation non linéaire.

On constate qu'il existe trois grandes techniques de résolution, pour plus de détails voir [45] :

2.4.1 Résolution directe

Dans cette stratégie de résolution, le programme fractionnaire (PF) est traité sous sa forme originale, c'est à dire sans modifier ni l'objectif ni l'ensemble des contraintes P. La méthode la plus connue est celle de A. Cambini et L. Martein [12] qui est une version améliorée de la méthode de B. Martos [43].

Méthode de Cambini et Martein

Cambini et Martein ont considéré le programme fractionnaire linéaire suivant :

$$(PFL') \begin{cases} \max \frac{c^t x+\alpha}{d^t x+\beta} \\ x \in P' = \{x \in \mathbb{R}^n / Ax \leq b,\ x \geq 0\}, \end{cases}$$

où $c, d \in \mathbb{R}^n$; $\alpha, \beta \in \mathbb{R}$; $A \in \mathbb{R}^{m \times n}$ de plein rang ; $b \in \mathbb{R}^m$ et $x \in \mathbb{R}^n$ avec $d^t x + \beta > 0$ sur P' et P' supposé borné non vide.

Définition 2.4.1 *Un point réalisable $\overline{x}$ est une solution niveau optimale pour le problème (PFL') si $\bar{x}$ est une solution optimale pour le programme linéaire suivant :*

$$(P_{\bar{x}}) \begin{cases} \max c^t x + \alpha \\ x \in P' \text{ et } d^t x = d^t \bar{x} \end{cases}$$

Si de plus $\bar{x}$ est un point extrême de P', $\bar{x}$ est dit solution niveau optimale de base.

L'algorithme de Cambini et Martein [12] génére une séquence finie de solutions niveau optimales dont la première x^0 est une solution optimale du programme linéaire suivant :

$$(P_L) \begin{cases} \min d^t x + \beta \\ x \in P', \end{cases}$$

- Si x^0 est unique, alors elle est aussi une solution niveau optimale de base pour (PFL').
- Sinon, résoudre le programme linéaire (P_{x^0}) pour obtenir une solution niveau optimale de base.

2.4.2 Résolution par paramétrisation

À l'inverse de la résolution directe, on construit un problème à objectif simplifié, combinaison linéaire du numérateur et du dénominateur de (PF) par l'intermédiaire d'un paramètre, sans faire aucun changement sur l'ensemble des contraintes. Une séquence de résolutions de ce type de problème fournit une solution du programme fractionnaire. Cette méthode a été utilisée pour les différents programmes fractionnaires linéaires et non linéaires, en variables continues.

Afin de simplifier l'objectif du programme fractionnaire linéaire (PFL), un paramètre est introduit permettant par exemple de ramener un programme fractionnaire en un programme paramétré linéaire, tout en gardant l'ensemble des contraintes inchangé [45]. Ainsi le programme obtenu peut être résolu paramétriquement, une séquence de résolutions de tels programmes à objectif simplifié engendre une suite de solutions convergeant vers une solution optimale du programme fractionnaire linéaire.

Afin d'assurer la convergence de la procédure, l'hypothèse de compacité du domaine défini par les contraintes est adoptée à cette approche.

Le programme paramétré (Q_λ) associé à (PF) consiste à simplifier l'objectif en combinant linéairement le numérateur f et le dénominateur h de l'objectif G par l'intermédiaire d'un paramètre $\lambda \in \mathbb{R}$. Dans le cas d'un programme fractionnaire linéaire, (Q_λ) est un programme linéaire dont l'objectif est une fonction du paramètre λ et il est défini comme suit :

$$(Q_\lambda) \begin{cases} \min v(\lambda) = (c - \lambda d)^t x + (\alpha - \lambda\beta) \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases} . \tag{2.1}$$

En notant λ^* la valeur optimale de (PFL), le résultat fondamental liant le programme fractionnaire au programme paramétré associé est donné par :

Proposition 2.4.1 *[16]*

Toute solution optimale de (Q_λ) est une solution optimale de (PFL).

En tant que fonction de la variable λ la valeur optimale $v(\lambda)$ du programme paramétré vérifie quelques propriétés [16] que nous résumons ci-après :

Proposition 2.4.2 *La fonction $\lambda \longrightarrow v(\lambda)$ est continue, strictement décroissante et convexe avec :*

$v(0) > 0$ et $v(\lambda)$ tend vers $-\infty$ quand λ tend vers $+\infty$.

Si de plus, le programme est fractionnaire, alors v est linéaire par morceaux.

En particulier, l'équation $v(\lambda) = 0$ admet une solution unique λ^*, plus précisément on a :

Proposition 2.4.3 *[16]*

(a) $v(\lambda) = 0 \Longleftrightarrow \lambda = \lambda^*$

(b) $v(\lambda) > 0 \Longleftrightarrow \lambda < \lambda^*$

(c) $v(\lambda) < 0 \Longleftrightarrow \lambda > \lambda^*$

Ainsi la résolution du programme fractionnaire (PFL) revient à trouver la racine de l'équation non linéaire à une seule variable, $v(\lambda) = 0$.

Pour ce faire, un catalogue d'algorithmes de la littérature est présenté dans (Nagih et al.) [45]. Il inclut des algorithmes de résolution itératifs (Newton, interpolation linéaire) et dichotomiques (recherche dichotomique, recherche dichotomique modifiée et interpolation convexe).

2.4.3 Résolution d'un programme équivalent

Un changement de variables permet lui aussi de simplifier l'objectif G du problème (PFL), mais en transportant la difficulté sur l'ensemble des contraintes P [45]. Par exemple, un programme fractionnaire concave-convexe est transformé en un programme concave, et un programme fractionnaire linéaire en un programme à objectif linéaire, avec des contraintes additionnelles.

La transformation du programme fractionnaire linéaire en un programme équivalent à objectif linéaire est obtenue par un changement de variables. Á l'inverse de l'approche paramétrée, ce changement de variables induit l'ajout des contraintes et des variables.

2.5 Formulation de (PFL) en (PL)

Soit le problème de programmation fractionnaire linéaire suivant :

$$(PFL) \begin{cases} \min \frac{c^t x+\alpha}{d^t x+\beta} \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases},$$

où $c, d \in \mathbb{R}^n$; $\alpha, \beta \in \mathbb{R}$; $A \in \mathbb{R}^{m \times n}$ de plein rang ; $b \in \mathbb{R}^m$ et $x \in \mathbb{R}^n$ avec $d^t x + \beta > 0$.

Avant de présenter notre nouvelle formulation, on va d'abord donner une formulation existante proposée par Charnes et Cooper 1962 [15] pour résoudre (PFL) en le transformant en un programme linéaire (PL).

2.5.1 Formulation de Charnes et Cooper (FC)

Charnes et Cooper ont considéré (PFL) en introduisant le changement de variables suivant :

$$y = tx \geq 0 \quad \text{avec} \quad t = \frac{1}{d^t x + \beta} > 0.$$

Ce qui induit aux deux nouvelles données suivantes :

1) Transformation de l'objectif

On a

$$\frac{c^t x + \alpha}{d^t x + \beta} = (c^t x + \alpha)\frac{1}{d^t x + \beta} = (c^t x + \alpha)t.$$

On remplace x par $\frac{y}{t}$, on obtient un objectif linéaire équivalent.

$$\frac{c^t x + \alpha}{d^t x + \beta} = c^t y + \alpha t.$$

2) Transformation des contraintes

Les contraintes de (PFL) sont données par :

$$Ax = b.$$

On remplace x par $\frac{y}{t}$, on obtient

$$A\frac{y}{t} = b \Longleftrightarrow Ay - bt = 0.$$

On a aussi

$$\begin{aligned} d^t x + \beta = \tfrac{1}{t} \quad &\Longleftrightarrow \quad d^t \tfrac{y}{t} + \beta = \tfrac{1}{t} \\ &\Longleftrightarrow \quad d^t y + \beta t = 1. \end{aligned}$$

Alors, (PFL) devient équivalent au programme linéaire suivant :

$$(PL) \begin{cases} \min c^t y + \alpha t \\ Ay - bt = 0 \\ d^t y + \beta t = 1 \\ y, t \geq 0. \end{cases}$$

(PL) peut s'écrire sous la forme standard suivante :

$$(PL)\begin{cases} \min (c,\alpha)^t \begin{pmatrix} y \\ t \end{pmatrix} \\ \begin{pmatrix} A & -b \\ d^t & \beta \end{pmatrix}\begin{pmatrix} y \\ t \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ (y,t) \geq 0. \end{cases}$$

Cette notion d'équivalence entre (PFL) et (PL) est précisée par la proposition suivante :

Proposition 2.5.1 *[15] Si (y^*, t^*) est une solution optimale de (PL), alors $t^* > 0$ et $x^* = \frac{y^*}{t^*}$ est une solution optimale de (PFL).*

Remarque 2.5.1 *L'approche de l'équivalence de Charnes et Cooper nécessite l'augmentation des contraintes et des variables. Le problème linéaire obtenu (PL) est de taille $(m+1, n+1)$ contrairement au (PFL) qui est de taille (m,n).*

Dans cette partie, Nous présentons une nouvelle formulation permettant de transformer (PFL) en un programme linéaire (PL) équivalent tout en gardant la même dimension du problème initial (m,n).

Le problème linéaire obtenu sera traité et résolu par une approche de points intérieurs.

On note que la nouvelle formulation fonctionne bien pour tout $\beta \in \mathbb{R}$; contrairement à d'autres méthodes existantes dans la littérature qui échouent dans le cas $\beta = 0$ [4].

2.5.2 Nouvelle formulation de Bennani et Benterki (FB)

Lemme 2.5.1 *Le programme fractionnaire linéaire (PFL) est équivalent au programme linéaire (PL) suivant :*

$$(PL)\begin{cases} \min c'^t x + \alpha' \\ Ax = b \\ x \geq 0 \end{cases},$$

où

$$c' = c - z^k d \in \mathbb{R}^n,\ z^k = \frac{c^t x^k + \alpha}{d^t x^k + \beta} \in \mathbb{R} \text{ et } \alpha' = \alpha - z^k,\ \beta \in \mathbb{R}.$$

Où z^k est une borne supérieure de la valeur optimale de l'objectif de (PFL) et x^k est une solution strictement réalisable de (PFL) i.e., $Ax^k = b$ et $x^k > 0$.

Preuve. Soit z^* la valeur optimale de (PFL), alors :

$$(PFL)\begin{cases} \min \frac{c^t x+\alpha}{d^t x+\beta} = z^* \\ Ax = b \\ x \geq 0. \end{cases}$$

qui est équivalent à :

$$\begin{cases} \min c^t x + \alpha - z^*(d^t x + \beta) = 0 \\ Ax = b \\ x \geq 0. \end{cases}$$

Ce qui nous donne

$$\begin{cases} \min(c^t - z^* d^t)x + (\alpha - z^*\beta) = 0 \\ Ax = b \\ x \geq 0. \end{cases}$$

Puisque z^* est généralement inconnue, dans l'implémentation numérique on l'approxime à chaque itération par une borne supérieure z^k ; c'est à dire :

$$z^* \leq z^k$$

où

$$z^k = \frac{c^t x^k + \alpha}{d^t x^k + \beta}$$

avec x^k est une solution strictement réalisable connue de (PFL).

Alors, (PFL) devient équivalent au programme linéaire suivant :

$$(PL)\begin{cases} \min c'^t x + \alpha' \\ Ax = b \\ x \geq 0, \end{cases}$$

où :

$$c' = c - z^k d \text{ et } \alpha' = \alpha - z^k \beta.$$

■

Remarque 2.5.2 *Pour résoudre les deux formes des programmes linéaires obtenus précédemment via les deux formulations* **FC** *et* **FB**, *on utilise une approche des méthodes de points intérieurs de type projectif introduite originellement en 1984 par Karmarkar [30]. Cette approche concerne l'algorithme de Ye-Lustig. On présente son algorithme dans la sous-section ci-dessous. Pour plus de détails sur cet algorithme et les méthodes de points intérieurs pour la programmation linéaire voir [41, 42].*

2.5.3 Principe général de la méthode projective de Ye-Lustig

Le programme linéaire traité par Ye-Lustig est le suivant :

$$(PL)\begin{cases} \min c^t x = z^* \\ Ax = b \\ x \geq 0 \end{cases}$$

où $c \in \mathbb{R}^n$; $A \in \mathbb{R}^{m\times n}$ de plein rang $(rang(A) = m < n)$.

Dans ces méthodes, à chaque itération k, on utilise une transformation projective qui ramène la région admissible polyèdrique $\{Ax = b,\ x \geq 0\}$ à un simplexe $S_{n+1} = \left\{x \in \mathbb{R}^{n+1}_+,\ \sum_{i=1}^{n+1} x_i = 1\right\}$.

Cette transformation est définie par : $T_a : \mathbb{R}^n \longrightarrow S_{n+1}$

$$T_a(x_k) = y = \begin{cases} y_i = \dfrac{\frac{x_i^k}{ai}}{1+\sum_{i=1}^{n}\frac{x_i^k}{ai}},\ i = 1, ..., n \\ y_{n+1} = 1 - \sum_{i=1}^{n} y_i, \end{cases}$$

avec $a \in \mathbb{R}^n_+$ une solution strictement réalisable et $T_a(a) = \frac{e_{n+1}}{n+1}$ et $e_{n+1} = (1, ..., 1)^t \in \mathbb{R}^{n+1}$.

Donc, (PL) se transforme via la transformation T_a au problème linéaire suivant :

$$PL(z^*)\begin{cases} \min \begin{pmatrix} D_k c \\ -z^* \end{pmatrix}^t y \\ B_k y = 0 \\ y \in S_{n+1} = \left\{y \in \mathbb{R}^{n+1},\ y \geq 0,\ \sum_{i=1}^{n+1} y_i = 1\right\}. \end{cases}$$

Où $B_k = [AD_k, -b] \in \mathbb{R}^{m\times(n+1)}$ $D_k = diag(x^k)$ matrice diagonale, x^k est une solution strictement réalisable et z^* la valeur optimale de (PL).

Comme le calcul d'une solution optimale d'un programme linéaire, sur une sphère est évident, on introduit à chaque itération la plus grande sphère inscrite dans le simplexe S_{n+1}.

On obtient le sous problème de $PL(z^*)$ suivant :

$$PL_s(z^*)\begin{cases} \min \begin{pmatrix} D_k c \\ -z^* \end{pmatrix}^t y \\ B_k y = 0 \\ \|y - e_{n+1}\|^2 \leq r^2 < 1 \end{cases},$$

où

- $r = \frac{1}{\sqrt{n(n+1)}}$ est le rayon de la sphère.
- $e_{n+1} = (1, ..., 1)^t \in \mathbb{R}^{n+1}$.

Comme la valeur de z^* est généralement inconnue, Ye-Lustig approxime z^* à chaque itération par des bornes supérieures $z^k = c^t x^k \geq z^*$.

