République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohammed Seddik Ben Yahia - Jijel

Faculté des Sciences Exactes et Informatique

Département de Mathématiques

Mémoire de fin d'études

Présenté pour l'obtention du diplôme de

Master

Spécialité : Mathématiques.

Option : EDP et applications.

Thème

Etude théorique d'un problème d'optimisation linéaire

Présenté par :

Boukehil Narimane.

Kehal Khaoula.

Devant le jury :

Président	: I. Touil	M.C.A Université de Jijel.
Encadreur	: L. Menniche	M.C.B Université de Jijel.
Examinateur	: S. Djemai	M.C.B Université de Jijel.

Promotion **2020/2021**

Remerciements

En premier lieu, nous tenons à remercier ***Allah****, notre créateur pour nous avoir donné la force pour accomplir ce travail.*

Nous remercions chaleureusement notre encadreur ***L. Menniche*** *pour tout le soutien et l'encadrement qu'il nous a donnée.*

Nous remercions vivement ***I. Touil*** *pour avoir accepter de présider le jury de soutenance.*

Nous adressons également nos vifs remerciements à L'examinatrice ***S. Djemai*** *pour avoir accepté d'être membre de ce jury.*

Nous voulons aussi remercier tous les enseignants qui ont contribué à notre formation, ainsi qu'à toute l'équipe du département de mathématiques.
Enfin, c'est avec beaucoup de plaisir que nous pouvons remercier profondément nos familles pour leurs conseils et encouragement durant toutes les années de notre étude.

Dédicace

Je dédie ce mémoire

À *mes chères parents ma mère et mon père*

Pour leur patience, leur amour, leur soutient et leurs encouragements.

À *mon chère frère.*

À *mes chères sœurs.*

À *tous les membres de ma grande famille.*

À *mes amies et mes camarades.*

À *tout ceux qui sont proches à mon cœur.*

B. Narimane

Dédicace

Je dédie ce mémoire

À *mes chères parents ma mère et mon père*

Pour leur patience, leur amour, leur soutient et leurs encouragements.

À *mes frères.*

À *mes chères sœurs.*

À *tous les membres de ma grande famille.*

À *mes amies et mes camarades.*

À *tout ceux qui sont proches à mon cœur.*

K. Khaoula.

Table des matières

Introduction

Les méthodes de points intérieurs forment une classe d'algorithmes qui permettent de résoudre des problèmes d'optimisation convexes. Les méthodes de points intérieurs se répartissent en plusieurs familles :

• La méthode affine.

• La méthode de réduction du potentiel (notion de barrière logarithmique, trajectoire centrale).

• La méthode projective basée sur la transformation projective de $\mathbb{R}^n_+$ dans un simplexe S_n.

Les méthodes de points intérieurs ont fait l'objet de plusieurs travaux de recherches, ceux effectueés par Den Hertog [7], Nestrov et Nemirovski [17], Roos, Terlaky, Vial [20], Ye, Tse [22],...etc. Plusieurs algorithmes ont été proposé pour résoudre les problème de programmation linéaire, par les méthodes projectives [10], les méthodes de trajectoire centrale [9] et les méthodes barrières logarithmique [4]. Les algorithmes développés ont été inspirés par l'algorithme de Narendra Karmarkar, pour l'optimisation linéaire [8]. Cet algorithme est le premier réellement efficace qui résout ces problèmes en un temps polynomial. La méthode des ellipsoïdes fonctionne aussi en temps polynomial mais est inefficace en pratique. En utilisant une fonction potenciel, Karmarkar montre que son algorithme converge après $O(nq + n \log n)$ itérations pour un pas de déplacement $\alpha = \frac{1}{4}$.

Depuis, plusieurs chercheurs s'intéressent à cet algorithme dans le but d'améliorer au mieux son comportement numérique. Les deux éléments de base visés sont la direction de

descente qui domine le coût du calcul à chaque itération et le pas de déplacement qui joue un rôle important dans la vitesse de convergence. Pedberg [18] a pu réduire le nombre des itérations à $O(nq)$ itérations pour $\alpha = \frac{1}{2}$. De son côté Scherijver [21], en gardant la même fonction potentiel de Karmarkar a pu montrer que l'algorithme converge après $\frac{n}{1-\log 2}\log\left(\frac{c^t e_n}{\varepsilon}\right)$ itérations pour $\alpha_s = \frac{1}{(1+nr)}$.
Ce travail est consacré à détailler les deux articles de Bouafia et al. [2, 3], ils ont proposé deux nouveaux pas de déplacement permettant d'améliorer le comportement numérique de l'algorithme de Karmarkar. Notre étude est d'ordre théorique, algorithmique et numérique.
Le mémoire présenté est organisé comme suit : Dans le premier chapitre, on présente des résultats fondamentaux nécessaires pour notre travail : il s'agit de l'analyse convexe, la programmation linéaire. Le deuxième chapitre est consacré à une brève étude des propriétés fondamentales de la méthode de Karmarkar et sa variante de Ye-Lustig, puis on propose les deux nouveau pas de déplacement et on montre que ces deux pas meilleurs que celui de Scherijver. D'autre part, on donne les résultats de convergence amélioré par rapport à celui de Scherijver. Enfin, on termine par des tests numériques sur plusieurs exemples où on montre qu'on a une amélioration légère du comportement numérique de l'algorithme en question.

