République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université Hadj Lakhdar-Batna-

Faculté des Sciences

Département de Mathématiques

Thèse

Présentée en vue de l'obtention du Diplôme de

DOCTORAT EN SCIENCES

Option : Mathématiques appliquées

Par

El Amir DJEFFAL

Thème

ETUDE DE QUELQUES ALGORITHMES DE POINTS INTERIEURS POUR LA PROGRAMMATION CONVEXE

Soutenue le : 10 septembre 2013

Devant le jury d'examen :

Président :	Rachid Benacer	Pr. Université Hadj Lakhdar. Batna
Rapporteur :	Lakhdar Djeffal	M.C.A. Université Hadj Lakhdar. Batna
Examinateurs :	Abdelhamid Ayadi	Pr. Université Larbi ben M'hidi. O.E.B
	Khaled Melkemi	Pr. Université Mohamed Kheidar. Biskra
	Nacer Adjeroud	M.C.A. Université Abbas Laghror. Khenchela
	Nacer Khelil	M.C.A. Université Mohamed Kheidar. Biskra

A Thesis
Presented to
The Academic Faculty

by

In Partial Fulfillment
of the Requirements for the Degree

Georgia Institute of Technology
October 2013

Approved by:

Date Approved _____

Table des matières

Liste des tableaux

Table des figures

Introduction

La programmation mathématique, branche de l'optimisation, s'occupe de la minimisation (maximisation) sous contraintes d'une fonction à plusieurs variables, schéma très général s'appliquant à de nombreuses situations pratiques dans beaucoup de domaines (minimisation de coûts, de durées, ... etc.)

Dans le cas d'une fonction et de contraintes linéaires (programmation linéaire), on dispose d'une méthode efficace de résolution : l'algorithme du simplexe, découvert par Dantzig en 1947 [17]. Cet algorithme a connu depuis lors de nombreuses améliorations, et est utilisé dans la majorité des logiciels commerciaux.

Cependant, un nouveau type de méthodes de résolution a fait son apparition en 1984 [43] : les méthodes de points intérieurs. La plupart des idées sous-jacentes à ces méthodes proviennent du domaine de la programmation non linéaire. Parmi leurs avantages, citons

1. Efficacité théorique : il est possible de prouver que ces méthodes s'exécutent en temps polynomial (ce qui n'est pas le cas de l'algorithme du simplexe, de nature exponentielle).
2. Efficacité pratique : les temps de calcul et de réponse de ces méthodes sont compétitifs (et battent l'algorithme du simplexe dans le cas de problèmes de grande taille).
3. Traitement de très grands problèmes : ces méthodes permettent de résoudre des problèmes de très grande taille qu'aucun autre algorithme connu ne pourrait traiter en un temps acceptable.
4. Adaptation au cas non linéaire : il est possible d'adapter ces méthodes à la programmation non linéaire, plus particulièrement à la programmation convexe, ce qui permet de traiter de nouveaux types de problèmes pour lesquels on ne

connaissait jusqu'à présent aucune méthode de résolution efficace.

Nous allons à présent donner un bref aperçu historique du domaine de la programmation mathématique, afin de situer la découverte des méthodes de points intérieurs dans son contexte.

Débuts de la programmation linéaire - Algorithme du simplexe

$1930 - 1940$: Premières formulations de problèmes de programmation linéaire.

$1939 - 1945$: Seconde guerre mondiale. La recherche opérationnelle fait ses débuts, applications militaires.

1947 : Georges B. Dantzig publie un article relatif à l'algorithme du simplexe de programmation linéaire [17].

1956 : A. J. Goldman et A. W. Tucker démontrent l'existence d'une solution strictement complémentaire au problème de programmation linéaire [39].

1972 : V. Klee et G. Minty démontrent le caractère exponentiel de la complexité algorithmique de la méthode du simplexe [48].

Débuts des méthodes de points intérieurs et complexité polynomiale

1955 : K. R. Frisch invente une méthode de points intérieurs pour résoudre un problème non linéaire (méthode de potentiel logarithmique, utilisation d'une fonction barrière) [38].

1968 : A. V. Fiacco et G. P. McCormick développent l'utilisation des méthodes de points intérieurs en programmation convexe non linéaire [36].

1978 : L. G. Kachian applique la méthode de l'ellipsoïde (développée par N. Shor en 1970, [74]) à la programmation linéaire et prouve qu'elle est de complexité polynomiale [47]. Cette méthode, bien que faisant converger une suite d'itérés, n'est cependant pas une méthode de points intérieurs au sens accepté actuellement.

Il faut remarquer qu'à cette époque, on utilise les méthodes de points intérieurs dans le cadre non linéaire uniquement. Bien que Fiacco et McCormick notent que leurs méthodes sont également applicables au cas linéaire, ils ne les considèrent pas

sérieusement comme une alternative viable à l'algorithme du simplexe.

De plus, la méthode de Kachian, bien que théoriquement supérieure à l'algorithme du simplexe (du point de vue de la complexité de pire cas), est en pratique bien plus lente, parce qu'elle atteint principalement son pire cas pour la majorité des problèmes, alors que le simplexe n'exhibe qu'exceptionnellement son comportement de pire cas exponentiel [13].

Explosion des méthodes de points intérieurs

1984 : N. K. Karmarkar découvre une méthode de points intérieurs polynomiale plus efficace que celle de Kachian, qu'il affirme également comme supérieure à l'algorithme du simplexe [43].

1994 : Y. Nesterov et A. Nemirovsky publient leurs études sur les méthodes de points intérieurs polynomiales appliquées à la programmation convexe [62].

1997 : Depuis l'annonce de Karmarkar, plus de 2000 articles de recherche portant sur les méthodes de point intérieur ont été publiés par la communauté scientifique [37]. Les premiers ouvrages récapitulatifs paraissent. Les recherches se dirigent vers la programmation non linéaire.

Ces méthodes ont fait leurs preuves dans le domaine de la programmation linéaire (PL) notamment par leur bonnes propriétés théoriques (complexité polynomiale et convergence superlinéaire) et leur bon comportement numérique. Aussitôt, des variantes sont introduites pour la programmation quadratique et la complémentarité linéaire. Ceci étant, il faut signaler tout de même que (sur le plan technique), ces méthodes présentent des inconvénients d'ordre théorique et numérique entre autre : le problème d'initialisation et le coût excessif de l'itération lié aux choix de la direction et du pas de déplacement.

Notre étude est répartie en trois volets :

Le premier est un étude comparative entre trois algorithmes de points intérieurs, celle de la trajectoire centrale. L'effort est concentré sur le problème d'initialisation

pour lequel nous proposons trois alternatives : trajectoire centrale non réalisable, trajectoire centrale réalisable et trajectoire centrale réalisable amélioré.

Le deuxième volet concerne la comparaison entre l'extension d'une méthodes projective et une méthode de la trajectoire centrale basée sur une nouvelle classe de la direction de descente pour un problème de complémentarité linéaire.

Le troisième volet concerne la programmation semi définie

La thèse est présentée en quatres chapitres.

Le premier chapitre, est une introduction générale. Celle ci contient un survol rapide : historique, éléments d'analyse convexe, programmation mathématique, conditions d'optimalité de la programmation mathématique.

Le deuxième chapitre est réservé pour une étude comparative théorique et numérique d'une méthode Newtonienne basée sur trois algorithmes pour la programmation quadratique convexe.

Dans le troisième chapitre, on s'intéresse à l'extension d'une méthode projective et étude d'une méthode de la trajectoire centrale pour résoudre un problème de complémentarité linéaire basée sur une nouvelle classe pour déterminer la direction de descente.

Dans le dernier chapitre, nous proposons une nouvelle fonction de proximité pour un problème d'optimisation semi défini (SDO) par une nouvelle fonction noyau. En outre, nous formulons un algorithme de points intérieurs primal-dual pour (SDO) en utilisant la fonction de proximité et de donner son analyse de la complexité.

Enfin, nous clôturerons ce travail par une conclusion et perspectives suivi par Annexe et Bibliographie.

Chapitre I

PRÉLIMINAIRES ET NOTIONS FONDAMENTALES

Dans ce chapitre, nous rappelons brièvement certaines propriétés de l'analyse convexe, et quelques notions fondamentales de la programmation mathématique.

1.1 Eléments d'analyse convexe

1.1.1 Notions de convexité

La notion de convexité prend deux formes : un ensemble convexe, et une fonction convexe.

Définition 1 - *Un ensemble D est convexe si pour toute paire de points x et y appartenant à D alors :*

$$tx \ + (1-t)y \in D \ , \forall \ 0 \leq t \leq 1$$

c'est-a-dire : le segment de droite reliant deux points de D est entièrement contenu dans D.

- Un ensemble est dit affine si pour toute paire de points x et y appartenant à D, $t \in \mathbb{R}$ alors :

$$tx \ + (1-t)y \in D$$

- Un ensemble D est un polyèdre s'il s'écrit comme suit :

$$D = \{x \in \mathbb{R}^n : p_i^t x \leq \alpha_i \ , i = 1, ..., m\},$$

où p_i est un vecteur non nul de $\mathbb{R}^n$ et α_i est un scalaire pour $i = 1, ..., m$.

- Une fonction f est convexe sur un ensemble convexe D si :

$$f \ (tx + (1-t)y) \leq t \ f(x) + (1-t)f(y) \ , \forall \ 0 \leq t \leq 1$$

c'est-a-dire : l'épigraphe $\{(x, y) : x \in D, y \geq f(x)\}$de la fonction f est un ensemble convexe.

- Une fonction f est strictement convexe sur un ensemble convexe D si :

$$f \ (tx + (1-t)y) < t \ f(x) + (1-t)f(y), \forall \ 0 < t < 1.$$

- Une fonction f est (strictement) concave sur un ensemble convexe D si $(-f)$ est (strictement) convexe sur D.

- Le gradient d'une fonction $f : \mathbb{R}^n \to \mathbb{R}$ continûment différentiable évalué au point $x \in \mathbb{R}^n$ s'écrit :

$$\nabla f(x) = \left(\begin{array}{ccccc} \frac{\partial f(x)}{\partial x_1} & \frac{\partial f(x)}{\partial x_2} & \frac{\partial f(x)}{\partial x_3} & . \ . \ . & \frac{\partial f(x)}{\partial x_n} \end{array} \right)^t$$

et l'élément de la $i^{ème}$ ligne et la $j^{ème}$ colonne de la matrice Hessienne s'écrit :

$$\left[\nabla^2 f(x)\right]_{i\ j} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}.$$

1.1.2 Convexité et Dérivée

Les deuxièmes dérivées peuvent servir à déterminer la convexité ou la concavité d'une fonction.

Définition 2 *Soit $f : \mathbb{R}^n \to \mathbb{R}$ une fonction continûment différentiable sur un domaine convexe D.*

f est une fonction convexe sur D si et seulement si la matrice Hessienne est semi-définie positive, c'est-a-dire : pour tout $x \in D$, $y^t \nabla^2 f(x) y \geq 0$, $\forall y \in \mathbb{R}^n$. De même, si la matrice Hessienne est définie positive, c'est-a-dire : pour tout $x \in D$, $y^t \nabla^2 f(x) y > 0$, $\forall y \neq 0 \in \mathbb{R}^n$, alors f est une fonction strictement convexe sur D.

Une matrice A est semi-définie (définie) positive si et seulement si les déterminants des mineurs principaux de A sont (strictement) positifs.

Il s'ensuit qu'on peut vérifier si une fonction est convexe en examinant le signe des mineurs principaux de la matrice Hessienne. De même, on peut vérifier la concavité d'une fonction f en analysant la convexité de $(-f)$. Il n'est pas vrai qu'une fonction est concave si les déterminants des mineurs principaux sont tous négatifs.

Exemple

considérons la fonction $f(x_1, x_2) = -(x_1 - 2x_2)^2 - \exp(x_1)$. Le gradient et la matrice Hessienne s'écrivent :

$$\nabla f(x) = \begin{bmatrix} -2(x_1 - 2x_2) - \exp(x_1) \\ 4(x_1 - 2x_2) \end{bmatrix}, \nabla^2 f(x) = \begin{bmatrix} -2 - \exp(x_1) & 4 \\ 4 & -8 \end{bmatrix}$$

Les déterminants des mineurs principaux sont respectivement :

$$-2 - \exp(x_1) < 0 \text{ et } 8\exp(x_1) > 0$$

On ne peut pas conclure que la fonction est convexe. Cependant, les déterminants des mineurs principaux de la matrice Hessienne $\nabla^2(-f(x))$ sont :

$2 + \exp(x_1) > 0$ et $8\exp(x_1) > 0$. Donc on peut conclure que $(-f)$ est convexe et par conséquent f est concave.

1.2 Programmation mathématique

La programmation mathématique, et plus particulièrement l'optimisation vise à résoudre des problèmes où l'on cherche à déterminer parmi un grand nombre de solutions candidates celle qui donne le meilleur rendement. Plus précisément, on cherche à trouver une solution satisfaisant un ensemble de contraintes qui minimise ou maximise une fonction donnée. L'application de la programmation mathématique est en expansion croissante et se retrouve dans plusieurs domaines.

Un programme mathématique (PM) est un problème d'optimisation de la forme :

$$(PM) \begin{cases} \min \ f(x) \\ s.c \\ D = \{x \in \Omega \subseteq \mathbb{R}^n \ / \ g_i(x) \leq 0, \ i = 1, ...n \ \text{ et } \ h_j(x) = 0, \ j = 1, ..., m\} \end{cases}$$

où $f, \ g_i, \ hj$ sont des fonctions définies de $\mathbb{R}^n$ dans $\mathbb{R}$.

La formulation mathématique (PM) signifie que l'on cherche à trouver la solution $x^* \in D$ dont la valeur de la fonction objectif est la plus petite.

Définition 3 *-Une solution $x^* \in D$ est un minimum global de (PM) si $f(x^*) \leq f(x), \forall x \in \mathbb{R}^n$. La valeure optimale est $f(x^*)$.*

- Une solution $x^* \in D$ est un minimum local de (PM) si $f(x^*) \leq f(x)$ $\forall x \in D \cap B_\epsilon(x^*)$ où $B_\epsilon(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| < \epsilon\}$est une boule ouverte de rayon ε $(\varepsilon > 0)$ centrée en x^* .

Remarque 4 *Notons que le minimum global n'est pas nécessairement unique, mais la valeure optimale l'est. Par exemple, $\min_{x \in \mathbb{R}} \sin x$ possède une infinité de minima globaux.*

1.2.1 Classification des problèmes d'optimisation

Il existe beaucoup d'algorithmes d'optimisation dans différentes applications scientifiques. Cependant beaucoup de méthodes ne sont valables que pour certains types de problèmes. Ainsi, il est important de bien connaitre les caractéristiques du problème posé afin d'identifier la technique approprieé pour sa résolution.

les problèmes d'optimisation sont classés en fonction des caractéristiques mathématiques de la fonction objectif, des contraintes et des variables d'optimisation (Tableau1). Il existe une classe particulière de problèmes qui concerne notamment le domaine de la recherche opérationnelle, où le but est de trouver la permutation optimale des variables d'optimisation (Tableau1). Ces problèmes sont connus sous le nom de problème d'*optimisation combinatoire.*

Caractéristiques	Propriétés	Classifications
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 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

Tableau $-1-$ *Classification des problèmes d'optimisations*

1.2.2 Qualification des contraintes

• Si un ensemble D est un polyèdre convexe (c'est-a-dire : toutes les fonctions contraintes sont affines), alors par définition les contraintes sont qualifiées en tout point réalisable.

• Si un ensemble D est convexe et $intD \neq \emptyset$, les contraintes sont qualifiées partout. C'est la condition de Slater.

• Une contrainte d'inégalité $g_i(x) \leq 0$ est dite saturée en $x^* \in D$ si $g_i(x^*) = 0$

Remarque 5 *La résolution complète de* (PM) *traite dans l'ordre les points suivants :*

• *L'existence d'une solution optimale.*

• *La caractérisation de la solution (il s'agit des conditions d'optimalité).*

• *L'élaboration d'algorithmes pour calculer cette solution.*

1.3 Résolution d'un programme mathématique

Soit le programme mathématique suivant :

$$(PM)\begin{cases} \min\ f(x) \\ s.c \\ D=\{x\in\Omega\subseteq\mathbb{R}^n\ /\ g_i(x)\leq 0,\ i=1,...n\ \text{ et }\ h_j(x)=0,\ j=1,...,m\} \end{cases}$$

où f, g_i, h_i sont des fonctions définies de $\mathbb{R}^n$ dans $\mathbb{R}$.

1.3.1 Existence & Unicité

Théorème 6 *[8]**(Weierstrass)***

Si D est compact non vide de $\mathbb{R}^n$ et si f est continue sur D alors (PM) admet au moins une solution optimale globale $x^* \in D$.

Théorème 7 *[8]*

Si D est non vide et fermé de $\mathbb{R}^n$, f est continue et coercive sur D, (c'est-a-dire : $\lim_{\|x\|\to+\infty} f(x) = +\infty$) alors (PM) admet au moins une solution optimale globale.

1.3.2 Conditions d'optimalité

Pourquoi avons-nous besoin de conditions d'optimalité ?

Afin d'analyser ou de résoudre d'une manière efficace un problème d'optimisation, il est fondamental de pouvoir disposer des conditions d'optimalité. En effet, celles-ci nous servent non seulement à vérifier la validité des solutions obtenues, mais souvent l'étude de ces conditions aboutit au développement des algorithmes de résolution eux-mêmes. L'approche considérée ici pour l'obtention de conditions est basée sur les notions de descente et de direction admissible.

Le développement des conditions d'optimalité en présence des contraintes est basé sur la même intuition que dans le cas sans contraintes : il est impossible de descendre

à partir d'un minimum.

On considère le programme mathématique suivant :

$$(PM)\begin{cases} \min\ f(x) \\ s.c \\ D=\{x\in\Omega\subseteq\mathbb{R}^n\ /\ g_i(x)\leq 0,\ i=1,...n\ \text{ et }\ h_j(x)=0,\ j=1,...,m\} \end{cases}$$

où f, g_i, h_i sont des fonctions définies de $\mathbb{R}^n$ dans $\mathbb{R}$.

Théorème 8 ***[64] (Karush-kuhn-tucker. contraintes linéaires)***

Soit $f:\mathbb{R}^n\to\mathbb{R}$ une fonction différentiable sur D. Si x^**un minimum local du problème* (PM)*, alors, il existe un vecteur* $y\in\mathbb{R}^m$*et* $\lambda\in\mathbb{R}^n_+$ *tel que :*

$$\begin{cases} \nabla f(x^*)+\sum_{i=1}^{n}\lambda_i\nabla g_i(x^*)+\sum_{j=1}^{m}y_j\nabla h_j(x^*)=0 \\ \lambda_i g_i(x^*)=0,\ i=1....,n \\ h_j(x^*)=0,\ j=1,...,m \end{cases}$$

1.3.3 Méthode de Newton pour résoudre un système non linéaire

La résolution du (PM) revient à résoudre le système d'équation non linéaire suivant :

$$\begin{cases} \nabla f(x)+\sum_{i=1}^{n}\lambda_i\nabla g_i(x)+\sum_{j=1}^{m}y_j\nabla h_j(x)=0 \\ \lambda_i g_i(x^*)=0,\ i=1....,n \\ h_j(x^*)=0,\ j=1,...,m \end{cases}$$

Posons

$$F(x,\lambda,y) = \begin{pmatrix} \nabla f(x) + \sum_{i=1}^{n} \lambda_i \nabla g_i(x) + \sum_{j=1}^{m} y_j \nabla h_j(x) = 0 \\ \lambda_i g_i(x) = 0,\ i = 1....,n \\ h_j(x) = 0,\ j = 1,...,m \end{pmatrix}$$

où

$$\begin{aligned} F \quad : \quad & \Re^{2n+m} \to \Re^{2n+m} \\ (x,\lambda,y) \quad \mapsto \quad & \left(\nabla f(x) + \sum_{i=1}^{n} \lambda_i \nabla g_i(x) + \sum_{j=1}^{m} y_j \nabla h_j(x), \lambda_i g_i(x^*), h_j(x) \right). \end{aligned}$$

Les méthodes les plus populaires appliquées pour la résolution d'un système non linéaire est la méthode de Newton, dans ce qui suit décrivons son principe.

Soit $f : \Re^n \to \Re^n$ une fonction continue, différentiable et soit $J(x)$ est la matrice jacobiènne de la fonction f.Alors nous considéroons le système non linéaire suivant :

$$f(x) = 0$$

A partir d'un vecteur x^0 de $\Re^n$ et l'utilisation de la formule suivante

$$x^{k+1} = x^k - J(x^k)^{-1} f(x^k),$$

on construit une suite de point définie par

$$x^{k+1} = x^k + d^k,$$

où d^k est le vecteur de direction, avec

$$J(x^k)d^k = -f(x^k).$$

1.3.4 Les méthodes de directions admissibles

Cette classe de méthodes résout un problème de minimisation non linéaire en se déplaçant d'un point de D vers un autre de ses points au coût inférieur. Elles fonctionnent selon le principe suivant : étant donné un élément x^k de D, une direction d^k est générée telle que pour un $\alpha^k > 0$ et suffisamment petit, les propriétés suivantes sont assurées :

1. $x^k + \alpha^k d^k$ appartient toujours à D
2. $f(x^k + \alpha^k d^k)$ est inférieur à $f(x^k)$

Une fois d^k déterminée, α^k s'obtient par minimisation monodimentionnelle pour que le déplacement dans la direction d^k soit optimal, mais cette fois-ci il est nécessaire d'imposer une borne supérieure sur la valeur de α^k afin de ne pas sortir de D. Cela définit le nouveau point x^{k+1} et le processus est recommencé.

1.3.5 Méthodes de résolution d'un programme mathématique

On peut classifier les méthodes de résolutions un programme mathématique en trois catégories

1.3.5.1 Méthodes de type gradient.

Gradient conjugué Cette méthode a été proposée par Hestenes (1952) pour résoudre un système linéaire à matrice définie positive, puis généralisée par Fletcher et Reeves (1964) pour résoudre des problèmes d'optimisation non linéaires, elle est connue par son efficacité pour minimiser une fonction quadratique sans contraintes. Dans le cas contraint un changement de variable simple permet de se ramener au cas sans contraintes, en effet : x^0 un point satisfaisant les contraintes ($Ax^0 = 0$) et posons $x = x^0 + P_A y$ tel que $P_A = I - A^t(AA)^{-1}A$ est l'opérateur de la projection sur le noyau de la matrice A. Le principe de cette méthode est de construire progressivement des directions $d_0, d_1, ..., d_k$ mutuellement conjuguées par rapport à

la matrice Hessienne ($\nabla^2 f(x)$) de la fonction objectif du problème d'optimisation : $d_i \nabla^2 f(x) d_j = 0,\ \forall i \neq j,\ i, j = \{0, 1, ..., k\}$

Gradient projeté (Rosen 1960). le principe de cette méthode est de projeter à chaque itération le gradient sur la frontière du domaine réalisable. Il faut signaler que cette méthode est conçue pour un programme plus général de la forme :

$$\min \{f(x) : Ax = b, x \geq 0\},$$

où f est différentiable non nécessairement convexe.

1.3.5.2 Méthodes simpliciales.

Parmi les méthodes simpliciales on cite, celle de gradient réduit dûe à Wolfe. C'est une extension directe de la méthode du simplexe, appliquée à la programmation quadratique. De ce fait elle présente les mêmes inconvénients à savoir cyclage et complexité exponentielle.

1.3.5.3 Méthodes de points intérieurs.

Conjointement aux méthodes décrites précédemment, il existe actuellement des méthodes de points intérieurs pour la résolution d'un problème d'optimisation convexe. Ce sont des extensions des méthodes développées pour la programmation linéaire (affines, projectives et de trajectoire centrale). Les problèmes d'initialisation, le coût de l'itération et le choix de la direction de descente deviennent plus pesants.

On distingue trois classes fondamentales de méthodes de points intérieurs à savoir :

Méthodes Affines Il s'agit pratiquement de l'algorithme de Karmarkar sans fonction potentiel et sans transformation projective, on utilise une transformation affine et on remplace la contrainte de non négativité par un ellipsoide qui contient le nouveau itéré. L'algorithme est d'une structure simple, malheureusement, il n'est pas facile de démontrer la polynômialité.

Méthodes de réduction du potentiel La fonction potentiel joue un grand rôle dans le développement des méthodes de points intérieurs. L'algorithme de Karmarkar appliqué au programme linéaire sous forme standard utilise une fonction potentiel de la forme : $(n+1)\log(c^t x - z) - \sum_{i=1}^{n} \log(x_i)$ où z est une borne inférieure de la valeur optimale de l'objectif. Karmarkar prouve la convergence et la polynômialité de son algorithme par monter que cette fonction est réduit à chaque itération par au moins une constante. Depuis 1987, les chercheurs introduisent des fonctions du potentiel de type primales-duales, parmi lesquelles, celle de Tanabe, Todd et Ye [81] défnie par : $\Phi_\rho(x,s) = \rho\log(x^t s) - \sum_{i=1}^{n} \log(x_i s_i)$ pour $\rho > n$. Cette fonction a joué un rôle trés important dans le développement des algorithmes de réduction du potentiel aprés 1988. Les algorithmes correspondants à ces méthodes possèdent une complexité polynômiale, ils nécessitent $O(\sqrt{n}\,|\log\varepsilon|)$ itérations pour réduire le saut de dualité.

Méthodes de la trajectoire centrale Elles ont été introduites à la même époque que les méthodes de réduction du potentiel et pleinement développées au début des années 90. Elles possèdent de bonnes propriétés théoriques : une complexité polynômiale et une convergence superlinéaire.

Les algorithmes de la trajectoire centrale restreintent les itérés à un voisinage du chemin central, ce dernier est un arc de points strictement réalisables.

Plusieurs chercheurs essaient de généraliser le principe de ces méthodes pour la programmation non linéaire. En 1989, Megiddo [58] a proposé un algorithme primal-dual de trajectoire centrale pour la programmation linéaire avec une généralisation pour le problème de complémentarité linéaire (LCP). Kojima et al. [52] ont développé un algorithme primal-dual pour la programmation linéaire, une extension pour le (LCP) est proposée par les mêmes chercheurs en 1989 avec la complexité $O(\sqrt{n}\log\frac{1}{\varepsilon})$ itérations.