Donc, $PL_s(z^*)$ devient PL_k :

$$PL_k\begin{cases} \min \begin{pmatrix} D_k c \\ -c^t x^k \end{pmatrix}^t y \\ B_k y = 0 \\ \|y - e_{n+1}\|^2 \leq r^2 < 1. \end{cases}$$

En utilisant les conditions d'optimalité de (K.K.T), la solution optimale est donnée par :

$$y^{(k+1)} = \frac{e_{n+1}}{n+1} - \alpha^k r d_k.$$

Où

- α^k est le pas de déplacement ; $0 < \alpha^k < 1$.
- d_k est la direction telle que $d_k = \frac{P_k}{\|P_k\|}$; avec P_k la projection du vecteur coût $(D_k c \quad - c^t x^k)^t$ de (PL_k) sur le noyau de la matrice des contraintes B_k définie par :

$$P_k = \left[I - B_k^t \left(B_k B_k^t\right)^{-1} B_k\right] \begin{pmatrix} D_k c \\ -c^t x^k \end{pmatrix}$$

On revient au problème initial (PL) par la transformation inverse T_a^{-1}, on obtient un candidat tel que :

$$x^{k+1} = T_a^{-1}\left(y^{(k+1)}\right) = \frac{D_k y^{(k+1)}[n]}{y_{n+1}^{(k+1)}}$$

Calcul d'une solution réalisable initiale

Le calcul d'une solution réalisable initiale, est un problème difficile qui se manifeste dans les différentes variantes des méthodes de points intérieurs. Plusieurs alternatives sont proposées pour le régler. Dans notre cas, on s'est intéressé à la procédure de la variable artificielle.

En effet, trouver une solution strictement réalisable pour (PL) revient à résoudre le problème suivant :

$$(F)\begin{cases} Ax = b \\ x > 0 \end{cases},$$

qui est équivalent au problème auxiliaire suivant :

$$(APL)\begin{cases} \min \lambda \\ Ax + \lambda(b - Ax^0) = b;\ x^0 \in \mathbb{R}^n_+ \text{ (arbitraire)} \\ (x, \lambda) \geq 0. \end{cases}$$

Le problème (APL) peut s'écrire aussi sous la forme :

$$(APL)\begin{cases} \min \bar{c}^t \bar{x} \\ B\bar{x} = b \\ \bar{x} \geq 0 \end{cases},$$

où

– $\bar{c} = (0, 0, ..., 1)^t \in \mathbb{R}^{n+1}$.

– $B = [A \quad b \quad -Ax^0] \in \mathbb{R}^{(n+1)\times m}$.

– $\bar{x} = (x, \lambda)^t \geq 0$, tel que $\bar{x} \in \mathbb{R}^{n+1}$.

Le problème (APL) possède une solution strictement réalisable triviale $\bar{x} = (x^0, 1)^t,\ x^0 \in \mathbb{R}^n_+$ (l'orthant positif), exemple $x^0 = (1, 1, ..., 1)^t$.

Le problème (APL) est équivalent au problème (F), au sens :

Lemme 2.5.2 *(x^*, λ^*) est une solution optimale de (APL) si et seulement si x^* est une solution de (F), (avec $\lambda^* \leq \varepsilon$, ε étant une très petite précision positive donnée). En pratique, si λ^* reste loin de zéro, on conclue que le problème (F) n'a pas de solution et alors (PL) est non réalisable.*

Maintenant, on résume l'algorithme de Ye-Lustig.

Description de l'algorithme :

Début d'algorithme

- **Initialisation**

  $k = 0$, $x^0 > 0$ strictement réalisable ($Ax^0 = b,\ x^0 > 0$) pour (PL), ε une précision donnée.

  - **Tant que** $\frac{\|P_k\|}{|c^t x^0|} > \varepsilon$ faire :
    - **Etape 0 :** (Construire la matrice des contraintes)

$$B_k = [A_k\ ,\ -b] \text{ où } A_k = AD_k \text{ et } D_k = diag(x^k)$$

    - **Etape 1 :** (Calculer la projection P_k)

$$P_k = \left[I - B_k^t \left(B_k B_k^t\right)^{-1} B_k\right] \begin{pmatrix} D_k c \\ -c^t x^k \end{pmatrix} \text{ et poser } d_k = \frac{P_k}{\|P_k\|}$$

    - **Etape 2 :** Calculer l'itéré suivant :

$$y^{k+1} = \frac{e_{n+1}}{n+1} - \alpha_k r d_k, \quad r = \frac{1}{\sqrt{n\,(n+1)}} \quad \text{avec} \quad 0 < \alpha_k < 1$$

      α_k est le pas de déplacement.
    - **Etape 3 :** (Revenir au problème initial (PL) par T_a^{-1})

$$x^{k+1} = T_a^{-1}\left(y^{(k+1)}\right) = \frac{D_k y^{(k+1)}\,[n]}{y_{n+1}^{(k+1)}} \quad , \quad k = k+1$$

  - **Fin tant que**

Fin d'algorithme.

2.6 Expérimentations numériques

Les tests numériques réalisés sont effectués sur des exemples de la littérature. La précision choisie étant $\varepsilon = 10^{-6}$. On résout les exemples (à taille fixe et variable) via les deux formulations équivalentes **FC** et **FB** de (PFL) afin de distinguer l'efficacité de l'une par rapport à l'autre. Dans chaque exemple, on donne la solution optimale x^* trouvée par l'algorithme de Ye-Lustig et la valeur optimale z^* ainsi que le nombre des itérations qu'on note k et le temps d'exécution nécessaire pour trouver une solution optimale qu'on note *temps* en secondes.

Signalons que lorsqu'il s'agit d'un programme fractionnaire linéaire à contraintes d'inégalités, on utilise des variables d'écart pour l'écrire sous sa forme standard. De même s'il s'agit d'un problème de maximisation la relation suivante permet de le convertir en un problème de minimisation : $\max(f(x) = -\min(-f(x))$.

2.6.1 Exemples à taille fixe

Exemple 2.6.1 *On considère le programme fractionnaire linéaire suivant :*

$$(PFL1)\begin{cases} \min\left(\frac{-2x_1+x_2+2}{x_1+3x_2+4}\right) \\ -x_1+x_2 \leq 4 \\ 2x_1+x_2 \leq 14 \\ x_2 \leq 6 \\ x_1, x_2 \geq 0. \end{cases}$$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 4$ itérations

$$x^* = (6.99999, 0.00000)^t.$$

La valeur optimale est $z^* = 1,09090$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 5$ itérations

$$x^* = (7.00000, 0.00000)^t.$$

La valeur optimale est $z^* = 1,09090$.

Exemple 2.6.2 *[4] On considère le programme fractionnaire linéaire suivant :*

$$(PFL2)\begin{cases} \max\left(\frac{x_1+2x_2+3.5x_3+x_4+1}{2x_1+2x_2+3.5x_3+3x_4+4}\right) \\ 2x_1+x_2+3x_3+3x_4 \leq 10 \\ x_1+2x_2+x_3+x_4 \leq 14 \\ x_1, x_2, x_3, x_4 \geq 0. \end{cases}$$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 7$ itérations

$$x^* = (0.00001, 6.00004, 1.19984, 0.00002)^t.$$

La valeur optimale est $z^* = 0,85713$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 6$ *itérations*

$$x^* = (0.00000, 6.39999, 1.20000, 0.00000)^t.$$

La valeur optimale est $z^* = 0,85714$.

Exemple 2.6.3 *On considère le programme fractionnaire linéaire suivant :*

$$(PFL3)\begin{cases} \max\left(\frac{x_1+2x_2+4x_3+5x_4+8x_5}{2x_1+5x_2+3x_3+4x_4+6x_5+1}\right) \\ 2x_1 + x_2 + 3x_3 + x_4 + x_5 \leq 15 \\ x_1 + 2x_2 + x_5 \leq 8 \\ 3x_{1+}5x_2 + 2x_4 \leq 10 \\ 2x_2 + 4x_3 + 3x_4 + x_5 \leq 21 \\ x_1, x_2, x_3, x_4, x_5 \geq 0. \end{cases}$$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 7$ *itérations*

$$x^* = (0.00001, 0.00002, 3.66665, 0.00059, 3.99924)^t.$$

La valeur optimale est $z^* = 1,29628$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 13$ *itérations*

$$x^* = (0.00000, 0.00000, 2.33333, 0.00001, 7.99999)^t.$$

La valeur optimale est $z^* = 1,30952$.

Exemple 2.6.4 *[45] On considère le programme fractionnaire linéaire suivant :*

$$(PFL4)\begin{cases} \min\left(\frac{\sum_{j=1}^{n} c_j x_j}{\sum_{j=1}^{n} d_j x_j}\right) \\ \sum_{j=1}^{n} x_j = N \\ 0 \leq x_j \leq 1, \ j = 1, ..., n. \end{cases}$$

1) n=5

$c^t = (2, -1, -3, 5, -2),\ d^t = (1, 2, 2, 3, 4),\ N = 3$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 8$ *itérations*

$$x^* = (0.00000, 0.99999, 0.99999, 0.00000, 0.99999)^t.$$

La valeur optimale est $z^* = -0, 74999$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 5$ *itérations*

$$x^* = (0.00000, 1.00000, 1.00000, 0.00000, 1.00000)^t.$$

La valeur optimale est $z^* = -0, 74999$.

2) n=10

$c^t = (1, 1, 1, 2, 2, 3, 0, 0, 1, 1),\ d^t = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1),\ N = 5$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 5$ *itérations*

$$\begin{aligned} x^* \ = \ & (0.59999, 0.59999, 0.59999, 0.00000, 0.00000, 0.00001, \\ & 0.99999, 0.99999, 0.59999, 0.59999)^t. \end{aligned}$$

La valeur optimale est $z^* = 0, 60000$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 5$ *itérations*

$$\begin{aligned} x^* \ = \ & (0.60000, 0.60000, 0.60000, 0.00001, 0.00001, 0.00001, \\ & 1.00000, 1.00000, 0.60000, 0.60000)^t. \end{aligned}$$

La valeur optimale est $z^* = 0, 60000$.

3) n=15

$c^t = (1, 2, 3, 3, 0, 0, 1, 6, 0, 4, 0, 1, 5, 2, 1),$

$d^t = (1, 2, 2, 4, 6, 1, 1, 0, 0, 5, 8, 2, 3, 1, 1),\ N = 8$

Résultats obtenus via* FC *:

Solution optimale trouvée après $k = 10$ *itérations*

$$\begin{aligned} x^* \ = \ & (0.99995, 0.00009, 0.00000, 0.00003, 0.99998, 0.99998, 0.99995, 0.00000, \\ & 0.99999, 0.00001, 0.99998, 0.99998, 0.00000, 0.00002, 0.99995)^t. \end{aligned}$$

La valeur optimale est $z^* = 0,20001$.

Résultats obtenus via* FB *:

Solution optimale trouvée après $k = 7$ *itérations*

$$x^* = (0.99994, 0.00019, 0.00000, 0.00003, 1.00000, 0.99999, 0.99994, 0.00000, 0.99998, 0.00001, 0.99999, 0.99997, 0.00000, 0.00002, 0.99994)^t.$$

La valeur optimale est $z^* = 0,20001$.

Le tableau suivant résume les résultats obtenus via les 2 formulations **FC** et **FB** pour les exemples testés précédement. On note que le temps d'execution est négligeable pour les exemples précédents.

$Exemple$	$Taille(m,n)$	k(**FC**)	k(**FB**)
$PFL1$	$(3,2)$	4	5
$PFL2$	$(2,4)$	7	6
$PFL3$	$(4,5)$	7	13
$PFL4$	$(6,5)$	8	5
	$(11,10)$	5	5
	$(16,15)$	10	7

2.6.2 Exemples à taille variable

Exemple 2.6.5 *On considère le programme fractionnaire linéaire à taille variable suivant :*

$$(PFL_m) \quad \begin{cases} \max \frac{c^t x+2m}{d^t x+1}, \\ Ax = b \\ x \geq 0 \end{cases}$$

$$o\grave{u} \quad \begin{cases} c_i = \begin{cases} -1+2m, & i = 1,...,m \\ 2m, & i = m+1,...,n \end{cases} \\ d_i = 1, \qquad i = 1,...,n \qquad n = 2m \\ A[i,i] = A[i,m+i] = 1, \quad i = 1,...,m \\ b_i = 2, \qquad i = 1,...,m. \end{cases}$$

La solution optimale obtenue par les deux formulations étant :

$$x_i^* = \begin{cases} 2 & i = 1, ..., m \\ 0 & i = m+1, ..., n. \end{cases}$$

Le tableau suivant résume les résultats obtenus via les 2 formulations **FC** et **FB** suivant des valeurs différentes de m.

$Taille$	**FC**		**FB**	
(m, n)	k	$temps$	k	$temps$
$(50, 100)$	3	$*$	3	$*$
$(100, 200)$	3	$*$	3	$*$
$(200, 400)$	2	1	3	$*$
$(300, 600)$	2	1	3	1
$(500, 1000)$	2	7	3	3
$(1000, 2000)$	2	52	3	29

La notation $*$ designe que le temps d'exécution dans ce cas est négligeable.

Commentaires

A travers les tests numériques et pour les différentes dimensions, les résultats montrent l'efficacité de la nouvelle formulation **FB** en particulier pour les exemples de grande taille; exprimée soit par la réduction de nombre des itérations ou le temps d'exécution.

L'introduction des méthodes de points intérieurs pour résoudre les programmes linéaires connues par son efficacité, reste préservée pour la programmation fractionnaire linéaire pour les exemples à grande taille.

Conclusion

Dans ce chapitre on a proposé une nouvelle formulation adéquate permettant de transformer (PFL) en un (PL) équivalent, puis on a résolu ce dernier en utilisant une méthode projective de points intérieurs de Ye-Lustig. Les résultats obtenus sont très encouragents et montrent clairement l'efficacité des méthodes de points intérieurs. L'ensemble de ces résultats a fait l'objet d'une publication à paraitre dans la revue International Journal of Computing Science and Mathematics [8].

Dans le prochain chapitre on résout à nouveau (PFL) via un problème d'inégalités variationnelles équivalent (VIP).

Chapitre 3

Résolution d'un problème fractionnaire linéaire via les inégalités variationnelles

Résumé : Dans ce chapitre, on présente une étude détaillée sur le problème d'inégalités variationnelles (VIP). On présente tout d'abord la formulation de ce problème suivie par quelques résultats d'existence et d'unicité, puis on passe à une classe très importante des méthodes de résolution dites de projection. En se basant sur le lien existant entre (PFL) et (VIP), on arrive à écrire (PFL) sous forme d'un (VIP) équivalent qu'on résout par une méthode de projection. Des tests numériques seront présentés à la fin de ce chapitre.

Contenu :

3.1 Introduction

Le problème d'inégalités variationnelles (VIP) dans son cadre général, a fait sa première apparition au milieu des années soixante. Il a été introduit originalement par Hertman et Stampacchia (1966) dans le domaine des équations aux dérivées partielles (EDP). Un nombre considérable de ces équations a été formalisé et résolu par cette méthodologie.

Ce problème traduit des modèles pratiques très intéressants.

Le problème (VIP) est d'une importance primordiale, car il est considéré comme étant le modèle approprié pour de nombreux problèmes pratiques tels que :

- Modèles d'équilibre en économie.
- Réseaux de transport en recherche opérationnelle.
- Des problèmes mécaniques.
- La théorie des jeux.