Chapitre 1

Préliminaires et notions fondamentales

Dans ce chapitre, nous rappelons quelques notions de base sur l'analyse convexe, programmation linéaire. Nous rappelons également, les résultats du dualité, en programmation linéaire.

Pour plus d'informations voir [14, 15].

1.1 Notions de convexité

Définition 1.1.1. (*Ensemble affine*)
Un sous ensemble $D \subset \mathbb{R}^n$ est dit affine si

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

Définition 1.1.2. (*Ensemble convexe*)
Un sous ensemble $D \subset \mathbb{R}^n$ est dit convexe si

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

Définition 1.1.3. *Un ensemble convexe de la forme*

$$D = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\},$$

où A est une matrice de $\mathbb{R}^{m \times n}$ et b un vecteur de $\mathbb{R}^m$, est appelé un polytope.

Définition 1.1.4. *Un ensemble convexe de la forme*

$$D = \{x \in \mathbb{R}^n_+ : \ A^t x \leq b\}$$

est appelé un polyèdre convexe.

Définition 1.1.5. (*Ensemble borné*)
Un sous ensemble $D \subset \mathbb{R}^n$ est borné, s'il existe $k \in \mathbb{R}_+$ tel que la valeur absolue de chaque composante d'un élément de D est plus petite que k, i.e., $\forall x = (x_1, \cdots, x_n) \in D$,
$\forall i = \overline{1,n} : \exists k \in \mathbb{R}_+, |x_i| \leq k$.

Remarque 1.1.1. *Un polytope borné sera appelé polyèdre convexe.*

Définition 1.1.6. *L'ensemble S_n est un n-simplexe, s'il est de la forme*

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

Définition 1.1.7. (*Fonction convexe*)
Une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ est convexe sur un ensemble convexe D si

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

Définition 1.1.8. (*Fonction affine*)
Une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ est affine sur un ensemble convexe D si

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

Définition 1.1.9. (*Fonction coercive*)
Une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ est dite coercive sur un ensemble convexe D si

$$\lim_{\|x\| \to +\infty} f(x) = +\infty, \ \forall x \in D.$$

Définition 1.1.10. (*Combinaison linéaire convexe*)
Un vecteur $y \in D \subset \mathbb{R}^n$ est une combinaison linéaire des points $\{x_1, \ \ldots, \ x_p\}$, s'il existe des coefficients réels λ_i, $i \in \{1, \cdots, p\}$, tels que

$$y = \sum_{i=1}^{p} \lambda_i x_i \ \ avec \ \sum_{i=1}^{p} \lambda_i = 1.$$

1.2 Convexité et dérivée

Définition 1.2.1. *Le gradient d'une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ continûment différentiable évalué au point $x \in \mathbb{R}^n$ s'écrit*

$$\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \cdots, \frac{\partial f(x)}{\partial x_n} \right)^t.$$

Définition 1.2.2. *Soient U un ouvert de $\mathbb{R}^n$ et $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction deux fois différentiable en $x \in U$ dont toutes les dérivées partielles d'ordre deux sont définies en x, alors la matrice*

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

est appelée la matrice Hessienne de f en x.

Définition 1.2.3. *Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction continûment différentiable sur un ensemble convexe D.*

1. *f est une fonction convexe sur D si et seulement si la matrice Hessienne $\nabla^2 f(x)$ est semi-définie positive, c'est-à-dire*

$$\forall x \in D,\ y^t \nabla^2 f(x) y \geq 0, \forall y \in \mathbb{R}^n.$$

Ou encore toutes les valeurs propres de $\nabla^2 f(x)$ sont positives.

2. *f est une fonction strictement convexe sur D si*

$$\forall x \in D,\ y^t \nabla^2 f(x) y > 0, \forall y \in \mathbb{R}^n - \{0\}.$$

Ou encore toutes les valeurs propres de $\nabla^2 f(x)$ sont strictement positives.

1.3 Formule de Taylor

Soient $\Omega \subset \mathbb{R}^n$ ouvert, $f : \Omega \to \mathbb{R}$, $a \in \Omega$ et $h \in \mathbb{R}^n$ tel que $[a, a + h] \subset \Omega$, alors

1) Si $f \in \mathbb{C}^1(\Omega)$ alors

i) Formule de Taylor à l'ordre 1 avec reste intégral

$$f(a + h) = f(a) + \int_0^1 \langle \nabla f(a + th), h \rangle dt.$$

ii) Formule de Taylor-Maclaurin à l'ordre 1

$\exists \theta \in [0,1]$, tel que $f(a+h) = f(a) + \langle \nabla f(a+\theta h), h \rangle$.

iii) Formule de Taylor-Young à l'ordre 1

$$f(a+h) = f(a) + \langle \nabla f(a), h \rangle + o(\|h\|).$$

2) Si $f \in \mathbb{C}^2(\Omega)$

i) Formule de Taylor à l'ordre 2 avec reste intégral

$$f(a+h) = f(a) + \langle \nabla f(a), h \rangle + \int_0^1 (1-t)\langle \nabla^2 f(a+th)h, h \rangle dt.$$

ii) Formule de Taylor-Maclaurin à l'ordre 2

$\exists \theta \in [0,1]$, tel que $f(a+h) = f(a) + \langle \nabla f(a), h \rangle + \frac{1}{2}\langle \nabla^2 f(a+\theta h)h, h \rangle$.