Chapitre II

MÉTHODES DE POINTS INTÉRIEURS DE TYPE NEWTONIENNE POUR RÉSOUDRE UN PROGRAMME QUADRATIQUE CONVEXE (CQP)

Dans ce chapitre, on s'intéresse à la résolution d'un problème d'optimisation quadratique convexe sous contraintes linéaires suivant :

$$\min\left\{\frac{1}{2}x^tQx + c^tx : Ax = b, x \geq 0\right\}, \qquad (CQP)$$

et son dual

$$\max\left\{b^ty - \frac{1}{2}x^tQx + c^tx : A^ty + s = Qx + c, s \geq 0\right\}, \qquad (CQD)$$

où $Q \in \Re^{n\times n}$ est une matrice symétrique semi définie positive, $A \in \Re^{m\times n}$ est une matrice, c, x et $s \in \Re^n$, y et $b \in \Re^m$.

On note par :

$\mathcal{F}_{(CQP)} = \{x \in \Re^n, Ax = b, x \geq 0\}$, l'ensemble des solutions réalisables de (CQP).

$\overset{o}{\mathcal{F}}_{(CQP)} = \{x \in \Re^n, Ax = b, x > 0\}$, l'ensemble des solutions strictement réalisables de (CQP).

$\mathcal{F}_{(CQD)} = \{y \in \Re^m, s \in \Re^n, A^ty + s = Qx + c, s \geq 0\}$, l'ensemble des solutions réalisables de (CQD).

$\overset{o}{\mathcal{F}}_{(CQD)} = \{y \in \Re^m, s \in \Re^n, A^ty + s = Qx + c, s > 0\}$, l'ensemble des solutions strictement réalisables de (CQD).

Au long de ce chapitre, on suppose que :

Hypothèse 1 : $\overset{o}{\mathcal{F}} = \overset{o}{\mathcal{F}}_{(CQP)} \times \overset{o}{\mathcal{F}}_{(CQD)} \neq \emptyset$.

Hypothèse 2 : $rg(A) = m < n$

Dans ce chapitre, on présente une étude comparative entre trois algorithmes d'une méthode de points intérieurs de type trajectoire centrale pour résoudre un problème quadratique convexe sous contraintes linéaires, citons :

1. Algorithme de la trajectoire centrale non réalisable.
2. Algorithme de la trajectoire centrale réalisable.
3. Algorithme de la trajectoire centrale réalisable amélioré.

2.1 Méthode de la trajectoire centrale non réalisable pour la (CQP)

On considère le programme quadratique convexe (CQP) suivant :

$$\min \left\{ \frac{1}{2} x^t Q x + c^t x : Ax = b, x \geq 0 \right\}, \qquad (CQP)$$

où $Q \in \Re^{n \times n}$ est une matrice symétrique semi définie positive, $A \in \Re^{m \times n}$ est une matrice de rang plein, $c, x \in \Re^n$ et $b \in \Re^m$.

2.1.1 Présentation et principe de la méthode

On associe à (CQP) le problème paramétrisé (CQP_μ) suivant :

$$\min \left\{ f_\mu(x) : Ax = b, x > 0 \right\}, \qquad (CQP_\mu)$$

où

$$f_\mu(x) = \frac{1}{2} x^t Q x + c^t x - \mu \sum_{i=1}^{n} \ln(x_i), \ \mu > 0$$

Lemme 9 *La fonction $f_\mu(x)$ est strictement convexe.*

Preuve. *La fonction $f_\mu(x)$ est écrite sous la forme :*

$$f_\mu(x) = \frac{1}{2} x^t Q x + c^t x - \mu \sum_{i=1}^{n} \ln(x_i), \ \mu > 0$$

En effet, $f_\mu(x) \in C^\infty$ *et nous avons en particulier* :

$$\nabla f_\mu(x) = Qx + c - \mu\ X^{-1}e,\ \textit{avec}\ \ e = (1, ...,\ 1)^t \in \mathbb{R}^n$$

$$\nabla^2 f_\mu(x) = Q + \mu\ X^{-2}.$$

Où $Q \in \Re^{n\times n}$*est une matrice symétrique semi définie positive,* $X\ = diag(x_1, ..., x_n)$ *est une matrice définie positive, car* $x_i > 0$*, et* $\mu > 0$*, alors* $\nabla^2 f_\mu(x)$ *est une matrice définie positive, donc* $f_\mu(x)$ *est convexe.* ■

La solution $x(\mu)$ est définie d'une façon unique par des conditions d'optimalité de Karush-Kuhn-Tucker (KKT) suivant :

$$\begin{cases} Qx + c - \mu X^{-1}e - A^t y = 0 \\ Ax = b \end{cases} \qquad (KKT)$$

Où $y \in \mathbb{R}^m$est le multiplicateur de Lagrange associé à la contrainte $Ax = b$ du problème (CQP_μ).

Posons $s = \mu X^{-1}$ avec $s \in \mathbb{R}^n_{++}$, le système (KKT) devient :

$$(S_\mu) \begin{cases} Xs = \mu\ e \\ Ax = b,\ x > 0 \\ A^t y + s = Qx + c,\ s > 0 \end{cases}$$

Notons que (S_μ) correspond aux conditions de complémentarité pour un problème de programmation quadratique primal-dual.

Le système (S_μ) désigne aussi les conditions d'optimalité du problème dual paramétrisé (CQD_μ) suivant :

$$\max \left\{ b^t y\ -\frac{1}{2}x^t Q x + \mu \sum_{i=1}^{n} \ln(s_i) : A^t y + s = Qx + c, s > 0 \right\} \qquad (CQD_\mu)$$

En effet, les conditions d'optimalité de (KKT) pour ce dernier problème sont données par :

$$(S'_\mu)\begin{cases} b - Ax = 0 \\ \mu S^{-1}e - x = 0 \\ A^t y + s = Qx + c \end{cases}$$

Où $x \in \mathbb{R}^n_{++}$ est le multiplicateur associé à la contrainte $A^t y + s = Qx + c$ et $S = diag(s_1, ..., \ s_n)$. D'où (S'_μ) est équivalent à (S_μ).

Le système (S_μ) est un système d'équations non linéaire. Pour cela, la méthode de Newton est envisageable pour sa résolution. A chaque $\mu > 0$, on trouve une solution approchée $(x(\mu), y(\mu), \ s(\mu))$ qui doit être faisable (Conditions de faisabilité).

Pour contrôler la faisabilité et l'optimalité, on introduit la fonction de mérite suivante

$$\Phi(x, y, s) = x^t s + \|Ax - b\|_2 + \left\|A^t y + s - Qx - c\right\|_2 \tag{1}$$

Nous résolvons le système d'équations non linéaire (S_μ) à partir d'un point initial $(x^0, s^0) > 0$, $y \in \Re^m$ et $\mu^0 = \frac{(x^0)^t s^0}{n}$, par la méthode de Newton, on obtient le système suivant :

$$\begin{cases} -Q\triangle x + A^t \triangle y + \triangle s = A^t y + s - Qx - c \\ A\triangle x = b - Ax \\ X\triangle s + S\triangle x = -Xs + \mu e \end{cases} \tag{2}$$

dont la solution est : $(\triangle x, \triangle y, \triangle s)$.

Le nouvel itéré est :

$$(x^+, y^+, s^+) = (x, y, s) + \alpha(\triangle x, \triangle y, \triangle s).$$

où $\alpha > 0$ est le pas de déplacement choisi de telle manière que $(x^+, s^+) > 0$ et $\Phi(x^+, y^+, s^+) < \Phi(x, y, s)$.

Maintenant, on donne l'algorithme générique de cette méthode.

2.1.2 Algorithme *(Méthode de la trajectoire centrale non réalisable*)

Algorithme 1

Début algorithme

Soit $\varepsilon > 0$ (un paramètre de précision),

on choisi (x^0, y^0, s^0) tel que : $(x^0, s^0) > 0$.

$k = 0$ (indice des itérations)

While $(\Phi(x^k, y^k, s^k)) > \varepsilon$ **do**

1. $\mu^k = \frac{(x^k)^t s^k}{n}$
2. calculer $(\triangle x, \triangle y, \triangle s)$, solution du système (2)
3. trouver un pas de déplacement α_k tel que $(x^k, s^k) > 0$ et $\Phi(x^{k+1}, y^{k+1}, s^{k+1}) < \Phi(x^k, y^k, s^k)$
4. mettre à jour $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\triangle x^k, \triangle y^k, \triangle s^k)$
5. posé $k = k + 1$.

End While.

Fin algorithme.

2.1.3 Convergence de l'algorithme

Sous les **Hypothèse 1** et **Hypothèse 2** citées dans l'introduction de ce chapitre, la convergence de l'algorithme est donnée par les résultats suivantes

Proposition 10 *[40] Si le pas de déplacement* $0 < \alpha_k \leq 1$*, alors la suite* (Φ^k) *engendrée par l'algorithme satisfait :*

$$\Phi^{k+1} = (1 - \rho^k)\Phi^k$$

où

$$\rho^k = \rho(\alpha_k) = \frac{\left(\alpha_k(1 - \sigma_k)(x^k)^t s^k + \alpha_k v^k r^0 - \alpha_k^2(\triangle x^k)^t \triangle s^k\right)}{(x^k)^t s^k + v^k r^0}$$

Lemme 11 *[40] Soit* (x^k, y^k, s^k) *une suite générée par l'**algorithme 1**, alors :*

1) $A(x^k + \alpha_k \triangle x^k) - b = (1 - \alpha_k)(Ax^k - b) = v^{k+1}(Ax^0 - b)$

2) $A^t(y + \alpha_k \triangle y^k) + (s^k + \alpha_k \triangle s^k) - Q(x^k + \alpha_k \triangle x^k) = v^{k+1}(A^t y^0 + s^0 - Qx^0 - c)$

3) $(x^k + \alpha_k \triangle x^k)(s^k + \alpha_k \triangle s^k) = (x^k)^t s^k (1 - \alpha_k + \alpha_k \sigma_k) + \alpha_k^2 (\triangle x^k)^t \triangle s^k$

où $v^{k+1} = (1 - \alpha_k)v^k = \prod_{i=1}^{k}(1 - \alpha_i)v^0$, *avec* $v^0 = 1$.

La fonction de mérite décroît d'une itération à l'autre d'un montant égal à $(1-\rho^k)$, de plus on a le corollaire suivant :

Corollaire 12 *[40] La suite* (Φ^k) *converge au moins linéairement si* $0 < \alpha_k \leq 1$ *auquel cas nous avons* $0 < \sigma_k < 1$ *et si* σ_k *tend vers* 1 *la convergence est superlinéaire.*

Théorème 13 *[40] Soit* $\varepsilon > 0$ *(un paramètre de précision),* $(x^0, y^0, s^0) = \xi(e, 0, e)$ *est le point initial où* $\xi > 0$, *alors : l'algorithme converge au bout de* $O(n^2 \left|\log(\varepsilon)\right|)$ *itérations.*

2.1.4 Tests Numériques

Les programmes sont réalisés en **DEV C++** avec une précision $\varepsilon = 10^{-6}$, on note par x^* la solution optimale primale, (y^*, s^*) la solution optimale duale, $f(x^*)$ la valeur optimale primale, $f(y^*, s^*)$ la valeur optimale duale et (**iter**) le nombre d'itérations, pour différents types de problème d'optimisation quadratique convexe sous contraintes linéaires.

Exemple 1 :

$$\begin{cases} \min 0.5x_1^2 + 6.5x_1 - x_2 - 2x_3 - 3x_4 - 2x_5 - x_6 \\ x_1 + x_2 + 8x_3 + x_4 + 3x_5 + 5x_6 \leq 26 \\ -8x_1 - 4x_2 - 2x_3 + 2x_4 + 4x_5 - x_6 \leq -11 \\ 2x_1 + 0.5x_2 + 0.2x_3 - 3x_4 - x_5 - 4x_6 \leq 24 \\ 0.2x_1 + 2x_2 + 0.1x_3 - 4x_4 + 2x_5 + 2x_6 \leq 12 \\ -0.1x_1 - 0.5x_2 + 2x_3 + 5x_4 - 5x_5 + 3x_6 \leq 3 \\ 0 \leq x_i \leq 1, \ i = 4, 5 \\ 0 \leq x_6 \leq 2 \end{cases}.$$

Les résultats numériques de cet exemple sont configurés dans le tableau suivant :

x^*	$(0 \quad 7.987342 \quad 0.253165 \quad 2 \quad 2 \quad 0)$
y^*	$(-0.246835 \quad 0 \quad 0 \quad -0.253165 \quad 0)$
$f(x^*)$	-18.493672
$f(y^*, s^*)$	-18.483671
$iter$	12

Exemple 2 :

$$\begin{cases} \min \sum_{i=1}^{9} x_i x_{i+1} + \sum_{i=1}^{8} x_i x_{i+2} + x_1 x_9 + x_1 x_{10} + x_2 x_{10} + x_1 x_5 + x_4 x_7 \\ \sum_{i=1}^{10} x_i = 1, \ x_i \geq 0 \end{cases}$$

Les résultats numériques de cet exemple sont configurés dans le tableau suivant :

x^*	$\begin{pmatrix} 0.000000 & 0.249335 & 0.250000 & 0.000000 & 0.000665 \\ 0.017431 & 0.000000 & 0.250000 & 0.232568 & 0.000000 \end{pmatrix}$
y^*	0.25
s^*	$\begin{pmatrix} 1.333990 & 2.464458 & 3.798448 & 1.333990 & 2.464458 \\ 3.549113 & 3.549113 & 0.000000 & 0.000000 & 0.249335 \end{pmatrix}$
$f(x^*)$	0.125010
$f(y^*, s^*)$	0.125000
$iter$	5

Exemple 3 :

$\min\left\{\frac{1}{2}x^tQx + c^tx : Ax = b, x \geq 0\right\}$,

où

$$Q = \begin{pmatrix} 20 & 1.2 & 0.5 & 0.5 & -1 \\ 1.2 & 32 & 1 & 1 & 1 \\ 0.5 & 1 & 14 & 1 & 1 \\ 0.5 & 1 & 1 & 15 & 1 \\ -1 & 1 & 1 & 1 & 16 \end{pmatrix},\ A = \begin{pmatrix} 1 & 1.2 & 1 & 1.8 & 0 \\ 3 & -1 & 1.5 & -2 & 1 \\ -1 & 2 & -3 & 4 & 2 \end{pmatrix},$$

$$b = \begin{pmatrix} 9.31 \\ 5.45 \\ 6.60 \end{pmatrix},\ c = \begin{pmatrix} 1 & -1.5 & 2 & 1.5 & 3 \end{pmatrix}.$$

Les résultats numériques de cet exemple sont configurés dans le tableau suivant :

x^*	(2.632276 0.701827 1.399507 2.464458 1.084655)
y^*	(25.271033 11.733714 5.257142)
s^*	(55.335041 29.065817 27.160177 42.968998 22.287996)
$f(x^*)$	172.733206
$f(y^*, s^*)$	172.733215
$iter$	28

Exemple 4 :

$\min\left\{\frac{1}{2}x^tQx + c^tx : Ax = b, x \geq 0\right\}$,

où

$$Q = \begin{pmatrix} 30 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 21 & 0 & 1 & -1 & 1 & 0 & 1 & 0.5 & 1 \\ 1 & 0 & 15 & -0.5 & -2 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & -0.5 & 30 & 3 & -1 & 1 & -1 & 0.5 & 1 \\ 1 & -1 & -2 & 3 & 27 & 1 & 0.5 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 16 & -0.5 & 0.5 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0.5 & -0.5 & 8 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 0.5 & 1 & 24 & 1 & 1 \\ 1 & 0.5 & 1 & 0.5 & 1 & 0 & 1 & 1 & 39 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 11 \end{pmatrix},$$

$$A = \begin{pmatrix} 1 & -1 & 1.9 & 1.25 & 1.2 & 0.4 & -0.7 & 1.06 & 1.5 & 1.05 \\ 1.3 & 1.2 & 0.15 & 2.15 & 1.25 & 1.5 & 0.4 & 1.52 & 1.3 & 1 \\ 1.5 & -1.1 & 3.5 & 1.25 & 1.8 & 2 & 1.95 & 1.2 & 1 & -1 \end{pmatrix},$$

$$b = \begin{pmatrix} 11.651 \\ 16.672 \\ 21.295 \end{pmatrix},\ c = \begin{pmatrix} -0.5 & -1 & 0 & 0 & -0.5 & 0 & 0 & -1 & -0.5 & -1 \end{pmatrix}.$$

Les résultats numériques de cet exemple sont configurés dans le tableau suivant :

x^*	$\begin{pmatrix} 0.963886 & 0.509607 & 1.739953 & 1.905056 & 1.243511 \\ 2.626820 & 1.322918 & 1.617087 & 0.824013 & 0.897582 \end{pmatrix}$
y^*	$(4.243380 \quad 22.362785 \quad 5.192083)$
s^*	$\begin{pmatrix} 41.103130 & 16.380672 & 29.589130 & 59.874329 & 42.391300 \\ 45.625690 & 16.599314 & 44.719913 & 40.628777 & 21.626253 \end{pmatrix}$
$f(x^*)$	264.148690
$f(y^*, s^*)$	264.148682
$iter$	27

2.2 Méthode de la trajectoire centrale réalisable pour (CQP)

La programmation non linéaire convexe a des applications importantes dans la programmation mathématique. Plus de vingt ans de nombreux auteurs ont présenté différentes méthodes de points intérieurs pour la programmation non linéaire convexe [56][60][75][84].

Dans ce qui suit, nous proposons un algorithme primal-dual de points intérieurs à court pas (Short step) pour résoudre le problème convexe quadratique. L'algorithme utilise à chaque itération une étape de Newton et une mesure de proximité adaptée pour tracer le chemin central (trajectoire centrale).

Des notations sont utilisées tout au long de cette section et ils sont comme suit. $\Re^n$, $\Re^n_+$ et $\Re^n_{++}$ et qui désignent respectivement l'ensemble des vecteurs de n des composants, l'ensemble des vecteurs non négatifs et l'ensemble de vecteurs positifs. $\Re^{n \times n}$ désigne l'ensemble des $n \times n$ matrices réelles. $\|.\|_2$ désigne la norme euclidienne, $e = (1, 1, ..., 1)^t$ est le vecteur identité de $\Re^n$, $r \in \Re^n_+$ est le paramètre poids. Enfin, pour $x,\ s \in \Re^n$, nous définissons $xs = (x_1 s_1, x_2 s_2, ..., x_n s_n)^t$, $\frac{x}{s} = (\frac{x_1}{s_1}, \frac{x_2}{s_2}, ..., \frac{x_n}{s_n})^t, s^{-1} = (\frac{1}{s_1}, \frac{1}{s_2}, ..., \frac{1}{s_n})^t$ et $\sqrt{x} = (\sqrt{x_1}, \sqrt{x_2}, ..., \sqrt{x_n})^t$.

2.2.1 Déscription et principe de la méthode

Nous considérons le problème quadratique convexe (CQP) et son dual (CQD) sous forme standard

$$\min\left\{ \frac{1}{2}x^tQx + c^tx : Ax = b, x \geq 0 \right\}, \qquad (CQP)$$

$$\max\left\{ b^ty - \frac{1}{2}x^tQx + c^tx : A^ty + s - Qx = c, s \geq 0, y \in \Re^m \right\}, \qquad (CQD)$$

où $A \in \Re^{m\times n}$ est une matrice, $Q \in \Re^{n\times n}$ est une matrice symétrique semi-définie positive, $x, s,\ c \in \Re^n$, et $b \in \Re^m$.

Nous supposons que les problèmes (CQP) et (CQD) satisfait aux conditions de points strictement intérieurs $(IPCs)$ i.e., il existe une solution initiale $(x^0,\ y^0,\ s^0)$ de telle sorte que :

$$Ax^0 = b,\ x^0 > 0,\ A^ty^0 + s^0 = Qx^0 + c,\ s^0 > 0,\ y^0 \in \Re^m.$$

Les conditions d'optimalité pour (CQP) et (CQD) sont données par le système suivant

$$\begin{cases} Ax = b, x \geq 0, \\ A^ty + s - Qx = c, s \geq 0, \\ x^ts = 0. \end{cases} \qquad (3)$$

Si la condition (IPC) détiennent, le $\mu-$central de (CQP) et (CQD) est définie par la solution $(x(\mu),\ y(\mu),\ s(\mu))$ du système suivant

$$\begin{cases} Ax = b, x > 0, \\ A^ty + s - Qx = c, s > 0 \\ xs = \mu e, \end{cases} \qquad (4)$$

avec $\mu > 0$. L'ensemble de $\mu-$central donne un trajet homotopie, que l'on appelle le chemin (trajectoire) centrale du (CQP) et (CQD).

Si $\mu \longrightarrow 0$, la limite de la voie centrale existe depuis le point limite satisfont à la condition de complémentarité, la limite d'un conduit $\varepsilon-$solution approchée pour (CQP) et (CQD).

En appliquant la méthode de Newton pour le système (4), nous determinons les équations suivantes pour les directions de descente $(\triangle x,\ \triangle y,\ \triangle s)$

$$\begin{cases} A(x+\triangle x) = b, \\ A^t(y+\triangle y) + (s+\triangle s) - Q(x+\triangle x) = c, \\ (x+\triangle x)(s+\triangle s) = \mu e - xs. \end{cases} \tag{5}$$

Le système (5) peut s'écrire sous la forme :

$$\begin{cases} A\triangle x = 0, \\ A^t\triangle y + \triangle s - Q\triangle x = 0, \\ s\triangle x + x\triangle s = \mu e - xs. \end{cases} \tag{6}$$

Ce dernièr peut s'écrire sous forme matricielle :

$$\begin{pmatrix} A & 0 & 0 \\ -Q & A^t & I \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \triangle x \\ \triangle y \\ \triangle s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu e - xs \end{pmatrix}, \tag{7}$$

où $X = diag(x),\ S = diag(s),\ I$ est la matrice identité d'ordre n.

Le format dans (7) est adapté à la mise en œuvre numérique pour determiner $(\triangle x, \triangle y, \triangle s)$ et par consequent le nouvel itéré est donné par :

$$x^+ = x + \triangle x,\ y^+ = y + \triangle y,\ s^+ = s + \triangle s.$$

Maintenant, nous introduisons une norme de mesure de proximité définie comme suit :

$$\delta(xs, \mu) = \frac{1}{2} \left\| \sqrt{\left(\frac{xs}{\mu}\right)^{-1}} - \sqrt{\frac{xs}{\mu}} \right\|_2,$$

pour mesurer la proximité de points réalisables pour la trajectoire centrale. Nous utilisons également une valeur seuil $0 < \beta < 1$ et nous supposons que le point initial (x^0, y^0, s^0) est strictement réalisable de telle sorte que $\delta(x^0 s^0, \mu^0) < \beta$ pour $\mu^0 = \frac{(x^0)^t s^0}{n}$.

Dans ce qui suit, on présente l'algorithme primal-dual de la méthode de la trajectoire centrale réalisable pour résoudre un programme quadratique convexe.

2.2.2 Algorithme (Méthode de la trajectoire centrale réalisable)

Algorithm 2

Début Algorithme

$\varepsilon > 0$ (un paramètre de précision),

un paramètre de mise à jour θ, $0 < \theta < 1$

un paramètre de seuil $\beta, 0 < \beta < 1$

un point initial strictement réalisable (x^0, y^0, s^0) tel que $\delta(x^0 s^0, \mu^0) \leq \beta$ et $\mu^0 = \frac{(x^0)^t \ s^0}{n}$.

$k = 0$

While $(n\mu^k) > \varepsilon$ **do**

1. calculer $(\triangle x, \triangle y, \triangle s)$ solution du système (7)
2. mettre à jour $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + (\triangle x, \triangle y, \triangle s)$
3. poser $\mu^{k+1} = (1-\theta)\mu^k = (1-\theta)\frac{x^k s^k}{n}$ et $k = k + 1$.

End While.

Fin Algorithme.

Remarque 14 *Si le point de départ ne vérifié pas la mesure de proximité i.e.,* $\delta(x^0 s^0, \mu^0) > \beta$, *alors il n'y a aucune garantie concernant la convergence de l'algorithme. Pour surmonter cette difficulté, nous introduisons le paramètre de poids pour*

forcer le point de départ dans le voisinage de la trajectoire centrale i.e., $\delta(x^0 s^0, \mu^0) = 0$. *D'où le recours à l'amélioration de cette méthode est necessaire qu'on décrit dans le paragraphe suivant*

2.3 Méthode de la trajectoire centrale réalisable améliorée pour (CQP)

2.3.1 Description de la méthode

On associé à (CQP) le problème perturbé défini comme suit :

$$\left\{ \min f_{\mu r}(x) = \left(f(x) - \mu \sum_{i=1}^{n} r_i \ln x_i \right) : Ax = b, x > 0 \right. \qquad (CQP_{\mu r})$$

$\mu > 0, r = (r_1, r_2, ..., r_n)^t \in \mathbb{R}^n_+$ est le vecteur poids associé à la fonction barrière.

$f_{\mu r}(x)$ est strictement convexe et les contraintes sont linéaires, alors les conditions de KKT correspondantes sont nécessaires et suffisantes et s'écrivent comme suit :

$$\begin{cases} \nabla f(x) - \mu X^{-1} r - A^t y = 0 \\ Ax = b \\ x > 0, y \in \mathbb{R}^m \end{cases}$$

$$\Leftrightarrow \begin{cases} Ax = b \\ A^t y + s - Qx = c \\ x^t s = \mu r, \end{cases} \qquad (8)$$

où $r \in \Re^n_+$ est le paramètre poids, avec $\mu > 0$. L'ensemble de $\mu-$central donne un trajet homotopie, que l'on appelle la trajectoire centrale du (CQP) et (CQD).

Si $\mu \longrightarrow 0$, la limite de la voie centrale existent depuis le point limite satisfont à la condition de complémentarité, la limite d'un conduit $\varepsilon-$solution approchée pour (CQP) et (CQD).