D'autre part, il englobe plusieurs problèmes mathématiques trés importants, du fait qu'il représente une formulation unificatrice de ces derniers tels que :

- Systèmes d'équations non linéaires.
- Problème de complémentarité.
- Certains problèmes d'optimisation.

3.2 Position du problème d'inégalités variationnelles

On commence cette partie par la définition du problème des inégalités variationnelles.

Définition 3.2.1 *Soit $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ un opérateur et C un convexe fermé non vide de $\mathbb{R}^n$. Le problème d'inégalités variationnelles noté (VIP) ou bien $VIP(F, C)$, consiste à :*

$$(VIP) \begin{cases} \text{Trouver } \overline{x} \in C \text{ tel que} \\ \langle F(\overline{x}), x - \overline{x} \rangle \geq 0 \quad \forall\, x \in C. \end{cases}$$

Notons par $Sol(VIP)$ l'ensemble des solutions associées au problème (VIP).

Dans cette partie, on s'intèresse principalement aux problèmes d'inégalités variationnelles sous contraintes linéaires où C est un convexe fermé de $\mathbb{R}^n$ de la forme $C = \{x \in \mathbb{R}^n / Ax = b\}$ avec A une matrice de $\mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$.

3.2.1 Résultats théoriques sur l'existence et l'unicité d'une solution de $VIP(F, C)$

Les résultats d'existence et d'unicité relatifs aux problèmes $VIP(F, C)$ sont inspirés de la théorie fondamentale de l'optimisation. Ils dépendent évidemment des propriétés de l'opérateur F et de l'ensemble C.

Alors avant de donner les principaux résultats, il convient de rappeler quelques notions utiles à savoir : la hémi-continuité inférieure et la coercivité des opérateurs.

On considère dans ce qui suit, $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ un opérateur et C un sous ensemble convexe non vide de $\mathbb{R}^n$.

Définition 3.2.2 *F est dit hémi-continu sur C si : $\forall x, y \in C$, et $t \in [0, 1]$, l'application unidimensionnelle :*

$$t \longrightarrow (y - x)^t F(x + t(y - x)) \text{ est continue à droite du point } t = 0.$$

Définition 3.2.3 *F est dit coercif sur C s'il existe $x_0 \in C$ tel que :*

$$\lim_{\substack{\|x\|\longrightarrow+\infty \\ x\in C}} \frac{\langle F(x), x - x_0\rangle}{\|x\|} = \infty.$$

Théorème 3.2.1 *[34] Soit C un convexe compact et F un opérateur continu sur C, alors il existe une solution de $VIP(F, C)$.*

Théorème 3.2.2 *[3] Soit C un convexe compact et F un opérateur monotone, hémi-continu et coercif sur C, alors $VIP(F, C)$ admet au moins une solution.*

Proposition 3.2.1 *[20] Soit C un convexe fermé et F un opérateur monotone et continu sur C, alors $\overline{x}$ est une solution de $VIP(F, C)$ si et seulement si :*

$$\langle F(x), x - \overline{x}\rangle \geq 0, \ \forall x \in C.$$

De plus, l'ensemble des solutions de $VIP(F, C)$ est un ensemble convexe fermé non vide.

Remarque 3.2.1 *Ce résultat a été démontré à l'origine par Minty pour les problèmes (VIP) monotones dans les espaces de Hilbert [46], et par la suite, il a été généralisé par Karamardian pour le cas où F est pseudomonotone [29].*

Proposition 3.2.2 *[20] Soit C un convexe fermé et F un opérateur strictement monotone, alors $VIP(F, C)$ admet au plus une solution.*

Remarque 3.2.2 *[20] La forte monotonie est une propriété assez forte qui implique la coercivité et la stricte monotonie. Par conséquent, on peut combiner les deux propositions précédentes pour garantir à la fois l'existence et l'unicité de la solution de $VIP(F, C)$.*

Corollaire 3.2.1 *Soit C un convexe fermé et F fortement monotone et continu sur C, alors il existe une solution unique pour $VIP(F, C)$.*

3.3 Méthodes de résolution d'un (VIP)

Depuis leur apparition, les problèmes d'inégalités variationnelles ont attiré l'attention de plusieurs chercheurs et ils n'ont cessé guère de faire l'objet d'études intensives de grande qualité dans le but de trouver une théorie riche et élaborer des algorithmes convenables pour leur résolution.

Pour résoudre un problème variationnel général, on peut distinguer cinq catégories fondamentales :

La première représente les algorithmes de projection, ces derniers sont basés essentiellement sur la reformulation du problème originel en un problème de point fixe.

La deuxième catégorie concerne les techniques d'optimisation développées pour la première fois par Auslender [3]. Son principe est de construire à partir du problème variationnel un problème d'optimisation équivalent (différentiable ou non différentiable, avec contraintes ou sans contraintes) équivalent, ensuite on résout ce dernier au lieu du problème d'origine comme dans le papier de Z. Kebaili et M. Achache [31].

La troisième comprend les méthodes de points intérieurs [13, 22, 23] dont la plus célébre est celle de Censor. En premier temps, ces méthodes ont été proposées comme des alternatives après les difficultés théoriques et numériques rencontrées dans les premières méthodes de projection développées.

La quatrième catégorie est particulièrement conçue pour la classe de $VIP(F, C)$ à contraintes d'égalités ou d'inégalités (linéaires ou non linéaires), cette catégorie est bien connue sous le nom : méthodes de directions alternées.

Pour la dernière catégorie, il s'agit des méthodes de pénalité où (VIP) en question est transformé à un problème d'égalités variationnelles (VEP), cette idée est inspirée principalement de l'optimisation. Pour les travaux relatifs à cette catégorie, on trouve celui de Auslender, Z. Kebaili et D. Benterki [32].

Il faut quand même signaler que les premières méthodes de ces catégories présentent des inconvénients d'ordre théorique et numérique majeurs ; les hypothèses de convergence trop fortes, le coût excessif des projections nécessaires à effectuer à chaque itération et le choix de la procédure de recherche linéaire appropriée pour déterminer un pas de déplacement.

Les méthodes basées sur les techniques d'optimisation en particulier les méthodes de points intérieurs ont rivalisé les méthodes de projection car elles offrent un aspect algorithmique plus performant.

3.3.1 Méthodes de projection

Ces méthodes sont basées essentiellement sur la reformulation de (VIP) en un problème du point fixe.

Plus tard, les méthodes de projection ont réapparu une autre fois où elles ont connu un fort relancement après les travaux réalisés par Iusem [27], Solodov et Svaiter [56], Wang et al. [60], **Grar et Benterki** [24]. Généralement les algorithmes proposés dans ces derniers travaux possèdent des propriétés théoriques meilleures avec des hypothèses assez relaxées et un comportement numérique performant où le coût des calculs a été réduit d'une manière très significative.

Dans ce qui suit, on donne un survol sur les principaux travaux réalisés dans cet axe ainsi qu'une description détaillée de la méthode de projection de Solodov et Svaiter [56] et un algorithme amélioré de ce dernier donné par **Grar et Benterki** [24] qui fait l'objet de cette étude.

Procédure de base des méthodes de projection

La procédure de base pour ce type de méthodes a été développée à l'origine au début des années 70 comme conséquence systématique de la reformulation de

$VIP(F,C)$ en un problème du point fixe.

$$\overline{x} = Proj_C(\overline{x} - \alpha F(\overline{x})) \Longleftrightarrow \overline{x} \in Sol(VIP(F,C)).$$

Le schéma itératif de cette procédure sera défini par :

$$x^{k+1} = Proj_C(x^k - \alpha F(x^k)),\ \alpha > 0. \tag{3.1}$$

Cet algorithme converge globalement vers $\bar{x}$ sous des conditions assez fortes, à savoir l'opérateur F doit être fortement monotone et satisfait la condition de Lipschitz.

On rappelle que :

F satisfait la condition de Lipschitz s'il existe $L \geq 0$, tel que :

$$\|F(x) - F(y)\| \leq L \|x - y\| \ \forall\ x, y \in C.$$

Si les conditions précédentes sont satisfaites, alors la suite générée par (3.1) converge vers la solution unique de $VIP(F,C)$ qui est le point fixe de l'opérateur contractant $Proj_C(I - \alpha F)$ pour $\alpha \in \left]0, \frac{2\mu}{L^2}\right[$, où μ est le coefficient de la forte monotonie et L la constante de Lipschitz.

Méthode de Korpolevich

C'est une variante de la procédure de base qui a été proposée par Korpolevich connue également par le nom : algorithme de l'extragradient [35]. Ce dernier converge si F satisfait la condition de Lipschitz et non nécessairement fortement monotone.

Pour cette alternative, on fait appel à deux projections à chaque itération ce qui rend en revanche le schéma itératif correspondant suivant plus coûteux.

$$\begin{cases} y^k = Proj_C(x^k - \alpha F(x^k)) \\ x^{k+1} = Proj_C(x^k - \alpha F(y^k)). \end{cases} \tag{3.2}$$

La preuve de convergence de (3.2) ne repose plus sur un argument de contraction, mais on a pour les valeurs de $\alpha \in \left]0, \frac{1}{L}\right[$, le vecteur $(x^k - F(y^k))$ est la projection de x^k sur un hyperplan qui sépare x^k de l'ensemble des solutions $Sol(VIP)$, ensuite x^{k+1} est obtenu à l'aide d'une autre projection sur C. Autrement dit, la distance entre le point courant x^k et l'ensemble $Sol(VIP)$ décroît à chaque itération, d'où la décroissance monotone de la suite

$$\left\{\left\|x^k - \overline{x}\right\|\right\},\ \forall \overline{x} \in Sol(VIP).$$

On rappelle que si $Sol(VIP)$ est non vide, alors il est également convexe fermé, ce qui justifie les opérations de séparation.

Méthode de Korpolevich modifiée

Dans le même ordre d'idée, les chercheurs ont pu éliminer l'hypothèse de Lipschitz, car il n'est pas facile de vérifer cette propriété pour tout opérateur ni d'estimer la valeur de la constante L. Même dans le cas où L est connue, elle risque d'être une valeur assez grande de sorte que l'intervalle $\left]0, \frac{1}{L}\right[$ soit trop restreint pour donner des valeurs convenables de α, ce qui entraînera une convergence lente.

Pour surmonter cet obstacle, Iusem [27] a proposé une procédure de recherche linéaire dite de relaxation inspirée par des méthodes d'optimisation classiques tout en gardant le même principe de la méthode de Korpolevich [35].

Le schéma itératif est le suivant :

$$\begin{cases} y^k = Proj_C(x^k - \alpha_k F(x^k)) \\ x^{k+1} = Proj_C(x^k - \lambda_k F(y^k)) \end{cases}. \tag{3.3}$$

Et la procédure de recherche linéaire pour déterminer α_k est donnée par :

$$\begin{cases} \text{Soit } \varepsilon_1 \in]0,1[\,; \text{ on démmare de } y^k = Proj_C(x^k - \alpha_{\max} F(x^k)) \\ \text{Si } \left\|F(y^k) - F(x^k)\right\| \leq \frac{\left\|y^k - x^k\right\|}{2\alpha_{\max}\left\|F(x^k)\right\|^2}, \text{ on prend } \alpha_k = \alpha_{\max} \\ \text{Sinon, trouver } \alpha \in]0, \alpha_{\max}[\text{ tel que :} \\ \varepsilon_1 \frac{\left\|y^k - x^k\right\|}{2\alpha_{\max}\left\|F(x^k)\right\|^2} \leq \left\|F(Proj_C(x^k - \alpha F(x^k))) - F(x^k)\right\| \leq \frac{\left\|y^k - x^k\right\|}{2\alpha_{\max}\left\|F(x^k)\right\|^2}. \\ \text{Alors, on prend } \alpha_k = \alpha \text{ et on calcule } y^k = Proj_C(x^k - \alpha_k F(x^k)) \end{cases}$$

Le principe est de démarrer d'une valeur initiale $\alpha_{\max}$ ensuite la réduire de telle façon que la nouvelle valeur obtenue satisfait la double inégalité donnée dans la procédure de recherche linéaire. Ainsi, cette procédure nous garanti que l'hyperplan H_k de normale $F(y^k)$ supposée non nulle passant par y^k sépare x^k de l'ensemble $Sol(VIP)$, et $(x^k - \lambda_k F(y^k))$ est la projection orthogonale de x^k sur H_k.

Dans ce cas, l'hyperplan H_k est défini par :

$$H_k = \left\{x \in \mathbb{R}^n : \left\langle F(y^k), x - y^k \right\rangle = 0\right\}.$$

Et un calcul simple nous donne l'expression du deuxième pas :

$$\lambda_k = \frac{\langle F(y^k), x^k - y^k \rangle}{\|F(y^k)\|^2}.$$

De même, ceci implique la décroissance monotone de la suite

$$\left\{ \left\| x^k - \overline{x} \right\| \right\}, \ (\forall \overline{x} \in Sol(VIP)).$$

Il est clair qu'avant de trouver le pas de déplacement α_k satisfaisant la procedure de recherche linéaire proposée par Iusem, il est nécessaire d'évaluer $Proj_C(x^k - \alpha_k F(x^k))$. Ceci signifie que si la recherche linéaire à l'itération k exige m_k étapes, alors nous devrons évaluer m_k projections sur C afin de trouver y^k, plus évidemment une autre projection pour le calcul de x^{k+1}, ce qui rend le calcul très coûteux.

Méthode de Iusem et Svaiter dite de double projection

Dans le but de réduire le nombre des projections nécessaires à calculer, Iusem et Svaiter [28] ont proposé une nouvelle variante des méthodes de projection pour la résolution de $VIP(F, C)$ monotone. Cette variante est basée sur l'introduction d'une procédure de recherche linéaire plus appropriée pour déterminer α_k tout en assurant la séparation de x^k de $Sol(VIP)$ qui ne nécessite que deux projections à chaque itération d'où le nom de la méthode. Ceci mènera certainement à une réduction considérable en volume calculatoire.

Description de l'algorithme de Iusem et Svaiter

Début algorithme

- **Initialisation**

  $k = 0$, $x^0 \in C$, $\delta \in (0, 1)$, $0 < \widehat{\beta} \leq \widetilde{\beta}$, une suite $\{\beta_k\} \subset [\widehat{\beta}, \widetilde{\beta}]$.

  ε une précision donnée.

  Calculer $z^k = Proj_C(x^k - \beta_k F(x^k))$.

- **Tant que** $\left\| x^k - z^k \right\| > \varepsilon$ faire :
  - **Etape 0 :** Trouver j le plus petit entier positif tel que

$$\left\langle F(2^{-j} z^k) + (1 - 2^{-j}) x^k), x^k - z^k \right\rangle \geq \frac{\delta}{\beta_k} \left\| x^k - z^k \right\|^2.$$

  - **Etape 1 :** Prendre $\alpha_k = 2^{-j}$.