iii) Formule de Taylor-Young à l'ordre 2

$$f(a+h) = f(a) + \langle \nabla f(a), h \rangle + \frac{1}{2}\langle \nabla^2 f(a)h, h \rangle + o(\|h\|^2).$$

1.4 Programmation mathématique

Définition 1.4.1. (*Problème d'optimisation*)

a) Un problème d'optimisation sans contraintes s'écrit sous la forme

$$\begin{cases} \textit{trouver } x^* \in \mathbb{R}^n \textit{ tel que} \\ f(x^*) \leq f(x), \forall x \in \mathbb{R}^n. \end{cases}$$

b) Un problème d'optimisation avec contrainte s'écrit sous la forme

$$\begin{cases} \textit{trouver } x^* \in D \textit{ tel que} \\ f(x^*) \leq f(x), \forall x \in D. \end{cases}$$

Où $D \subsetneq \mathbb{R}^n$ appelé l'ensemble des contraintes.

Définition 1.4.2. (*Programme mathématique*)

Un programme mathématique (PM) est un problème d'optimisation s'écrit sous la forme

$$(PM) \begin{cases} \min \ (ou \max) \ f(x) \\ g_i(x) \leq 0, \ i = \overline{1,n}, \\ h_j(x) = 0, \ j = \overline{1,m}, \\ x \in S, \end{cases}$$

avec

$$D = S \cap D_{h_j} \cap D_{g_i} \cap D_f$$

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

- *Un point* x *de* D *est appelé solution réalisable de (PM).*
- D *est l'ensemble des solutions réalisables de* (PM).
- *On appelle solution optimale de* (PM), *toute solution réalisable* x *réalisant le minimum de* f *sur* D, $f(x)$ *sera appelée valeur optimale de* (PM).

Remarque 1.4.1. *On classifie le problème* (PM) *à partir des propriétés fondamentales à savoir la convexité, la différentiabilité et la linéarité des fonctions du problème. A ce propos,*

- (PM) *est convexe si les fonctions* f, g_i *sont convexes et les fonctions* h_i *sont affines.*
- (PM) *est linéaire si la fonction* f *est linéaire et les fonctions* g_i, h_i *sont affines.*
- (PM) *est différentiable si les fonctions* f, g_i *et* h_i *sont différentiables.*

Définition 1.4.3. (*Minimum local*)
Une solution réalisable x^* *est un minimum local de* (PM) *s'il existe* $\varepsilon > 0$ *tel que* $f(x^*) \leq f(x), \forall x \in \mathbf{B}(x^*, \varepsilon) \subset D$, *où* $\mathbf{B}(x^*, \varepsilon) = \{x \in \mathbb{R}^n : \parallel x - x^* \parallel \leq \varepsilon\}$.
L'ensemble des minima locaux de (PM) *est noté par*

$$loc \min_D f(x).$$

Définition 1.4.4. (*Minimum global*)
Soit $f : D \subset \mathbb{R}^n \to \mathbb{R}$, *on dit que la fonction* f *admet un minimum global (le problème* (PM) *admet une solution optimale globale) en* $x^* \in D$ *si et seulement si*

$$f(x) \geq f(x^*), \forall x \in D.$$

L'ensemble des minima globaux de (PM) *est noté par*

$$arg \min_D f(x).$$

Remarque 1.4.2. *On a toujours*

$$arg \min_D f(x) \subseteq loc \min_D f(x).$$

1.4.1 Existence et unicité de la solution optimale du problème (PM)

Théorème 1.4.1. *Si D un ensemble compact (fermé et borné) non vide de $\mathbb{R}^n$ et f est une fonction continue sur D, alors (PM) admet au moins une solution optimale $x^* \in D$.*

Corollaire 1.4.1. *Si D est un ensemble non vide et fermé de $\mathbb{R}^n$ et f est une fonction continue et coercive sur D, alors (PM) admet au moins une solution optimale $x^* \in D$.*

Théorème 1.4.2. *Si D un ensemble convexe non vide de $\mathbb{R}^n$ et f est strictement convexe sur D, alors (PM) admet une solution optimale unique.*

1.4.2 Condition d'optimalité

Avant de donner les conditions d'optimalité de (PM), on exige que les contraintes doivent satisfaire certains critères dites critères de qualification.

Qualification des contraintes

1. Si D est un polyèdre convexe (i.e., g_i et h_j affines), alors par définition les contraintes sont qualifiées en tout point réalisable.
2. Si D est convexe (i.e., g_i convexes et h_j affines) et $intD \neq \emptyset$, alors les contraintes sont qualifiées partout. C'est la condition de Slater.
3. Une contrainte d'inégalité $g_i(x) \leq 0$ est dite saturée (ou active) en $x_* \in D$ si $g_i(x^*) = 0$. De ce fait, une contrainte d'égalité $h_j(x) = 0$ est par définition saturée en tout point $x \in D$. Les contraintes sont qualifiées en $x^* \in D$ si les gradients de toutes les fonctions contraintes saturées en x^* sont linéairement indépendantes.