En appliquant la méthode de Newton pour le système (8), nous determinons les équations suivantes pour les directions de descente $(\triangle x,\ \triangle y,\ \triangle s)$

$$\begin{cases} A(x+\triangle x) = b \\ A^t(y+\triangle y) + (s+\triangle s) - Q(x+\triangle x) = 0 \\ (x+\triangle x)(s+\triangle s) = \mu r - xs. \end{cases} \tag{9}$$

Le système (9) peut s'écrire sous la forme :

$$\begin{cases} A\triangle x = 0, \\ A^t\triangle y + \triangle s - Q\triangle x = 0, \\ s\triangle x + x\triangle s = \mu r - xs. \end{cases} \tag{10}$$

Ce dernier système peut aussi s'écrire comme suit :

$$\begin{pmatrix} A & 0 & 0 \\ -Q & A^t & I \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \triangle x \\ \triangle y \\ \triangle s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu r - xs \end{pmatrix}, \tag{11}$$

où $X = diag(x),\ S = diag(s),\ I$ est la matrice d'identité d'ordre n. Le format dans (11) est adapté à la mise en œuvre numérique. D'où le nouvel itéré est donné par :

$$x^+ = x + \triangle x,\ y^+ = y + \triangle y,\ s^+ = s + \triangle s,$$

et la condition de proximité est donnée comme suit :

$$\delta(xs, \mu) = \frac{1}{2} \left\| \sqrt{\left(\frac{xs}{\mu r}\right)^{-1}} - \sqrt{\frac{xs}{\mu r}} \right\|_2 .$$

Maintenant, on présente l'algorithme primal-dual de la méthode de la trajectoire centrale amélioré pour résoudre un programme quadratique convexe .

2.3.2 Algorithme (Méthode de la trajectoire centrale réalisable améliorée)

Algorithm 3.

Début Algorithme

$\varepsilon > 0$ (un paramètre de précision),

un paramètre de mise à jour θ, $0 < \theta < 1$

un paramètre de seuil $\beta, 0 < \beta < 1$

un point initial strictement réalisable (x^0, y^0, s^0) et $\mu^0 = \frac{(x^0)^t\ Rs^0}{n}$.
$\sigma = \dfrac{\|X^0 S^0 e\|}{\sqrt{n}}$, $r = \dfrac{X^0 S^0 e}{\sigma}$ (le vecteur poids). $R = diag(r_i)$
$k = 0$

While $(n\mu^k) > \varepsilon$ **do**

1. calculer $(\triangle x, \triangle y, \triangle s)$ solution de système (11)
2. mettre à jour $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + (\triangle x, \triangle y, \triangle s)$
3. posons $\mu^{k+1} = (1-\theta)\mu^k = (1-\theta)\frac{x^k R s^k}{n}$ et $k = k+1$.

End While.

Fin algorithme.

Dans la section suivante, on présente l'analyse de la complexité de cet algorithme. Lorsque $r = e$, on obtient la méthode classique (**algorithme 2**).

2.3.3 Etude de la convergence

Nous allons définir

$$d = \sqrt{\frac{x}{s}}, v = \sqrt{\frac{xs}{\mu r}} = \frac{d^{-1}x}{\sqrt{\mu r}} = \frac{ds}{\sqrt{\mu r}},$$

ainsi que pour la direction d'origine $\triangle x$ et $\triangle s$, nous définissons

$$d_x = \frac{d^{-1}\triangle x}{\sqrt{\mu r}} \text{ et } d_s = \frac{d\triangle s}{\sqrt{\mu r}}, \qquad (*)$$

nous obtenons

$$s\triangle x + x\triangle s = \mu r v(d_x + d_s)$$

et

$$\triangle x \triangle s = \mu r d_x d_s$$

Nous définissons aussi

$$\overline{A} = \sqrt{\mu}AD,\ \overline{Q}\ = \sqrt{\mu}DQD,\ \text{où } D = diag(d)$$

Le système (11) s'écrit comme suit

$$\begin{cases} \overline{A}d_x = 0 \\ \overline{A}^t \triangle y + d_s - \overline{Q}d_x = 0 \\ d_x + d_s = p_v \end{cases} \tag{12}$$

où

$$p_v = v^{-1} - v.$$

Pour l'analyse de notre algorithme, nous définissons la mesure de proximité $\delta(xs,\ \mu)$ comme suit :

$$\delta(v) = \delta(xs,\ \mu) = \frac{\|p_v\|}{2} = \frac{1}{2}\left\|v^{-1} - v\right\|_2.$$

En raison des deux premières équations du système (12), les direction d_x et d_s sont orthogonaux. Ainsi

$$d_x^t d_s = d_s^t d_x = 0,$$

et on peut facilement vérifier que

$$\delta(v) = 0 \Longleftrightarrow v = v^{-1} \Longleftrightarrow d_x = d_s = 0 \Longleftrightarrow xs = \mu r.$$

Ainsi, la valeur de $\delta(v)$ peut être considérée comme un mesure de la distance entre la paire donnée $(x,\ y,\ s)$ et μ-centre $(x(\mu),\ y(\mu),\ s(\mu))$. Les nouvelles directions de descente d_x et d_s sont obtenues en résolvant le système (12) avec $p_v = \frac{1}{2}(v^{-1} - v)$. Puis les directions de descente $\triangle x$ et $\triangle s$ sont calculées par $(*)$.

Lemme 15 *[65] Soit (d_x, d_s) la solution du système* (12) *et si la mésure de proximité $\delta = \delta(xs, \mu) < \beta$ et $\mu > 0$. Alors, on a*

$$0 < d_x^t d_s \leq 2\delta^2 \tag{13}$$

et

$$\|d_x d_s\|_\infty < \delta^2 \ et \ \|d_x d_s\|_2 \leq \sqrt{2}\delta^2 \tag{14}$$

Lemme 16 *Soit* (x, s) *une itération primal-dual strictement réalisable.*

Si $e + d_x d_s > 0$, *alors* $x^+ = x + \triangle x,\ s^+ = s + \triangle s$.

Preuve. Soit $0 < \alpha \leq 1$ est le pas de déplacement.

nous définissons

$$x(\alpha) = x + \alpha\triangle x,\ s(\alpha) = s + \alpha\triangle s,$$

on a

$$\begin{aligned} x(\alpha)s(\alpha) &= (x + \alpha\triangle x)(s + \alpha\triangle s) \\ &= xs + \alpha(x\triangle s + s\triangle x) + \alpha^2\triangle x\triangle s \\ &= xs + \alpha(\mu r - xs) + \alpha^2\triangle x\triangle s. \end{aligned}$$

Nous supposons que $e+d_x d_s > 0$, on déduit que $\mu r+\triangle x\triangle s > 0$, ce qui correspond à $\triangle x\triangle s > -\mu r$, par substitution, on obtient

$$\begin{aligned} x(\alpha)s(\alpha) &> xs + \alpha(\mu r - xs) - \alpha^2\mu r \\ &= (1-\alpha)xs + (\alpha - \alpha^2)\mu r \\ &= (1-\alpha)xs + \alpha(1-\alpha)\mu r. \end{aligned}$$

Puisque, $xs > 0$ et $\mu r > 0$, il s'ensuit que $x(\alpha)s(\alpha) > 0$ pour tous $\alpha \in]0, 1]$.

Ainsi, aucune des entrées de $x(\alpha)$ et $s(\alpha)$ disparaître pour $\alpha \in]0, 1]$.Comme x^0 et s^0 sont positifs, ce qui implique que $x(\alpha) > 0$ et $s(\alpha) > 0$ pour $\alpha \in]0, 1]$. Ainsi, par la continuité des vecteurs x^1 et s^1 doit être positif qui prouve que x^+ et s^+ sont positifs. Ceci termine la preuve. ■

Maintenant, pour plus de commodité, nous pouvons écrire

$$(v^+)^2 = \frac{x^+ s^+}{\mu r} = e + d_x d_s.$$

Lemme 17 *Si la mesure de proximité* $\delta(v) = \delta(xs, \mu) < 1$, *alors* $x^+ > 0$ *et* $s^+ > 0$.

Preuve. Dans le lemme (16), on a (x^+, s^+) sont strictement réalisable si

$$(e + d_x d_s) > 0.$$

$(e + d_x d_s) > 0$ est vrai si $(1 + (d_x d_s)_i) > 0$ pour $i \in \Re^n$.

On a

$$\begin{aligned}(1 + (d_x d_s)_i) &\geq (1 - |(d_x d_s)_i|, \text{ pour } i \in \Re^n \\ &\geq (1 - \delta^2).\end{aligned}$$

Ainsi

$$(e + d_x d_s) > 0 \text{ si } \delta(xs, \mu) < 1.$$

Ceci termine la preuve. ■

Dans le lemme suivant, on va prouver la convergence quadratique pour notre algorithme

Lemme 18 *Soit la mesure de proximité* $\delta = \delta(xs, \mu)$. *Si* $\delta(xs, \mu) < 1$ *alors*

$$\delta(x^+, s^+, \mu^+) \leq \frac{\delta^2}{\sqrt{2(1 - \delta^2)}}.$$

Preuve. On suppose que $\alpha = 1$,

on a

$$\begin{aligned}4\delta_+^2 &= \left\|(v^+)^{-1} - v^+\right\|^2 \\ &= \left\|(v^+)^{-1}(e - (v^+)^2)\right\|^2,\end{aligned}$$

où $(v^+)^2 = (e + d_x d_s)$ et $(v^+)^{-1} = \frac{1}{\sqrt{(e+d_x d_s)}}$, il s'ensuit que

$$\begin{aligned}4\delta_+^2 &= \left\|\frac{d_x d_s}{\sqrt{(e + d_x d_s)}}\right\|_2^2 \\ &= \left\|\frac{d_x d_s}{\sqrt{(e + d_x d_s)}}\right\|_2^2 \\ &\leq \frac{\|d_x d_s\|_2^2}{(1 - \|d_x d_s\|_\infty)}.\end{aligned}$$

On déduire

$$4\delta_+^2 \leq \frac{2\delta^4}{(1-\delta^2)},$$

d'où,

$$\delta_+ \leq \frac{\delta^2}{\sqrt{2(1-\delta^2)}}.$$

D'où le résultat. ■

Dans le lemme suivant, nous discutons de l'influence sur la mesure de proximité d'un paramètre de barrière $\mu^+ = (1-\theta)\mu$ pendant le processus de Newton le long de la trajectoire centrale.

Lemme 19 *Soit* $\delta(xs,\mu) < \frac{1}{\sqrt{2}}$ *et* $\mu^+ = (1-\theta)\mu,\ 0 < \theta < 1$. *Alors*

$$\delta^2(x^+s^+,\mu^+) \leq (1-\theta)\delta_+^2 + \frac{\theta^2(n+1)}{4(1-\theta)} + \frac{\theta}{2}.$$

En outre, si $\delta \leq \frac{1}{\sqrt{2}},\ \theta = \frac{1}{2\sqrt{n}}$ *et* $n \geq 2$, *alors, on a* $\delta(x^+s^+,\mu^+) \leq \frac{1}{\sqrt{2}}$.

Preuve. Soit $v^+ = \sqrt{\frac{x^+s^+}{\mu^+r}}$ et $\mu^+ = (1-\theta)\mu$, alors

$$\begin{aligned}
4\delta^2(x^+s^+,\mu^+) &= \left\| \sqrt{\frac{\mu^+r}{x^+s^+}} - \sqrt{\frac{x^+s^+}{\mu^+r}} \right\|_2^2 \\
&= \left\| \sqrt{1-\theta}(v^+)^{-1} - \frac{1}{\sqrt{1-\theta}} v^+ \right\|_2^2 \\
&= \left\| \sqrt{1-\theta}\left((v^+)^{-1} - v^+\right) - \frac{\theta}{\sqrt{1-\theta}} v^+ \right\|_2^2 \\
&= (1-\theta)\left\|(v^+)^{-1} - v^+\right\|_2^2 + \frac{\theta^2}{1-\theta}\left\|v^+\right\|_2^2 - 2\theta\left((v^+)^{-1} - v^+\right)^t v^+ \\
&= (1-\theta)\left\|(v^+)^{-1} - v^+\right\|_2^2 + \frac{\theta^2}{1-\theta}\left\|v^+\right\|_2^2 - 2\theta(v^+)^{-t}v^+ + v^{+^t}v^+ \\
&= 4(1-\theta)\delta_+^2 + \frac{\theta^2}{1-\theta}\left\|v^+\right\|_2^2 - 2\theta n + 2\theta\left\|v^+\right\|_2^2
\end{aligned}$$

puisque,

$(v^+)^{-t}v^+ = n$ et $(v^+)^t v^+ = \|v^+\|_2^2$, et on a $\delta_+^2 \leq \frac{\delta_+^4}{2(1-\delta_+^2)}$.Alors

$$4\delta^2(x^+s^+,\mu^+) \leq 4(1-\theta)\frac{\delta^4}{2(1-\delta_+^2)} + \frac{\theta^2}{1-\theta}\left\|v^+\right\|_2^2 - 2\theta n + 2\theta\left\|v^+\right\|_2^2,$$

comme,

$$x^+s^+ = \mu r(e + d_x d_s),$$

d'après le lemme (19), on a

$$\left\|v^+\right\|^2 = \frac{x^+s^+}{\mu r} = e + d_x d_s \leq 1 + n.$$

En conséquence,

$$4\delta^2(x^+s^+, \mu^+) \leq 4(1-\theta)\frac{\delta^4}{2(1-\delta_+^2)} + \frac{\theta^2}{1-\theta}\left\|v^+\right\|_2^2 - 2\theta n + 2\theta(n+1),$$

et

$$\delta^2(x^+s^+, \mu^+) \leq (1-\theta)\delta_+^4 + \frac{\theta^2(n+1)}{4(1-\theta)} + \frac{\theta}{2}.$$

La dernière déclaration de la preuve est la suivante. Si $\delta < \frac{1}{\sqrt{2}}$, alors $\delta_+^2 = \frac{1}{4}$ et cela donne la partie supérieure suivante à destination de $\delta(x^+s^+, \mu^+)$:

$$\delta^2(x^+s^+, \mu^+) \leq \frac{(1-\theta)}{4} + \frac{\theta^2(n+1)}{4(1-\theta)} + \frac{\theta}{2}.$$

Maintenant, en tenant $\theta = \frac{1}{2\sqrt{n}}$, alors $\theta^2 = \frac{1}{4n}$ il s'ensuit que

$$\delta^2(x^+s^+, \mu^+) \leq \frac{\frac{(n+1)}{4n}}{4(1-\theta)} + \frac{(1-\theta)}{4} + \frac{\theta}{2},$$

comme $\frac{(n+1)}{4n} \leq \frac{3}{8}$ pour $n \geq 2$, alors, on a

$$\delta^2(x^+s^+, \mu^+) \leq \frac{3}{32(1-\theta)} + \frac{\theta+1}{4}.$$

Pour $n \geq 2$, on a $0 < \theta \leq \frac{1}{2\sqrt{2}}$ et la fonction $f(\theta) = \frac{3}{32(1-\theta)} + \frac{\theta+1}{4}$ est continue et monotone croissante sur $0 < \theta \leq \frac{1}{2\sqrt{2}}$,

par conséquent,$f(\theta) \leq f(\frac{1}{2\sqrt{2}}) < \frac{1}{2}$, pour $0 < \theta \leq \frac{1}{2\sqrt{2}}$.

On obtient

$$\delta(x^+s^+, \mu^+) < \frac{1}{\sqrt{2}}.$$

d'où le résultat. ■

Le théorème suivant donne une borne supérieure pour le nombre total d'itérations pour notre algorithme

Théorème 20 *Soit* $(x^0,\ y^0,\ s^0)$ *une solution initiale strictement réalisable,* $\mu^0 = \frac{(x^0)^t R s^0}{n}$ *et* $\delta(x^0, s^0, \mu^0) \leq \frac{1}{\sqrt{2}}$. *Par ailleurs, soit* x^k *et* s^k *les vecteurs obtenus après* k *itération, alors*$(x^k)^t s^k \leq \varepsilon$ *est vérifiée pour tout* $k \geq \frac{1}{\theta}\left[\log(\frac{(x^0)^t R s^0 + \mu^0}{\varepsilon})\right]$.

Preuve. On a

$$(x^{k+1})^t s^{k+1} \leq \mu^k (n+1),$$

avec,

$$\mu^k = (1-\theta)\mu^{k-1} = (1-\theta)^k \mu^0,$$

alors,

$$(x^{k+1})^t s^{k+1} \leq (1-\theta)^k \mu^0 (n+1),$$

ainsi,

$$(1-\theta)^k \mu^0 (n+1) \leq \varepsilon.$$

Maintenant, en prenant logarithmes de $(1-\theta)^k \mu^0 (n+1) \leq \varepsilon$, nous pouvons écrire

$$\log((1-\theta)^k \mu^0 (n+1)) \leq \log \varepsilon$$

équivalent

$$k \log(1-\theta) \leq \log \varepsilon - \log \mu^0 (n+1),$$

on sait que $\log(1-\theta) \leq \theta$, pour $0 \leq \theta \leq 1$, alors l'inégalité ci-dessus est vérifiée si

$$k \geq \frac{1}{\theta}\left[\log(\frac{(x^0)^t R s^0 + \mu^0}{\varepsilon})\right].$$

Ceci termine la preuve. ■

2.3.4 Tests Numériques

Dans cette section, nous traitons l'**algorithme 2** et l'**algorithme 3** appliqué à certains problèmes convexes quadratiques. Ici, nous avons utilisé (x^0, y^0, s^0) pour désigner la stricte solution réalisable initiale de l'algorithme (voir **Annexe**), tels que $\delta(x^0, s^0, \mu^0) \leq \sqrt{2}$. **Iter** désigne le nombre d'itérations produit par l'algorithme pour

obtenir une solution optimale. $f(x^*)$ désigne la valeur optimale primale de la fonction objectif à x^*, $f(y^*, s^{*)})$ désigne la valeur optimale duale de la fonction objectif à (y^*, s^*) et μ^* désigne la valeur de controle de l'arrêt de l'algorithme. le programme est manipulé en **DEV C++**. Notre tolérance (précision) est $\varepsilon = 10^{-6}$. Pour le paramètre de mise à jour, nous avons d'abord utilisé la stratégie théorique de θ, $0 < \theta < 1$.

Exemple 1.

$$\min\left\{\frac{1}{2}x^tQx + c^tx : Ax = b, x \geq 0\right\},$$

où

$$Q = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix},\ A = \begin{pmatrix} -1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix},\ b = \begin{pmatrix} 1 \\ 2 \end{pmatrix},$$

$$c = \begin{pmatrix} -2 & -4 & 0 \end{pmatrix}.$$

La solution initiale primale strictement réalisable est :

$x^0 = (0.193812 \quad 1.193810 \quad 0.612381)$,

nombre d'itérations pour calculer la solution initiale primale : 8 itérations.

La solution initiale duale strictement réalisable est :

$y^0 = (-0.000003 \quad -1.664125)$,

$s^0 = (0.051764 \quad 0.051764 \quad 1.664132)$,

nombre d'itérations pour calculer la solution initiale duale : 17 itérations.

Les résultats numériques de cet exemple sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0 s^0, \mu^0) = 4.228485 > \sqrt{2}$	$\delta(x^0 s^0, \mu^0) = 0.000000 < \sqrt{2}$				
Diverge	θ	**Iter**	$f(x^*)$	$f(y^*, s^*)$	μ^*
	0.15	69	-4.499988	-4.499966	$9 \times 10e-06$
	0.20	50	-4.499986	-4.499965	$9 \times 10e-06$
	0.25	39	-4.499987	-4.499967	$9 \times 10e-06$
	0.30	32	-4.499990	-4.499967	$8 \times 10e-06$
	0.35	26	-4.499985	-4.499967	$10 \times 10e-06$
	0.45	19	-4.499987	-4.499968	$9 \times 10e-06$
	0.65	12	-4.499996	-4.499966	$4 \times 10e-06$
	0.70	10	-4.499989	-4.499966	$8 \times 10e-06$
	0.95	5	-4.499994	-4.499966	$5 \times 10e-06$

Exemple 2.

$$\min\left\{\frac{1}{2}x^t Q x + c^t x : Ax = b, x \geq 0\right\},$$

où

$$Q = \begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},\ A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 5 & 0 & 1 \end{pmatrix},\ b = \begin{pmatrix} 2 \\ 5 \end{pmatrix},$$

$$c = \begin{pmatrix} -4 & -6 & 0 & 0 \end{pmatrix}.$$

La solution initiale primale strictement réalisable est :

$x^0 = (0.605512 \quad 0.835697 \quad 0.558794 \quad 0.216010)$,

nombre d'itérations pour calculer la solution initiale primale : 7 itérations.

La solution initiale duale strictement réalisable est :

$y^0 = (-3.036075 \quad -0.260399)$,

$s^0 = (0.047168 \quad 0.469876 \quad 3.036084 \quad 0.260408)$,

nombre d'itérations pour calculer la solution initiale duale : 16 itérations.

Les résultats numériques de cet exemple sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0 s^0, \mu^0) = 4.228485 > \sqrt{2}$	$\delta(x^0 s^0, \mu^0) = 0.000000 < \sqrt{2}$				
	θ	**Iter**	$f(x^*)$	$f(y^*, s^*)$	μ^*
	0.15	72	-7.161293	-7.161268	$9 \times 10e-06$
	0.20	53	-7.161294	-7.161268	$8 \times 10e-06$
	0.25	41	-7.161293	-7.161268	$9 \times 10e-06$
	0.30	34	-7.161294	-7.161265	$7 \times 10e-06$
Diverge	0.35	28	-7.161293	-7.161270	$9 \times 10e-06$
	0.45	21	-7.161294	-7.161265	$7 \times 10e-06$
	0.65	13	-7.161294	-7.161260	$5 \times 10e-06$
	0.70	12	-7.161293	-7.161258	$3 \times 10e-06$
	0.95	7	-7.161294	-7.161252	$1 \times 10e-06$

Exemple 3.

$$\min\left\{\frac{1}{2}x^t Q x + c^t x : Ax = b, x \geq 0\right\},$$

où

$$Q = \begin{pmatrix} 20 & 1.2 & 0.5 & 0.5 & -1 \\ 1.2 & 32 & 1 & 1 & 1 \\ 0.5 & 1 & 14 & 1 & 1 \\ 0.5 & 1 & 1 & 15 & 1 \\ -1 & 1 & 1 & 1 & 16 \end{pmatrix}, \; A = \begin{pmatrix} 1 & 1.2 & 1 & 1.8 & 0 \\ 3 & -1 & 1.5 & -2 & 1 \\ -1 & 2 & -3 & 4 & 2 \end{pmatrix},$$

$$b = \begin{pmatrix} 9.31 \\ 5.45 \\ 6.60 \end{pmatrix},\ c = \begin{pmatrix} 1 & -1.5 & 2 & 1.5 & 3 \end{pmatrix}.$$

La solution initiale primale strictement réalisable est :

$x^0 = (2.797350 \quad 1.242208 \quad 1.149637 \quad 2.151306 \quad 0.878306)$,

nombre d'itérations pour calculer la solution initiale primale : 7 itérations.

La solution initiale duale strictement réalisable est :

$y^0 = (21.4412529.2574244.227999)$,

$s^0 = (14.224366\ 20.858749 \quad 1.121712 \quad 1.446803 \quad 1.085264)$,

nombre d'itérations pour calculer la solution initiale duale : 17 itérations.

Les résultats numériques avec ce problème sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0 s^0, \mu^0) = 4.228485 > \sqrt{2}$	$\delta(x^0 s^0, \mu^0) = 0.000000 < \sqrt{2}$				
Diverge	θ	**Iter**	$f(x^*)$	$f(y^*, s^*)$	μ^*
	0.15	90	172.732666	172.732544	$10 \times 10e-06$
	0.20	66	172.732773	172.732483	$9 \times 10e-06$
	0.25	51	172.732727	172.732468	$10 \times 10e-06$
	0.30	42	172.732697	172.732574	$7 \times 10e-06$
	0.35	35	172.732712	172.732483	$7 \times 10e-06$
	0.45	25	172.732742	172.732513	$8 \times 10e-06$
	0.65	15	172.732727	172.732498	$4 \times 10e-06$
	0.70	13	172.732697	172.732513	$5 \times 10e-06$
	0.95	6	172.732727	172.732529	$4 \times 10e-06$

Exemple 4.

$$\min\left\{\frac{1}{2}x^t Q x + c^t x : Ax = b, x \geq 0\right\},$$

où

$$Q = \begin{pmatrix} 30 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 21 & 0 & 1 & -1 & 1 & 0 & 1 & 0.5 & 1 \\ 1 & 0 & 15 & -0.5 & -2 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & -0.5 & 30 & 3 & -1 & 1 & -1 & 0.5 & 1 \\ 1 & -1 & -2 & 3 & 27 & 1 & 0.5 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 16 & -0.5 & 0.5 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0.5 & -0.5 & 8 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 0.5 & 1 & 24 & 1 & 1 \\ 1 & 0.5 & 1 & 0.5 & 1 & 0 & 1 & 1 & 39 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 11 \end{pmatrix},$$

$$A = \begin{pmatrix} 1 & -1 & 1.9 & 1.25 & 1.2 & 0.4 & -0.7 & 1.06 & 1.5 & 1.05 \\ 1.3 & 1.2 & 0.15 & 2.15 & 1.25 & 1.5 & 0.4 & 1.52 & 1.3 & 1 \\ 1.5 & -1.1 & 3.5 & 1.25 & 1.8 & 2 & 1.95 & 1.2 & 1 & -1 \end{pmatrix},$$

$$b = \begin{pmatrix} 11.651 \\ 16.672 \\ 21.295 \end{pmatrix}, \; c = \begin{pmatrix} -0.5 & -1 & 0 & 0 & -0.5 & 0 & 0 & -1 & -0.5 & -1 \end{pmatrix}.$$

La solution initiale primale strictement réalisable est :

$$x^0 = \begin{pmatrix} 1.475426 & 0.868316 & 1.752137 & 1.637813 & 1.560784 \\ 1.926482 & 1.538978 & 1.416004 & 1.223422 & 0.728767 \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale primale : 6 itérations.

La solution initiale duale strictement réalisable est :

$y^0 = (5.656631 \; 16.708544 \; 3.673377)$,

$$s^0 = \begin{pmatrix} 23.525881 & 12.616838 & 2.999578 & 7.234838 & 15.434731 \\ 0.836241 & 9.222806 & 5.6565592 & 3.057934 & 1.441667 \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale duale : 16 itérations.