– **Etape 2 :** Poser

$$y^k = \alpha_k z^k + (1 - \alpha_k)x^k.$$

– **Etape 3 :** Calculer

$$\lambda_k = \frac{\langle F(y^k), x^k - y^k \rangle}{\|F(y^k)\|^2}.$$

– **Etape 4 :** Calculer l'itéré suivant

$$x^{k+1} = Proj_C(x^k - \lambda_k F(y^k)).$$

– **Etape 5** : Poser $k = k + 1$ et réitérer.

– **Fin tant que.**

Fin algorithme.

Méthode de Solodov et Svaiter

Quelques années plus tard, en se basant essentiellement sur l'idée de Iusem et Svaiter concernant la méthode de double projection [28], Solodov et Svaiter [56] ont développé une autre méthode de projection qui converge sous des hypothèses assez faibles, la continuité et la pseudomonotonie de F, même moins que la pseudomonotonie, on exige seulement que la propriété suivante soit vérifiée :

$$\langle F(x), x - \overline{x} \rangle \geq 0, \ \forall \ x, y \in C \text{ et } \overline{x} \ \in Sol(VIP(F, C)) \tag{3.5}$$

On présente maintenant, l'algorithme de Solodov et Svaiter [56] pour un (VIP) général.

Description de l'algorithme de Solodov et Svaiter

Début algorithme

– **Initialisation**

$k = 0$, $x^0 \in C$, $\gamma, \sigma \in (0, 1)$, $\alpha_{-1} > 0$, $\theta > 0$, et ε une précision donnée.

Prendre $\beta_k = \min \{\theta \alpha_{k-1}, 1\}$.

Calculer $z^k = Proj_C(x^k - \beta_k F(x^k))$.

Calculer $r(x^k, \beta_k) = x^k - z^k$.

– **Tant que** $\left\|r(x^k, \beta_k)\right\| > \varepsilon$ faire :

– **Etape 0 :** Trouver j le plus petit entier positif tel que

$$\left\langle F(x^k - \gamma^j \beta_k r(x^k, \beta_k)), r(x^k, \beta_k) \right\rangle \geq \frac{\sigma}{\beta_k} \left\|r(x^k, \beta_k)\right\|^2.$$

- **Etape 1 :** Prendre $\alpha_k = \gamma^j \beta_k$.
- **Etape 2 :** Poser
$$y^k = x^k - \alpha_k r(x^k, \beta_k).$$
- **Etape 3 :** Calculer l'itéré suivant
$$x^{k+1} = Proj_{C \cap D_k}(x^k),$$
où :
$$D_k = \left\{ x \in \mathbb{R}^n / \left\langle F(y^k), x - y^k \right\rangle \leq 0 \right\}.$$
- **Etape 4 :** Poser $k = k + 1$ et réitérer.

– **Fin tant que.**

Fin algorithme.

Remarque 3.3.1 *Le symbole α_{-1} désigne une valeur initiale, elle est donnée sous cette forme, car on a besoin de α_{k-1} et α_k à chaque itération. Donc, pour la première itération, on a α_{-1} donnée comme une valeur initiale puis on calcule α_0, et ainsi de suite.*

Propriétés de la méthode de Solodov et Svaiter Les majeurs aménagements apportés par ces chercheurs se résument dans les points suivants :

* La convergence de l'algorithme de Iusem et Svaiter dépend en grande partie du choix de la suite $\{\beta_k\}$. A cet effet, Solodov et Svaiter ont proposé une autre technique pour déterminer la suite en question afin d'améliorer la performance algorithmique.

* Pour avoir le point y^k, on procède d'une manière presque analogue que l'algorithme précédent, où une légère modification est introduite au niveau de la procédure de recherche linéaire et l'opération de projection s'effectue toujours sur C.

* En ce qui concerne la dernière étape, autrement dit déterminer le nouvel itéré x^{k+1} on change la région de projection. Cette fois, on doit la calculer sur l'intersection de C et D_k où D_k désigne le demi plan contenant l'ensemble des solutions de $VIP(F, C)$ (on signale dans ce cas, que la frontière de D_k est l'hyperplan H_k défini auparavant). Cette opération a pour intérêt de déterminer un itéré plus proche de l'ensemble des solutions que celui calculé par l'algorithme de double projection.

Donnons le résultat suivant qui caractérise le calcul du nouvel itéré, ainsi que le théorème principal de convergence pour cette méthode.

Lemme 3.3.1 *[56] On suppose que la procédure de recherche linéaire pour déterminer le paramètre α_k est bien définie. Alors, on a :*

$$x^{k+1} = Proj_{C\cap D_k}(\overline{x}^k) \quad où \quad (\overline{x}^k) = Proj_{D_k}(x^k).$$

On note que la projection d'un point quelconque x^k n'appartenant pas au demi plan D_k sur D_k est la même que sur l'hyperplan H_k(la frontière de D_k) qu'on rappelle ici son expression :

$$Proj_{D_k}(x^k) = x^k - \frac{\left\langle F(y^k), x^k - y^k \right\rangle}{\|F(y^k)\|^2} F(y^k).$$

On voit clairement, d'après le lemme précédent que l'itéré x^{k+1} calculé par les deux algorithmes, de double projection et celui de Solodov est comme suit :

Algorithme de Iusem et Svaiter : $x^{k+1} = Proj_C(x^k - \lambda_k F(y^k))$.
Algorithme de Solodov et Svaiter : $x^{k+1} = Proj_{C\cap D_k}(x^k - \lambda_k F(y^k))$.

Où

$$\lambda_k = \frac{\left\langle F(y^k), x^k - y^k \right\rangle}{\|F(y^k)\|^2}$$

et y^k est calculé selon chaque algorithme.

Les deux expressions sont assez semblables, mais l'itéré x^{k+1} obtenu par l'algorithme de Solodov et Svaiter est plus proche de l'ensemble $Sol(VIP)$ que celui obtenu par l'algorithme de double projection.

Cette propriété a été expliquée par Solodov et Svaiter de point de vue géométrique et confirmée par ses tests numériques et démontrée théoriquement plus tard dans les travaux de Wang [60].

Théorème 3.3.1 *[56] Soit le problème $VIP(F, C)$ avec F continu pseudomonotone et $Sol(VIP)$ l'ensemble des solutions associé est non vide. Alors, la suite générée par l'algorithme de Solodov et Svaiter converge vers une solution de $VIP(F, C)$.*

Méthode de Wang et al.

Y. J. Wang et al. [61], de leur part ont contribué d'une manière claire dans le développement de ce type de méthode pour la résolution des problèmes d'inégalités variationnelles pseudomonotones. Leur algorithme converge sous les hypothèses de continuité et la pseudomonotonie de F.

Description de l'algorithme de Wang et al.

Début algorithme

- **Initialisation**

  $k = 0$, $x^0 \in C$; $\sigma, \gamma \in (0,1)$, et ε une précision donnée.

  Calculer $z^k = Proj_C(x^k - F(x^k))$.

  Calculer $r(x^k) = x^k - z^k$.

- **Tant que** $\left\|r(x^k)\right\| > \varepsilon$ faire :

  - **Etape 0 :** Trouver j le plus petit entier positif tel que

$$\left\langle F(x^k - \gamma^j r(x^k)), r(x^k)\right\rangle \geq \sigma \left\|r(x^k)\right\|^2.$$

  - **Etape 1 :** Prendre $\alpha_k = \gamma^j$.
  - **Etape 2 :** Poser

$$y^k = (1 - \alpha_k)x^k + \alpha_k z^k.$$

  - **Etape 3 :** Calculer l'itéré suivant

$$x^{k+1} = Proj_C(x^k - \lambda_k F(y^k)).$$

    Où λ_k est choisie telle que

$$\left\langle F(y^k), Proj_C(x^k - \lambda_k F(y^k)) - y^k\right\rangle \geq 0$$

    et

$$\lambda_k \geq \frac{\left\langle F(y^k), x^k - y^k\right\rangle}{\left\|F(y^k)\right\|^2}.$$

  - **Etape 4 :** Poser $k = k + 1$ et réitérer.

- **Fin tant que.**

Fin algorithme.

Méthode de Grar et Benterki

Grar et Benterki [24] ont proposé sous les mêmes hypothèses de convergence et dans la même forme des algorithmes de projections (algorithme de Iusem et Svaiter, algorithme de Solodov et Svaiter et celui de Wang et al.) une autre version où ils introduisent les modifications suivantes afin d'accélerer la convergence de l'algorithme obtenu :

- Le paramètre β utilisé dans le calcul de la première projection est pris constant entre 0 et1.
- La procédure de recherche linéaire utilisée diffère un peu de celle utilisée dans l'algorithme de Solodov et Svaiter et celui de Wang.
- Le paramètre variable λ utilisé dans le calcul de l'itéré x^{k+1}, est choisi de telle sorte qu'il satisfait la condition :

$$\left\langle F(y^k), Proj_C(x^k - \lambda_k F(y^k)) - y^k \right\rangle \leq 0$$

- Tous les itérés sont pris dans l'ensemble C et le demi plan D_k qui contient l'ensemble des solutions. Ce qui fait de x^{k+1} un point plus proche de $Sol(VIP)$.

On présente maintenant l'algorithme modifié nommé algorithme de **Grar-Benterki.**

Description de l'algorithme de Grar-Benterki

Début algorithme

- **Initialisation**

  $k = 0$, $x^0 \in C$; $\sigma, \gamma, \beta \in (0, 1)$, et ε une précision donnée.

  Calculer $z^k = Proj_C(x^k - \beta F(x^k))$.

  Calculer $r(x^k, \beta) = x^k - z^k$.

- **Tant que** $\left\|r(x^k, \beta)\right\| > \varepsilon$ faire :
  - **Etape 0 :** Trouver j le plus petit entier positif tel que :

$$\left\langle F(x^k - \gamma^j r(x^k, \beta)), r(x^k, \beta) \right\rangle \geq \sigma \left\|r(x^k, \beta)\right\|^2.$$

  - **Etape 1 :** Prendre $\alpha_k = \gamma^j$.
  - **Etape 2 :** Poser :

$$y^k = (1 - \alpha_k)x^k + \alpha_k z^k.$$

– **Etape 3 :** Calculer l'itéré suivant :

$$x^{k+1} = Proj_C(x^k - \lambda_k F(y^k)).$$

Où λ_k est choisie tel que :

$$\left\langle F(y^k), Proj_C(x^k - \lambda_k F(y^k)) - y^k \right\rangle \leq 0.$$

– **Etape 4 :** Poser $k = k + 1$ et réitérer.

– **Fin tant que.**

Fin algorithme.

Avant de donner les résultats théoriques de convergence, on résume quelques propriétés et résultats connus pour les opérateurs de projection dans les lemmes ci-dessous :

Lemme 3.3.2 *[59] Soit C un convexe fermé non vide de $\mathbb{R}^n$. Alors, pour tout $x \in \mathbb{R}^n$ et $z \in C$, on a :*

(i) $\langle Proj_C(x) - x, z - Proj_C(x)\rangle \geq 0$.

(ii) $\|Proj_C(x) - z\|^2 \leq \|x - z\|^2 - \|Proj_C(x) - x\|^2$.

(iii) $\langle z - x, z - Proj_C(x)\rangle \geq \|z - Proj_C(x)\|^2$.

Lemme 3.3.3 *[61] $\bar{x}$ est une solution de $VIP(F, C)$ si et seulement si*

$$\bar{x} = Proj_C(\bar{x} - \lambda F(\bar{x})),\ \lambda > 0$$

D'après [61], en vertu du lemme 3.3.2 (ii), et puisque F est pseudomonotone on a :

$$\begin{aligned} \left\|x^{k+1} - \bar{x}\right\|^2 &= \left\|Proj_C(x^k - \lambda_k F(y^k)) - \bar{x}\right\|^2 \\ &\leq \left\|x^k - \bar{x} - \lambda_k F(y^k)\right\|^2 - \left\|x^k - x^{k+1} - \lambda_k F(y^k)\right\|^2 \\ &\leq \left\|x^k - \bar{x}\right\|^2 + {\lambda_k}^2 \left\|F(y^k)\right\|^2 - 2\lambda_k \left\langle F(y^k), x^k - y^k \right\rangle \\ &\quad - \left\|x^k - x^{k+1} - \lambda_k F(y^k)\right\|^2. \end{aligned}$$

On définit, pour $\lambda \geq 0$

$$x^{k+1} = x(\lambda) = Proj_C(x^k - \lambda F(y^k)),$$

et la fonction :

$$\phi_k(\lambda) = 2\lambda \left\langle F(y^k), x^k - y^k \right\rangle + \left\|x^k - x^k(\lambda) - \lambda F(y^k)\right\|^2 - \lambda^2 \left\|F(y^k)\right\|^2,$$

dont sa dérivée :

$$\phi_k'(\lambda) = 2\left\langle F(y^k), x^k(\lambda) - y^k\right\rangle.$$

Dans ce qui suit on note :

- λ_I : le pas de déplacement associé à l'algorithme de Iusem et Svaiter de double projection, donné par la formule explicite suivante :

$$\lambda_I = \frac{\left\langle F(y^k), x^k(\lambda) - y^k\right\rangle}{\|F(y^k)\|^2}.$$

- λ_W : le pas de déplacement associé à l'algorithme de Wang et al. qui est choisi comme suit :

$$\lambda_W \geq \lambda_I \text{ et } \left\langle F(y^k), x^k(\lambda_W) - y^k\right\rangle \geq 0.$$

- λ_S : le pas de déplacement associé à l'algorithme de Solodov et Svaiter.

Toujours, d'après [61], on rappelle que ϕ_k est une fonction positive pour tout $\lambda \in [0, \lambda_S]$, en particulier pour λ_I, λ_W ($\lambda_I,\ \lambda_W \in]0, \lambda_S]$).

On peut remarquer la même chôse pour ϕ_k' au pas λ_W i.e.,

$$\phi_k'(\lambda_W) = 2\left\langle F(y^k), x^k(\lambda_W) - y^k\right\rangle \geq 0.$$

On peut remarquer que l'itéré x^{k+1} calculé par l'algorithme de Wang est : $x^{k+1} = Proj_C(x^k - \lambda_W F(y^k)) = x(\lambda_W)$ appartient au demi plan qui ne contient pas l'ensemble des solutions $Sol(VIP)$ (le complement de D_k).

Pour λ_S, la fonction ϕ_k atteint son maximum à cette valeur, alors on a pour ϕ_k'

$$\phi_k'(\lambda_S) = 2\left\langle F(y^k), x^k(\lambda_S) - y^k\right\rangle = 0.$$

Donc,

$$\left\langle F(y^k), x^k(\lambda_S) - y^k\right\rangle = 0. \tag{3.6}$$

Géométriquement, on voit que l'itéré x^{k+1} calculé par l'algorithme de Solodov est : $x^{k+1} = Proj_C(x^k - \lambda_S F(y^k)) = x(\lambda_S)$ est sur la frontière de D_k (appartient à H_k).