Définition 1.4.5. *Le lagrangien du programme mathématique (PM) est défini par*

$$L(x, \lambda, y) = f(x) + \sum_{i=1}^{n} \lambda_i g_i(x) + \sum_{j=1}^{m} y_j h_j(x), \ \lambda_i \in \mathbb{R}^+, y_j \in \mathbb{R}.$$

Théorème 1.4.3. *(Karush-Kuhn-Tucker (KKT))*
Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction différentiable sur D, si x^* est un minimum local du problème (PM), alors il existe des vecteurs $y \in \mathbb{R}^m$ et $\lambda \in \mathbb{R}^n_+$ (dits multiplicateurs de

Lagrange) *tel que*

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

Si de plus, f, g_i *et* h_j sont convexes, les conditions d'optimalité précédentes sont à la fois nécessaires et suffisantes pour que x^* soit un optimum global pour (PM).

1.5 Programmation linéaire

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

Définition 1.5.1. *Un programme linéaire* (PL) *est un système d'équations ou d'inéquations appelées contraintes qui sont linéaires et à partir de ses contraintes on doit optimiser une fonction également linéaire appelée objectif (économique) sous la forme*

$$(PL) \begin{cases} Opt(\sum_{j=1}^{n} c_j x_j) \\ \sum_{j=1}^{n} a_{ij} x_j (\leq, =, \geq) b_i, \ i = \overline{1,m}, \\ x_j \geq 0, \ j = \overline{1,n}. \end{cases}$$

Ou bien sous la forme matricielle

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

tels que

- $Opt = (\max \ ou \ \min)$.
- $c^t x = \sum_{j=1}^{n} c_j x_j$ *est la fonction à optimiser.*
- $c \in \mathbb{R}^n$ *est le vecteur coût.*

- $b \in \mathbb{R}^m$ *est le second membre.*
- $A = (a_{ij})_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \in \mathbb{R}^{m \times n}$ *est la matrice des contraintes.*

Exemple 1.5.1.

$$\begin{cases} \max(6x_1 + 15x_2) \\ x_1 + x_2 \leq 150, \\ x_1 + x_2 \geq 200, \\ x_1,\ x_2 \geq 0. \end{cases}$$

1.5.1 Formes usuelles d'un programme linéaire

Les programmes linéaire (PL) se présentent sous trois formes différentes sont

Forme standard

Un programme linéaire est sous forme standard si toutes les contraintes sont des égalités, c'est-à-dire : il s'écrit sous la forme

$$\begin{cases} Opt \quad c^t x \\ Ax = b, \\ x \geq 0. \end{cases}$$

Forme canonique

Un programme linéaire est sous forme canonique si toutes les contraintes sont des inégalités, c'est-à-dire : il s'écrit sous la forme

$$\begin{cases} Opt \quad c^t x \\ Ax \geq b \text{ (ou } \leq), \\ x \geq 0. \end{cases}$$

Forme Mixte

On dit que le problème de programmation linéaire (PL) est sous forme mixte, s'il n'écrit pas ni sous forme canonique, ni sous forme standard.

Remarque 1.5.1. *On peut toujours mettre un programme linéaire quelconque sous forme standard en introduisant des variables supplémentaires positives appelées variables d'écart.*

Exemple 1.5.2.

$$\begin{cases} \min 5x_1 - 2x_2 \\ x_1 - x_2 \geq 1, \\ 7x_1 + 3x_2 \leq 8, \\ -x_1 + 6x_2 = 10, \\ x_1,\ x_2 \geq 0. \end{cases}$$

En introduisant les variables d'écart $x_3 \geq 0,\ x_4 \geq 0$ *dans la première et la deuxième contrainte, on met le problème précédent sous la forme standard suivante*

$$\begin{cases} \min 5x_1 - 2x_2 + 0x_3 + 0x_4 \\ x_1 - x_2 - x_3 = 1, \\ 7x_1 + 3x_2 + x_4 = 8, \\ -x_1 + 6x_2 = 10, \\ x_1,\ x_2,\ x_3,\ x_4 \geq 0. \end{cases}$$

Théorème 1.5.1. *L'ensemble réalisable du problème de programmation linéaire est un polyèdre convexe.*

1.5.2 Dualité en programmation linéaire

La notion de dualité est un concept fondamental en programmation linéaire qui conduit à un résultat de grande portée théorique et pratique (le théorème de dualité faible et le théorème de dualité forte). Ainsi, étant donné un problème de programmation linéaire (P) appelé primal, on peut toujours lui associer un autre problème appelé dual (noté par (D)).

Soit le problème primal

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

Son dual est écrit sous la forme

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

Le problème linéaire (D) est très lié au problème linéaire (P), on remarque que

• La matrice des contraintes de (D) est la transposée de la matrice des contraintes de (P).

• Le vecteur coût de (P) est le vecteur de second membre de (D) et vice-versa.