Les résultats numériques avec ce problème sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0s^0,\mu^0)=3.484792>\sqrt{2}$	$\delta(x^0s^0,\mu^0)=0.000000<\sqrt{2}$				
Diverge	θ	**Iter**	$f(x^*)$	$f(y^*,s^*)$	μ^*
	0.15	89	264.542755	264.543365	$9\times 10e-06$
	0.20	65	264.542755	264.543549	$9\times 10e-06$
	0.25	50	264.542725	264.543518	$10\times 10e-06$
	0.30	41	264.542694	264.543488	$8\times 10e-06$
	0.35	34	264.542786	264.543518	$8\times 10e-06$
	0.45	25	264.542725	264.543518	$6\times 10e-06$
	0.65	14	264.542755	264.543549	$9\times 10e-06$
	0.70	13	264.542725	264.543579	$4\times 10e-06$
	0.95	6	264.542725	264.543640	$1\times 10e-06$

Exemple 5.

$$\min\left\{\frac{1}{2}x^tQx+c^tx : Ax=b, x\geq 0\right\},$$

où

$$Q = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$A = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 2 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \ b = \begin{pmatrix} 5 \\ 4 \\ 1.5 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$c = \begin{pmatrix} -1 & -3 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

La solution initiale primale strictement réalisable est :

$$x^0 = \begin{pmatrix} 0.130437 \ \ 1.421849 \ \ 0.709742 \ \ 0.000010 \ \ 1.316122 \ \ 0.055277 \\ 0.082906 \ \ 0.130437 \ \ 1.421849 \ \ 0.709742 \ \ 0.000010 \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale primale : 15 itérations.

La solution initiale duale strictement réalisable est :

$$y^0 = \begin{pmatrix} -0.222487 \quad -0.716910 \quad -0.016919 \quad -0.098474 \quad -0.072830 \\ -0.394631 \quad -0.084645 \end{pmatrix},$$

$$s^0 = \begin{pmatrix} 0.113781 & 0.080468 & 3.031929 & 0.733798 & 0.222491 & 0.716914 \\ 0.016924 & 0.098479 & 0.072834 & 0.394635 & 0.084649 & \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale duale : 25 itérations.

Les résultats numériques avec ce problème sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0 s^0, \mu^0) = 381.432709 > \sqrt{2}$	$\delta(x^0 s^0, \mu^0) = 0.000000 < \sqrt{2}$				
	θ	**Iter**	$f(x^*)$	$f(y^*, s^*)$	μ^*
	0.15	76	-4.155189	-4.155192	$9 \times 10e-06$
	0.20	56	-4.155189	-4.155187	$8 \times 10e-06$
	0.25	43	-4.155189	-4.155196	$10 \times 10e-06$
	0.30	35	-4.155189	-4.155194	$9 \times 10e-06$
Diverge	0.35	29	-4.155189	-4.155197	$10 \times 10e-06$
	0.45	22	-4.155188	-4.155176	$6 \times 10e-06$
	0.65	13	-4.155189	-4.155178	$6 \times 10e-06$
	0.70	12	-4.155189	-4.155164	$3 \times 10e-06$
	0.90	7	-4.155189	-4.155174	$6 \times 10e-06$

Exemple 6.

$$\min \{f(x) : Ax \leq b, x \geq 0\},$$

où $f(x) = 0.5x_1^2 + 6.5x_1 - x_2 - 2x_3 - 3x_4 - 2x_5 - x_6$,

$$A = \begin{pmatrix} 1 & 2 & 8 & 1 & 3 & 5 \\ -8 & -4 & -2 & 2 & 4 & -1 \\ 2 & 0.5 & 0.2 & -3 & -1 & -4 \\ 0.2 & 2 & 0.1 & -4 & 2 & 2 \\ -0.1 & -0.5 & 2 & 5 & -5 & 3 \end{pmatrix}, \; b = \begin{pmatrix} 26 \\ -11 \\ 24 \\ 12 \\ 3 \end{pmatrix},$$

La solution initiale primale strictement réalisable est :

$$x^0 = \begin{pmatrix} 0.298856 & 1.671936 & 2.310770 & 0.104127 & 0.669299 & 0.185316 \\ 0.832510 & 0.7839842 & 3.827036 & 7.072518 & 1.514232 & \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale primale : 11 itérations.

La solution initiale duale strictement réalisable est :

$$y^0 = (-0.590339 \quad -0.084560 \quad -0.010448 \quad -0.092916 \quad -0.049575),$$

$$s^0 = \begin{pmatrix} 6.747187 & 0.008723 & 7.389776 & 0.019236 & 0.036791 & 2.159913 \\ 0.590347 & 0.000008 & 0.010457 & 0.092924 & 0.049583 & \end{pmatrix},$$

nombre d'itérations pour calculer la solution initiale duale : 28 itérations.

Les résultats numériques avec ce problème sont résumés dans le tableau ci-dessous :

Algorithme 2.	**Algorithme 3.**				
$\delta(x^0 s^0, \mu^0) = 34.102821 > \sqrt{2}$	$\delta(x^0 s^0, \mu^0) = 0.000000 < \sqrt{2}$				
Diverge	θ	**Iter**	$f(x^*)$	$f(y^*, s^*)$	μ^*
	0.15	82	-17.695480	-17.694429	$9 \times 10e-06$
	0.20	60	-17.694048	-17.694429	$9 \times 10e-06$
	0.25	47	-17.693722	-17.694424	$8 \times 10e-06$
	0.30	38	-17.69411	-17.694429	$8 \times 10e-06$
	0.35	31	-17.695181	-17.694433	$10 \times 10e-06$
	0.45	23	-17.694525	-17.694433	$7 \times 10e-06$
	0.65	13	-17.692478	-17.694433	$10 \times 10e-06$
	0.70	12	-17.695179	-17.694433	$4 \times 10e-06$

2.3.4.1 Conclusion et commentaires

Les méthodes de points intérieurs de type Newtonienne sont connues par leurs simplicités algorithmiques, de complexité théorique polynomiale.

L'inconvénient majeur de ce type des algorithmes est l'initialisation c'est-à-dire, comment choisir le point de départ.

Le premier algorithme (**Algorithme 1**) le choix de ce point est arbitraire où le point initial ne vérifie pas la condition de faisabilité (contraintes), les résultats obtenus sont très satisfaisants et encourageants. Pour l'**algorithme 2** et l'**algorithme 3** le point initial calculer vérifie la condition de faisabilité (contraintes), mais l'algorithme 2 est divergent c'est-à-dire le point initial ne vérifie pas la condtion de proximité. Pour assurer la convergence de cet algorithme nous introduisons le paramètre de poids et en utilisant une stratégie pratique, ainsi on obtient un bon comportement théorique et numérique de ce type de méthode présenté dans l'algorithme 3.

Chapitre III

ALGORITHMES DE RÉSOLUTION D'UN PROBLÈME DE COMPLÉMENTARITÉ LINÉAIRE (LCP)

Dans ce chapitre, nous présentons deux types d'algorithmes de points intérieurs pour résoudre un problème de compléméntarité linéaire (LCP). Le premièr est une description d'une méthode projective, qui est une extension de la méthode de Karmarkar appliquée à la programmation linéaire simplifieé. Le deuxième algorithme est une méthode Newtonienne, où nous étudions l'analyse de la complexité polynomiale et la mise en œuvre d'un algorithme numérique pour petit pas de points intérieurs basé sur la fonction du noyau. Notons aussi que notre analyse est basée sur une nouvelle classe de directions de descente.

3.1 Méthode de Karmarkar

En 1984 Karmarkar [43] propose un algorithme de résolution d'un programme linéaire de la forme :

$$w^* = \min \left\{ c^t x : Ax = 0, x \in S_n \right\} \qquad (KP)$$

pour lequel on connait à priori la valeur optimale $w^* = 0$ et une solution réalisable $a = \frac{e_n}{n}$ (centre du simplexe S_n).

A est une matrice de $\Re^{m \times n}$ et de rang plein.

$S_n = \left\{ x \in \Re^n : x \geq 0, \sum_{i=1}^{n} x_i = 1 \right\}$ est le simplexe de dimension $(n-1)$.

On suppose que :

(**Hypothèse 1**) La matrice A est de plein rang ($rgA = m < n$).

(**Hypothèse 2**) On dispose d'un point x^0 strictement réalisable ($Ax^0 = b,\ x^0 > 0$)

(**Hypothèse 3**) La valeur optimale est connue au départ.

3.1.1 Principe de la méthode

A partir d'une solution initiale $x^0 = a$, l'algorithme construit une suite de points intérieurs qui converge vers la solution optimale du problème. Dans le but de ramener la valeur objectif à zéro, on le minimise localement sur une sphère inscrite dans l'orthant réalisable.

Pour chaque itération k, l'itéré $x^k > 0$ est ramené au centre de simplexe S_n par la transformation suivante :

$$\begin{aligned} T_k \quad &: \quad S_n \to S_n \\ x \quad &\mapsto \quad T_k(x) = y \end{aligned}$$

où

$$T_k(x) = \frac{D_k^{-1}x}{e_n^t D_k^{-1}x} = y \text{ et } T_k^{-1}(y) = \frac{D_k y}{e_n^t D_k y}, \text{ avec } D_k = diag(x^k).$$

Le problème (KP) s'écrit comme suit :

$$\min\left\{\frac{c^t D_k y}{e_n^t D_k y} : \frac{AD_k y}{e_n^t D_k y} = 0, \sum_{i=1}^{n} y_i = 1, y > 0\right\} \tag{15}$$

Les hypothèses de départ permettent d'obtenir le programme de Karmarkar simplifié :

$$\min\left\{c^t D_k y : AD_k y = 0, \sum_{i=1}^{n} y_i = 1, y > 0\right\}. \tag{16}$$

Pour la résolution du problème (16), on a le Lemme suivant :

Lemme 21 *[43] Si pour un programme linéaire donné on connait une solution réalisable y^0 tel que $(y_i^0 > 0, i = 1, ..., n+1)$, alors l'ellipsoide :*

$$ell = \left\{y \in \Re^{n+1} : \sum_{i=1}^{n+1} \frac{(y_i - y_i^0)^2}{(y_i^0)^2} \leq \beta^2, 0 < \beta < 1\right\}$$

est dans l'intérieur de l'orthant positif de $\Re^{n+1}$.

D'après le Lemme 21, si on ajoute au problème (16) la contrainte :

$$y \in \Re^n : \|y - a\| \leq \alpha r \text{ où } 0 < \alpha < 1 \text{ et } r = \frac{1}{\sqrt{n(n+1)}},$$

on obtient alors le problème suivant :

$$\min \left\{ c^t D_k y : AD_k y = 0, \sum_{i=1}^{n} y_i = 1, \|y - a\|^2 \leq (\alpha r)^2 \right\}, \tag{17}$$

ceci est un problème d'optimisation sur une sphère, dont la solution optimale est donnée à l'aide du théorème suivant :

Théorème 22 *[43] La solution optimale du problème* (17) *est donnée explicitement par :* $y^k = a - \alpha r d^k$, *où* $d^k = \frac{p^k}{\|p^k\|}$ *avec* $p^k = P_{B_k}(D_k c)$, $B_k = \begin{bmatrix} AD_k \\ e_n^t \end{bmatrix}$.

A chaque itération k, on revient à la variable initiale x en appliquant la transformation inverse T_k^{-1} et ainsi de suite jusqu'à ce que le test d'optimalité ($c^t x^k \leq \varepsilon$) soit verifié.

3.1.1.1 Algorithme de Karmarkar

Début Algorithme

Initialisation

$\varepsilon > 0$ est un paramètre de précision

$x^0 = a = \frac{e_n}{n}$ est une solution initiale, $e_n = (1, ..., 1) \in \Re^n$

$k = 0$

Tant que ($c^t x^k \geq \epsilon$) **faire**

1. Construire

$$D_k = diag(x^k), A_k = AD_k, B_k = \begin{bmatrix} A_k \\ e_n^t \end{bmatrix}$$

2. Calculer

$$\begin{aligned} P^k &= p_{B_k}(D_k c) = \left[I - B_k^t (B_k B_k^t)^{-1} B_k\right] D_k c, \text{ (la projection sur le noyau de } B_k) \\ d^k &= \frac{P^k}{\|P^k\|} \\ y^k &= a - \alpha r d^k \text{ où } r = \frac{1}{\sqrt{n(n-1)}}, 0 < \alpha < 1 \end{aligned}$$

3. Prendre

$$\begin{aligned} x^{k+1} &= T_k^{-1}(y^k) = \frac{D_k y^k}{e_n^t D_k y^k} \\ k &= k+1 \end{aligned}$$

Fin Tant que

Fin Algorithme

3.1.2 Etude de la convergence

Théorème 23 *[43] A chaque itération k, le point y^k vérifie :*

$$\frac{c^t D_k y^k}{c^t D_k a} \leq 1 - \frac{\alpha}{n-1}.$$

Pour établir la convergence de l'algorithme, Karmarkar introduit à l'objectif la fonction potentiel suivante :

$$f(x) = \sum_{i=1}^{n} \ln(\frac{c^t x}{x_i}), \text{ définie sur : } \left\{ x \in \Re^n : x > 0, Ax = 0, \sum_{i=1}^{n} x_i = 1 \right\}.$$

Lemme 24 *[43] Soit x^k le $k^{ième}$ itéré de l'algorithme, alors :*

$$\frac{c^t x^k}{c^t x^0} \leq \left(\exp\left(f(x^k) - f(x^0)\right)\right)^{\frac{1}{n}}.$$

Karmarkar montre que la convergence de l'algorithme est réalisée en $O(nq + n \ln n)$ itérations pour $0 < \alpha < \frac{1}{4}$.

Théorème 25 *[43] Si* $0 < \alpha < \frac{1}{4}$ *et* $x^0 = \frac{e_n}{n}$, *l'algorithme calcule la solution optimale après* $O(nq + n \ln n)$ *itérations tel que :*

1) $c^t x^k = 0$,

2) $\frac{c^t x^k}{c^t x^0} \leq \varepsilon = 2^{-q}$ *où* q *est une précision fixée.*

3.2 Extension de la méthode de karmarkar

3.2.1 Introduction

nous considérons le problème de complémentarité linéaire (LCP) : trouver des vecteurs x et y de $\Re^n$ qui satisfont aux conditions suivantes :

$$x \geq 0,\ y = Mx + q \geq 0 \text{ et } x^t y = 0,$$

où q est un vecteur donné dans $\Re^n$ et M est une matrice réelle donnée de $\Re^{n \times n}$. LCP a des applications importantes dans la programmation mathématique et les différents domaines de l'ingénierie [3][15]. Les méthodes de la trajectoire centrale primal-dual sont les méthodes les plus attrayants parmi les méthodes de points intérieurs pour résoudre un large éventail de problèmes d'optimisation en raison de leur complexité polynomiale et leur efficacité numérique [1][2][5][18][68][69][71]. D'après le papier de Karmarkar [43] un certain nombre de différents algorithmes de points intérieurs ont été proposées et analysées [73][81][82][57]. Les méthodes primal-dual de points intérieurs $(IPMs)$ pour les problèmes d'optimisation linéaires (LO) ont d'abord été introduit par Kojima et al. [52] et Megiddo [58]. Ils ont montré leurs compétences dans la résolution de grandes classes de problèmes d'optimisation.

L'idée principale de cette méthode est basé sur le remplacement d'un problème de complémentarité linéaire par un programme quadratique convexe. Après l'apparition de l'algorithme de Karmarkar [43], les chercheurs ont introduit des extensions pour la programmation quadratique convexe [4][45][59][61][78]. Nous proposons dans cette section une méthode de points intérieurs de type projectif pour résoudre un problème

plus général où la fonction objectif n'est pas forcément linéaire. Nous combinons l'approche de linéarisation avec des ingrédients apportés par Karmarkar.

3.2.2 Présentation du problème

Nous considérons le programme non linéaire convexe (CNP) sous la forme standard suivant :

$$\min\left\{f(x) : Ax = b,\ x \geq 0\right\}, \qquad (CNP)$$

où $f : \Re^n \to \Re$ est une fonction convexe non linéaire, $A \in \Re^{m\times n}$, $\mathrm{rg}(A) = m,\ b \in \Re^m$.

Le dual du (CNP) :

$$\max\left\{L(x,y,s) : A^t y + s = -\nabla f(x),\ s \geq 0,\ y \in \Re^m\right\},$$

où $L(x,y,s)$ est la fonction lagrangienne.

Le problème de complémentarité linéaire associé à la programmation non linéaire convexe (CNP) est rédigé comme suit :

$$\text{Trouver}\ \ z \in \Re^{n+m}\ \ : z^t w = 0,\ w = Mz + q,\ (w,\ z) \geq 0 \qquad (LCP)$$

où $w \in \Re^{n+m}$, $z = (x,y) \in \Re^{n+m}$, $M = \begin{pmatrix} \nabla^2 f(x) & A^t \\ -A & 0 \end{pmatrix} \in \Re^{(n+m)\times(n+m)}$ et $q \in \Re^{n+m}$.

Remarque 26 *En général, on ne peut transformer un problème de complémentarité linéaire arbitraire à un programme quadratique convexe si et seulement si la matrice M est semi-définie positive.*

Théorème 27 *[81]Un problème de complémentarité linéaire est équivalent au programme quadratique convexe suivant :*

$$\min\left\{z^t(Mz+q) : Mz + q \geq 0,\ z \geq 0\right\}, \qquad (18)$$

où $(z^*, Mz^* + q)$ *est une solution du problème de complémentarité linéaire si et seulement si* z^* *est une solution optimale du problème* (18) *avec* $(z^*)^t(Mz^* + q) = 0$.

Dans la section suivante, nous introduisons l'algorithme de Karmarkar pour résoudre le problème de complémentarité linéaire (LCP).

3.2.3 Préparation de l'algorithme

Nous pouvons écrire le problème (18) sous forme simplifiée qui suit Karmarkar :

$$\min \{g(t) : Bt = 0,\ t \in S_{n+m+1}\}, \tag{19}$$

où $g : \Re^{n+m+1} \to \Re$ est une fonction linéaire, convexe et différentiable, B est une matrice, t est un vecteur et $S_{n+m+1} = \left\{t \in \Re^{n+m+1} : e^t_{n+m+1}t = 1, t \geq 0\right\}$ est le simplexe de dimension $(n + m)$ et de centre $a_i = \frac{1}{n+m+1},\ i = 1, ...,\ n + m + 1$.

Nous introduisons une transformation projective de Karmarkar définie par :

$$T_k : \Re^{n+m} \to S_{n+m+1}$$

$$z \to t,$$

où

$$\begin{cases} t_i = \dfrac{\frac{z_i}{z^k_i}}{1+\sum\limits_{i=1}^{n+m}\frac{z_i}{z^k_i}},\ i = 1, ...,\ n + m \\ t_{n+m+1} = 1 - \sum\limits_{i=1}^{n+m} t_i. \end{cases}$$

Alors, on a

$$z = T_k^{-1}(t) = \frac{D_k t[n + m]}{t_{n+m+1}},$$

où

$$t[n + m] = (D_k^{-1}z)t_{n+m+1} = (z_i)_{i=1}^{n+m},\ D_k = diag(z_k).$$

Ainsi, le problème

$$\begin{aligned} & \min \left\{\ f(z) = z^t(Mz + q) : Mz + q \geq 0\ \right\} \\ \Leftrightarrow\ & \min \left\{\ f(z) = z^t(Mz + q) : Mz = l\ \right\}, \end{aligned}$$

est transformée comme suit

$$\min\left\{f(T_k^{-1}(t)) : M\frac{D_k t[n+m]}{t_{n+m+1}} = l,\ \sum_{i=1}^{n+m+1} t_i = 1,\ t[n+m] \geq 0,\ t_{n+m+1} > 0\right\},\ (20)$$

par conséquent, il est conseillé d'écrire (20) sous la forme :

$$\min\left\{g(t) = t_{n+m+1} f(D_k t[n+m]) : M_k t = 0,\ t \in S_{n+m+1}\right\}. \qquad (21)$$

où

$$M_k = [MD_k,\ -l],\ t = \begin{bmatrix} t[n+m] \\ t_{n+m+1} \end{bmatrix}.$$

Noté que la valeur optimale de g est égal à *zéro* et le centre du simplexe est réalisable pour (21), également noté que la fonction g est convexe sur l'ensemble :

$$\left\{t \in S_{n+m+1} : M_k t = 0\right\}.$$

En linéarisant la fonction g au voisinage de centre du simplexe a_i, et en introduisant une boule de centre a considérée comme voisinage de a , alors, on a :

$$g(t) = g(a) + \langle \nabla g(a), t-a \rangle,\ \text{pour } t \in \left\{t \in \Re^{n+m+1} : \|t-a\|^2 \leq \beta^2\right\}.$$

On obtient le sous problème suivant :

$$\min\left\{\nabla g(a)^t t : M_k t = 0, e_{n+m+1}^t t = 1, \|t-a\|^2 \leq \beta^2\right\}. \qquad (22)$$

Lemme 28 *La solution optimale du problème* (22) *est donnée explicitement par :*

$$t^k = a - \beta d^k,$$

où $d^k = \frac{P^k}{\|P^k\|},\ P^k = p_{B_k}\nabla g(a),\ B_k = \begin{bmatrix} M_k \\ e_{n+m+1}^t \end{bmatrix}.$

Preuve. On pose $z = t - a$, alors on a $B_k z = \begin{bmatrix} M_k \\ e^t_{n+m+1} \end{bmatrix} (t-a) = 0$, et le sous problème (22) est équivalent à :

$$\min \left\{ \nabla g(a)^t z : B_k z = 0, \|z\|^2 \leq \beta^2 \right\}, \tag{23}$$

z^* est une soulution de (23) si et seulement si $\exists \lambda \in \Re^{n+m+1},\ \exists \mu \geq 0$ tels que :

$$\nabla g(a) + B_k^t \lambda + \mu z^* = 0, \tag{7}$$

En multipliant les deux membres de (24) par B_k on obtient :

$$\begin{aligned} B_k \nabla g(a) + B_k B_k^t \lambda + \mu B_k z^* &= 0 \\ B_k \nabla g(a) + B_k B_k^t \lambda &= 0, \end{aligned}$$

alors, on obtient

$$\lambda = -(B_k B_k^t)^{-1} (B_k \nabla g(a)),$$

en substituant dans (24) on obtient

$$z^* = -\frac{1}{\mu} P^k, o\grave{u} P^k = \left[I - B_k^t (B_k B_k^t)^{-1} B_k \right] \nabla g(a),$$

et

$$\|z^*\| = \frac{1}{\mu} \left\| P^k \right\| = \beta \Longrightarrow z^* = -\beta \frac{P^k}{\|P^k\|} = -\beta d^k,$$

ainsi

$$t^k = t^* = a + z^* = a - \beta d^k.$$

d'où le résultat. ■

Dans la suite , on présente l'algorithme générique de notre extension.

3.2.4 Algorithme (Extension d'une méthode projective pour résoudre (LCP))

Début Algorithme

Initialisation :

$\varepsilon > 0$ est un paramètre de précision donné

$0 < \beta < 1,\ z^0$: est une solution strictement réalisable.

$k = 0$

Tant que $(f(z^k) - f(z^*) \geq \epsilon)$ **faire**

1. Construire

$$D_k = diag(z^k), M_k = [MD_k,\ -l], B_k = \begin{bmatrix} M_k \\ e^t_{n+m+1} \end{bmatrix}.$$

2. Calculer

$$\begin{aligned} P^k &= p_{B_k} \nabla g(a) = \left[I - B_k^t (B_k B_k^t)^{-1} B_k\right] \nabla g(a) \\ d^k &= \frac{P^k}{\|P^k\|} \\ t^k &= a - \beta d^k \end{aligned}$$

Prendre

$$\begin{aligned} z^{k+1} &= T_k^{-1}(t^k). \\ k &= k + 1 \end{aligned}$$

Fin Tant que

Fin Algorithme.

3.2.5 Convergence de l'algorithme

Pour établir la convergence de notre algorithme, nous introduisons une fonction potentielle associée au problème (18) définie par :

$$F(z) = (n + m + 1) \log(f(z) - f(z^*)) - \sum_{i=1}^{n+m} \log(z_i).$$

Nous avons le lemme suivant

Lemme 29 *Pour chaque itération, on obtient une réduction de la fonction g i.e :*

$$g(t^k) \leq g(a).$$

Preuve.

On a

$$g(t^k) = g(a) + \left\langle \nabla g(a),\ t^k - a \right\rangle, et\ \ t^k = a - \beta \frac{P^k}{\|P^k\|},$$

alors:

$$\begin{aligned} g(t^k) - g(a) &= \left\langle \nabla g(a),\ -\beta \frac{P^k}{\|P^k\|} \right\rangle \\ &= -\frac{\beta}{\|P^k\|} \left\langle \nabla g(a),\ P^k \right\rangle \\ &= -\frac{\beta}{\|P^k\|} \left\| P^k \right\|^2 < 0 \end{aligned}$$

d'où le résultat. ■

Théorème 30 *À chaque itération de l'algorithme, la fonction de potentiel est réduite à une valeur constante de telle sorte que :*

$$F(z^{k+1}) < F(z^k) - \delta, \quad avec\ 0 < \delta < 1.$$

Preuve.

On a

$$\begin{aligned} F(z^{k+1}) - F(z^k) &= (n+m+1) \log \left[\frac{f(z^{k+1}) - f(z^*)}{f(z^k) - f(z^*)} \right] - \sum_{i=1}^{n+m} \log \left(\frac{z_i^{k+1}}{z_i^k} \right), \\ &= (n+m+1) \log \frac{g(t^k)}{g(a)} - \sum_{i=1}^{n+m} \log \left(t_i^k \right), \\ &\leq (n+m+1) \log \left(1 - \frac{\beta}{n+m+1} + \frac{\beta^2}{2(1-\beta)^2} \right), \\ &\leq -\beta + \frac{\beta^2}{2(1-\beta)^2}, \end{aligned}$$

nous pouvons utiliser le résultat de Karmarkar [43] suivant :

$$-\sum_{i=1}^{n+m} \log \left(t_i^k \right) \leq \frac{\beta^2}{2(1-\beta)^2},$$

alors

$$F(z^{k+1}) < F(z^k) - \delta \text{ où } \delta \ = \beta - \frac{\beta^2}{2(1-\beta)^2}$$

d'où le résultat. ■

Sous les hypothèses suivantes :

Hypothèse 1. La solution initiale z^0 vérifié $z^0 \geq 2^{-2L} e_{n+m+1}$.

Hypothèse 2. La solution optimale z^* vérifié $z^* \leq 2^{2L} e_{n+m+1}$, pour chaque solution z, on a $-2^{3L} \leq f(z^*) \leq 2^{3L}$.

Dans le théorème suivant, nous étudions l'analyse de la complexité de notre extension.

Théorème 31 *L'algorithme trouve la solution optimale après $O((n+m+1)L)$ itérations.*

Preuve.