Pour l'algorithme de **Grar-Benterki**, le pas de déplacement doit vérifier l'inégalité suivante :

$$\left\langle F(y^k), Proj_C(x^k - \lambda_k F(y^k) - y^k\right\rangle \leq 0. \tag{3.7}$$

Cette condition nous assure que :

– L'itéré x^{k+1} calculé par l'algorithme de **Grar-Benterki** est :

$$x^{k+1} = Proj_C(x^k - \lambda_k F(y^k)) = x(\lambda_k).$$

– x^{k+1} appartient à D_k et

$$\phi_k'(\lambda_k) = 2\left\langle F(y^k), x^k(\lambda_k) - y^k \right\rangle \leq 0.$$

– Le pas de déplacement $\lambda_k > \lambda_W$ et même $\lambda_k > \lambda_S$ (puisque la fonction ϕ_k est décroissante)

Si un tel pas existe vraiment, alors l'algorithme de **Grar-Benterki** convergera plus rapidement par rapport aux autres algorithmes. Maintenant, le problème qui se pose pour ce pas est le suivant : Quelle est la condition suffisante qui nous garanti la décroissance de la suite $\left\{\left\|x^k - \bar{x}\right\|^2\right\}$?

Pour cet effet, on donne la proposition ci-dessous :

Proposition 3.3.1 *[24] Soient $x^k(\lambda_S)$ et $x^k(\lambda_k)$ les deux itérés suivants (correspondants à l'itération $(k+1)$) déterminés en fonction des algorithmes de Solodov et Svaiter et celui de **Grar-Benterki** respectivement.*

Si : $\left\|x^k - x^k(\lambda_k)\right\|^2 - \left\|x^k - x^k(\lambda_S)\right\|^2 \geq 2\lambda_k \left\langle F(y^k), y^k - x^k(\lambda_k)\right\rangle$, alors :

(i) $\left\|x^k - x^k(\lambda_k)\right\|^2 - \left\|x^k - x^k(\lambda_S)\right\|^2 \geq 0.$

(ii) $\phi_k(\lambda_k) \geq \phi_k(\lambda_S)$.

Donnons maintenant la proposition et le théorème établissant la convergence de l'algorithme de **Grar-Benterki.**

Proposition 3.3.2 *[24] Soit $\left\{x^k\right\}$ la suite générée par l'algorithme de **Grar-Benterki** et supposons que l'ensemble $Sol(VIP)$ est non vide, alors :*

(i) La suite $\left\{\left\|x^k - \bar{x}\right\|\right\}$ est décroissante pour tout $x \in Sol(VIP)$.

(ii) La suite $\left\{x^k\right\}$ est bornée.

(iii) $\lim_{k\longrightarrow\infty} \left\langle F(y^k), x^k - y^k \right\rangle = 0.$

(iiii) Si un point d'accumulation de la suite $\left\{x^k\right\}$ appartient à $Sol(VIP)$, alors $\left\{x^k\right\}$ converge vers une solution de l'ensemble $Sol(VIP)$.

Théorème 3.3.2 *[24]On suppose que F est continu et pseudomonotone et l'ensemble $Sol(VIP)$ est non vide, alors la suite générée par l'algorithme de **Grar-Benterki** converge vers une solution de $VIP(F, C)$ et* $\lim_{k\longrightarrow\infty} \left\|r(x^k, \beta)\right\| = 0.$

Calcul de l'itéré x^{k+1}

On peut distinguer au minimum 4 cas, mais il y a des cas qui restent pûrement théoriques.

- **Cas 1 :** Si C est un sous espace vectoriel, alors l'opérateur de projection $Proj_C$, sera une application linéaire. On peut alors choisir $\theta > 0$ et $\lambda_k = \lambda_S + \theta$, donc l'itéré x^{k+1} est comme suit :

  $$x^{k+1} = x(\lambda_k) = x(\lambda_S + \theta) = Proj_C(x^k - (\lambda_S + \theta)F(y^k)).$$

  On obtient

  $$x^{k+1} = Proj_C(x^k - \lambda_S F(y^k)) - \theta Proj_C(F(y^k)).$$

- **Cas 2 :** On démarre avec une valeur de λ_k qui doit être mise à jour de telle sorte qu'elle vérifie l'inégalité 3.7. Mais ceci nécessitera le calcul de plusieurs projections avant de tomber sur le bon choix de λ_k. Ceci éliminera l'avantage de cet algorithme (le calcul de que deux projections à chaque itération).
- **Cas 3 :** Pour surmonter ces difficultés, on peut prendre x^{k+1} comme une combinaison convexe de $x(\lambda_S)$ et z^k comme suit :

  $$x^{k+1} = \theta x(\lambda_S) + (1-\theta)z^k,\ \theta \in [0,1].$$

  D'après cette forme, on voit que x^{k+1} appartient toujours à C et satisfait l'inégalité (3.7) :

  $$\begin{aligned} & \left\langle F(y^k), \theta x(\lambda_S) + (1-\theta)z^k - y^k \right\rangle \\ = & \left\langle F(y^k), \theta x(\lambda_S) + (1-\theta)z^k - (\theta + (1-\theta))y^k \right\rangle \\ = & \theta \left\langle F(y^k), x(\lambda_S) - y^k \right\rangle + (1-\theta) \left\langle F(y^k), z^k - y^k \right\rangle \\ < & 0. \end{aligned}$$

  En utilisant la procédure de recherche linéaire et le fait que λ_S satisfait (3.6), alors la dernière inégalité est obtenue.
  On note que pour $\theta = 1$, on tombe sur l'itéré de Solodov.
- **Cas 4 :** On peut utiliser un pas fixe ($\lambda_k = \lambda,\ \forall k$) mais à chaque itération, on doit vérifier que ce pas satisfait l'inégalité (3.7) jusqu'à ce qu'on trouve le meilleur choix. Cette opération peut prendre du temps, mais une fois le pas convenable est trouvé, le coût de calcul sera inférieur au cas du pas de déplacement variable.

Remarque 3.3.2 *Le changement introduit à λ_k a permis d'obtenir une séquence d'itérés plus proche de l'ensemble des solutions que tous les algorithmes précédents. Le choix de la constante β dans la première projection dans le calcul de z^k, ainsi que la procédure de recherche linéaire qui est un peu différente à celle utilisée dans Solodov et Svaiter et celle de Wang et al. ont conduit à une grande amélioration dans les performances numériques de l'algorithme de* ***Grar-Benterki****.*

3.4 Lien entre programme fractionnaire linéaire (PFL) et problème d'inégalités variationnelles (VIP)

Dans cette partie, on présente une formulation de (PFL) qui permet de le convertir en un problème d'inégalites variationnelles (VIP) équivalent, puis résoudre ce dernier via l'algorithme de projection de **Grar et Benterki**.

Le lien entre les deux problèmes (PFL) et (VIP) repose essentiellement sur la condition d'optimalité du problème d'optimisation (PFL). Par la suite, on va établir clairement ce lien dans un cadre plus large, autrement dit avec un problème d'optimisation avec contraintes général.

Lemme 3.4.1 *Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction différentiable et C un convexe fermé. On définit le problème d'optimisation suivant :*

$$(PO)\begin{cases} \min f(x) \\ x \in C. \end{cases}$$

Alors, si $\overline{x}$ est un minimum local de (PO), alors $\overline{x}$ est solution du problème d'inégalités variationnelles suivant :

$$\langle \nabla f(\overline{x}), x - \overline{x} \rangle \geq 0, \ \forall \ x \in C. \qquad (VIP(\nabla f, C))$$

Preuve. Puisque C est convexe, alors $\forall x \in C, \forall t \in [0,1], tx + (1-t)\overline{x} \in C$,

on a :

$$f(tx + (1-t)\overline{x}) = f(\overline{x} + t(x - \overline{x})) \geq f(\overline{x}),$$

en utilisant le développement de Taylor

$$0 \leq f(\overline{x} + t(x - \overline{x})) - f(\overline{x}) = t\nabla f(\overline{x})^T (x - \overline{x})$$

Ce qui donne l'inégalité variationnelle $VIP(\nabla f, C)$. ■

Lemme 3.4.2 *Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction convexe différentiable, alors*

$$\overline{x} \text{ solution de } VIP(\nabla f, C) \Longleftrightarrow \overline{x} \text{ minimum global de } (PO)$$

Preuve. 1) $\Leftarrow$) Supposons que $\overline{x}$ est une solution de (PO), alors d'après le lemme 3.4.1, $\overline{x}$ est une solution de $VIP(\nabla f, C)$.

2) $\Rightarrow$) Réciproquement, on suppose que $\overline{x} \in Sol(VIP(\nabla f, C))$.

En se servant de la définition de la convexité de f en fonction de ∇f :

$$f(x) - f(y) \geq \langle \nabla f(y), x - y) \rangle \, , \forall \; x, y \in C.$$

On prend $y = \overline{x} \in C$, on obtient

$$f(x) - f(\overline{x}) \geq \langle \nabla f(\overline{x}), x - \overline{x} \rangle \geq 0, \; \forall \; x \in C.$$

alors

$$f(x) - f(\overline{x}) \geq 0, \forall \; x \in C.$$

Donc, $\overline{x}$ est une solution de (PO). ■

Remarque 3.4.1
- *On déduit que la condition nécessaire et suffisante pour que $\overline{x}$ soit une solution globale de (PO) convexe est équivalente à ce qu'il soit solution de (VIP) avec $F = \nabla f$.*
- *Le même résultat reste vrai dans le cas où f est pseudoconvexe.*
- *Le cas particulier qui nous intéresse dans ce travail est de prendre (PO) un (PFL) à objectif pseudoconvexe.*

Remarque 3.4.2 *La transformation de (PFL) en un (VIP) équivalent, a pour but d'appliquer des algorithmes connus pour cette classe de problèmes. Dans notre travail, on s'intèresse à la résolution de (PFL) par l'algorithme de projection de* ***Grar-Benterki****.*

3.5 Expérimentations numériques

Dans cette partie, nous présentons des éxpérimentations numériques comparatives entre les méthodes de projection en utilisant l'algorithme de **Grar-Benterki** et les méthodes de points intérieurs en utilisant l'algorithme de **Ye-Lustig**.

Les tests numériques réalisés sont effectués sur des exemples de la littérature. Le langage utilisé est le MATLAB. La précision choisie étant $\varepsilon = 10^{-6}$. Dans chaque exemple, on donne la solution optimale x^* trouvée par l'algorithme de **Ye-Lustig** et celui de **Grar-Benterki** ainsi que la valeur optimale z^*. On désigne par :

- k : le nombre des itérations nécessaire pour trouver la solution optimale.
- *temps* : le temps d'execution en secondes.

Nous avons considéré dans ces tests numériques des exemples à taille fixe issue de la littérature, et d'autres à taille variable et ceci pour mesurer l'efficacité des deux algorithmes précédents pour les problèmes de grande taille.

3.5.1 Exemples à taille fixe

Exemple 3.5.1 *On considère le programme fractionnaire linéaire suivant :*

$$(PFL1)\begin{cases} \min\left(\frac{7x_1+9x_2+3}{3x_1+4x_2+2}\right) \\ 2x_1+3x_2 \leq 6 \\ 3x_1+2x_2 \leq 5 \\ x_1, x_2 \geq 0. \end{cases}$$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 3$ *itérations et* $0.0008s$

$$x^* = (0.5999, 1.6000)^t.$$

La valeur optimale est $z^* = 2,1176$.

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 1$ *itération et* $0.0731s$

$$x^* = (0.6000, 1.6000)^t.$$

La valeur optimale est $z^* = 2,1176$.

Exemple 3.5.2 *On considère le programme fractionnaire linéaire suivant :*

$$(PFL2)\begin{cases} \max\left(\frac{x_1+2x_2+4x_3+5x_4+8x_5}{2x_1+5x_2+3x_3+4x_4+6x_5+1}\right) \\ 2x_1+x_2+3x_3+x_4+x_5 \leq 15 \\ x_1+2x_2+x_5 \leq 8 \\ 3x_{1+}5x_2+2x_4 \leq 10 \\ 2x_2+4x_3+3x_4+x_5 \leq 21 \\ x_1, x_2, x_3, x_4, x_5 \geq 0. \end{cases}$$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 12$ *itérations et* $0.0253s$

$$x^* = (0.00001, 0.00002, 2.33333, 0.00009, 8.00000)^t.$$

La valeur optimale est $z^* = 1,3$.

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 9$ *itérations et* $1.3309s$

$$x^* = (0.00000, 0.00003, 2.33333, 0.00009, 8.00000)^t.$$

La valeur optimale est $z^* = 1,3$.

Exemple 3.5.3 *On considère le programme fractionnaire linéaire suivant :*

$$(PFL3)\begin{cases} \max\left(\frac{x_1+x_2+1}{5x_1+x_2+1}\right) \\ -5x_1 - 2x_2 \leq -6 \\ x_1 \leq 3 \\ x_2 \leq 3 \\ x_1, x_2 \geq 0. \end{cases}$$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 5$ *itérations et* $0.0026s$

$$x^* = (3.00000, 0.00000)^t.$$

La valeur optimale est $z^* = 0,25$.

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 1$ *itération et* $0.0456s$

$$x^* = (3.00000, 0.00000)^t.$$

La valeur optimale est $z^* = 0,25$.

Exemple 3.5.4 *[45] On considère le programme fractionnaire linéaire suivant :*

$$(PFL4)\begin{cases} \min\left(\frac{\sum_{j=1}^{n} c_j x_j}{\sum_{j=1}^{n} d_j x_j}\right) \\ \sum_{j=1}^{n} x_j = N \\ 0 \leq x_j \leq 1, \ j = 1, ..., n. \end{cases}$$

1) n=5

$c^t = (2, -1, -3, 5, -2),\ d^t = (1, 2, 2, 3, 4),\ N = 3$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 5$ *itérations et* $0.0021s$

$$x^* = (0.00000, 0.99999, 0.99999, 0.00000, 0.99999)^t.$$

La valeur optimale est $z^* = -0, 74999.$

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 1$ *itération et* $0.0095s$

$$x^* = (0.00000, 1.00000, 1.00000, 0.00000, 1.00000)^t.$$

La valeur optimale est $z^* = -0, 75.$

2) n=10

$c^t = (1, 1, 1, 2, 2, 3, 0, 0, 1, 1),\ d^t = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1),\ N = 5$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 5$ *itérations et* $0.0031s$

$$x^* = (0.5999, 0.5999, 0.5999, 0.0000, 0.0000, 0.0001, 0.9999, 0.9999, 0.5999, 0.5999)^t.$$

La valeur optimale est $z^* = 0, 6.$

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 1$ *itération et* $0.0582s$

$$x^* = (0.6000, 0.6000, 0.6000, 0.0000, 0.0000, 0.0001, 1.0000, 1.0000, 0.6000, 0.6000)^t.$$

La valeur optimale est $z^* = 0, 6.$

3) n=15

$c^t = (1, 2, 3, 3, 0, 0, 1, 6, 0, 4, 0, 1, 5, 2, 1),\ d^t = (1, 2, 2, 4, 6, 1, 1, 0, 0, 5, 8, 2, 3, 1, 1),$
$N = 8$

Résultats obtenus via l'algorithme de Ye-Lustig :

Solution optimale trouvée après $k = 7$ *itérations et* $0.0066s$

$$\begin{aligned} x^* \ = \ & (0.9999, 0.0000, 0.0000, 0.0000, 0.9999, 0.9999, 0.9999, 0.0000, \\ & 0.9999, 0.0000, 0.9999, 0.9999, 0.0000, 0.0000, 0.9999)^t. \end{aligned}$$

La valeur optimale est $z^* = 0, 2.$

Résultats obtenus via l'algorithme de Grar-Benterki :

Solution optimale trouvée après $k = 2$ *itérations et* $0.0957s$

$$\begin{aligned} x^* &= (0.9999, 0.0000, 0.0000, 0.0000, 1.0000, 0.9999, 0.0000, 1.0000, \\ &\quad 0.0000, 1.0000, 1.0000, 0.0000, 0.0000, 0.9999, 0.0000)^t. \end{aligned}$$

La valeur optimale est $z^* = 0, 2$.