Définition 1.5.2. (*Définition du dual dans le cas général*)
Évidement, on peut définir le dual d'un problème linéaire quelconque (*pas nécessairement sous forme standard*)*, le tableau suivant résume les correspondances entre primal et dual et permet d'écrire directement le dual d'un problème linéaire quelconque.*

Primal	Dual
Fonction objectif (min)	Second membre
Second membre	Fonction objectif (max)
A matrice des contraintes	A^t matrice des contraintes
Contrainte $i \quad \geq$	Variable $y_i \geq 0$
Contrainte $i =$	Variable $y_i \lesseqgtr 0$
Variable $x_j \geq 0$	Contrainte $j \quad \leq$
Variable $x_j \lesseqgtr 0$	Contrainte $j =$

TABLE 1.1 – Relation primal-dual.

Remarque 1.5.2.

1. *Le dual du problème dual* (D) *est le problème primal* (P).
2. *On peut transformer un problème de maximisation à un problème de minimisation il suffit d'écrire* $\max f = -\min(-f)$.

Théorème 1.5.2. *[14]* (*Dualité faible*)
Si x *et* y *sont des solutions réalisables de problèmes* (P) *et* (D) *respectivement, alors* $c^t x \geq b^t y$.

Théorème 1.5.3. *[14]* (*Dualité forte*)
Si x^* et y^* sont des solutions réalisables de (P) et (D) respectivement tels que $b^t y^* = c^t x^*$, alors x^* et y^* sont des solutions optimales des problèmes (P) et (D) respectivement.

Chapitre 2

Algorithme de Karmarkar et ses propriétés pour la programmation linéaire

Dans ce chapitre, on s'intéresse à une méthode de points intérieurs de type projectif, où on donne une description générale de la méthode classique de Karmarkar [8] et la variante la plus utilisée traduite par Ye-Lustig [13] ainsi que les résultats de convergence.

2.1 Problème linéaire traité par Karmarkar

Dans son article [8], Karmarkar traite le problème suivant, sous la forme réduite

$$(PK)\begin{cases} \min c^t x = z^* = 0 \\ Ax = 0, \\ x \in S_n = \{x \in \mathbb{R}^n_+, e_n^t x = 1\}. \end{cases}$$

Ayant comme solution réalisable le centre du simplexe S_n, $x^0 = \dfrac{e_n}{n}$.

Tels que

- $c \in \mathbb{R}^n$ est le vecteur coût.
- A est une $(m \times n)$ matrice réelle de plein rang $(rang(A) = m \leq n)$.
- $e_n = (1, \ \ldots, 1)^t \in \mathbb{R}^n$.

2.1.1 Transformation de Karmarkar sur S_n

Cette transformation est définie à chaque itération par

$$\begin{array}{rcccl} T_k & : & S_n & \to & S_n \\ & & x & \mapsto & y = T_k(x) = \dfrac{D_k^{-1}x}{e_n^t D_k^{-1}x}, \end{array}$$

tel que

$D_k = diag(x^k)$ est une $(n \times n)$ matrice diagonale avec les éléments diagonaux sont les composantes de x^k.

Comme la transformation T_k est inversible nous avons

$$x = T_k^{-1}(y) = \frac{D_k y}{e_n^t D_k y}.$$

La transformation T_k applique le simplexe S_n dans lui même, en même temps l'itéré $x^k > 0$ est envoyé au centre de S_n, le transformé du programme linéaire (PK) est le programme fractionnaire suivant

$$(PKT) \begin{cases} \min (D_k c)^t y \\ AD_k y = 0, \\ y \in S_n = \{y \in \mathbb{R}^n_+, e_n^t y = 1\}. \end{cases}$$

Qui peut s'écrire aussi sous la forme

$$(PKT) \begin{cases} \min (D_k c)^t y \\ \begin{bmatrix} AD_k \\ e_n^t \end{bmatrix} y = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \\ y \geq 0. \end{cases}$$

Le problème transformé (PKT) est aussi mis sous la forme réduite de Karmarkar.

Lemme 2.1.1. *[11] Si pour un programme linéaire donné on connait une solution réalisable y^0, tel que $(y_i^0 > 0,\ i = \overline{1,m})$, alors l'ellipsoïde*

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

est dans l'intérieur de l'orthant positif de $\mathbb{R}^n$.

D'après le lemme précédent, si on ajoute au problème (PKT) la contrainte
$y \in \mathbb{R}^n : \| y - x^0 \| \leq \alpha r$, où $0 < \alpha < 1$ et $r = \dfrac{1}{\sqrt{n(n-1)}}$ (le rayon de la sphère),
alors la contrainte de positivité $y \geq 0$ devient redondante. On obtient alors le problème

$$(P0) \begin{cases} \min (D_k c)^t y \\ AD_k y = 0, \\ e_n^t y = 1, \\ \| y - x^0 \|^2 \leq (\alpha r)^2. \end{cases}$$

Théorème 2.1.2. *La solution optimale du problème* $(P0)$ *est donnée explicitement par*

$$y_k = x^0 - \alpha r d_k,$$

où $d_k = \dfrac{p_k}{\| p_k \|}$ *et* $p_k = P_{B_k}(D_k c)$ *(la projection du vecteur* $(D_k c)$ *sur le noyau de* B_k*) et* $B_k = \begin{bmatrix} AD_k \\ e_n^t \end{bmatrix}$.

Démonstration.

Posons $x = y - x^0$, alors $B_k x = 0$, où B_k est la matrice des contraintes de $(P0)$.