On a

$$\frac{f(z^k) - f(z^*)}{f(z^0) - f(z^*)} = \eta(z^k) \exp\left[\frac{F(z^k) - F(z^0)}{n+m+1}\right],$$

d'après les hypothèses on a : $\eta(z^k) \leq 2^{2L}$ alors :

$$\begin{aligned} f(z^k) - f(z^*) &\leq 2^{2L}(f(z^0) - f(z^*)) \exp\left[\frac{F(z^k) - F(z^0)}{n+m+1}\right] \\ &\leq 2^{2L} 2^{3L} \exp\left(\frac{-k\delta}{n+m+1}\right) \end{aligned}$$

d'où,

$$k \geq \xi(n+m+1)L, \text{ où } \xi \in \Re_+^*.$$

d'où le résultat ■

3.2.6 Tests Numériques

Dans cette section, nous traitons notre algorithme appliqué à certains problèmes de LCP monotone. Nous avons utilisé $(z^0)^t = ((x^0)^t, (y^0)^t)^t$ pour désigner la solution initiale réalisable de l'algorithme, z^* est la solution optimale de problème LCP et

Iter désigne le nombre d'itérations pour trouver la solution optimale. Le programme est manipulé sur **DEV C++**. La précision est $\varepsilon = 10^{-6}$.

Exemple 1 :

$$M = \begin{pmatrix} 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ -2 & -1 & 0 & 0 & 0 \\ -1 & -2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \end{pmatrix}, \; q = \begin{pmatrix} -4 & -5 & 8 & 7 & 3 \end{pmatrix}^t$$

La solution initiale est :

$z^0 = \begin{pmatrix} 2 & 2 & 2 & 2 & 2 \end{pmatrix}^t$

La solution optimale est :

$z^* = \begin{pmatrix} 2 & 3 & 2 & 1 & 1 \end{pmatrix}^t$.

Iter : 6.

Exemple 2 :

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 3 & 0.8 & 0.32 & 1.128 & 0.0512 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 & 0.32 & 1.128 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 & 0.32 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1.28 & -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0512 & -1.28 & -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$q = \begin{pmatrix} -0.0256 & -0.064 & -0.16 & -0.4 & -1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}^t$$

La solution initiale est :

$z^0 = \begin{pmatrix} 0.18 & 0.18 & 0.18 & 0.18 & 0.25 & 3 & 4 & 5 & 6 & 9 \end{pmatrix}^t$

La solution optimale est :

$z^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^t$

Iter : 11.

3.3 Algorithme à petit pas pour (LCP)

3.3.1 Introduction

nous considérons le problème de complémentarité linéaire (LCP) : trouver des vecteurs x et $y \in \Re^n$ qui satisfont aux conditions suivantes :

$$x \geq 0,\ y = Mx + q \geq 0 \text{ et } x^t y = 0, \tag{24}$$

où q est un vecteur donné dans $\Re^n$ et $M \in \Re^{n \times n}$ est une matrice. Le (LCP) a des applications importantes dans la programmation mathématique et les différents domaines de l'ingénierie. Les méthodes de points inérieurs $(IPMs)$ pour résoudre l'optimisation linéaire (LO) ont été initiées par Karmarkar [44]. Ils ont non seulement une complexité polynomiale, mais sont également très efficaces dans la pratique. Les méthodes de ponits intérieurs $(IPMs)$ réalisables commence par une étude de faisabilité strictement réalisable et de maintenir pendant le processus de solution. Les méthodes de ponits intérieurs $(IPMs)$ réalisables exigent que les points de départ doivent satisfaire exactement les contraintes et sont strictement positifs, i.e., ils se trouvent à l'intérieur d'une région définie par les contraintes. L'extension des méthodes de l'optimisation linéaire aux problèmes (LCP) a été un succès dans de nombreux cas. voir, e.g.,[42][70].

Dans cette partie, nous étudions l'analyse de la complexité et de mise en œuvre d'un algorithme à petit pas par une méthode de points intérieurs. Cet algorithme est basé sur la stratégie de la trajectoire centrale pour trouver une nouvelle directions de descente.

3.3.2 Présentation du problème

Soit le problème de compléméntarité linéaire suivant :

$$x \geq 0,\ y = Mx + q \geq 0 \text{ et } x^t y = 0, \tag{25}$$

L'ensemble de réalisabilité, l'ensemble strictement réalisable et l'ensemble de la solution du problème (25) sont désignés respectivement par

$$\mathcal{F} = \left\{(x, y) \in \Re^{2n} : y = Mx + q, x \geq 0, y \geq 0\right\},$$

$$\overset{o}{\mathcal{F}} = \left\{(x, y) \in \mathcal{F} : x > 0, y > 0\right\},$$

et

$$\Omega = \left\{(x, y) \in \mathcal{F} : x \geq 0, y \geq 0, x^t y = 0\right\}.$$

Dans ce qui suit, nous supposons que les hypothèses suivantes sont vérifiées.

Hypothèse 1. $\overset{o}{\mathcal{F}} \neq \emptyset$.

Hypothèse 2. M est une matrice semi-définie positive.

Le problème (25), est équivalent au problème d'optimisation suivant : voir [81].

$$\min\left\{x^t y : x \geq 0, y \geq 0, y = Mx + q\right\}. \tag{26}$$

Par conséquent, trouver la solution de (25) est équivalent à résoudre le problème (26) avec sa valeur objectif est zéro.

Afin d'introduire une méthode de points intérieurs pour résoudre (26), nous introduisons le problème pénalisé suivant :

$$\min\left\{f_{\mu r}(x, y) : y = Mx + q, x > 0, y > 0\right\}, \tag{27}$$

où $f_{\mu r}(x, y) = x^t y - \mu \sum_{i=1}^{n} r_i \log(x_i y_i)$, $\mu > 0$ est le paramètre barrière et $r = (r_1, r_2, ..., r_n) \in \Re^n_+$ est un vecteur poids présenter à faire en sorte que le point initial (x^0, y^0) vérifie $\delta(x^0 y^0, \mu^0) = 0$ (mesure de proximité définit ci-dessous).

Si $r_i = 1, i = 1, ..., n$, alors la méthode de la trajectoire centrale avec poids coïncide avec la méthode classique.

Puisque le problème (27) est un problème d'optimisation convexe, les contraintes sont linéaires, alors les conditions d'optimalité sont données par :

$$\begin{cases} Mx + q = y, \\ xy = \mu r, \ x > 0, y > 0. \end{cases} \tag{28}$$

D'après les hypothèses 1 et 2, alors pour chaque $\mu > 0$, le problème (27) et le système (28) admet une solution unique [81] que l'on note par $(x(\mu), y(\mu))$, avec $x(\mu) > 0$ et $y(\mu) > 0$. Nous appelons $(x(\mu), y(\mu))$, $\mu > 0$, le μ-centre de du problème (28). L'ensemble de μ-centres définit ce que l'on appelle le chemin (trajectoire) centrale du problème (25).

Dans la section suivante, nous introduisons une méthode pour tracer la trajectoire centrale basée sur une nouvelle classe de directions de descente.

3.3.3 Direction de descente

Maintenant, l'idée de base de cette approche consiste à remplacer l'équation non linéaire :

$$\frac{xy}{\mu r} = e,$$

dans le système (28) par

$$\psi(\frac{xy}{\mu r}) = \psi(e),$$

où ψ, est une fonction à valeurs réelles sur $[0, \infty[$ et dérivable sur $[0, \infty[$ tels que $\psi(t)$ et $\psi^{'}(t) > 0$, pour $t > 0$. Alors le système (28) peut s'écrire sous forme équivalente suivante :

$$\begin{cases} Mx + q = y, \ x > 0, y > 0 \\ \psi(\frac{xy}{\mu r}) = \psi(e). \end{cases} \tag{29}$$

Supposons que $(x, y) \in \overset{o}{\mathcal{F}}$. L'application de la méthode de Newton pour le système (29), nous donne une nouvelle classe de directions de descente :

$$\begin{cases} M\triangle x = \triangle y, \\ \frac{y}{\mu r}\psi^{'}(\frac{xy}{\mu r})\triangle x + \frac{x}{\mu r}\psi^{'}(\frac{xy}{\mu r})\triangle y = \psi(e) - \psi(\frac{xy}{\mu r}). \end{cases} \tag{30}$$

Maintenant, les notations suivantes sont utiles pour étudier la complexité de l'algorithme proposé.

Les vecteurs

$$v = \sqrt{\frac{xy}{\mu r}},\ d = \sqrt{xy^{-1}},$$

cette notation conduit à écrire

$$\frac{d^{-1}x}{\sqrt{\mu r}} = \frac{dy}{\sqrt{\mu r}} = v.$$

On note par

$$d_x = \frac{d^{-1}\triangle x}{\sqrt{\mu r}},\ d_y = \frac{d\triangle y}{\sqrt{\mu r}}, \tag{31}$$

et, par conséquent, on a

$$\mu r v(d_x + d_y) = y\triangle x + x\triangle y, \tag{32}$$

et

$$d_x d_y = \frac{\triangle x \triangle y}{\mu r} \tag{33}$$

En utilisant (31) et(32), le système (30) devient

$$\begin{cases} \overline{M} d_x = d_y, \\ d_x + d_y = p_v, \end{cases}$$

où $\overline{M} = MDM$ avec $D = diag(d)$

et

$$p_v = \frac{\psi(e) - \psi(v^2)}{v\psi^{'}(v^2)}.$$

Nous considérons la fonction (kernel function) suivante :

$$\psi(t) = \frac{1}{2}(t^2 - 1), \text{ avec } \psi^{'}(t) = t \text{ pour } t > 0.$$

Par conséquent, les directions de Newton (30) sont la solution du système suivant

$$\begin{cases} M\triangle x = \triangle y, \\ d_x + d_y = \frac{1}{2}(v^{-1} - v), \end{cases} \tag{34}$$

avec

$$p_v = \frac{1}{2}(v^{-1} - v),$$

et on définit pour tout vecteur v la mesure de proximité par

$$\begin{aligned} \delta(xy, \mu) &= \frac{\|p_v\|_2}{2}, \\ &= \left\|v^{-1} - v\right\|_2, \\ &= \left\| \left(\sqrt{\frac{xy}{\mu r}}\right)^{-1} - \sqrt{\frac{xy}{\mu r}} \right\|_2. \end{aligned}$$

L'algorithme générique de petit pas primal-dual pour résoudre le (LCP) est présenté comme suit :

3.3.4 Algorithme (Méthode de la trajectoire centrale à petit pas pour résoudre LCP)

Début Algorithme

$\varepsilon > 0$ est un paramètre de précision donné

un parmètre θ, $0 < \theta < 1$ (par défaut $\theta = \frac{1}{2\sqrt{n}}$)

une solution initiale strictement réalisable (x^0, y^0) et $\mu^0 = \frac{(x^0)^t \ Ry^0}{n}$.

$\sigma = \dfrac{\|X^0Y^0e\|}{\sqrt{n}}$, $r = \dfrac{X^0Y^0e}{\sigma}.R = diag(r_i)$

$k = 0$

Tant que $(n\mu^k > \varepsilon\ )$ **faire**

1. Calculer

$(\triangle x, \triangle y)$ les directions de descente

2. Nouvel itéré

$$(x^{k+1}, y^{k+1}) = (x^k, y^k) + (\triangle x, \triangle y)$$

3. Prendre

$$\begin{aligned} \mu^{k+1} &= (1-\theta)\mu^k = (1-\theta)\frac{x^k R y^k}{n} \\ k &= k+1. \end{aligned}$$

Fin Tant que.

Fin Algorithme.

3.3.5 Etude de la convergence

Soit

$$p_v = d_x + d_y,\ q_v = d_x - d_y,$$

et on a

$$d_x = \frac{1}{2}(p_v + q_v),\ d_y = \frac{1}{2}(p_v - q_v),$$

alors

$$d_x d_y = \frac{1}{4}(p_v^2 - q_v^2) \text{ et } \|q_v\|_2 \leq \|p_v\|_2 .$$

Ce qui donne

$$\|p_v\|_2^2 = \|q_v\|_2^2 + 4 d_x^t d_y,$$

lorsque

$$d_x^t d_y = d_x^t \overline{M} d_x \geq 0, \text{car la matrice } \overline{M} \text{ est semi définie positive.}$$

On a

$$\delta(v, \mu) \geq \|q_v\|_2$$

Dans le lemme suivant, nous énonçons une condition qui assure la faisabilité de l'itéré de Newton, soit :

$$x^+ = x + \triangle x \text{ et } y^+ = y + \triangle y,$$

le nouvel itéré.

Lemme 32 *Soit* (x, y) *une solution strictement réalisable du problème* (25) .

Si $(e + d_x d_y) > 0$ *alors* $(x^+, y^+) = (x + \triangle x, y + \triangle y)$

Preuve. Soit $0 < \alpha \leq 1$ le pas de déplacement.

Nous définissons :

$$x(\alpha) = x + \alpha \triangle x \text{ et } \ y(\alpha) = y + \alpha \triangle y,$$

alors, on a

$$\begin{aligned} x(\alpha)y(\alpha) &= (x + \alpha\triangle x)(y + \alpha\triangle y) \\ &= xy + \alpha(x\triangle y + y\triangle x) + \alpha^2 \triangle x \triangle y \\ &= xy + \alpha(\mu r - xy) + \alpha^2 \triangle x \triangle y. \end{aligned}$$

Nous supposons que $(e + d_x d_y) > 0$, on en déduit que $(\mu r + \triangle x \triangle y) > 0$, ce qui correspond à $\triangle x \triangle y > -\mu r$ et par substitution, on obtient

$$\begin{aligned} x(\alpha)y(\alpha) &> xy + \alpha(\mu r - xy) - \alpha^2 \mu r \\ &= (1 - \alpha)xy + (\alpha - \alpha^2)\mu r \\ &= (1 - \alpha)xy + \alpha(1 - \alpha)\mu r. \end{aligned}$$

Lorsque $xy > 0$ et $\mu r > 0$, il s'ensuit que $x(\alpha)y(\alpha) > 0$ pour $\alpha \in]0, 1]$. ■

Maintenant, pour plus de commodité, nous pouvons écrire

$$(v^+)^2 = \frac{x^+ y^+}{\mu r} = e + d_x d_y.$$

Lemme 33 *Si la fonction de proximité* $\delta(xy, \mu) < 1$, *Alors* $x^+ > 0$ *et* $y^+ > 0$.

Preuve. D'après le résultat précédent, on a (x^+, y^+) est strictement réalisable si $(e + d_x d_y) > 0$. Alors $(e + d_x d_y) > 0$ est vrai si $(1 + (d_x d_y)_i) > 0$ pour $i \in \Re^n$.

On a

$$\begin{aligned}(1+(d_x d_y)_i) &\geq (1-|(d_x d_y)_i|\,, \text{ pour } i \in \Re^n \\ &\geq (1-\delta^2).\end{aligned}$$

Ainsi $(e + d_x d_y) > 0$ si $\delta(xy, \mu) < 1$. ■

Le théorème suivant donne une borne supérieure pour le nombre total d'itérations pour l'algorithme.

Remarque 34 *Pour prouver la convergence quadratique de notre algorithme nous appliquons les lemmes* (18) *et* (19) *cité dans le chapitre* 2.

Théorème 35 *Soit $\varepsilon > 0$ un paramètre de précision. L'algorithme a une complexité liée de $O(\sqrt{n}L)$ itérations et la complexité totale de l'algorithme lié est $O(n^{2.5}L)$, où $L = \log \frac{\mu^0}{\varepsilon}$.*

Preuve. On a

$$\mu^k = (1-\theta)^k \mu^0,$$

ainsi,

$$(1-\theta)^k \mu^0 \leq \varepsilon.$$

Maintenant, en prenant le logarithme de $(1-\theta)^k \mu^0 \leq \varepsilon$, on obtient

$$\log((1-\theta)^k \mu^0) \leq \log \varepsilon$$

équivalent à :

$$k \log(1-\theta) \leq \log \varepsilon - \log \mu^0,$$

en utilisant $\log(1-\theta) \leq \theta$, pour $0 \leq \theta \leq 1$,alors l'inégalité ci-dessus est vérifiée si

$$k \geq \frac{1}{\theta}\left[\log \frac{\mu^0}{\varepsilon}\right].$$

Soit $L = \log \frac{\mu^0}{\varepsilon}$, alors $k \geq 2\sqrt{n} \log \frac{\mu^0}{\varepsilon} = O(\sqrt{n} \log \frac{\mu^0}{\varepsilon}) = O(\sqrt{n}L)$ itérations dans l'algorithme. Cependant, à chaque étape, la complexité liée pour calculer le système linéaire est de compléxité $O(n^2)$. Par conséquent, la complexité totale de l'algorithme est $O(n^{2.5}L)$. ■

3.3.6 Tests numériques

Dans cette section, nous traitons cet algorithme appliqué à certains problèmes de $LCPs$ monotone. On note par (x^0, y^0) la solution initiale strictement réalisable tels que $\delta(x^0y^0, \mu^0) < \frac{1}{\sqrt{2}}$, (x^*, y^*) est la solution optimale de LCP et **Iter** est le nombre d'itérations effectuer pour calculer la solution optimale. z^* est la valeur objectif à (x^*, y^*) et μ^* désigne la valeur lorsque la solution optimale est determinée. La mise en œuvre est manipulé en **DEV C++**. La précision est $\varepsilon = 10^{-6}$. On varie le paramètre θ, $0 < \theta < 1$. Enfin, nous notons que le système linéaire de Newton est résolu grâce à la procédure d'élimination de Gauss.

Exemple 1.

$$M = \begin{pmatrix} 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ -2 & -1 & 0 & 0 & 0 \\ -1 & -2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \end{pmatrix}, q = \begin{pmatrix} -4 & -5 & 8 & 7 & 3 \end{pmatrix},$$

La solution initiale est

$$x^0 = \begin{pmatrix} 2 & 2 & 2 & 2 & 2 \end{pmatrix}, y^0 = \begin{pmatrix} 2 & 3 & 2 & 1 & 1 \end{pmatrix}$$

$\delta(x^0y^0, \mu^0) = 2.517539 > \frac{1}{\sqrt{2}}$,

Les résultats numériques pour ce problème sont résumés dans le tableau ci-dessous :

	Résultats de l'algorithme
	$\delta(x^0y^0, \mu^0) = 0.000000 < \frac{1}{\sqrt{2}}$
$\theta = 0.15$	**iter** 87 x^* $\begin{pmatrix} 2.999995 & 2.000002 & 1.000001 & 2.000001 & 0.000005 \end{pmatrix}$ y^* $\begin{pmatrix} 0.000003 & 0.000007 & 0.000009 & 0.000002 & 0.999998 \end{pmatrix}$ z^* 0.000041 μ^* 0.000009
$\theta = 0.20$	**iter** 65 x^* $\begin{pmatrix} 2.999994 & 2.000002 & 1.000001 & 2.000000 & 0.000005 \end{pmatrix}$ y^* $\begin{pmatrix} 0.000003 & 0.000008 & 0.000010 & 0.000002 & 0.999998 \end{pmatrix}$ z^* 0.000045 μ^* 0.000010
$\theta = 0.60$	**iter** 24 x^* $\begin{pmatrix} 2.999994 & 2.000002 & 1.000001 & 2.000001 & 0.000004 \end{pmatrix}$ y^* $\begin{pmatrix} 0.000003 & 0.000006 & 0.000008 & 0.000002 & 0.999998 \end{pmatrix}$ z^* 0.000037 μ^* 0.000008
$\theta = 0.95$	**iter** 19 x^* $\begin{pmatrix} 2.999993 & 2.000003 & 1.000002 & 2.000000 & 0.000004 \end{pmatrix}$ y^* $\begin{pmatrix} 0.000002 & 0.000006 & 0.000012 & 0.000002 & 0.999997 \end{pmatrix}$ z^* 0.000036 μ^* 0.000008

Exemple 2.

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 3 & 0.8 & 0.32 & 1.128 & 0.0512 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 & 0.32 & 0.128 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 & 0.32 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1.128 & -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0512 & -1.128 & -0.32 & -0.8 & -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$q = \begin{pmatrix} -0.0256 & -0.064 & -0.16 & 5.59 & -1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

La solution initiale est

$$x^0 = \begin{pmatrix} 0.18 & 0.18 & 0.18 & 0.18 & 0.25 & 3 & 4 & 5 & 6 & 9 \end{pmatrix},$$

$$y^0 = \begin{pmatrix} 21.0032 & 11.008 & 12.52 & 12.8 & 8 & 0.46 & 0.676 & 0.6184 & 0.41536 & 0.336144 \end{pmatrix}.$$

$\delta(x^0 y^0, \mu^0) = 2.278802 > \frac{1}{\sqrt{2}}$,

Les résultats numériques pour ce problème sont résumés dans le tableau ci-dessous :

	Résultats de l'algorithme
	$\delta(x^0y^0, \mu^0) = 0.000000 < \frac{1}{\sqrt{2}}$
$\theta = 0.15$	**iter** 84 x^* $\begin{pmatrix} 0.000036 & 0.886520 & 0.000003 & 0.000001 & 0.000001 \\ 0.000005 & 0.000088 & 0.000039 & 0.000013 & 8.035358 \end{pmatrix}$ y^* $\begin{pmatrix} 0.385925 & 0.000008 & 2.411364 & 12.028297 & 7.035360 \\ 0.999892 & 0.113452 & 0.290770 & 0.716269 & 0.000001 \end{pmatrix}$ z^* 0.000085 μ^* 0.000009
$\theta = 0.20$	**iter** 64 x^* $\begin{pmatrix} 0.000036 & 0.886520 & 0.000003 & 0.000001 & 0.000001 \\ 0.000005 & 0.000088 & 0.000039 & 0.000013 & 8.035358 \end{pmatrix}$ y^* $\begin{pmatrix} 0.385925 & 0.000008 & 2.411364 & 12.028297 & 7.035360 \\ 0.999892 & 0.113452 & 0.290770 & 0.716269 & 0.000001 \end{pmatrix}$ z^* 0.000095 μ^* 0.000010

$\theta = 0.60$	**iter** 24 x^* $\begin{pmatrix} 0.000024 & 0.886521 & 0.000002 & 0.000001 & 0.000001 \\ 0.000003 & 0.000058 & 0.000026 & 0.000009 & 8.035394 \end{pmatrix}$ y^* $\begin{pmatrix} 0.385887 & 0.000005 & 2.411360 & 12.028322 & 7.035394 \\ 0.999927 & 0.113460 & 0.290773 & 0.716283 & 0.000001 \end{pmatrix}$ z^* 0.000065 μ^* 0.000007
$\theta = 0.95$	**iter** 19 x^* $\begin{pmatrix} 0.000047 & 0.886521 & 0.000003 & 0.000001 & 0.000000 \\ 0.000001 & 0.000032 & 0.000025 & 0.000004 & 8.035417 \end{pmatrix}$ y^* $\begin{pmatrix} 0.385854 & 0.000002 & 2.411361 & 12.028336 & 7.035416 \\ 0.999859 & 0.113441 & 0.290765 & 0.716257 & 0.000000 \end{pmatrix}$ z^* 0.000048 μ^* 0.000005

Exemple 3. Soit $M \in \Re^{n\times n}$ et $q \in \Re^n$ donnés par :

$$M = \begin{pmatrix} 1 & 2 & 2 & . & . & . & 2 \\ 0 & 1 & 2 & . & . & . & 2 \\ 0 & 0 & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & 2 \\ 0 & 0 & 0 & . & . & 0 & 1 \end{pmatrix}, q = \begin{pmatrix} -1 & . & . & . & -1 \end{pmatrix},$$

Cas 1 : $n = 10$.

La solution initiale est

$$x^0 = \begin{pmatrix} 0.0009 & 0.0009 & 0.0009 & 0.0009 & 0.0009 \\ 0.0009 & 0.0009 & 0.0009 & 0.0009 & 1.0009 \end{pmatrix},$$

$$y^0 = \begin{pmatrix} 1.0171 & 1.0153 & 1.0135 & 1.0108 & 1.0099 \\ 1.0081 & 1.0063 & 1.0045 & 1.0027 & 0.0009 \end{pmatrix}.$$

$\delta(x^0y^0, \mu^0) = 0.032154 < \frac{1}{\sqrt{2}}$,

Les résultats numériques pour ce problème sont résumés dans le tableau ci-dessous :

	Résultats de l'algorithme ($r = e$)
	$\delta(x^0y^0, \mu^0) = 0.032154 < \frac{1}{\sqrt{2}}$
$\theta = 0.15$	**iter** 31 x^* $\begin{pmatrix} 0.000009 & 0.000009 & 0.000009 & 0.000009 & 0.000009 \\ 0.000009 & 0.000009 & 0.000009 & 0.000009 & 1.000009 \end{pmatrix}$ y^* $\begin{pmatrix} 1.000168 & 1.000151 & 1.000133 & 0.999215 & 1.000097 \\ 1.000080 & 1.000062 & 1.000044 & 1.000027 & 0.000009 \end{pmatrix}$ z^* 0.000089 μ^* 0.000009
$\theta = 0.20$	**iter** 23 x^* $\begin{pmatrix} 0.000009 & 0.000009 & 0.000009 & 0.000009 & 0.000009 \\ 0.000009 & 0.000009 & 0.000009 & 0.000009 & 1.000010 \end{pmatrix}$ y^* $\begin{pmatrix} 1.000180 & 1.000161 & 1.000142 & 0.999223 & 1.000104 \\ 1.000085 & 1.000066 & 1.000047 & 1.000028 & 0.000009 \end{pmatrix}$ z^* 0.000095 μ^* 0.000009

$\theta = 0.60$	**iter** 9 x^* $\begin{pmatrix} 0.000007 & 0.000007 & 0.000007 & 0.000007 & 0.000007 \\ 0.000007 & 0.000007 & 0.000007 & 0.000007 & 1.000007 \end{pmatrix}$ y^* $\begin{pmatrix} 1.000129 & 1.000115 & 1.000102 & 0.999188 & 1.000074 \\ 1.000061 & 1.000047 & 1.000034 & 1.000020 & 0.000007 \end{pmatrix}$ z^* 0.000068 μ^* 0.000007
$\theta = 0.95$	**iter** 7 x^* $\begin{pmatrix} 0.000007 & 0.000007 & 0.000007 & 0.000007 & 0.000007 \\ 0.000007 & 0.000007 & 0.000007 & 0.000007 & 1.000007 \end{pmatrix}$ y^* $\begin{pmatrix} 1.000138 & 1.000124 & 1.000109 & 0.999194 & 1.000080 \\ 1.000065 & 1.000050 & 1.000036 & 1.000021 & 0.000007 \end{pmatrix}$ z^* 0.000073 μ^* 0.000007

Cas 2 : $n = 15$.