Le tableau suivant résume les résultats obtenus via les deux algorithmes : l'algorithme de ***Ye-Lustig*** *et l'algorithme de* ***Grar-Benterki*** *pour les exemples testés précédemment.*

		Ye − Lustig		**Grar − Benterki**	
$Exemple$	$Taille(m,n)$	k	$temps(s)$	k	$temps(s)$
$PFL1$	$(2,4)$	3	0.0008	1	0.0731
$PFL2$	$(4,9)$	12	0.0253	9	1.3309
$PFL3$	$(3,5)$	5	0.0026	1	0.0456
$PFL4$	$(6,10)$	5	0.0021	1	0.0095
	$(11,20)$	5	0.0031	1	0.0582
	$(16,30)$	7	0.0066	2	0.0957

.

3.5.2 Exemples à taille variable

Exemple 3.5.5 *On considère le programme fractionnaire linéaire à taille variable suivant :*

$$(PFL_m) \quad \begin{cases} \max Z = \frac{c^t x + 2m}{d^t x + 1} \\ Ax = b \\ x \geq 0 \end{cases},$$

$$où \quad \begin{cases} c_i = \begin{cases} -1 + 2m, & i = 1, ..., m \\ 2m, & i = m+1, ..., n \end{cases} \\ d_i = 1, \qquad i = 1, ..., n \qquad n = 2m \\ A[i,i] = A[i, m+i] = 1, \quad i = 1, ..., m \\ b_i = 2, \qquad i = 1, ..., m. \end{cases}$$

Le tableau suivant résume les résultats obtenus via les deux algorithmes : l'algorithme de **Ye-Lustig** et l'algorithme de **Grar-Benterki** suivant des valeurs différentes de m.

$Taille$	**Ye − Lustig**		**Grar − Benterki**	
(m, n)	k	$temps(s)$	k	$temps(s)$
$(50, 100)$	3	0.0516	2	0.1793
$(100, 200)$	3	0.2941	2	1.0173
$(200, 400)$	2	1.5522	3	8.5241
$(300, 600)$	2	5.9558	4	26.0963
$(500, 1000)$	2	31.0989	5	85.7836
$(1000, 2000)$	2	247.4904	9	617.5747
$(1500, 3000)$	2	461.9488	13	2270.8168

Commentaires

A travers les tests numériques et pour les différentes dimensions, les résultats montrent que :

- Pour les exemples à petite et moyenne taille, le temps d'exécution par **Ye-Lustig** est sensiblement bas par rapport à celui de **Grar-Benterki**, bien qu'il necessite moins d'itérations.
- Pour les exemples à grande taille, on remarque la supériorité de l'algorithme de **Ye-Lustig** par rapport à celui de **Grar-Benterki**. En effet, le nombre des itérations et le temps d'exécution par **Ye-Lustig** sont significativement inférieurs à ceux obtenus par **Grar-Benterki.**
- On remarque une réduction considérable en temps d'exécution par **Ye-Lustig**. Ceci montre clairement l'efficacité de l'introduction des méthodes de points intérieurs pour résoudre les programmes linéaires déjà connue pour les exemples à grande taille.

Conclusion

Dans ce chapitre, nous avons appliqué deux approches numériques pour résoudre un programme fractionaire linéaire. Les résultats numériques obtenus favorisent davantage l'approche des points intérieurs, lorsque la taille du problème devient importante. Ce travail a fait l'objet d'une publication dans la revue Rairo Operations Researchs [9].

Dans le chapitre suivant, on envisage de résoudre un programme fractionnaire quadratique (PFQ) via deux approches. Une approche basée sur la transformation de (PFQ) en un (PL) équivalent, et l'autre sert à ramener (PFQ) à un programme quadratique (PQ) équivalent.

Chapitre 4

Résolution d'un problème fractionnaire quadratique via l'algorithme DCA

Résumé : Dans ce chapitre, on s'intèresse à la résolution numérique des problèmes fractionnaires quadratiques (PFQ). On présente tout d'abord la formulation mathématique du problème fractionnaire quadratique (PFQ) où on considère le cas des contraintes linéaires. Puis on présente deux formulations de (PFQ). La première permet de transformer (PFQ) en une suite de (PL) équivalents. La deuxième transforme (PFQ) en un programme quadratique (PQ) équivalent. Nous achevons ce chapitre par la résolution numérique du (PFQ) par deux algorithmes différents. Des tests numériques seront présentés à la fin de ce chapitre.

Contenu :

4.1 Introduction

Dans plusieurs applications de la programmation non linéaire, on rencontre souvent des problèmes dans lesquels le rapport de deux fonctions données doit être optimisé, en particulier le cas où ces deux fonctions sont quadratiques qu'on appelle problèmes de programmation fractionnaire quadratique.

Ces derniers ont attiré l'attention de nombreux chercheurs puisqu'ils se manifestent dans plusieurs domaines tels que : La planification de production, planification financière et corporative, planification des soins de santé et des hôpitaux etc...

Plusieurs méthodes sont proposées pour résoudre cette catégorie de problèmes. Beck et al. [7] en 2006, ont trouvé une solution optimale globale pour un problème fractionnaire quadratique sous contraintes quadratiques avec applications du total régularisé des moindres carrés. Khurana et Arora [2] en 2011, ont présenté un algorithme pour résoudre (PFQ) en le transformant en un (PFQ) équivalent avec moins de contraintes. En 2013, Nadjmaddin Suleiman et Maher Nawkhas [44] ont résolu (PFQ) en utilisant la méthode de Wolfe et une approche modifiée du Simplexe. En 2014, dans leurs papier [58], Van Bong Nguyen et al. ont résolu (PFQ) via une approche de SDP et M. Rashidul Hasan et al. [51] ont procédé à une extension de la méthode de Simplexe pour la programmation fractionnaire quadratique. Parmi les approches les plus récentes, il y a celle proposée par Sivri et al. en 2018 [55], où ils ont résolu (PFQ) via une approche basée sur le développement de Taylor de la fonction objectif. Cette dernière est l'approche qui nous intèresse dans ce travail.

4.2 Position du programme fractionnaire quadratique

Les programmes fractionnaires quadratiques consistent à optimiser un objectif mis sous la forme d'un rapport de deux fonctions quadratiques, soumis à un ensemble de contraintes linéaires ou non linéaires.

On considère le programme fractionnaire quadratique (PFQ) suivant :

$$(PFQ)\begin{cases} \min G(x) = \frac{\frac{1}{2}x^tHx+c^tx+c_0}{\frac{1}{2}x^tDx+d^tx+d_0} \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases}$$

Où H et D sont deux matrices symétriques de $\mathbb{R}^{n\times n}$; $c, d \in \mathbb{R}^n$; $c_0, d_0 \in \mathbb{R}$; $A \in \mathbb{R}^{m\times n}$ de plein rang ; $b \in \mathbb{R}^m$ et $\frac{1}{2}x^tDx + d^tx + d_0 > 0$ pour tout $x \in P$.

Remarque 4.2.1 *Puisque la fonction G est continue sur P un compact (fermé et borné), alors d'après le théorème de Weirstass (PFQ) admet au moins une solution optimale.*

4.2.1 Résultats fondamentaux de (PFQ)

Lemme 4.2.1 *Si H est une matrice symétrique semi-définie positive, D une matrice symétrique semi-définie négative et $\frac{1}{2}x^tHx+c^tx+c_0 \geq 0$, alors la fonction G est semi-strictement quasiconvexe.*

Preuve. Il suffit d'appliquer la proposition 1.2.6 à la fonction G. ■

Lemme 4.2.2 *Si H est une matrice symétrique semi-définie positive, D une matrice symétrique semi-définie négative et $\frac{1}{2}x^tHx + c^tx + c_0 \geq 0$, la fonction G est quasiconvexe.*

Preuve. Puisque la fonction G est continue, alors elle est semi continue inférieurement. De plus elle est semi-strictement quasiconvexe d'après le lemme 4.2.1, alors d'après théorème 1.2.10 elle est quasiconvexe sur P. ■

Lemme 4.2.3 *Si H est une matrice symétrique semi-définie positive, D une matrice symétrique semi-définie négative et $\frac{1}{2}x^tHx+c^tx+c_0 \geq 0$, tout minimum local de G sur P est un minimum global.*

Preuve. Puisque G est semi-strictement quasiconvexe, il suffit d'appliquer le théorème 1.2.11 ■

Avant de présenter notre nouvelle formulation, on va d'abord donner une formulation existante proposée en 2018 par Sivri et al. [55] pour résoudre (PFQ) en le transformant en une suite de programmes linéaires (PL). L'approche de Ye-Lustig de points intérieurs est une des meilleures méthodes qui résout cette suite de (PL).

4.3 Formulation de (PFQ) en une suite de (PL) équivalents

Cette formulation est basée essentiellement sur le développement de Taylor de l'objectif, c'est à dire, au lieu de résoudre (PFQ) on résout une suite de programmes linéaires.

Développement de Taylor d'une fonction F à l'ordre 1 Soit $F : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction différentiable de classe C^1.

Les deux premiers termes du développement de Taylor de la fonction F au voisinage du point $a = (a_1, a_2, ..., a_n)$ sont :

$$F(a) + \nabla^t F(a)(x - a),$$

qui linéarisent la fonction F en n variables, autrement dit :

$$F(x) \simeq F(a) + \nabla^t F(a)(x - a) \tag{4.1}$$

avec $\nabla^t F(x) = \left(\frac{\partial F(x)}{\partial x_1}, \frac{\partial F(x)}{\partial x_2}, ..., \frac{\partial F(x)}{\partial x_n}\right)$, alors :

$$\begin{aligned} F(x) &\simeq F(a) + \nabla^t F(a)(x - a) \\ &\simeq F(a) + \nabla^t F(a)x - \nabla^t F(a)a \\ &\simeq \nabla^t F(a)x + (F(a) - \nabla^t F(a)a) \end{aligned} \tag{4.2}$$

Dans ce qui suit on donne une description détaillée des étapes de cette formulation.

4.3.1 Description de la formulation de (PFQ) en une suite de (PL) équivalents

– **Etape 0 :** On considère le problème de programmation fractionnaire quadratique suivant :

$$(PFQ) \begin{cases} \min G(x) = \frac{\frac{1}{2}x^t Hx + c^t x + c_0}{\frac{1}{2}x^t Dx + d^t x + d_0} \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases}.$$

Où H et D sont deux matrices symétriques de $\mathbb{R}^{n\times n}$; $c, d \in \mathbb{R}^n$; $c_0, d_0 \in \mathbb{R}$; $A \in \mathbb{R}^{m\times n}$ de plein rang ; $b \in \mathbb{R}^m$ et $\frac{1}{2}x^t Dx + d^t x + d_0 > 0$ pour tout $x \in P$.

- **Etape 1 :** On choisit un point initial arbitraire strictement réalisable $\widetilde{x} = (\widetilde{x}_1, \widetilde{x}_2, ..., \widetilde{x}_n)^t$.
- **Etape 2 :** On linéarise la fonction objectif G de (PFQ) via le développement de Taylor au point $\widetilde{x}$ en utilisant (4.2). On obtient le programme linéaire suivant :

$$(PL_{\widetilde{x}}) \begin{cases} \min[\nabla^t G(\widetilde{x})x + (G(\widetilde{x}) - \nabla^t G(\widetilde{x})\widetilde{x})] \\ x \in P = \{x \in \mathbb{R}^n / Ax = b \ ; \ x \geq 0\} \end{cases}$$

avec :

$$\nabla G(x) = \frac{(\frac{1}{2}x^t Dx + d^t x + d_0)(Hx + c) - (\frac{1}{2}x^t Hx + c^t x + c_0)(Dx + d)}{(\frac{1}{2}x^t Dx + d^t x + d_0)^2}$$

- **Etape 3 :** On résout $(PL_{\widetilde{x}})$ par l'algorithme de **Ye-Lustig** pour obtenir sa solution optimale qu'on note $\overline{x} = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_n)^t$.
- **Etape 4 :** On développe la fonction objectif G de (PFQ) en utilisant le développement de Taylor cette fois au point $\overline{x}$ (solution optimale de $(PL_{\widetilde{x}})$) en utilisant (4.2). On obtient le programme linéaire suivant :

$$(PL_{\overline{x}}) \begin{cases} \min[\nabla G(\overline{x})^t x + (G(\overline{x}) - \nabla G(\overline{x})\overline{x})] \\ x \in P = \{x \in \mathbb{R}^n / Ax = b \ ; \ x \geq 0\} \end{cases}$$

- **Etape 5 :** On résout le nouveau programme linéaire $(PL_{\overline{x}})$ obtenu, par l'algorithme de **Ye-Lustig** pour trouver sa solution optimale qu'on note $\underline{x} = (\underline{x}_1, \underline{x}_2, ..., \underline{x}_n)^t$.
- **Etape 6 :**

  Si les deux solutions successives $\overline{x}$ et $\underline{x}$ se coïncident, alors c'est la solution optimale de (PFQ).

  Sinon, affecter $\underline{x}$ à $\overline{x}$ et aller à l'étape 4.

Remarque 4.3.1 *Dans le papier original de Sivri et al. [55], ils résolvent les programmes linéaires obtenus par la méthode de Simplexe, par contre dans ce travail on les résout via une approche de* ***Ye-Lustig*** *des méthodes de points intérieurs de type projectif introduite précédemment dans le chapitre 2.*

Dans ce qui suit, nous présentons une nouvelle formulation permettant de transformer (PFQ) en un programme quadratique (PQ) équivalent. Le programme quadratique (PQ) obtenu sera traité et résolu par un algorithme **DCA** (Algorithme de Différence de deux fonctions Convexes).

4.4 Formulation de (PFQ) en un (PQ) équivalent

On considère le problème de programmation fractionnaire quadratique suivant :

$$(PFQ)\begin{cases} \min G(x) = \frac{\frac{1}{2}x^tHx+c^tx+c_0}{\frac{1}{2}x^tDx+d^tx+d_0} \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases}.$$

Où H et D sont deux matrices symétriques de $\mathbb{R}^{n\times n}$; $c, d \in \mathbb{R}^n$; $c_0, d_0 \in \mathbb{R}$; $A \in \mathbb{R}^{m\times n}$ de plein rang ; $b \in \mathbb{R}^m$ et $\frac{1}{2}x^tDx + d^tx + d_0 > 0$ pour tout $x \in P$.