$(P0)$ est alors équivalent au problème suivant

$$(P0) \begin{cases} \min (D_k c)^t x \\ B_k x = 0, \\ \| x \|^2 = \sum_{i=1}^n x_i^2 \leq (\alpha r)^2. \end{cases}$$

D'après (KKT), on a x^* est une solution optimale de $(P0)$ si et seulement s'il existe $\lambda \in \mathbb{R}^{m+1}$ et $\mu \geq 0$, telle que

$$D_k c + B_k^t \lambda + \mu x^* = 0. \tag{2.1}$$

En multipliant les deux membres de (2.1) par B_k, on trouve

$B_k D_k c + B_k B_k^t \lambda + \mu B_k x^* = B_k D_k c + B_k B_k^t \lambda = 0$ (puisque $B_k x = 0$),

alors

$$\lambda = -(B_k B_k^t)^{-1} (B_k D_k c),$$

d'où en substituant dans (2.1)

$$x^* = -\frac{1}{\mu}[I - B_k^t((B_k B_k^t)^{-1})B_k]D_k c = -\frac{1}{\mu}p_k,$$

mais x^* vérifie

$$\| x^*\| = \frac{1}{\mu} \| p_k\| = \alpha r,$$

ce qui implique

$$\frac{1}{\mu} = \frac{\alpha r}{\| p_k\|},$$

alors

$$x^* = -\alpha r \frac{p_k}{\| p_k\|},$$

finalement,

$$y^k = y^* = x^0 - \alpha r \frac{p_k}{\| p_k\|}.$$

■

2.1.2 Algorithme de Karmarkar

Début Algorithme

Initialisation

$\varepsilon > 0$ (le paramètre de précision) ;

$e_n = (1, \ \ldots, 1)$;

$x^0 = \dfrac{e_n}{n}$;

$k = 0$;

Tant que $(c^t x^k) > \varepsilon$ **faire**

1. Construire

   $D_k = diag(x_k)$;

   $B_k = \begin{bmatrix} AD_k \\ e_n^t \end{bmatrix}$;

2. Calculer

   $P_k = [I - B_k^t(B_k B_k^t)^{-1}B_k]D_k c$ (la projection du vecteur $D_k c$ sur le noyau de B_k) ;

   $d_k = \dfrac{p_k}{\| p_k\|}$ (la direction) ;

   $y^k = x^0 - \alpha r d_k$ où $0 < \alpha < 1$ (le pas de déplacement) et $r = \dfrac{1}{\sqrt{n(n-1)}}$;

3. Prendre

   $x^{k+1} = T_k^{-1}(y^k) = \dfrac{D_k y^k}{e_n^t D_k y^k}$ (on applique la transformation inverse pour obtenir l'itéré x^{k+1}) ;

   $k = k + 1$;

Fin Tant que

Fin Algorithme.

2.1.3 Généralisation de l'algorithme de Karmarkar

On peut ramener un programme linéaire générale quelconque mis sous la forme standard suivante

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

sous forme réduite de Karmarkar.

Remarque 2.1.1.

1) Si la valeur optimale z^* est non nulle, alors l'égalité $e_n^t x = 1$ permet de se ramener à un objectif nul. En effet, soit x^* une solution optimale du problème et z^* la valeur optimale de l'objectif, alors $c^t x^* = z^* = z^* e_n^t x^* \Longrightarrow (c - z^* e_n)^t x^* = 0$ et pour le système de contraintes $Ax = b, b \neq 0$, on se ramène facilement à un système homogène, il suffit d'écrire $Ax = be_n^t x \Longrightarrow (A - be_n^t)x = 0$.

2) Si z^* n'est pas connue, il existe des techniques d'approximation fiables permettant de remplacer z^* par des bornes inférieures ou supérieures convenables.

Pour ce faire, Karmarkar [10] a utilisé la transformation projective suivante

$$\begin{array}{rcccl} T_a & : & \mathbb{R}_+^n & \to & S_{n+1} \\ & & x & \mapsto & y = T_a(x), \end{array}$$

telle que

$$T_a(x) = \begin{cases} y_i = \dfrac{\dfrac{x_i}{a_i}}{1 + \sum_{i=1}^{n} \left(\dfrac{x_i}{a_i}\right)} \ , \ i = \overline{1,n} \\ y_{n+1} = 1 - \sum_{i=1}^{n} y_i. \end{cases}$$

Le problème (PL) se transforme via la transformation T_a au problème linéaire (PLT) suivant

$$(PLT) \begin{cases} \min(c')^t y = 0 \\ A'y = 0, \\ y \in S_{n+1} = \{y \in \mathbb{R}_+^{n+1} \ , \ e_{n+1}^t y = 1\}. \end{cases}$$

Où

- $a = (a_1, a_2, \ldots, a_n)^t$ est une solution strictement réalisable de (PL).
- $D_a = diag(a)$ est la matrice diagonale dont les éléments diagonaux sont les composantes du vecteur a.
- $A' = [AD_a, -b] \in \mathbb{R}^{m \times (n+1)}$.
- $c' = [(D_a c)^t, -z^*]^t \in \mathbb{R}^{n+1}$.
- $y = \begin{bmatrix} y[n] \\ y_{n+1} \end{bmatrix}$.