La solution initiale est

$$x^0 = \begin{pmatrix} 0.0009 & 0.0009 & 0.0009 & 0.0009 & 0.0009 \\ 0.0009 & 0.0009 & 0.0009 & 0.0009 & 0.0009 \\ 0.0009 & 0.0009 & 0.0009 & 0.0009 & 1.0009 \end{pmatrix},$$

$$y^0 = \begin{pmatrix} 1.0261 & 1.0243 & 1.0225 & 1.0198 & 1.0189 \\ 1.0171 & 1.0153 & 1.0135 & 1.0108 & 1.0099 \\ 1.0081 & 1.0063 & 1.0045 & 1.0027 & 0.0009 \end{pmatrix}.$$

$\delta(x^0y^0, \mu^0) = 0.059169 < \frac{1}{\sqrt{2}}$,

Les résultats numériques pour ce problème sont résumés dans le tableau ci-dessous :

	Résultats de l'algorithme $(r = e)$
	$\delta(x^0y^0, \mu^0) = 0.032154 < \frac{1}{\sqrt{2}}$
$\theta = 0.15$	**iter** 15 x^* $\begin{pmatrix} 0.000097 & 0.000097 & 0.000097 & 0.000097 & 0.000097 \\ 0.000097 & 0.000097 & 0.000097 & 0.000097 & 0.000097 \\ 0.000097 & 0.000097 & 0.000097 & 0.000097 & 1.000097 \end{pmatrix}$ y^* $\begin{pmatrix} 1.002816 & 1.002622 & 1.002428 & 1.001334 & 1.002040 \\ 1.001846 & 1.001652 & 1.001458 & 1.000363 & 1.001069 \\ 1.000875 & 1.000680 & 1.000486 & 1.000292 & 0.000097 \end{pmatrix}$ z^* 0.001458 μ^* 0.000097
$\theta = 0.20$	**iter** 12 x^* $\begin{pmatrix} 0.000084 & 0.000084 & 0.000084 & 0.000084 & 0.000084 \\ 0.000084 & 0.000084 & 0.000084 & 0.000084 & 0.000084 \\ 0.000084 & 0.000084 & 0.000084 & 0.000084 & 1.000084 \end{pmatrix}$ y^* $\begin{pmatrix} 1.002448 & 1.002280 & 1.002111 & 1.001042 & 1.001773 \\ 1.001605 & 1.001436 & 1.001267 & 1.000198 & 1.000929 \\ 1.000760 & 1.000591 & 1.000423 & 1.000253 & 0.000085 \end{pmatrix}$ z^* 0.001268 μ^* 0.000085

$\theta = 0.60$	**iter** 5 x^* $\begin{pmatrix} 0.000061 & 0.000061 & 0.000061 & 0.000061 & 0.000061 \\ 0.000060 & 0.000060 & 0.000060 & 0.000060 & 0.000060 \\ 0.000060 & 0.000060 & 0.000060 & 0.000060 & 1.000060 \end{pmatrix}$ y^* $\begin{pmatrix} 1.001750 & 1.001629 & 1.001508 & 1.000486 & 1.001265 \\ 1.001145 & 1.001024 & 1.000903 & 0.999882 & 1.000662 \\ 1.000541 & 1.000421 & 1.000300 & 1.000180 & 0.000060 \end{pmatrix}$ z^* 0.000906 μ^* 0.000060
$\theta = 0.95$	**iter** 4 x^* $\begin{pmatrix} 0.000060 & 0.000059 & 0.000059 & 0.000059 & 0.000059 \\ 0.000059 & 0.000058 & 0.000058 & 0.000058 & 0.000058 \\ 0.000058 & 0.000057 & 0.000057 & 0.000057 & 1.000057 \end{pmatrix}$ y^* $\begin{pmatrix} 1.001688 & 1.001569 & 1.001450 & 1.000432 & 1.001214 \\ 1.001097 & 1.000980 & 1.000863 & 0.999847 & 1.000631 \\ 1.000515 & 1.000400 & 1.000286 & 1.000171 & 0.000057 \end{pmatrix}$ z^* 0.000874 μ^* 0.000058

3.3.6.1 Conclusion

Dans l'algorithme de Karmarkar le calcul de la projection P^k constitue l'opération la plus coûteuse dans l'algorithme. L'efficacité pratique de l'algorithme dépend, en grande partie, de la manière du calcul de P^k.

Chapitre IV

ALGORITHME DE POINTS INTÉRIEURS POUR UN PROBLÈME D'OPTIMISATION SEMI DÉFINI (SDO) BASÉ SUR UNE NOUVELLE FONCTION NOYAU

Les méthodes de points intérieurs $IPMs$ démarre avec un point positif arbitraire et la faisabilité est atteint quand l'optimalité est approché. Le choix du point de départ dans IPM est cruciale pour la performance. Lustig [53] et Tanabe [77] ont été les premiers à présenter $IPMs$ pour l'optimisation linéaire (LO). Kojima et al. [51] ont été les premiers qui ont prouvé la convergence globale d'un primal - dual IPM pour LO. Zhang [83] a été le premier qui a présenté la compléxité polynomiale pour l'optimisation linéaire (LO). Les méthodes primale-duale de points intérieurs (IPM) pour la programmation semi définie (SDO) ont été largement étudiés, le lecteur est renvoyé à Klerk [49] pour plus d'eclaircissement. Récemment, une méthode de points intérieurs non réalisable de programmation linéaire (PL) a été présenté par Roos [72]. Certaines extensions des méthodes de points intérieurs ont été réalisées par Mansouri et Roos [55]. Mansouri et Roos [54] ont étendu cet algorithme de programmation semi-définie en utilisant une étape de faisabilité spécifique. La fonction de barrière est déterminée par une fonction simple unidimensionnel, appelé fonction noyau. Bai et al. [7] introduit une nouvelle fonction de barrière qui n'est pas une fonction de barrière dans le sens habituel du terme.

Dans ce chapitre, nous définissons une nouvelle fonction de proximité pour (SDO)

par une nouvelle fonction noyau. En outre, nous formulons un algorithme de points intérieurs primal-dual pour (SDO) en utilisant la fonction de proximité et de donner son analyse de complexité. De plus, la complexité obtenue par l'algorithme de long pas est $O(m^{\frac{3m+1}{2m}} n^{\frac{m+1}{2m}} \log \frac{Tr(X^0 S^0)}{\varepsilon})$ et de petit pas est $O(m^{\frac{3m+1}{2m}} \sqrt{n} \log \frac{Tr(X^0 S^0)}{\varepsilon})$.

Les notations utilisées tout au long du chapitre sont les suivantes : $\Re^n$, $\Re^n_+$ et $\Re^n_{++}$ désignent respectivement l'ensemble des vecteurs à n composantes, l'ensemble de vecteurs positifs ou nuls et l'ensemble de vecteurs strictement positifs. $\Re^{n\times n}$ désigne l'ensemble des matrices réelles. $\|.\|_F$ et $\|.\|_2$ désigne respectivement la norme de Frobenius et la norme spectrale de matrices. S_n, S_n^+ et S_n^{++} désigne respectivement le cône de matrices symétrique, matrices symétrique semi-définie positive et matrices symétrique définie positive de dimension $n \times n$. E représente la matrice identité. $A \succeq B$ (or $A \succ B$) si $A - B$ est semi-définie positive (ou bien definie positive). Nous utilisons le produit de matrice interne $A \bullet B = Tr(A^T B)$. Pour $Q \in S_n^{++}$, l'expression $Q^{\frac{1}{2}}$ indique sa racine carrée symétrique. pour $V \in S_n^+$, on note par $\lambda_{\min}(V)$ la valeur propre minimale de V.

4.1 Déscription de l'algorithme

4.1.1 Méthode de la trajectoire centrale

Nous considérons le problème d'optimisation semi définie (SDO), dont la forme primale est donnée par :

$$\min \{Tr(CX) : \ Tr(A_i X) = b_i, i = 1, ..., m, \ X \succeq 0\}, \qquad (SDO)$$

et son dual

$$\max \left\{ b^t y : \ \sum_{i=1}^{m} y_i A_i + S = C, \ S \succeq 0 \right\}, \qquad (SDD)$$

où $A_i, \ i = 1, ..., m$ et C sont des matrices symétriques de $\Re^{n\times n}$, b et $y \in \Re^m$.

Tout au long de ce chapitre, nous formulons les hypothèses suivantes :

Hypothèse 1 : les matrices $A_i, i = 1, ..., m$ sont linéairement indépendantes

Hypothèse 2 : une solution initiale (X^0, y^0, S^0) tels que :

$$Tr(A_i X^0) = b_i, i = 1, ..., m,\ X^0 \succ 0,\ \sum_{i=1}^{m} y_i^0 A_i + S^0 = C,\ S \succ 0.$$

Nous avons le lemme suivant qui est bien connu :

Lemme 36 *[67] Les propriétés suivantes sont équivalentes :*

1. $X \succeq 0,\ S \succeq 0$ *et* $Tr(XS) = 0$,
2. $X \succeq 0,\ S \succeq 0$ *et* $\left\| X^{\frac{1}{2}} S^{\frac{1}{2}} \right\|^2 = 0$,
3. $X \succeq 0,\ S \succeq 0$ *et* $XS = 0$.

Il est bien connu que la recherche d'une solution optimale (X^*, y^*, S^*) de (SDO) et (SDD) est équivalent à résoudre le système suivant :

$$\begin{cases} Tr(A_i X) = b_i, i = 1, ..., m,\ X \succeq 0, \\ \sum_{i=1}^{m} y_i A_i + S = C,\ S \succeq 0, \\ XS = 0. \end{cases} \tag{35}$$

L'idée de base des méthodes de points intérieurs $(IPMs)$ de type primale-duale consiste à remplacer dans le système (35) la troisième équation, la condition de complémentarité par l'équation paramétrisée $XS = \mu E$ avec $\mu > 0$, où E est une matrice identité d'ordre n. Ainsi on considère le système suivant

$$\begin{cases} Tr(A_i X) = b_i, i = 1, ..., m,\ X \succeq 0, \\ \sum_{i=1}^{m} y_i A_i + S = C,\ S \succeq 0, \\ XS = \mu E. \end{cases} \tag{36}$$

Si le problème (SDO) et (SDD) satisfait les conditions de points strictement intérieur $(IPCs)$, alors pour chaque $\mu > 0$ le système (36) admet une solution unique $(X(\mu), y(\mu), S(\mu))$ (voir [50][63][79][76]), que l'on appelle μ-centre de (SDO) et de

(SDD). L'ensemble de μ-centres, $\Lambda = \{(X(\mu), y(\mu), S(\mu)) / \mu > 0\}$, construit la trajectoire centrale de (SDO) et de (SDD).

En générale, les méthodes de points intérieurs $(IPMs)$ pour (SDO) se composent de deux stratégies : La première stratégie consiste à déterminer une solution paramétrisée strictement réalisable et vérifie certaine conditions (condition de proximité) et la seconde consiste à diminuer le paramètre μ à $\mu^+ = (1-\theta)\mu$, $0 < \theta < 1$ pour déterminer la qualité de la solution.

4.1.2 Direction de descente

Sans perte de généralité de $IPMs$, nous supposons que $(X(\mu), y(\mu), S(\mu))$ est une solution pour $\mu > 0$. Par exemple, en raison des hypothèses citées dans le paragraphe 4.1.1, on peut supposer cela pour $\mu^0 = \frac{Tr(X^0 S^0)}{n}$, avec $X^0 \succ 0$ et $S^0 \succ 0$. Nous dimunions ensuite μ à $\mu^+ = (1-\theta)\mu$, pour $0 < \theta < 1$, et on résout le système suivant :

$$\begin{cases} Tr(A_i \triangle X) = 0, i = 1, ..., m, \\ \sum\limits_{i=1}^{m} \triangle y_i A_i + \triangle S = 0, \\ X \triangle S + \triangle X S = \mu E - XS. \end{cases} \tag{37}$$

Nous considérons le schéma de symétrisation qui donne la direction de Nestrov-Toold (NT), on définit la matrice :

$$P = X^{\frac{1}{2}}(X^{\frac{1}{2}} S X^{\frac{1}{2}})^{-\frac{1}{2}} X^{\frac{1}{2}} = S^{-\frac{1}{2}}(S^{\frac{1}{2}} X S^{\frac{1}{2}})^{\frac{1}{2}} S^{-\frac{1}{2}}, \tag{38}$$

définir aussi

$$D = P^{\frac{1}{2}}, V = \frac{1}{\sqrt{\mu}} D^{-1} X D^{-1} = \frac{1}{\sqrt{\mu}} DSD$$

On remarque que les matrices D et V sont semi définies positives et symétriques, similaire à (SDO) [55]. On peut conclure que le système (35) a une unique solution symétrique, on définit

$$\overline{A}_i := \frac{1}{\sqrt{\mu}} D A_i D, \ i = 1, ..., m. \ D_X := \frac{1}{\sqrt{\mu}} D^{-1} \triangle X D^{-1} \ \text{ et } \ D_S := \frac{1}{\sqrt{\mu}} D \triangle S D. \tag{39}$$

La direction de descente de (NT) est obtenu à partir du système suivant :

$$\begin{cases} Tr(\overline{A}_i D_X) = 0, \ i = 1, ..., m, \\ \sum\limits_{i=1}^{m} \triangle y_i \overline{A}_i + D_S = 0, \\ D_X + D_S = V^{-1} - V = -\nabla \Psi_l(V). \end{cases} \tag{40}$$

Nous pouvons dire que $Tr(D_X D_S) = 0$, qui provient des première et seconde équations de (40) ou à partir de l'orthogonalité de $\triangle X$ et $\triangle S$. La fonction noyau classique est définie comme suit :

$$\psi_l(t) = \frac{1}{2}(t^2 - 1) - \log t.$$

4.1.3 Algorithme générique de points intérieurs pour (SDO)

On appelle $\psi_l(t)$ la fonction noyau de la fonction barrière logarithmique $\Psi_l(V)$. Dans ce chapitre on remplace $\psi_l(t)$ par une nouvelle fonction $\psi(t)$ qui est défini dans la section suivante et supposons que $\tau \geq 1$.

L'algorithme proposé dans ce chapitre se déroule comme suit : supposons que l'on donne un point strictement réalisable (X, y, S) qui est dans un τ-voisinage de μ-centre. Ensuite, nous diminuons μ à $\mu^+ = (1 - \theta)\mu$, pour $0 < \theta < 1$ puis on résout le système de Newton (40) pour obtenir la direction de descente. La condition de positivité d'une nouvelle itération est assurée avec le bon choix de la taille du pas α qui est défini dans la section suivante. Cette procédure est répétée jusqu'à ce qu'on trouve une nouvelle itération (X^+, y^+, S^+) qui est dans un τ-voisinage de μ^+-centre. Ce processus est répété jusqu'à ce que μ est suffisamment petit, i.e., $\mu \leq \frac{\varepsilon}{n}$.

Les paramètres τ, θ et le pas de déplacement α doivent être choisie de telle sorte que l'algorithme est optimisée dans le sens où le nombre d'itérations requises est le plus petit possible. Le choix de paramètre θ joue un rôle important dans la théorie et la pratique des $IPMs$. Si θ est une constante indépendante de la dimension n du problème, par exemple $\theta = \frac{1}{2}$, nous appelons l'algorithme de grand pas. Si θ

dépend de la dimension du problème, telles que $\theta = \frac{1}{\sqrt{n}}$, alors l'algorithme est appelé algorithme de petit pas.

L'algorithme générique de points intérieurs primal-dual pour (SDO) est donnée comme suit :

Primal-Dual IPM **pour** (SDO)

Début Algorithme

Initialisation

$\varepsilon > 0$ un paramètre de précision

un paramètre θ, $0 < \theta < 1$

un paramètre de seuil $\tau, 0 < \tau < 1$,

une solution initiale strictement réalisable (X^0, y^0, S^0) et $\mu^0 = \frac{Tr(X^0 S^0)}{n}$ tels que $\Psi(X^0 S^0, \mu^0) \leq \tau$.

$k = 0$

$X := X^0, S := S^0, \mu := \mu^0,$

Tant que $(\mu \geq \frac{\varepsilon}{n})$ **faire**

$\mu = (1 - \theta)\mu$

Tant que $(\Psi(V) > \tau)$ **faire**

Résoudre le système (40) pour obtenir $(\triangle X, \triangle y, \triangle S)$,

Déterminer le pas de déplacement α

Prendre

$X := X + \alpha \triangle X$

$y := y + \alpha \triangle y$

$S := S + \alpha \triangle S$

Fin Tant que.

Fin Tant que.

Fin Algorithme.

Dans la section suivante, nous définissons une nouvelle fonction noyau et donner ses propriétés qui sont essentielles à notre analyse de la complexité.

4.2 Fonction noyau et ses propriétés

Soit $\psi : \Re_{++} \rightarrow \Re_{+}$ est une fonction noyau, si ψ est deux fois dérivable et satisfait les conditions suivantes [6] :

$$\begin{aligned} &\psi^{'}(1) = \psi(1) = 0, \\ &\psi^{''}(t) > 0. \end{aligned} \tag{41}$$

Dans cette étude, nous définissons une nouvelle fonction noyau suivante :

$$\psi(t) = (m+1)t^2 - (m+2)t + \frac{1}{t^m}, \text{ pour } t > 0, \tag{42}$$

où $m > 4$.

Alors, on a les dérivées successives de la fonction ψ :

$$\begin{aligned} &\psi^{'}(t) = 2(m+1)t - (m+2) - mt^{-m-1}, \\ &\psi^{''}(t) = 2(m+1) - m(-m-1)t^{-m-2}, \\ &\psi^{'''}(t) = -m(-m-1)(-m-2)t^{-m-3}. \end{aligned} \tag{43}$$

D'après les conditions (41), $\psi(t)$ est bien définie.

$$\psi^{''}(t) > 2(m+1), \text{ pour } t > 0. \tag{44}$$

On remplace la fonction $\Psi_l(V)$ de (40) par la fonction $\Psi(V)$ suivante :

$$D_X + D_S == -\nabla\Psi(V) \tag{45}$$

où $\Psi(V) = Tr(\psi(V)) = \sum_{i=1}^{n} \psi(\lambda_i(V))$, $\psi(t)$ définie dans (42). Ainsi, la direction de descente $(\triangle X, \triangle y, , \triangle S)$ est obtenue à l'aide de la résolution du système suivant :

$$\begin{cases} Tr(A_i \triangle X) = 0, i = 1, ..., m, \\ \sum\limits_{i=1}^{m} \triangle y_i A_i + \triangle S = 0, \\ X\triangle S + \triangle X S = -\mu V \nabla\Psi(V). \end{cases} \tag{46}$$

On a

$$D_X = D_S = 0 \Leftrightarrow \nabla\Psi(V) = 0 \Leftrightarrow V = E \Leftrightarrow \nabla\Psi(V) = 0 \Leftrightarrow X = X(\mu), S = S(\mu). \tag{47}$$

Nous utilisons $\Psi(V)$ comme fonction de proximité pour mesurer la distance entre l'itération en cours et μ-centrale pour $\mu > 0$. Nous définissons aussi la mesure de proximité basé sur la norme, $\delta(V) : \Re_{++} \rightarrow \Re_{+}$, comme suit :

$$\delta(XS, \mu) = \delta(V) = \frac{1}{2} \left\| \nabla\Psi(V) \right\| = \frac{1}{2} \left\| D_X + D_S \right\|. \tag{48}$$

Lemme 37 *Pour une fonction de noyau $\psi(t)$, $t > 0$, nous avons ce qui suit.*

(i) $\psi(t)$ est exponentielle convexe pour tout $t > 0$,

(ii) $\psi^{''}(t)$ est monotone décroissante pour tout $t > 0$,

(iii) $t\psi^{''}(t) - \psi^{'}(t) > 0$, pour tout $t > 0$.

Preuve. Pour (i), par le Lemme 2.1.2 dans [63], il suffit de montrer que la fonction $\psi(t)$ satisfait $t\psi^{''}(t) + \psi^{'}(t) \geq 0$, pour $t > 0$. En utilisant (43), on a

$$\begin{aligned} t\psi^{''}(t) + \psi^{'}(t) &= t\left(2(m+1) - m(-m-1)t^{-m-2}\right) + \left(2(m+1)t - (m+2) - mt^{-m-1}\right) \\ &= 4(m+1)t + m^2t^{-m-1} - (m+2). \end{aligned}$$

Soit

$$g(t) = 4(m+1)t + m^2t^{-m-1} - (m+2).$$

Alors

$$g^{'}(t) = 4(m+1) - m^2(m+1)t^{-m-2}$$

$$g^{''}(t) = m^2(m+1)(m+2)t^{-m-3} > 0, \text{ pour } t > 0.$$

Soit $g^{'}(t) = 0$, nous obtenons $t = (\frac{m^2}{4})^{\frac{1}{m+2}}$. Puisque $g(t)$ est strictement convexe et a un minimum global, $g((\frac{m^2}{4})^{\frac{1}{m+2}}) > 0$, et par le Lemme 2.1.2 dans [67], on déduit que $\psi(t)$ est exponentiellement convexe

Pour (ii), en utilisant (43), on a $\psi^{'''}(t) > 0$, d'où le résultat

Pour (iii), en utilisant (43), on a

$$t\psi^{''}(t) - \psi^{'}(t) = m(m+2)t^{-m-1} + (m+2) > 0, \text{ pour } t > 0.$$

D'où le résultat. ■

Lemme 38 *Pour une fonction de noyau $\psi(t)$, $t > 0$, nous avons ce qui suit.*

$$(m+1)(t-1)^2 \leq \psi(t) \leq \frac{1}{4(m+1)}\psi^{'}(t)^2, \text{ pour } t > 0, \tag{49}$$

$$\psi(t) \leq \frac{(m+1)(m+2)}{2}(t-1)^2, \text{ pour } t \geq 1. \tag{50}$$

Preuve. Pour (49), en utilisant (41) et (44), on a

$$\psi(t) = \int_1^t \int_1^\xi \psi^{''}(\zeta)d\zeta d\xi \geq 2(m+1)\int_1^t \int_1^\xi d\zeta d\xi = (m+1)(t-1)^2,$$

et on a aussi,

$$\begin{aligned}
\psi(t) &= \int_1^t \int_1^\xi \psi^{''}(\zeta)d\zeta d\xi \\
&\leq \frac{1}{2(m+1)}\int_1^t \int_1^\xi \psi^{''}(\xi)\psi^{''}(\zeta)d\zeta d\xi \\
&= \frac{1}{2(m+1)}\int_1^t \psi^{''}(\xi)\psi^{'}(\xi)\ d\xi \\
&= \frac{1}{2(m+1)}\int_1^t \psi^{'}(\xi)\ d(\psi^{'}(\xi)) \\
&= \frac{1}{4(m+1)}\psi^{'}(t)^2.
\end{aligned}$$

Pour (50), en utilisant le théorème de Taylor, nous avons

$$\begin{aligned}
\psi(t) &= \psi(1) + \psi^{'}(1)(t-1) + \frac{1}{2}\psi^{''}(1)(t-1)^2 + \frac{1}{6}\psi^{'''}(\xi)(\xi-1)^3 \\
&= \frac{1}{2}\psi^{''}(1)(t-1)^2 + \frac{1}{6}\psi^{'''}(\xi)(\xi-1)^3 \\
&\leq \frac{1}{2}\psi^{''}(1)(t-1)^2 \\
&= \frac{(m+1)(m+2)}{2}(t-1)^2.
\end{aligned}$$

D'où le résultat. ■

Maintenant, nous définissons $\gamma : (0, \infty) \to (1, \infty)$, la fonction inverse de $\psi(t)$ pour $t \geq 1$,

et $\rho : (0, \infty) \to (0, 1)$, la fonction inverse de $-\frac{1}{2}\psi^{'}(t)$ pour $t \in (0, 1)$. Alors nous avons le lemme suivant.

Lemme 39 *Pour une fonction de noyau $\psi(t)$, $t > 0$, nous avons ce qui suit.*

$$\sqrt{\frac{s}{m+1} + 1} \leq \gamma(s) \leq 1 + \sqrt{\frac{s}{m+1}}, \ s \geq 0, \tag{51}$$

et

$$\rho(s) \geq (\frac{m}{2s+m})^{\frac{1}{m+1}}, \ s \geq 0. \tag{52}$$

Preuve. Pour (51), on pose $s = \psi(t)$, $t \geq 1$, i.e., $\gamma(s) = t, t \geq 1$, alors on a

$$(m+1)t^2 = s + (m+2)t - t^{-m}.$$

à fin que $(m+2)t - t^{-m}$ soit monotone croissante par rapport à $t \geq 1$, on a

$$(m+1)t^2 \geq s + m + 1,$$

ce qui implique que

$$t = \gamma(s) \geq \sqrt{\frac{s}{m+1} + 1}.$$

De (42), on a

$$s = \psi(t) \geq (m+1)(t-1)^2,$$

alors

$$t = \gamma(s) \leq 1 + \sqrt{\frac{s}{m+1}}.$$

Pour (52), on pose $z = -\frac{1}{2}\psi^{'}(t)$ pour $t \in (0, 1)$.Par la définition de ρ, on a $\rho(z) = t$ et $2z = -\psi^{'}(t)$. Alors

$$mt^{-m-1} = 2z + 2(m+1)t - (m+2),$$

à fin que $2(m+1)t-(m+2)$ soit monotone croissante par rapport à $t \in (0,1)$, on a

$$mt^{-m-1} \leq 2z+m,$$

ce qui implique que

$$\rho(z) = t \geq (\frac{m}{2z+m})^{\frac{1}{m+1}}.$$

D'où le résultat. ■

Lemme 40 *Soit* $\gamma : (0,\infty) \to (1,\infty)$,*la fonction inverse de* $\psi(t)$ *pour* $t \geq 1$. *Alors on a*

$$\Psi(\beta V) \leq n\psi\left(\beta\gamma\left(\frac{\Psi(V)}{n}\right)\right),\ V \in,\ \beta \geq 1. \tag{53}$$

Preuve. En utilisant le théorème 3.2 dans [6], nous obtenons le résultat. ■

Lemme 41 *Soit* $0 \leq \theta \leq 1,\ V^+ = \frac{1}{\sqrt{1-\theta}}V$. *Si* $\Psi(V) \leq \tau$, *alors on a*

$$\Psi(V^+) \leq \frac{(m+1)(m+2)}{2(1-\theta)}\left(\sqrt{n}\theta + \sqrt{\frac{\tau}{m+1}}\right)^2. \tag{54}$$

Preuve. Lorsque $\frac{1}{\sqrt{1-\theta}} \geq 1$ et $\gamma\left(\frac{\Psi(V)}{n}\right) \geq 1$, on a $\frac{\gamma\left(\frac{\Psi(V)}{n}\right)}{\sqrt{1-\theta}} \geq 1$.