Lemme 4.4.1 *Le programme fractionnaire quadratique (PFQ) est équivalent au programme quadratique (PQ) suivant :*

$$(PQ)\begin{cases} \min \frac{1}{2}x^tQx + q^tx + q_0 \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases},$$

où

$$\begin{aligned} Q &= H - z^kD \in \mathbb{R}^{n\times n}, \\ q^t &= c^t - z^kd^t \in \mathbb{R}^n, \\ q_0 &= c_0 - z^kd_0 \in \mathbb{R}, \end{aligned}$$

avec z^k une borne supérieure de la valeur optimale de l'objectif de (PFQ), on prend

$$z^k = G(x^k),$$

tel que x^k est une solution strictement réalisable de (PFQ) i.e., $Ax^k = b$ et $x^k > 0$.

Preuve. Soit z^* la valeur optimale de (PFQ), alors :

$$(PFQ)\begin{cases} \min \frac{\frac{1}{2}x^tHx+c^tx+c_0}{\frac{1}{2}x^tDx+d^tx+d_0} = z^* \\ Ax = b \\ x \geq 0 \end{cases},$$

qui est équivalent à :

$$\begin{cases} \min \frac{1}{2}x^tHx + c^tx + c_0 - z^*(\frac{1}{2}x^tDx + d^tx + d_0) = 0 \\ Ax = b \\ x \geq 0. \end{cases}$$

Ce qui nous donne

$$\begin{cases} \min \frac{1}{2}x^t(H - z^*D)x + (c^t - z^*d^t)x + (c_0 - z^*d_0) = 0 \\ Ax = b \\ x \geq 0. \end{cases}$$

Puisque z^* est généralement inconnue, on l'approxime par une borne supérieure z^k ; c'est à dire :

$$z^* \leq z^k = G(x^k).$$

Donc

$$\begin{aligned} Q &= H - z^*D \simeq H - z^kD, \\ q^t &= c^t - z^*d^t \simeq c^t - z^kd^t, \\ q_0 &= c_0 - z^*d_0 \simeq c_0 - z^kd_0, \end{aligned}$$

telle que :

$$z^* \leq \frac{\frac{1}{2}(x^k)^t H x_k + c^t x^k + c_0}{\frac{1}{2}(x^k)^t D x_k + d^t x^k + d_0} = z^k,$$

avec x^k une solution strictement réalisable connue de (PFQ).

Alors, (PFQ) devient équivalent au programme quadratique suivant :

$$(PFQ) \Longleftrightarrow (PQ) \begin{cases} \min \frac{1}{2}x^t Q x + q^t x + q_0 \\ x \in P = \{x \in \mathbb{R}^n / Ax = b;\ x \geq 0\} \end{cases}$$

■

Remarque 4.4.1 *Pour résoudre le programme quadratique obtenu (PQ) (généralement non convexe), on utilise un algorithme dit **DCA** (Algorithme de Différence de deux fonctions Convexes), proposé pour la première fois par Pham Dinh Tao [47] en 1985. On présente un rappel de cet algorithme dans la sous-section ci-dessous.*

4.4.1 Principe général de DCA

DCA (DC Algorithm) est une méthode itérative basée sur l'optimalité locale et la dualité en programmation DC. Cet algorithme a été introduit par Pham Dinh Tao [47] en 1985 et puis intensivement développé depuis les environs des

années 90 par lui même [48], A. Yassine [62], Le Thi Hoai An et Pham Dinh Tao [36] et Le Thi Hoai An [37].

La popularité de la programmation DC et des algorithmes **DCA** réside dans leur simplicité, flexibilité, robustesse, rapidité, performance et leur adaptation aux structures des problèmes traités ainsi que leur capacité à résoudre des problèmes industriels de grande dimension.

Avant de présenter la description de l'algorithme **DCA**, on rappelle quelques définitions fondamentales.

Définition 4.4.1 *Une fonction f est dite propre si elle ne prend jamais la valeur $-\infty$ ou $+\infty$.*

Définition 4.4.2 *On définit par $\Gamma_0(X)$ l'ensemble des fonctions convexes, s.c.i. et propres sur X.*

Définition 4.4.3 *La fonction conjuguée de $f \in \Gamma_0(X)$,*
notée $f^* : Y \longrightarrow \mathbb{R} \cup \{+\infty\}$, est définie par

$$f^*(y) = \sup \{\langle x, y\rangle - f(x) : \ x \in X\}.$$

Définition 4.4.4 *Soit $f : X \longrightarrow \mathbb{R} \cup \{+\infty\}$ une fonction convexe. On appelle domaine effectif de f, noté $dom(f)$, l'ensemble*

$$dom(f) = \{x \in X, \ f(x) < +\infty\}.$$

C'est un ensemble convexe de X.

Définition 4.4.5 *Soit f une fonction convexe et propre sur X. Un vecteur y dans Y est appelé sous-gradient de f au point $x^0 \in dom(f)$ si pour tout $x \in X$ on a :*

$$f(x) \geq f(x^0) + \left\langle y, x - x^0 \right\rangle.$$

Définition 4.4.6 *L'ensemble de tous les sous-gradients de f au point $x^0 \in dom(f)$ est dit le sous-différentiel de f au point x^0, noté $\partial f(x^0)$. C'est un ensemble convexe fermé de Y.*

Remarque 4.4.2 *$f \in \Gamma_0(X)$ est différentiable en $x \iff \partial f(x)$ se réduit au singleton $\{\nabla f(x)\}$.*

On appelle un programme DC tout problème d'optimisation de la forme

$$(P_{dc}) : \ \alpha = \inf \{ f(x) = g(x) - h(x), \ x \in \mathbb{R}^n \},$$

où, $g, h \in \Gamma_0(X)$ sont appelées composantes DC de f et $g-h$ la décomposition DC de la fonction f.

(P_{dc}) est un problème d'optimisation sans contraintes. Cependant un problème d'optimisation DC avec C convexe fermé de la forme

$$\inf \{ f(x) = g(x) - h(x), \ x \in C \},$$

peut être ramené sous la forme (P_{dc}) en ajoutant la fonction indicatrice de C à la première composante de f, i.e.,

$$(P_{dc}) : \ \alpha = \inf \{ F(x) = \varphi(x) - h(x), \ x \in \mathbb{R}^n \},$$

avec

$$\varphi(x) = g(x) + \chi_C(x)$$

et

$$\chi_C(x) = \begin{cases} 0 & \text{si } x \in C \\ \infty & \text{ailleurs} \end{cases}$$

Remarque 4.4.3 – *Il est important de noter qu'une fonction DC possède plusieurs décompositions.*

– *Il est clair que* ***DCA*** *s'applique aux fonctions convexes g, h et non pas à la fonction f elle même.*

– *Le choix de la décomposition DC appropriée est crucial car il conditionne les qualités essentielles (rapidité, robustesse, globalité des solutions calculées) du* ***DCA*** *résultant.*

Définition 4.4.7 *On appelle point critique ou point KKT généralisé, tout point $x^* \in X$ vérifiant*

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset.$$

Théorème 4.4.1 *Si x^* est un minimum local de $g - h$ alors*

$$\partial h(x^*) \subset \partial g(x^*).$$

Description de l'algorithme DCA

Début algorithme

- **Initialisation**

  $k = 0$, x^0 donné, ε une précision donnée.
- **Etape 1 :** On calcule $y^k \in \partial h(x^k)$.
- **Etape 2 :** On calcule

$$x^{k+1} \in \partial g^*(y^k) = \arg\min(g(x) - h(x^k) - \langle x - x^k, y^k \rangle : x \in \mathbb{R}^n).$$

- **Etape 3 :**
  - Si les conditions d'arrêt sont vérifiées alors on s'arrête : $x^* = x^{k+1}$ est solution optimale du problème (P_{dc})
  - Sinon on pose $k = k + 1$ et on retourne à l'étape 1.

Fin algorithme

Interprétation de DCA

Dans cette partie nous donnons une interprétation de chaque étape de l'algorithme **DCA,** qui ne fonctionne qu'avec les composantes DC, g et h. Son principe est simple : à chaque itération k de **DCA**, on approxime la composante h par sa minorante affine $h_k(x) = h(x^k) + \langle x - x^k, y^k \rangle$ au voisinage de x^k (qui correspond à prendre $y^k \in \partial h(x^k)$) et on minimise la fonction convexe obtenue (ce qui est équivalent à determiner $x^{k+1} \in \partial g^*(y^k)$).

- **Initialisation :** Puisque nous utilisons une approche de descente, la convergence de l'algorithme est indépendante du point de départ des suites que l'algorithme a créé. Ainsi, nous pouvons initialiser l'algorithme avec un choix arbitraire du point initial x^0.
- **Etape 1 :** $\partial h(x^k) = \arg\min \left\{ h^*(y) - g^*(y^{k-1}) - (x^k)^t(y - y^{k-1}) : y \in Y \right\}$. Puisque nous minimisons par rapport à y, nous considérons que x^k et y^{k-1} sont des constantes, alors on a $\partial h(x^k) = \arg\max \left\{ (x^k)^t y - h^*(y) : y \in Y \right\}$ En général, le calcul de $\partial h(x^k)$ est explicite et facile à calculer.
- **Etape 2 :** On a $x^{k+1} \in \arg\min \left\{ g(x) - h(x^k) - (y^k)^t(x - x^k) : x \in X \right\}$ identiquement à l'étape 1, on trouve $x^{k+1} \in \arg\max \left\{ (y^k)^t x - g(x) : \ x \in X \right\}$. En général, pour calculer x^{k+1}, il faut résoudre à chaque itération un problème d'optimisation convexe.

– **Etape 3 :** Cette étape est un test d'arrêt, souvent en pratique, on utilise l'une des deux conditions, soit

$$\left\|x^{k+1}-x^k\right\| \leq \varepsilon \text{ ou } \left\|(g-h)(x^{k+1})-(g-h)(x^k)\right\| \leq \varepsilon.$$

Les résultats de convergence de **DCA** sont mentionnés dans les références [38, 40, 49, 50].

Choix de la décomposition

Pour un programme quadratique général pas nécessairement convexe, plusieurs décompositions sont proposées dans [38, 39], c'est pour cela qu'il faut choisir la meilleure décomposition qui accélère la convergence de l'algorithme.

On considère le programme quadratique suivant :

$$\begin{cases} \min f(x) = \frac{1}{2}x^tQx + q^tx \\ x \in C, \end{cases} \tag{4.3}$$

où $Q \in \mathbb{R}^{n\times n}$ une matrice symétrique, $q \in \mathbb{R}^n$.

On prend dans notre travail

$$f(x) = g(x) - h(x),$$

où

$$g(x) = \chi_C(x) + \frac{1}{2}x^t(Q+\rho I_n)x + q^tx \text{ et } h(x) = \frac{1}{2}\rho\left\|x\right\|^2,$$

avec $\rho \geq \rho_{\min} = \max(0, -\lambda_{\min})$ où $\lambda_{\min}$ est la plus petite valeur propre de la matrice Q et $\chi_C(.)$ désigne la fonction indicatrice sur l'ensemble C. Il est clair que les fonctions g et h sont deux fonctions convexes.

Donc (4.3) est un programme **DC** dans la forme standard

$$\min\left\{f(x) = g(x) - h(x),\ x \in \mathbb{R}^n\right\}. \tag{4.4}$$

En suivant l'algorithme **DCA** et ses propriétés et ses bases théoriques décrites dans [39, 40], à chaque itération $k \geq 0$, on calcule $y^k = \left(\nabla h(x^k)\right)^t = \rho x^k$, et on trouve la solution notée x^{k+1} du problème de minimisation convexe

$$\min\left\{[g(x) - (h(x^k) + (y^k)^t(x-x^k)],\ x \in \mathbb{R}^n\right\},$$

qui est équivalent au problème quadratique convexe suivant

$$\min\left\{\frac{1}{2}x^t(Q+\rho I_n)x + x^t(q-y^k)\ ,\ x \in C\right\}.$$

Donc **DCA** appliqué au problème (4.4) est décrit comme suit :

Description de l'algorithme DCA avec la décomposition précèdente

Début algorithme

- **Initialisation**

  $k = 0$, $x^0 \in C$ un point connu, ε une précision donnée.

- **Tant que** le critère d'arrêt n'est pas vérifié faire
  - **Etape 1 :** Calculer $y^k = \rho x^k$.
  - **Etape 2 :** Calculer x^{k+1} une solution optimale du problème quadratique convexe suivant

$$\min \left\{ \frac{1}{2} x^t (Q + \rho I_n) x + x^t (q - y^k) \ , \ x \in C \right\}.$$

  - **Etape 3 :** Poser $k = k + 1$.

Fin algorithme

4.5 Illustrations numériques

Dans cette partie, nous présentons des éxpérimentations numériques préliminaires comparatives entre les méthodes de points intérieurs en utilisant l'algorithme de **Ye-Lustig** pour la première approche de Sivri et al. basée sur le développement de Taylor et notre approche basée sur l'algorithme **DCA**.

Les tests numériques réalisés sont effectués sur des exemples de la littérature. Le langage utilisé est le MATLAB. La précision choisie étant $\varepsilon = 10^{-6}$. Dans chaque exemple, on donne la solution optimale x^* trouvée par l'algorithme de **Ye-Lustig** et par celui de **DCA** ainsi que la valeur optimale z^*. On désigne par :

- k est le nombre des itérations nécessaire pour trouver une solution optimale.

Exemple 4.5.1 *On considère le programme fractionnaire quadratique suivant :*

$$(PFQ1) \begin{cases} \max \left(\frac{x_2^2 + x_1 x_2 + x_1 + 4x_2 + 3}{x_1^2 + x_1 x_2 + 6x_1 + 4x_2 + 8} \right) \\ -x_1 + x_2 \leq 1 \\ x_1 + 2x_2 \leq 7 \\ x_1, x_2 \geq 0. \end{cases}$$

Résultats obtenus par Ye-Lustig via l'approche de Sivri et al. :

Solution optimale trouvée après $k = 1$ *itération et* $0.0057s$ *est :*

$$x^* = (1.6667, 2.6667)^t.$$

La valeur optimale est : $z^* = 0.7492$.

Résultats obtenus par DCA via notre approche :

Solution optimale trouvée après $k = 2$ *itérations et* $0.029736s$ *est :*

$$x^* = (1.6667, 2.6667)^t.$$

La valeur optimale est : $z^* = 0.7492$.

Exemple 4.5.2 *On considère le programme fractionnaire quadratique suivant :*

$$(PFQ2)\begin{cases} \max\left(\frac{2x_1^2+2x_2^2+4x_1x_2+46x_1+46x_2+204}{-4x_1^2-x_2^2-4x_1x_2+8x_1-4x_2+165}\right) \\ x_1 + 2x_2 \leq 2 \\ 3x_1 + x_2 \leq 4 \\ -5x_1 + 3x_2 \leq 0 \\ -x_1 + x_2 \leq 0 \\ x_{1,}x_2 \geq 0. \end{cases}$$

Résultats obtenus Ye-Lustig via l'approche de Sivri et al. :

Solution optimale trouvée après $k = 1$ *itération et* $0.0067s$ *est :*

$$x^* = (1.2, 0.4)^t.$$

La valeur optimale est : $z^* = 1.6729$.