Remarque 2.1.2. *La transformation T_a est inversible et on a*

$$x = T_a^{-1}(y) = \frac{D_a y[n]}{y_{n+1}}.$$

Dans ce qui suit, on donne l'algorithme de Ye-Lustig appliqué pour z^* inconnu.

2.1.4 Algorithme de Ye-Lustig

En utilisant la transformation T_a et en remplaçant la contrainte $y \geq 0$ par la sphère $S\left(\frac{e_{n+1}}{n+1}, \alpha r\right)$, on obtient le problème suivant

$$(PY)\begin{cases} \min \begin{pmatrix} D_k c \\ -c^t x^k \end{pmatrix}^t y \\ B_k y = 0, \\ e_{n+1}^t y = 1, \\ \| y - \frac{e_{n+1}}{n+1} \|^2 \leq (\alpha r)^2. \end{cases}$$

Où $B_k = [AD_k, -b] \in \mathbb{R}^{m \times (n+1)}$.

Algorithme de Ye-Lustig

Début algorithme

Initialisation

$k = 0; x^0 > 0\,;\ \varepsilon > 0\,;$

$\| P_0 \| = (1+\varepsilon)\,|c^t x^0|;$

Tant que $\left(\dfrac{\| P_k \|}{|c^t x^0|} > \varepsilon\right)$ **faire**

1) Construire

$D_k = diag(x_k)\,;$

$B_k = \begin{pmatrix} AD_k, & -b \end{pmatrix};$

2) Calculer

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

$d_k = \dfrac{P_k}{\| P_k \|}\,;$

$y^k = \dfrac{e_{n+1}}{n+1} - \alpha r d_k,$

où $0 < \alpha < 1$ (le pas de déplacement) et $r = \dfrac{1}{\sqrt{n(n-1)}}\,;$

3. Prendre

   $x^{k+1} = T_k^{-1}(y^k) = \dfrac{D_k y^k[n]}{y^k_{n+1}}$ (on applique la transformation inverse pour obtenir l'itéré x^{k+1}) ;

   $k = k + 1$;

Fin tant que

Fin algorithme

Remarque 2.1.3.

1) Le pas de déplacement α peut être fixé durant l'exécution. Karmarkar a pris le pas de déplacement entre 0 et 1. Ceci assure que tous les itérés restent à l'intérieur du simplexe et permet de montrer la convergence polynomiale de l'algorithme.

2) Du point de vue numérique, il s'aver que plus α est grand plus l'algorithme converge vite (α peut être > 1). Pour cette raison plusieurs chercheurs suggèrent d'utiliser des pas variables moyennant les méthodes de recherche linéaire. Ces méthodes consiste à effecteur une recherche linéaire suivant la direction $(-d_k)$ minimisant approximativement la fonction $\varphi(\alpha) = f(\frac{e_n}{n} + \alpha r d_k)$, où f est la fonction potentiel de Karmarkar, malheureusement ces procédures sont coûteuses et inapplicable dans les problème de programmation semi-définie linéaire.

3) D'autre part, pour améliorer les résultats théoriques portés par Karmarkar sur α, Bouafia et al. [2, 3] ont utilisé des pas qui dépendent de la taille du problème (travaux de Schrijver [21], et Padberg [18]).

Dans ce qui suit, on donne le théorème de convergence polynomiale de l'algorithme de Karmarkar.

2.1.5 Convergence

Pour étudier la convergence polynomiale de l'algorithme, Karmarkar a utilisé la fonction potentiel définie sur D_x par

$$\begin{aligned} f(x) &= \sum_{i=1}^{n} \log\left(\frac{c^t x}{x_i}\right) \\ &= n \log(c^t x) - \sum_{i=1}^{n} \log x_i. \end{aligned}$$

Où

$$D_x = \{x \in \mathbb{R}^n : x > 0, Ax = 0, e_n^t x = 1\}.$$

Lemme 2.1.3. *[11] Soit y^k une solution strictement réalisable pour (PKT), alors on a*

$$\frac{c^t D_k y^k}{c^t D_k \frac{e_n}{n}} \leq \left(1 - \alpha \frac{1}{n-1}\right) = \left(1 - \alpha \frac{r}{R}\right) = \left(1 - \alpha n r^2\right),$$

tel que $r = \dfrac{1}{\sqrt{n(n-1)}}$, $R = \sqrt{\dfrac{n-1}{n}}$ *et* $0 < \alpha < 1$.

Théorème 2.1.4. (*Théorème de convergence de Karmarkar*) *[8]*
Si $0 < \alpha < \dfrac{1}{4}$, *alors en partant de* $x^0 = \dfrac{e_n}{n}$, *après* $O(nq + n \log n)$ *itérations, l'algorithme trouve un point réalisable* x *tel que*

1) $c^t x = 0$.

Ou

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

Remarque 2.1.4. *Dans ces travaux, Padberg [18] a pu améliorer la convergence de l'algorithme classique de Karmarkar en modifiant la fonction potentiel de Karmarkar ainsi*

$$h(x) = \frac{c^t x}{\left(\prod_{i=1}^n x_i\right)^{\frac{1}{n}}}.$$

Définie sur

$$D_x = \{x \in \mathbb{R}^n : x > 0, Ax = b, e_n^t x = 1\}.$$

Dans le lemme suivant, Padberg [18] a montré la convergence de son algorithme.