En utilisant le Lemme 42, avec $\beta = \sqrt{1-\theta}$, (49), (50) et $\Psi(V) \leq \tau$, on a

$$\begin{aligned}
\Psi(V^+) &\leq n\psi\left(\frac{1}{\sqrt{1-\theta}}\gamma\left(\frac{\Psi(V)}{n}\right)\right), \\
&\leq n\frac{(m+1)(m+2)}{2}\left(\frac{1}{\sqrt{1-\theta}}\gamma\left(\frac{\Psi(V)}{n}\right)-1\right)^2, \\
&= n\frac{(m+1)(m+2)}{2(1-\theta)}\left(\gamma\left(\frac{\Psi(V)}{n}\right)-\sqrt{1-\theta}\right)^2, \\
&\leq n\frac{(m+1)(m+2)}{2(1-\theta)}\left(1+\sqrt{\frac{\Psi(V)}{(m+1)n}}-\sqrt{1-\theta}\right)^2, \\
&\leq n\frac{(m+1)(m+2)}{2(1-\theta)}\left(\theta+\sqrt{\frac{\tau}{(m+1)n}}\right)^2, \\
&\leq n\frac{(m+1)(m+2)}{2(1-\theta)}\left(\sqrt{n}\theta+\sqrt{\frac{\tau}{(m+1)n}}\right)^2.
\end{aligned}$$

D'où le résultat. ■

On note par

$$\Psi_0 = L(n, \theta, \tau) = n\frac{(m+1)(m+2)}{2(1-\theta)}\left(\sqrt{n}\theta + \sqrt{\frac{\tau}{(m+1)n}}\right)^2, \tag{55}$$

alors Ψ_0 est une borne supérieure pour $\Psi(V)$ pendant le processus de l'algorithme.

4.3 Convergence de l'algorithme

Le but de ce chapitre est de définir une nouvelle fonction noyau et d'obtenir des nouveaux résultats pour la complexité d'un problème (SDO), en utilisant la fonction de proximité défini par la fonction noyau. En utilisant le concept d'une fonction de matrice [9], la définition de la fonction noyau peut être étendue à toute matrice diagonalisable de valeurs propres positives. En particulier, pour une décomposition de la matrice V donnée par

$$V = Q_V^{-1}\, diag\left(\lambda_1(V), \lambda_2(V), ..., \lambda_n(V)\right) Q_V,$$

Q_V une matrice non singulière, la fonction de matrice $\psi(V)$ est définie par

$$\psi(V) = Q_V^{-1}\, diag\left(\psi(\lambda_1(V)), \psi(\lambda_2(V)), ..., \psi(\lambda_n(V))\right) Q_V. \tag{56}$$

Dans cette section, nous calculons un pas de déplacement α efficace et la diminution de la fonction de proximité au cours d'une itération interne et améliorer les résultats de la complexité de l'algorithme, pour $\mu > 0$.

4.3.1 Le pas de déplacement

Soit α est le pas de déplacement, le nouvel itéré est donné comme suit

$$X^+ = X + \alpha\triangle X,\ y^+ = y + \alpha\triangle y \text{ et } S^+ = S + \alpha\triangle S.$$

Soit

$$X^+ = X\left(I + \alpha\frac{\triangle X}{X}\right) = X\left(I + \alpha\frac{D_X}{V}\right) = \frac{X}{V}\left(V + \alpha D_X\right),$$

$$S^+ = S\left(I + \alpha\frac{\triangle S}{S}\right) = S\left(I + \alpha\frac{D_S}{V}\right) = \frac{S}{V}\left(V + \alpha D_S\right).$$

Alors, on a

$$V^+ = \left(\left(V + \alpha D_X\right)^{\frac{1}{2}}\left(V + \alpha D_S\right)\left(V + \alpha D_X\right)^{\frac{1}{2}}\right)^{\frac{1}{2}}.$$

Après une étape la proximité est définie par :

$$\Psi(V^+) = \Psi\left(\left(\left(V + \alpha D_X\right)^{\frac{1}{2}}\left(V + \alpha D_S\right)\left(V + \alpha D_X\right)^{\frac{1}{2}}\right)^{\frac{1}{2}}\right).$$

Par (i) dans le Lemme (38), on a

$$\begin{aligned}\Psi(V^+) &= \Psi\left(\left(\left(V + \alpha D_X\right)\left(V + \alpha D_S\right)\right)^{\frac{1}{2}}\right), \\ &\leq \frac{1}{2}\left(\Psi\left(V + \alpha D_X\right) + \Psi(V + \alpha D_S)\right).\end{aligned}$$

Définir, pour $\alpha > 0$

$$f(\alpha) = \Psi(V^+) - \Psi(V).$$

on a $f(\alpha) \leq f_1(\alpha)$, où

$$f_1(\alpha) = \frac{1}{2}\left(\Psi\left(V + \alpha D_X\right) + \Psi(V + \alpha D)\right) - \Psi(V). \tag{57}$$

évidemment,

$$f(0) = f_1(0) = 0.$$

Maintenant, nous traitons avec d'autre concepts pertinents aux fonctions matricielles dans la théorie des matrices [41].

Définition 42 *[41] Une matrice $X(t)$ est une matrice de fonctions si chaque entrée de $X(t)$ est une fonction de t, c'est-à-dire, $X(t) = [X_{ij}(t)]$.*

Les notions de continuité, dérivabilité et intégrabilité s'étend naturellement aux fonctions de matrice d'une valeur, d'un scalaire en interprétant les composants sage. Ainsi, nous pouvons dire que

$$\frac{d}{dt}X(t) = \frac{d}{dt}\left[X_{ij}(t)\right] = X^{'}(t).$$

Supposons que la matrice des fonctions de valeur $H(t)$ et $G(t)$ sont différentiables par rapport à t. Alors on a

$$\frac{d}{dt}\left(Tr(G(t)\right) = Tr\left(\frac{d}{dt}G(t)\right) = Tr\left(G^{'}(t)\right),$$

$$\frac{d}{dt}\left((G(t)H(t)\right) = G^{'}(t)H(t) + (G(t)H^{'}(t).$$

Pour chaque fonction $\psi(t)$, notons par $\triangle\psi(t)$ la différence divisée de $\psi(t)$:

$$\triangle\psi(t_1, t_2) = \frac{\psi(t_1) - \psi(t_2)}{t_1 - t_2},\ \forall t_1 \neq t_2 \in \Re^*.$$

Si $t_1 = t_2$, nous écrivons simplement $\triangle\psi(t_1, t_2) = \psi^{'}(t)$.

Nous allons définir que Q_α est la matrice orthogonale de telle sorte que

$$V + \alpha D_X = Q_\alpha^T\ diag\left(\lambda_1(V + \alpha D_X), \lambda_2(V + \alpha D_X), ..., \lambda_n(V + \alpha D_X)\right) Q_\alpha$$

et notons aussi par D_i une matrice identité. Il résulte de [67][41][67] que

$$\frac{d}{d\alpha}\left(\psi(V + \alpha D_X)\right) = Q_\alpha^T\left(\sum_{j,k=1}^{n} \triangle\psi(\lambda_j(V + \alpha D_X), \lambda_k(V + \alpha D_X)) D_j\left(Q_\alpha(V + \alpha D_X)^{'} Q_\alpha^T\right) D_k\right) Q_\alpha. \tag{58}$$

Maintenant par le choix de D_i, il contient $Tr\left(D_j\left(Q_\alpha(V + \alpha D_X)^{'} Q_\alpha^T\right) D_k\right) = 0$, pour $j \neq k$. Il s'ensuit donc que

$$\begin{aligned}\frac{d}{d\alpha}\left(\psi(V+\alpha D_X)\right) &= Tr\left(\sum_{i=1}^{n}\psi^{'}(\lambda_i(V+\alpha D_X))D_i\left(Q_\alpha(V+\alpha D_X)^{'}Q_\alpha^T\right)D_i\right),\\ &= Tr\left(Q_\alpha^T\left(\sum_{i=1}^{n}D_i\psi^{'}(\lambda_i(V+\alpha D_X))D_i\right)Q_\alpha(V+\alpha D_X)^{'}\right),\\ &= Tr\left(\psi^{'}(V+\alpha D_X)D_X\right).\end{aligned}$$

Et

$$\frac{d}{d\alpha}\left(\psi(V+\alpha D_S)\right) = Q_\alpha^T\left(\sum_{j,k=1}^{n}\triangle\psi(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))D_j\left(Q_\alpha(V+\alpha D_S)^{'}Q_\alpha^T\right)D_k\right)Q_\alpha. \tag{59}$$

Et $Tr\left(D_j\left(Q_\alpha(V+\alpha D_S)^{'}Q_\alpha^T\right)D_k\right)=0$, pour $j\neq k$. Il s'ensuit donc que

$$\begin{aligned}\frac{d}{d\alpha}\left(\psi(V+\alpha D_S)\right) &= Tr\left(\sum_{i=1}^{n}\psi^{'}(\lambda_i(V+\alpha D_S))D_i\left(Q_\alpha(V+\alpha D_S)^{'}Q_\alpha^T\right)D_i\right),\\ &= .Tr\left(Q_\alpha^T\left(\sum_{i=1}^{n}D_i\psi^{'}(\lambda_i(V+\alpha D_S))D_i\right)Q_\alpha(V+\alpha D_S)^{'}\right),\\ &= Tr\left(\psi^{'}(V+\alpha D_S)D_S\right).\end{aligned}$$

Maintenant, nous pouvons écrire

$$f_1^{'}(\alpha)=\frac{1}{2}Tr\left(\psi^{'}\left(V+\alpha D_X\right)D_X\right)+\psi^{'}(V+\alpha D_S)D_S\Big).$$

On obtient

$$f_1^{'}(0)=-2\delta^2.$$

En outre,

$\frac{d^2}{d\alpha^2}\left(Tr\left(\psi\left(V+\alpha D_X\right)\right)\right)=Tr\left(\frac{d}{d\alpha}\left(\psi^{'}\left(V+\alpha D_X\right)D_X\right)\right),$

$$= Tr\left(Q_\alpha^T\left(\sum_{j,k=1}^{n}\triangle\psi^{'}(\lambda_j(V+\alpha D_X),\lambda_k(V+\alpha D_X))D_j\left(Q_\alpha(V+\alpha D_X)^{'}Q_\alpha^T\right)D_k\right)Q_\alpha D_X\right),$$

$$= Tr\left(\sum_{j,k=1}^{n} \triangle\psi^{'}(\lambda_j(V+\alpha D_X),\lambda_k(V+\alpha D_X))D_j(Q_\alpha D_X Q_\alpha^T)D_k(Q_\alpha D_X Q_\alpha^T)\right),$$
$$= Tr\left(\sum_{j,k=1}^{n} \triangle\psi^{'}(\lambda_j(V+\alpha D_X),\lambda_k(V+\alpha D_X))(Q_\alpha D_X Q_\alpha^T)^2_{jk}\right),$$
$$\leq \max\left\{\left|\triangle\psi^{'}(\lambda_j(V+\alpha D_X),\lambda_k(V+\alpha D_X))\right|,\ j,k=1,...,n\right\}\|D_X\|^2.$$

Aussi, on a

$$\tfrac{d^2}{d\alpha^2}\left(Tr\left(\psi\left(V+\alpha D_S\right)\right)\right) = Tr\left(\tfrac{d}{d\alpha}\left(\psi^{'}\left(V+\alpha D_S\right)D_S\right)\right),$$
$$= Tr\left(Q_\alpha^T\left(\sum_{j,k=1}^{n} \triangle\psi^{'}(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))D_j\left(Q_\alpha(V+\alpha D_S)^{'}Q_\alpha^T\right)D_k\right)Q_\alpha D_S\right),$$
$$= Tr\left(\sum_{j,k=1}^{n} \triangle\psi^{'}(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))D_j(Q_\alpha D_S Q_\alpha^T)D_k(Q_\alpha D_S Q_\alpha^T)\right),$$
$$= Tr\left(\sum_{j,k=1}^{n} \triangle\psi^{'}(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))(Q_\alpha D_S Q_\alpha^T)^2_{jk}\right),$$
$$\leq \max\left\{\left|\triangle\psi^{'}(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))\right|,\ j,k=1,...,n\right\}\|D_S\|^2.$$

On note par

$$\omega_1 = \max\left\{\left|\triangle\psi^{'}(\lambda_j(V+\alpha D_X),\lambda_k(V+\alpha D_X))\right|,\ j,k=1,...,n\right\},$$

et

$$\omega_2 = \max\left\{\left|\triangle\psi^{'}(\lambda_j(V+\alpha D_S),\lambda_k(V+\alpha D_S))\right|,\ j,k=1,...,n\right\}.$$

Par conséquent,

$$\begin{aligned} f^{''}(\alpha) &= \tfrac{1}{2}\tfrac{d^2}{d\alpha^2}Tr\left(\psi\left(V+\alpha D_X\right)+\psi\left(V+\alpha D_S\right)\right), \\ &\leq \tfrac{1}{2}\left(\omega_1\|D_X\|^2+\omega_2\|D_S\|^2\right). \end{aligned} \tag{60}$$

Lemme 43 *Soit $\delta(V)$ celui défini au* (48)*. Alors on a*

$$\delta(V) \geq \sqrt{(m+1)\Psi(V)}.$$

Preuve. En utilisant (49), on a

$$\begin{aligned}\Psi(V) &= Tr(\psi(V)),\\ &= \sum_{i=1}^{n}\psi(\lambda_i(V)),\\ &\leq \sum_{i=1}^{n}\frac{1}{4(m+1)}\psi^{'}(\lambda_i(V))^2,\\ &= \frac{1}{4(m+1)}\left\|\nabla\Psi(V)\right\|^2,\\ &= \frac{1}{m+1}\delta(V)^2.\end{aligned}$$

on obtient

$$\delta(V) \geq \sqrt{(m+1)\Psi(V)}.$$

D'où le résultat. ■

Nous supposons que $\tau \geq 1$. En utilisant le Lemme (45) et l'hypothèse que $\Psi(V) \geq \tau$, on a $\delta(V) \geq \sqrt{(m+1)}$.

Nous avons le lemme suivant.

Lemme 44 *Soit $f_1(\alpha)$ celui défini au (57), $\delta(V)$ celui défini au (48) et ψ est la fonction noyau défini au (42). Alors on a*

$$f_1^{''}(\alpha) \leq 2\delta^2\psi^{''}(\lambda_{\min}(V) - 2\alpha\delta).$$

Preuve. On a

$$f_1^{''}(\alpha) \leq 2\max\left\{\omega_1, \omega_2\right\}\delta^2.$$

Il suffit de montrer l'inégalité suivante :

$$\max\left\{\omega_1, \omega_2\right\} \leq \psi^{''}(\lambda_{\min}(V) - 2\alpha\delta).$$

Nous pouvons choisir $j^*, k^* \in \{1, 2, ..., n\}$ tel que

$$\omega_1 = \left|\frac{\psi^{'}(\lambda_{j^*}(V+\alpha D_X) - \psi^{'}(\lambda_{k^*}(V+\alpha D_X)}{\lambda_{j^*}(V+\alpha D_X) - \lambda_{k^*}(V+\alpha D_X)}\right|.$$

En utilisant le théorème de la valeur moyenne, il existe

$$\eta \in [\min(\lambda_{j^*}(V+\alpha D_X), \lambda_{k^*}(V+\alpha D_X)),\ \max(\lambda_{j^*}(V+\alpha D_X), \lambda_{k^*}(V+\alpha D_X))],$$

tel que

$$\psi^{''}(\eta) = \triangle\psi^{'}(\lambda_{j^*}(V+\alpha D_X), \lambda_{k^*}(V+\alpha D_X)).$$

Lorsque D_X est une matrice symétriques, à partir de la définition de δ et norme de Frobenius, nous avons

$$\begin{aligned} \eta &\geq \min\{\lambda_{j^*}(V+\alpha D_X), \lambda_{k^*}(V+\alpha D_X)\}, \\ &\geq \lambda_{\min}(V) - 2\alpha\delta, \end{aligned}$$

à fin que, $\psi^{''}$ soit monotone décroissante, on obtient

$$\omega_1 \leq \psi^{''}(\lambda_{\min}(V) - 2\alpha\delta).$$

Par la même méthode pour obtenir ω_1, on obtient aussi

$$\omega_2 = \left| \frac{\psi^{'}(\lambda_{j^*}(V+\alpha D_S) - \psi^{'}(\lambda_{k^*}(V+\alpha D_S)}{\lambda_{j^*}(V+\alpha D_S) - \lambda_{k^*}(V+\alpha D_S)} \right|.$$

En utilisant le théorème de la valeur moyenne, il existe

$$\eta \in [\min(\lambda_{j^*}(V+\alpha D_S), \lambda_{k^*}(V+\alpha D_S)),\ \max(\lambda_{j^*}(V+\alpha D_S), \lambda_{k^*}(V+\alpha D_S))],$$

tel que

$$\psi^{''}(\eta) = \triangle\psi^{'}(\lambda_{j^*}(V+\alpha D_S), \lambda_{k^*}(V+\alpha D_S)).$$

Lorsque D_S est une matrice symétriques, à partir de la définition de δ et norme de Frobenius, nous avons

$$\begin{aligned} \eta &\geq \min\{\lambda_{j^*}(V+\alpha D_S), \lambda_{k^*}(V+\alpha D_S)\}, \\ &\geq \lambda_{\min}(V) - 2\alpha\delta, \end{aligned}$$

à fin que, $\psi^{''}$ est monotone décroissante, on obtient

$$\omega_2 \leq \psi^{''}(\lambda_{\min}(V) - 2\alpha\delta).$$

Par conséquent,

$$\max\{\omega_1, \omega_2\} \leq \psi''(\lambda_{\min}(V) - 2\alpha\delta).$$

on obtient

$$f_1''(\alpha) \leq 2\delta^2\psi''(\lambda_{\min}(V) - 2\alpha\delta).$$

D'où le résultat. ■

puisque $f_1(0) = 0$ et $f_1'(0\ ) = -2\delta(V)^2$, on a

$$\begin{aligned} f(\alpha) &\leq f_1(\alpha) := f_1(0) + f_1'(0)\alpha + \int_0^\alpha \int_0^\xi f_1''(\zeta)d\zeta d\xi, \\ &\leq f_2(\alpha) := f_1(0) + f_1'(0)\alpha + 2\delta^2 \int_0^\alpha \int_0^\xi \psi''(\lambda_{\min}(V) - 2\zeta\delta)d\zeta d\xi. \end{aligned}$$

On remarque que $f_2(0) = 0$. Alors

$$f_2'(\alpha) = -2\delta^2 + \delta\left(\psi'(\lambda_{\min}(V)) - \psi'(\lambda_{\min}(V) - 2\alpha\delta)\right).$$

on a $f_2'(0) = -2\delta^2$, qui est la même valeur de $f_1'(0)$ et $f_2''(\alpha) = 2\delta^2\psi''(\lambda_{\min}(V) - 2\alpha\delta)$, ce qui augmente pour $\alpha \in \left[0, \frac{\lambda_{\min}(V)}{2\delta}\right]$. Ainsi, nous pouvons réécrire $f_2(\alpha)$ comme suit :

$$f_2(\alpha) = f_2(0) + f_2'(0)\alpha + 2\delta^2 \int_0^\alpha \int_0^\xi \psi''(\lambda_{\min}(V) - 2\zeta\delta)d\zeta d\xi.$$

Maintenant, en utilisant $f_1'(0) = f_2'(0)$ et $f_1''(\alpha) \leq f_2''(\alpha)$, on peut facilement vérifier que

$$f_1'(\alpha) = f_1'(0) + \int_0^\alpha f''(\xi)d\xi \leq f_2'(\alpha).$$

Cela donne que

$$f_1^{'}(\alpha) \leq 0, \text{ si } \ f_2^{'}(\alpha) \leq 0.$$

Pour calculer le pas de déplacement α telle que la mesure de proximité soit décroit quand nous prenons une nouvelle itération pour $\mu > 0$. Nous voulons calculer le pas de déplacement α qui s'assure que $f_2^{'}(\alpha) \leq 0$ tient avec α le plus grand possible. Puisque $f_2^{''}(\alpha) > 0$, c'est-à-dire $f_2^{'}(\alpha)$ est monotone croissante à α, la plus grande valeur possible à α satisfaisant $f_2^{'}(\alpha) \leq 0$ se produit lorsque $f_2^{'}(\alpha) = 0$, c'est-à-dire

$$\psi^{'}(\lambda_{\min}(V)) - \psi^{'}(\lambda_{\min}(V) - 2\alpha\delta) = 2\delta \tag{61}$$

Lorsque $\psi^{''}(t)$ est monotone décroissante, la dérivée de la partie gauche de (61) par rapport à $\lambda_{\min}(V)$ est

$$\psi^{''}(\lambda_{\min}(V)) - \psi^{''}(\lambda_{\min}(V) - 2\alpha\delta) \leq 0.$$

Ainsi, le membre gauche de (61) diminue à $\lambda_{\min}(V)$. Cela implique que si $\lambda_{\min}(V)$ devient plus petit, alors α devient plus petit avec δ. On note que

$$\begin{aligned} \delta &= \sqrt{\sum_{i=1}^{n} \left(\psi^{'}(\lambda_i(V))\right)^2} \\ &\geq \left|\psi^{'}(\lambda_{\min}(V))\right| \\ &\geq -\psi^{'}(\lambda_{\min}(V)), \end{aligned}$$

et l'égalité est vraie si et seulement si $\lambda_{\min}(V)$ est les coordonnées de $(\lambda_1(V), \lambda_2(V), ..., \lambda_n(V))$ sont différent de 1 et $\lambda_{\min}(V) < 1$, c'est-à-dire, $\psi^{'}(\lambda_{\min}(V)) < 0$.

Par conséquent, la pire situation pour la plus grande valeur de α se produit lorsque $\lambda_{\min}(V)$ satisfait

$$-\psi^{'}(\lambda_{\min}(V)) = \delta \tag{62}$$

Pour notre propos, nous devons faire face à la pire des cas et si nous supposons que (62) est vérifie. Cela implique

$$\lambda_{\min}(V) = \rho(\delta) \tag{63}$$

En utilisant (61) et (62) on obtient immédiatement

$$-\psi^{'}(\lambda_{\min}(V) - 2\alpha\delta) = 4\delta.$$

Par la définition de ρ et l'utilisation de (63), la valeur de pas maximal est donnée comme suit :

$$\alpha^* = \frac{\rho(\delta) - \rho(2\delta)}{2\delta}. \tag{64}$$

Lemme 45 *On considère la définition de*ρ *et* α^* *celui défini au* (64), *alors on a*

$$\alpha^* \geq \frac{1}{(m+1)(m+2)^{\frac{m+2}{m+1}}}.$$

Preuve. En utilisant le Lemme 4.4 in [6], la définition de $\psi^{''}(t)$ et (52), on a

$$\begin{aligned}
\alpha^* &\geq \frac{1}{\psi^{''}(\rho(2\delta))}, \\
&= \frac{1}{2(m+1) + \frac{m(m+1)}{\rho(2\delta)^{m+2}}}, \\
&\geq \frac{1}{2(m+1) + m(m+1)(\frac{4\delta+m}{m})^{\frac{m+2}{m+1}}}, \\
&\geq \frac{1}{2(m+1)\delta + 3m(m+2)\delta^{\frac{m+2}{m+1}}}, \\
&\geq \frac{1}{3(m+1)(m+2)\delta^{\frac{m+2}{m+1}}}.
\end{aligned}$$

D'où le résultat.. ■

Pour utiliser $\overline{\alpha}$ comme un pas de déplacement par defaut dans l'algorithme, nous définissons $\overline{\alpha}$ comme suit :

$$\overline{\alpha} = \frac{1}{3(m+1)(m+2)\delta^{\frac{m+2}{m+1}}}. \tag{65}$$

4.3.2 Diminution de la fonction de proximité

Maintenant, on montre que la fonction de proximité Ψ avec le pas de déplacement $\overline{\alpha}$ est diminue. Alors on a le résultat suivant :

Lemme 46 *[66] Soit $h(t)$ une fonction deux fois différentiable convexe, avec $h(0) = 0,\ h^{'}(0) < 0$ et soit $h(t)$ atteint son minimum (globale) au $t > 0$. Si $h^{''}(t)$ augmente pour $t \in [0, t^*]$, alors*

$$h(t) = \frac{th^{'}(0)}{2}.$$

Soit la fonction h tel que

$$h(0) = f_1(0) = 0,\ h^{'}(0) = f_1^{'}(0) = -2\delta^2,\ h^{''}(\alpha) = 2\delta^2\psi^{''}(\lambda_{\min}(V) - 2\alpha\delta).$$

Lorsque $f_2(\alpha)$ maintient la condition du lemme ci-dessus, , alors :

$$f(\alpha) \leq f_1(\alpha) \leq f_2(\alpha) \leq \frac{f_2^{'}(0)}{2}\alpha,\ \text{pour } 0 \leq \alpha \leq \alpha^*.$$

Nous pouvons obtenir la limite supérieure de la valeur décroissante de la proximité dans l'itération interne par le lemme ci-dessus

Théorème 47 *Soit $\overline{\alpha}$ le pas de déplacement définie dans* (65) *et $\delta = \Psi(V) \geq \tau = 1$. Alors, on a*

$$f(\overline{\alpha}) \leq -\frac{(m+1)^{\frac{-m-2}{2(m+1)}}}{3(m+2)}\Psi(V)^{\frac{m}{2(m+1)}}.$$

Preuve. Pour tous $\overline{\alpha} \leq \alpha^*$, on a

$$\begin{aligned}
f(\overline{\alpha}) &\leq -\overline{\alpha}\delta^2, \\
&= -\frac{1}{3(m+1)(m+2)\delta^{\frac{m+2}{m+1}}}\delta^2, \\
&= -\frac{1}{3(m+1)(m+2)}\delta^{\frac{m}{m+1}}, \\
&\leq -\frac{1}{3(m+1)(m+2)}\left(\sqrt{(m+1)\Psi(V)}\right)^{\frac{m}{m+1}}, \\
&\leq -\frac{(m+1)^{\frac{m}{2(m+1)}}}{3(m+1)(m+2)}\Psi(V)^{\frac{m}{2(m+1)}}, \\
&\leq \frac{(m+1)^{\frac{-m-2}{2(m+1)}}}{3(m+2)}\Psi(V)^{\frac{m}{2(m+1)}}.
\end{aligned}$$

D'où le résultat. ■

4.3.3 Les bornes d'itération

Nous avons besoin de compter le nombre d'itérations internes nécessaires pour revenir à la situation où $\Psi(V) \leq \tau$ après une mise à jour de μ. On note Ψ_0 la valeur de $\Psi(V)$ après une mise à jour de μ ainsi les valeurs suivantes de la même itération extérieure notée $\Psi_k,\ k = 1, ...$ Si K désigne le nombre total d'itérations internes dans l'itération extérieure, puis nous avons

$$\Psi_0 \leq L = O(n, \theta, \tau),\ \Psi_{K-1} > \tau\,,\ 0 \leq \Psi_K \leq \tau.$$

et selon la (53),

$$\Psi_{k+1} \leq \Psi_k - \frac{(m+1)^{\frac{-m-2}{2(m+1)}}}{3(m+2)} \Psi_k^{\frac{m}{2(m+1)}}.$$

Alors, nous invoquons le lemme suivant du Lemme 14 en [66]

Lemme 48 *[66] Soient $t_0, t_1, ..., t_k$ une sequence de nombres positives telle que*

$$t_{k+1} \leq t_k - \beta t_k^{1-\nu},\ k = 0, 1, ..., K-1.$$

Où $\beta > 0,\ 0 < \nu \leq 1$, alors

$$K \leq \frac{t_0^\nu}{\beta\nu}.$$

Par consequent

$$t_k = \Psi_k,\ \beta = \frac{(m+1)^{\frac{-m-2}{2(m+1)}}}{3(m+2)} \text{ et } \nu = \frac{m+2}{2(m+1)},$$

Lemme 49 *Soit K le nombre total d'itérations internes dans l'itération extérieure. Alors, on a*

$$K \leq 6(m+1)^{\frac{3m+4}{2(m+1)}} \Psi_0^{\frac{m+2}{2(m+1)}}.$$

Preuve. En utilisant le Lemme 49, on obtient

$$K \leq \frac{\Psi_0^\nu}{\beta\nu} = 6(m+1)^{\frac{3m+4}{2(m+1)}} \Psi_0^{\frac{m+2}{2(m+1)}}.$$

D'où le résultat. ■

Maintenant, nous estimons le nombre total d'itérations de notre algorithme.