Résultats obtenus par DCA via notre approche :

Solution optimale trouvée après $k = 2$ *itérations et* $0.053218s$ *est :*

$$x^* = (1.2, 0.4)^t.$$

La valeur optimale est : $z^* = 1.6729$.

Commentaires

Les tests numériques préliminaires réalisés donnent la solution optimale de (PFQ) dans peu de temps avec un petit nombre d'itérations dans les deux algorithmes.

L'efficacité de ces méthodes se manifeste beaucoup plus dans les exemples de grande taille. Notre ambition est de confirmer cet avantage consolidé par des tests numériques en cherchant des exemples de grande taille.

Remarque 4.5.1 *Malgré que les tests numériques préliminaires effectués favorisent l'approche de Sivri et al., la convergence de son algorithme n'est pas garantie : aucun résultat de convergence n'est mentionné dans les travaux de Sivri et al. [55].*

Conclusion

Dans cette thèse, nous nous sommes intéréssés à la résolution d'un problème de programmation fractionnaire. Dans un premier temps, nous nous sommes focalisés sur la programmation fractionnaire linéaire, où nous avons proposé deux formulations de (PFL) en programmes équivalents.

- La première formulation consiste à convertir (PFL) en un (PL) équivalent, et puis résoudre ce dernier via une variante projective de points intérieurs de Ye-Lustig. L'avantage de cette formulation réside dans la conservation de la taille du problème (pas de changement ni d'ajout de variables ou de contraintes).
- La deuxième formulation consiste à convertir (PFL) en un (VIP) équivalent, et puis résoudre ce dernier en utilisant l'algorithme de projection de Grar et Benterki qui est considéré parmi les algorithmes les plus efficaces pour résoudre un (VIP).

Les tests numériques présentés montrent clairement l'efficacité des méthodes de points intérieurs pour résoudre les programmes linéaires déjà connus surtout pour les problèmes à grande taille.

Ensuite, nous nous sommes aussi intéressés à la résolution d'un problème fractionnaire quadratique (PFQ) en le transformant en deux programmes équivalents.

- Le premier programme équivalent est un (PL), obtenu sur la base de développement de Taylor de l'objectif, que nous résolvons par la suite à l'aide de l'algorithme projectif de points intérieurs de Ye-Lustig utilisé dans la première partie.
- Le deuxième programme équivalent est un programme quadratique (PQ), que nous résolvons à l'aide de l'algorithme DCA conçu pour résoudre les programmes quadratiques non convexes.

Les résultats numériques obtenus sont encourageants, ce qui nous motive à généraliser nos travaux à des problèmes fractionnaires non linéaires quelconques non nécessairement quadratiques.

Bibliographie

[1] S. Addoune : Une courte introduction à l'optimisation. Université Mohamed El Bachir El Ibrahimi Bordj Bou arréridj - Algérie (2017).

[2] K. Archana, S. R. Arora : An algorithm for solving quadratic fractional program with linear homogeneous constraints. Vietnam Journal of Mathematics **39** (2011), 391–404.

[3] A. Auslender : Optimisation (Méthodes numériques). Masson - Paris (1976).

[4] M. Babul Hasan, S. Acharjee : Solving LFP by converting it into a single LP. International Journal of Operations Research **8** (2011), 1–14.

[5] E. B. Bajalinov : Linear fractional programming theory, Methods, Applications and Software. Boston, Kluwer Academic Publishers (2003).

[6] M. S. Bazaraa, H. D. Sherali and C.M. Shetty : Nonlinear programming. Theory and Algorithms, Third Edition, Willey Sons (2006).

[7] A. Beck, A. Ben-Tal, M. Teboulle : Finding a global optimal solution for a quadratically constrained fractional quadratic problem with applications to the regularized total least squares. Society for Industrial and Applied Mathematics Siam Journal Matrix Analysis. Appl **2** (2006), 425–445.

[8] A. Bennani, D. Benterki : Efficient projective algorithm for linear fractional programming problem based on a linear programming formulation. International Journal of Computing Science and Mathematics, (à paraitre).

[9] A. Bennani, D. Benterki D, H. Grar : adaptive projection methods for linear fractional programming. RAIRO Operations Research. 55, (2021), S2383–S2392.

[10] G. R. Bitrand, A. J. Novas : Linear programming with a fractional objective function. Journal of Operations Research **21** (1973), 22–29.

[11] S. Boyd, L. Vandenberghe : Convex optimization, Cambridge University Press (2004).

[12] A. Cambini, L. Martein : A modified version of Martos's algorithm for the linear fractionnal problem. Methods of Operations Research **53** (1986), 33-44.

[13] Y. Censor, A. N. Iusem, S. A. Zenios : An interior point method with Bregman functions for variational inequality problem with paramonotone operators. Mathematical Programming **81** (1998), 373–400.

[14] A. Charnes, W. W. Cooper : Management models and industrial applications of linear programming, John Wiley, New York (1961).

[15] A. Charnes, W. W. Cooper : Programming with linear fractional functional. Naval Research Logistics Quarterly **9** (1962), 181–186.

[16] W. Dinkelbach : On nonlinear fractional programming. Management Science **13** (1967), 492–498.

[17] B. C. Eaves : On the basic theorem of complementarity. Mathematical Programming (1971), 68–75.

[18] C. Ferris, J. S. Pang : Complementarity and variational problems. State of Art, SIAM Publications, Philadelphia, Pennsylvania (1997).

[19] P. C. Gimore, R. E. Gomory : Linear programming approach to the cutting stock problem-part 2. Operations Research **11** (1963), 863–867.

[20] N. Gould : An introduction to algorithms for continuous optimization. Oxford University Computing Laboratory and Rutherford Appleton Laboratory (2006).

[21] H. Grar : Etude Comparative de quelques méthodes pour la résolution du problème varitaionnel à contraintes linéaires. Thèse de doctorat, Université Ferhat Abbas, Sétif (2012).

[22] H. Grar, A. Keraghel : Etude théorique et numérique d'une méthode de points intérieurs pour la résolution du problème d'inégalités variationnelles. Revue Sciences et Technologie A **26** (2007), 39–48.

[23] H. Grar, A. Keraghel : A new interior point method for complementarity. Applied Mathematical Sciences **41-42** (2008), 2081–2094.

[24] H. Grar, D. Benterki : New effective projection method for variational inequalities problem. RAIRO Operations Research **49** (2015), 805–820.

[25] P. T. Harker : Finite dimensional variational inequality and nonlinear complementarity problem : A survey of theory, algorithms and applications. Mathematical Programming **48** (1990), 161–220.

[26] J. R. Isbell, W. H. Marlow : Attrition games. Naval Research Logistics Quarterly **3** (1956), 1–99.

[27] A. N. Iussem : An iterative algorithm for variational inequalities problem. Comput. Appl. Math. **13** (1994), 103–114.

[28] A. I. Iusem, B. F. Svaiter : A variant of Korpolevich's method for variational inequalities with a new search strategy, Optimization **42** (1997), 309–321.

[29] S. Karamardian : An existence theorem for the complementarity problem. Journal of Optimization Theory and Applications **18** (1976), 445-454.

[30] N. K. Karmarkar : A new polynomial-time algorithm for linear programming. Proceedings of the 16^{th} Annual ACM Symposium on Theory of computing **4** (1984), 373–395.

[31] Z. Kebaili, M. Achache : On solving non monotone affine variational inequalities problem by DC programming and DCA. Asian-European Journal of Mathematics **1** (2018), 1–8.

[32] Z. Kebaili, D. Benterki : A penality approach for a box constrained variational inequality problem. Applications of mathematics **4** (2018), 439-454.

[33] A. Keraghel : Analyse convexe : théorie fondamentale et exercices, Editions Dar el'Houda, Ain M'lila, Algérie, (1999).

[34] D. Kinderlehrer, G. Stampacchia : An introduction to variational inequalities and their applications. Academic Press, New York (1980)

[35] G. M. Korpolevich : The extragradient method for finding saddle points and other problems. Matecon **12** (1976), 747–756.

[36] H. A. Le Thi, T. Pham Dinh : A continuous approach for large-scale constrained quadratic zero-one programming. (In Honor of Professor Elster, Founder of the Journal Optimization). Optimization **45** (2001), 1–28.

[37] H. A. Le Thi : An efficient algorithm for globally minimizing a quadratic function under convex quadratic constraints. Mathematical Programming **87** (2000), 401–426.

[38] H. A. Le Thi, T. Pham Dinh : On solving linear complementarity problems by DC programming and DCA. Computational Optimization and applications **50** (2011), 507–524.

[39] H. A. Le. Thi, T. Pham Dinh : Solving a Class of Linearly Constrained Indefinite Quadratic Problems by D.C. Algorithms. Journal of global optimization **11(3**) (1997), 253-285.

[40] H. A. Le Thi, T. Pham Dinh : The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems. Annals of Operations Research **133(3)** (2005), 23-46.

[41] I. J. Lustig : A practical approach to Karmarkar's algorithm. Technical report sol **85-5** System optimization laboratory ; department of Operations Research Stanford. university Stanford California 94305 (1985).

[42] I. J. Lustig : Feasibility issues in a primal-dual interior point method for linear programming. Mathematical programming **49** (1991), 145–162.

[43] B. Martos : Hyperbolic programming. Naval Research Logistics Quarterly **11** (1964), 135–155.

[44] A. Nejmaddin Suleiman, A. Maher Nawkhas : Solving quadratic fractional programming problem. International Journal of Applied Mathematical Research **2** (2013), 303–309.

[45] A. Nagih, G. Plateau : Problèmes fractionnaires : Tour d'horizon sur les applications et méthodes de résolution. RAIRO Operations Research **33** (1999), 383–419.

[46] A. Nagurney : Computational comparisons of spatial price equilibrium methods. Journal of Regional Science **27** (1987), 55-76.

[47] T. Pham Dinh : Algorithme de calcul d'une forme quadratique sur la boule unité de la norme maximum. Numerische Mathematik **45** (1985), 377–440.

[48] T. Pham Dinh : Algorithms for solving a class of non convex optimization problems. Methods of subgradients. Fermat days 85. Mathematics for Optimization, J.B. Hiriart Urruty (ed.), Elsevier Science Publishers, B.V. North-Holland, (1986).

[49] T. Pham Dinh, H. A. Le Thi : A DC optimization algorithm for solving the trust-region subproblem. SIAM Journal on Optimization **8(2)** (1998), 476–505.

[50] T. Pham Dinh, H. A. Le Thi : Convex analysis approach to DC programming. Theory, algorithms and applications. Acta Math. Vietnam **22(1)** (1997), 289-355.

[51] M. Rashidul Hasan, M. Babul Hasan : An alternative method for solving quadratic fractional programming problems with homogenous constraints.

Journal of Emerging Trends in Engineering and Applied Sciences **5** (2014), 11–19.

[52] R. T. Rockafellar : Convex analysis, published by Princeton University Press **41** William Street, Princeton, New Jersey.

[53] S. K. Saha, M. R. Hossain, M. K. Uddin, R. N. Mondal : A new approach of solving linear fractional programming problem (LFP) by using computer algorithm. Open Journal of Optimization **4** (2015), 74–86.

[54] J. K. Sharma, A. K. Gupta, M. P. Gupta : Extension of Simplex technic for solving programming problems. Indian Journal of Pure and Applied Mathematics **11** (1980), 961–968.

[55] M. Sivri, I. Albayrak, G. Temelcan : A novel approach for solving quadratic fractional programming problems. Croation Operational Research Review **9** (2018), 199–209.

[56] M. V. Solodov, B. F. Svaiter : A new projection method for variational inequality problems, SIAM Journal on Control and Optimization **37** (3) (1999), 765–776.

[57] K. Swarup : Linear fractional functionals programming. Operations Research **13 (**6**)** (1965), 1029–1036.

[58] N. Van-Bong, S. Ruey-Lin, X. Yong : An sdp approach for solving quadratic fractional programming problems. Cornel university (2014).

[59] C. Y. Wang, F. Zhao : Directional derivatives of optimal value functions in mathematical programming. Journal of Optimization Theory and Applications **82** (1994), 397–404.

[60] J. Wang, N. Xiu, C. Y. Wang : A new version of extragradient method for variational inequality problems. Computers and Mathematics with Applications (2001), 969–979.

[61] J. Wang, N. Xiu, C. Y. Wang : Unified framework of extragradient type method for pseudomonotone variational inequalities. Journal of Optimization Theory and Applications, **3** (2001), 641–656.

[62] A. Yassine : Méthode de région de confiance et optimisation DC, théorie, algorithmes et applications, HDR-Université Henri Poincaré, Nancy I, (1998).

[63] O. Zerdani : L'optimisation non linéaire multiobjectif, Thèse de doctorat, Université Mouloud Mammeri, Tizi-ouzou (2013).

ملخص :

تتعلق هذه الأطروحة بحل مسألة البرمجة الكسرية ذات النمط الخطي. في البداية، قمنا بتحويل المسألة المدروسة إلى برنامج خطي مكافئ من خلال صياغة مناسبة مع تجنب زيادة أبعاد المسألة الأصلية، ثم حلها بطريقة الإسقاط للنقاط الداخلية. في الخطوة الثانية، باستخدام مسائل المتباينات التغيرية، قمنا بحل برنامج الكسر الخطي بطريقة حديثة للإسقاط. أخيرًا ، نجحنا في الاختبارات العددية للخوارزميات الناتجة. نتائج هذه الاختبارات التي حصلنا عليها مرضية، مما يدل على فعالية طرق النقاط الداخلية.

الكلمات المفتاحية : مسألة البرمجة الكسرية ذات النمط الخطي، مسألة البرمجة الخطية، مسألة المتباينات التغيرية، طرق النقاط الداخلية، طرق الإسقاط.

Abstract:

This thesis concerns the resolution of an optimization problem of linear fractional programming type. Initially, the considered problem will be transformed into an equivalent linear program via an adequate formulation while avoiding the increase of the dimension of the initial problem, then solved by means of a interior point projective methods. In a second step, using the variational inequalities problems, we solved the linear fractional program by a recent projection method. Finally, we have successfully established the numerical implementation of the resulting algorithms. The numerical results obtained are satisfactory, highlighting the effectiveness of the approach of interior point methods.

Keywords: Linear fractional programming, Linear programming, Variational inequalities problem, Interior point methods, Projection methods.

Résumé :

Cette thèse concerne la résolution d'un problème d'optimisation de type programmation fractionnaire linéaire. Dans un premier temps, le problème considéré sera transformé en un programme linéaire équivalent via une formulation adéquate tout en évitant l'augmentation de la dimension du problème initial, puis résolu moyennant une méthode projective de points intérieurs. Dans une deuxième étape, moyennant les problèmes des inégalités variationnelles, nous avons résolu le programme fractionnaire linéaire par une méthode de projection récente. Enfin, nous avons établi avec succès l'implémentation numérique des algorithmes obtenus. Les résultats numériques obtenus sont satisfaisants mettant en évidence l'efficacité de l'approche des méthodes de points intérieurs.

Mots clés : Programmation fractionnaire linéaire, Programmation linéaire, Problème d'inégalités variationelles, Méthodes de points intérieurs, Méthodes de projection.