Théorème 50 *Si $\tau \geq 1$, le nombre total d'itérations est d'au plus*

$$6(m+1)^{\frac{3m+4}{2(m+1)}} \Psi_0^{\frac{m+2}{2(m+1)}} \frac{1}{\theta} \log \frac{n\mu^0}{\varepsilon}.$$

Preuve. Dans l'algorithme, $n\mu \leq \varepsilon,\ \mu^k = (1-\theta)^k \mu^0$ et $\mu^0 = \frac{Tr(X^0 S^0)}{n}$. Par simple calcul, nous avons

$$K \leq \frac{1}{\theta} \log \frac{n\mu^0}{\varepsilon}.$$

Par conséquent, le nombre d'itérations extérieures est majorée par $\frac{1}{\theta} \log \frac{n\mu^0}{\varepsilon}$.

multipliant le nombre d'itérations extérieures par le nombre d'itérations internes, nous obtenons une borne supérieure pour le nombre total d'itérations, à savoir,

$$6(m+1)^{\frac{3m+4}{2(m+1)}} \Psi_0^{\frac{m+2}{2(m+1)}} \frac{1}{\theta} \log \frac{n\mu^0}{\varepsilon}.$$

Pour l'algorithme de grand pas, on obtient $O(m^{\frac{3m+1}{2m}} n^{\frac{m+1}{2m}} \log \frac{Tr(X^0 S^0)}{\varepsilon})$ itérations et $O(m^{\frac{3m+1}{2m}} \sqrt{n} \log \frac{Tr(X^0 S^0)}{\varepsilon})$ pour petit pas.

D'où le résultat. ■

Conclusion

Les méthodes de points intérieurs sont connues par leur efficacité, rapidité de convergence, simplicité algorithmique et capacité de résoudre des problèmes de grandes tailles. L'inconvénient principal dans ce type de méthodes est l'initialisation, c'est-à-dire la détermination d'un point initial qui se trouve à l'intérieur du domaine. La plupart des résultats théoriques supposent que ce point initial est connu, mais numériquement l'obtention de ce point initial prend beaucoup de temps.

Dans cette thèse, nous avons apporté des contributions d'ordre algorithmique, théorique et numérique. En effet, l'introduction du poids et également l'approche non réalisable constituent un remède appréciable pour le problème d'initialisation au niveau des méthodes de trajectoire centrale. Et d'autre part, en utilisant la fonction noyau d'objective d'atteint une nouvelle classe de direction de descente, condition de proximité et la complexité théorique.

Les résultats obtenus sont très encourageants et donnent lieu à d'autres perspectives dans le domaine de l'optimisation numérique. Les recherches futures pourraient se concentrer sur l'extension de l'optimisation cône symétrique. Enfin, les tests numériques pour (SDO) est un travail intéressant pour étudier le comportement d' algorithme afin d'être comparée avec d'autres approches existantes.

Annexe

1. Calcul d'une solution initiale strictement réalisable pour la programmation mathematiques sous contraintes linéaires

1.1. Solution initiale primale strictement réalisable

Un point primal strictement réalisable d'un programme mathematique est une solution du problème :

$$(PF)\ \{Ax = b,\ x > 0\}$$

En utilisant la téchnique de la variable artificielle [53], le problème (PF) est équivalent au problème suivant :

$$(P_1)\begin{cases} \min\ \lambda \\ s.c \\ Ax + \lambda(b - Aa) = b \\ x > 0,\ \lambda > 0 \end{cases}$$

Tel que $a \in \mathbb{R}^n_{++}$ est choisi arbitrairement dans l'orthant positif et λ une variable artificielle. Posons $z = (x,\ \lambda)$, le problème (P_1) est équivalent au programme linéaire suivant :

$$(P_2)\begin{cases} \min\ \overline{c}^t z = w^* \\ s.c \\ \overline{A}z = b \\ z > 0 \end{cases}$$

Où $\overline{c} = (0, ...,\ 1) \in \Re^{n+1},\ \overline{A} = (A,\ b - Aa) \in \Re^{m\ \times n+1}$.

Le problème (P_2) vérifie les hypothèses de Karmarkar :

1) $w^* = 0$.

2) $z = (a,\ 1)$ est une solution strictement réalisable de (P_2).

3) la matrice $\overline{A}$ est de plein rang ($rg\ \overline{A}\ = m < n+1$). Plus précisement Karmarkar [43] démontre le théorème suivant :

Théorème : $\exists\ \varepsilon > 0$, *telle que les deux propositions suivantes sont équivalentes :*

1) x *solution de* (PF).

2) (P_2) *admet une solution optimale* $(x,\ \lambda)$ / $\lambda \leq \varepsilon$.

Remarque : *Pour résoudre le problème* (P_2)*, on utilise la variante de Ye-Lustig [53] décrite dans la suite.*

1.2. Solution initiale duale strictement réalisable

Un point strictement réalisable pour le problème dual est une solution du problème :

$$(DF)\ \left\{A^t y + s = \nabla f(x),\ y \in \Re^m,\ s > 0\right\}$$

On pose : $y = y^+ - y^-$ tel que $y^+ > 0$ et $y^- > 0$.

Le problème (DF) est équivalent au problème suivant :

$$\begin{aligned} & (DF)\ \left\{A^t(y^+ - y^-) + s = \nabla f(x),\ y^+ > 0,\ y^- > 0,\ s > 0\right\} \\ = \ & \left\{A^t y^+ - A^t y^- + s = \nabla f(x),\ y^+ > 0,\ y^- > 0,\ s > 0\right\} \end{aligned}$$

En utilisant la technique de la variable artificielle, le problème (DF) est équivalent au problème suivant :

$$(D_1) \begin{cases} \min\ \lambda \\ s.c \\ Dd + \lambda(\nabla f(x) - Da) = \nabla f(x) \\ d > 0,\ \lambda > 0 \end{cases}$$

Tel que :

$a \in \mathbb{R}^{n+2m}_{++}$ est choisi arbitrairement dans l'orthant positif.

λ une variable artificielle.

$D = (A^t,\ -A^t,\ I)$ est une matrice de $\Re^{n\ \times\ 2m+n}$.

$d = (y^+,\ y^-,\ s)$ un vecteur de $\Re^{\ 2m+n}$.

Posons $t = (d,\ \lambda) \in \mathbb{R}^{n+2m+1}$, le problème (D_1) est équivalent au programme linéaire suivant :

$$(D_2)\begin{cases} \min\ \overline{\overline{c}}^t t = v^* \\ s.c \\ \overline{\overline{D}} t = \nabla f(x) \\ t > 0 \end{cases}$$

Où $\overline{\overline{c}} = (0, ..., 0,\ 1) \in \Re^{2m+n+1}$, $\overline{\overline{D}} = (D,\ c - Da) \in \Re^{n\ \times\ 2m+n+1}$.

Le problème (D_2) vérifie les hypothèses de Karmarkar :

1) $v^* = 0$.

2) $t = (a,\ 1)$ est une solution strictement réalisable de (D_2).

3) la matrice $\overline{\overline{D}}$ est de plein rang ($rg\ \overline{\overline{D}}\ = n < 2m + n + 1$).

De la même façon que le théorème précédent on a aussi :

(y, s) solution de $(DF) \Longleftrightarrow (D_2)$ admet une solution optimale $(d,\ \lambda)\ /\ \lambda \leq \varepsilon$.

Algorithme (*Variante de Ye-Lustig*)

Début

$\varepsilon > 0$ est un paramètre de précision, $\widehat{x}^0 = (x^0,\ \lambda^0)^t$, $\widehat{c} = (0, ...,\ 0,\ 1)^t$

$k = 0$

Si $(\|Ax^0 - b\| < \varepsilon)$ **alors** :

Stop, on dit que x^0 est une solution approximative de (PF) et aller à **Fin.**

Sinon

b/

Si $(\lambda^k < \varepsilon)$ **alors** :

Stop, on dit que x^k est une solution approximative de (PF) et aller à **Fin.**

Sinon

Poser :

$$D_k = diag(\widehat{x}^k)$$

$$B = [A,\ b - Aa]$$

$$r = \frac{1}{\sqrt{(n+1)(n+2)}}$$

$$B_k = [BD_k,\ -b]$$

$$p_k = [I - B_k^t(B_k B_k^t)^{-1}B_k][D_k\widehat{c},\ -\widehat{c}^t\widehat{x}^k]^t$$

Prendre :

$$y^{k+1} = \frac{e_{n+2}}{n+2} - \alpha_k r \frac{p_k}{\|p_k\|}$$

$$\widehat{x}^{k+1} = \frac{1}{y^{k+1}[n+2]} D_k\ y^{k+1}\,[n+1]$$

Poser :

$\lambda^k = \widehat{x}^{k+1}\,[n]$, et aller à **b.**

fin si.

fin si.

Fin.

RÉFÉRENCES

[1] M. Achache, A new primal-dual path-following method for convex quadratic programming, Computational and Applied Mathematics $25(1)(2006)97-110$.

[2] M. Achache, H. Roumili, A. Keraghel, A numerical study of an infeasible primal-dual path-following algorithm for linear programming, Applied Mathematics and Computation $(186)(2007)472-1479$.

[3] M. Achache, A weighted path-following method for the linear complementarity problem. Universitatis Babes-Bolyai, Series Informatica $49(1)(2004)61-73$.

[4] M. Achache, Complexity analysis and numerical implementation of a short-step primal-dual algorithm for linear complementarity problems, Applied Mathematics and Computation 216 $(2010)1889-1895$.

[5] Y.Q. Bai, G. Lesaja, C. Roos, G.Q. Wang, M. El'Ghami, A class of large-update and small-update primal-dual interior point algorithms for linear optimization, Journal of Optimization Theory and Application $(138)(2008)341-359$.

[6] Y.Q. Bai, M El. Ghami, C. Roos, A comparative study of kernel functions for primal-dual interior-point algorithms in linear optimization, SIAM Journal on Optimization $15(1)(2004)101-128$.

[7] Y.Q. Bai, M. El Ghami, C. Roos, A new efficient large-update primal-dual interior-point method based a finite barrier, SIAM J. Optim. $13(3)(2002)766-782$.

[8] S. Bazarra, H.D. Sherali and C.M. Shetty, " Nonlinear programming, theory and algorithms ", Second edition, 1993.

[9] R. Bellman, Introduction to Matrix Analysis, in : Classics in Applied Mathematics, vol. 12, SIAM, Philadelphia, 1995.

[10] D. Benterki, A. Leulmi. An improving procedure of the interior projective method for linear programming. Applied Mathematics and Computation 199 (2008) $811 - 819$

[11] D. Benterki, B. Merikhi, A modified algorithm for the strict feasibility, RAIRO Oper. Res.$35(2001)395 - 399$.

[12] D. Benterki, A. Leulmi. An improving procedure of the interior projective method for linear programming. Applied Mathematics and Computation $199(2008)811 - 819$

[13] K. H. Borgwardt, The Simplex Method : A Probabilistic Analysis, Springer-Verlag, Berlin, 1987.

[14] N. Boudiaf, Problème de complémentarité linéaire semi défini. Etude théorique et algorithmique, thèse de doctorat, Université Hadj Lakhdar, Batna, Algérie, 2012.

[15] X. Chen, Smoothing methods for complementarity problem and their applications. a survey, Journal of the Operations Research Society of Japan $(43)(2000)2 - 47$.

[16] R. W. Cottle, J.S. Pang, R.E. Stone, The Linear Complementarity Problem, Academic Press, San Diego, 1992.

[17] G. B. Dantzig, Linear Programming and extensions, Princeton University Press, Princeton, N. J, 1963.

[18] Z. Darvay, A new algorithm for solving self-dual linear programming problems. Studia Universitatis Babes-Bolyai, Series Informatica $47(2)(2002)15 - 26$.

[19] Z. Darvay, New interior-point algorithms in linear optimization, Adv. Model. Optim. $5(1)(2003)51 - 92$.

[20] El Amir Djeffal, Etude numérique d'une méthode de la trajectoire centrale avec poids pour la programmation convexe à contraintes linéaires, Mémoir de magister, Université Hadj Lakhdar, Batna, Algérie, 2009.

[21] El-Amir Djeffal, Djamel Benterki, Minimisation d'une fonction convexe sous contraintes linéaires, Conférences Internationales de Mathématiques et Applications , $CIMA'09$ du $26 - 28$ Oct 2009

[22] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Etude de quelques Algorithmes de points intérieurs pour la programmation quadratiques, $Workshop\ SDDC'10$ du $09-13\ Mai$ 2010.

[23] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, A Procedure improving interior point Alghorithms for the linear programming, $24^{th} MEC\ EUROPT$ 2010, $IZMIR\ TURKEY\ du\ 23-26\ jun$ 2010.

[24] DJEFFAL El-Amir, DJEFFAL Lakhdar, Design and implementation of operational research techniques for making schedules . $International\ Conference\ EURO\ XXIV\ LISBON\ du\ 11-14\ july$ 2010.

[25] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Extension et implémentation numérique d'une méthode de points intérieurs pour la programmation non linéaire, $RAMA'07\ du$ 24 au 26 $octobre$ 2010, $Batna$, $Algérie$.

[26] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, A midified algorithm for the linear programming, $DIMACOS\ du$ 05 au 08 May 2011, $Mohammedia$, $Maroc$.

[27] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Comparative study between two interior point methods for solving a mathematical programming, $JNMA'11du$ 29 au 30 $novembre$ 2011, $Skikda$, $Algérie$.

[28] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Une approche barrière logarithmique pour résoudre un problème d'optimisation convexe, $WTCO$ 2011du 18 au 20 $Décembre$ 201, $Mostaganem$, $Algérie$

[29] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Analyse de complexité et implémentation numérique d'une approche pénalisée pour un problème d'optimisation convexe, SMA 2012 $du\ 7-8\ Mars$ 2012, $Annaba$, $Algérie$.

[30] El Amir Djeffal, Lakhdar Djeffal, Complexity analysis predictor-corrector interior-point algorithm for convex nonlinear programming, $COSI$ 2012, $12-15\ Mai$ 2012, $Tlemcen$, $Algeria$.

[31] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Infeasible interior point methods for quadratic semidefinite programming with NT steps based on kernel function, $JIOEdp'12$, ENSET d'Oran, $4-5$ $Novembre$ 2012, $Algérie$.

[32] El Amir Djeffal, Lakhdar Djeffal. Finding a Strict Feasible Dual Initial Solution for Quadratic Semidefinite Programming. Applied Mathematical Sciences, $Vol.6$, 2012, $no.41$, $2029-2034$.

[33] El Amir Djeffal, Lakhdar Djeffal. A New Approach for Solving an Optimization Problem. Applied Mathematical Sciences, $Vol.6$, 2012, $no.41$, $2035-2042$.

[34] El Amir Djeffal, Lakhdar Djeffal, Djamel Benterki, Extension of Karmarkar's algorithm for solving an optimization problem. Advanced in Applied Mathematics and approximation Theory, Springer, vol 41, 2013

[35] El Amir Djeffal, Lakhdar Djeffal, Feasible short-step interior point algorithm for linear complementarity problem based on kernel function, Advanced Modeling and Optimization, $Volume15, Number$ 2, 2013.

[36] A. V. Fiacco, G. P. Mccormick, Nonlinear Programming : Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968. Réimprimé par SIAM Publications, 1990.

[37] R. M. Freund, S. Mizuno, Interior Point Methods : Current Status and Future Directions, Optima, 51, October 1996, $pp.1-9$.

[38] K. R. Frisch, The logarithmic potential method of convex programming, Technical Report, University Institue of Economics, Oslo, Norway, 1955.

[39] A. J. Goldman, A. W. Tucker, Theory of linear programming, in Linear Equalities and Related Systems, H. W. Kuhn and A. W. Tucker eds., Princeton University Press, Princeton N.J., 1956, $pp.53-97$.

[40] Y. Hang, D.Zhang, Superlinear convergence of infeasible interior point methods for linear programming, Mathematical programming 66(1994) $pp.361-377$

[41] R. A. Horn, C.R. Johnson, Topics in Matrix Analysis, Cambridge University Press, 1991.

[42] J. Ji, F.A. Potra, S. Huang, Predictor–corrector method for linear complementarity problems with polynomial complexity and superlinear convergence, J. Optim. Theory Appl. 85 (1) (1995) $187 - 199$.

[43] N. Karmarkar, A new polynomial-time algorithm for linear programming, Combinatorica, $4(1984), pp.373 - 395$.

[44] N. Karmarkar, A new polynomial-time algorithm for linear programming, in : Proceedings of the 16th Annual ACM Symposium on Theory of Computing, 1984, *pp.* $302 - 311$.

[45] Z. Kebbiche, A. Keraghel, A. Yassine, Extension of a projective interior point method for linearly constrained convex programming. Applied mathematics and computation 193 (2007) $553 - 559$.

[46] Z. Kebbiche, Etude et extensions d'algorithmes de points intérieurs pour la programmation non linéaire, thèse de doctorat, Université Ferhat Abbas, Sétif, Algérie, 2007.

[47] L. G. Khachiyan, A polynomial algorithm in linear programming, Soviet Mathematics Doklady, $20(1979), pp.191 - 194$.

[48] V. Klee, G. J. Minty, How good is the simplex algorithm, in Inequalities, O. Shisha, ed., Academic Press, New York, $1972, pp.159 - 175$.

[49] E. Klerk, Aspects of Semidefinite Programming : Interior Point Methods and Selected Applications, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[50] M. Kojima, M. Shida, S. Hara, Interior point methods for the monotone semidefinite linear complementarity problem in symmetric matrices, SIAM J. Optim. 7 (1997) $86 - 125$.

[51] M. Kojima, N. Megiddo, S. Mizuno, A primal–dual infeasible-interior-point algorithm for linear programming, Math. Program. $61(1993)\ 263 - 280$.

[52] M. Kojima, S. Mizuno, A. Yoshise, A primal-dual interior point algorithm for linear programming, in : N. Megiddo (Ed.), Progress in Mathematical Programming : Interior Point and Related Methods, Springer Verlag, New York, 1989, $pp.29 - 47$.

[53] I. J. Lustig, Feasible issues in a primal–dual interior point method for linear programming, Math. Program. $49(1990 - 1991)145 - 162$.

[54] H. Mansouri, C. Roos, A new full-Newton step $O(n)$ infeasible interior-point algorithm for semidefinite optimization, Numer. Algorithms $52(2)(2009)225 - 255$.

[55] H. Mansouri, C. Roos, Simplified $O(nL)$ infeasible interior-point algorithm for linear optimization using full Newton steps, Optim. Methods Softw. $22(3)(2007)519 - 530$.

[56] G. P. McCormick, The projective SUMT method for convex programming, Mathematics of Operations Research $14(1989)\ 203 - 223$.

[57] S. Mehrotra, On the implementation of a (primal-dual) interior point method, SIAM Journal on Optimization $2(1992)575 - 601$.

[58] N. Megiddo, Pathways to the optimal set in linear programming, in : N. Megiddo (Ed.), Progress in Mathematical Programming : Interior Point and Related Methods, Springer Verlag, New York, 1989, $pp.313 - 158$.

[59] B. Merikhi, Etude comparative de l'extension de l'algorithme de Karmarkar et des méthodes simpliciales pour la programmation quadratique convexe, Thèse de Magister, Institut de Mathématiques, University Ferhat Abbas, Sétif, Octobre 1994.

[60] R. D. C. Monteiro, I. Adler, An extension of Karmarkar type algorithm to a class of convex separable programming problems with global linear rite of convergence, Mathematics of Operations Research $15(1990)\ 408 - 422$.

[61] A. Nemirovskii, K. Scheinberg, Extension of Karmarkar's algorithm onto convex quadratically constrained quadratic problems, Mathematical Programming : Series A $72(3)(1996)273 - 289$

[62] Y. E. Nestrov, A. S. Nemrovesky, Interior Point Polynomial Methods in Convex Programming, SIAM Publications, Philadelphia, 1994.

[63] Y.E. Nesterov, A.S. Nemirovskii, Interior Point Polynomial Algorithms in Convex Programming, in : SIAM Stud. Appl. Math., vol. 13, SIAM, Philadelphia, 1994.

[64] J. Nocedal, S.J. Wright, " Numerical optimization ", Springer series in operations research.

[65] J. Peng, C. Roos and T.Terlaky. New complexity analysis of the primal-dual Newton method for linear optimization. Annals of operations research.(49) $pp.23-39.(2000)$.

[66] J. Peng, C. Roos, T. Terlaky, Self-regular functions and new search directions for linear and semidefinite optimization, Math. Program. 93 (1) (2002) $129-171$.

[67] J. Peng, C. Roos, T. Terlaky, Self-Regularity. A New Paradigm for Primal–Dual Interior-Point Algorithms, Princeton University Press, 2002.

[68] J. Peng, C. Roos, Terlaky, A new sufficient large-update interior point method for linear optimization, Journal of Computational and Technologies $6(4)(2001)61-80$.

[69] J. Peng, C. Roos, T. Terlaky, A New Paradigm for Primal-dual Interior-Point Methods, Princeton University Press, Princeton, NY, 2002.

[70] F. A. Potra, A superlinearly convergent predictor–corrector method for degenerate LCP in a wide neighborhood of the central path with $O(\sqrt{n}L)$ iteration complexity, Math. Program, 100 (2) (2004) $317-337$.

[71] Y. Qian, A polynomial predictor–corrector interior point algorithm for convex quadratic programming, Acta Mathematica Scientia, 26 $B(2)$ (2006) $265-279$.

[72] C. Roos, A full-Newton step $O(n)$ infeasible interior-point algorithm for linear optimization, SIAM J. Optim. $16(4)(2006)1110-1136$.

[73] C. Roos, T. Terlaky, J.Ph. Vial, Theory and algorithms for linear optimization, in : An Interior Approach, John Wiley and Sons, Chichester, UK, 1997.

[74] N. Z. Shor, Utilization of the operation of space dilatation in the minimization of convex functions, Kibernetika, $1(1970), pp.6-12$. Traduction anglaise : Cybernetics,$6, pp.7-15$.

[75] G. Sonnevend, J. Stoer, Global ellipsoidal approximation and homotopy methods for solving convex analytic programming, Applied Mathematics and optimization $21(1991)\ 139-165$.

[76] J. Sturm, Theory and algorithms of semidefinite programming, in : H. Frenk, C. Roos, T. Terlaky, S. Zhang (Eds.), High Performance Optimization, Kluwer Academic Publishers, Dordrecht, The Netherlands,

[77] K. Tanabe, Centered Newton method for linear programming : interior and 'exterior' point method, in : K. Tone (Ed.), New Methods for Linear Programming, vol. 3, The Institute of Statistical Mathematics, Tokyo, Japan, $1990, pp.98-100$. (in Japanese).

[78] Y. Ye, E. Tse, An extension of Karmarkar's projective algorithm for convex quadratic programming, Mathematical Programming, 44 $(1989)157-179$.

[79] L. Tuncel, Potential reduction and primal dual methods, in : H. Wolkowicz, R. Saigal, L. Vandenberghe (Eds.), Theory, Algorithms and Applications, in : Handbook of Semidefinite Programming, Kluwer Academic Publishers, Boston, MA, 2000.

[80] G.Q. Wang ,Y.Q.Bai, A new primal-dual path-following interior-point algorithm for semidefinite optimization, J. Math. Anal. Appl, 353 $(2009)339-349$

[81] S.J. Wright, Primal-dual Interior-Point Methods, SIAM, Philadelphia, USA, 1997.

[82] Yinyu Ye, Interior Point Algorithms : Theory and Analysis, Wiley-Interscience, 1997.

[83] Y. Zhang, On the convergence of a class of infeasible-interior-point methods for the horizontal linear complementarity problem, SIAM J. Optim. $4(1994)208-227$.

[84] J. Zhu, A path following algorithm for a class of convex programming problems, ZOR-Methods and Models of Operations Research, 36 $(1992)\ 359-377$.