Lemme 2.1.5. *[18] Pour* $0 < \alpha < 1$ *et* y^k *défini par l'algorithme de Karmarkar, nous avons*

$$\frac{1}{n\left(\prod_{i=1}^n y_i^k\right)^{\frac{1}{n}}} \leq \frac{1}{1 + \dfrac{\alpha}{n-1}} \left(\frac{1 + \dfrac{\alpha}{n-1}}{1 - \alpha}\right)^{\frac{1}{n}}, \quad n \geq 3.$$

Et a montré que pour un pas de déplacement $\alpha_p = \dfrac{1}{2}$, *l'algorithme converge après* $O(nq)$ *itérations.*

Théorème 2.1.6. *[21] Shrijver a montré que pour* $\alpha_s = \dfrac{1}{1 + nr}$, *l'algorithme converge après* $\dfrac{n}{1 - \log 2} \log\left(\dfrac{c^t e_n}{\varepsilon}\right)$ *itérations.*
Tel que n *est la taille du problème* (PL) *et* $r = \dfrac{1}{\sqrt{n(n-1)}}$.

2.2 Amélioration de la convergence de l'algorithme de Karmarkar

Vu que le calcul du pas de déplacement joue un rôle important dans la vitesse de convergence de l'algorithme, pour cette raison Bouafia et al. dans [1, 2, 3] ont proposé deux nouveaux pas de déplacement, meilleurs que celui introduit par Schrijver [21].
Dans cette partie, on s'intéresse au développement du pas de déplacement.
Avant cela, nous avons besoin des lemmes, utilisé par Schrijver [21], pour facilité au lecteur la compréhension des propos établis le long des différents preuves.

Lemme 2.2.1. *[20] Soit ψ_n une fonction définie par*

$$\begin{array}{rccl} \psi_n & : &]-1,\infty[^n & \to & \mathbb{R}_+ \\ & & x & \mapsto & \psi_n(x) = e_n^t x - \sum_{i=1}^n \log(1+x_i). \end{array}$$

Alors, pour $\|x\| < 1$, on a

$$\psi_1(-\|x\|) \geq \psi_n(-x).$$

Lemme 2.2.2. *[11] Soit x^k le k-ième itéré de l'algorithme de Karmarkar, alors*

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

Où f est la fonction potentiel du Karmarkar et $x^0 = \dfrac{e_n}{n}$.

Démonstration.

On a

$$f(x) = n\log c^t x - \sum_{i=1}^n \log x_i = \sum_{i=1}^n \log\left(\frac{c^t x}{x_i}\right),$$

alors

$$\begin{aligned} f(x^k) - f(x^0) &= \sum_{i=1}^n \log\left(\frac{c^t x^k}{x_i^k}\right) - \sum_{i=1}^n \log\left(\frac{c^t x^0}{x_i^0}\right) \\ &= \sum_{i=1}^n \left(\log\left[\frac{c^t x^k}{c^t x^0}\right] + \log\left[\frac{x_i^0}{x_i^k}\right]\right) \\ &= n\log\left(\frac{c^t x^k}{c^t x^0}\right) + \log\left[\prod_{i=0}^n \left(\frac{x_i^0}{x_i^k}\right)\right], \end{aligned}$$

donc

$$\exp\left(f(x^k)-f(x^0)\right)=\exp\left[n\cdot\log\left(\frac{c^tx^k}{c^tx^0}\right)+\log\left(\prod_{i=0}^{n}\left(\frac{x_i^0}{x_i^k}\right)\right)\right]$$
$$=\left(\frac{c^tx^k}{c^tx^0}\right)^n\prod_{i=0}^{n}\left(\frac{x_i^0}{x_i^k}\right),$$

d'où

$$\left(\frac{c^tx^k}{c^tx^0}\right)=\left[\prod_{i=0}^{n}nx_i^k\right]^{\frac{1}{n}}\left[\exp\left(f(x^k)-f(x^0)\right)\right]^{\frac{1}{n}}.$$

Si $\left[\prod_{i=0}^{n}nx_i^k\right]^{\frac{1}{n}}\leq 1$, alors $\left(\frac{c^tx^k}{c^tx^0}\right)\leq\left[\exp\left(f(x^k)-f(x^0)\right)\right]^{\frac{1}{n}}$.

En effet,

pour démontrer que $\left[\prod_{i=0}^{n}nx_i^k\right]^{\frac{1}{n}}\leq 1$, on utilise la fonction

$$\begin{array}{rcccc} f & : & S_n-\{0\} & \rightarrow & \mathbb{R}^n \\ & & x^k & \mapsto & -\log(x^k), \end{array}$$

on a

$$f'(x^k)=-\frac{1}{x^k}\Longrightarrow f''(x^k)=\frac{1}{(x^k)^2}>0$$

comme f est convexe, c-à-d

$f\left(\sum_{i=1}^{n}\lambda_ix_i^k\right)\leq\sum_{i=1}^{n}\lambda_if(x_i^k)$, *telle que* $\sum_{i=1}^{n}\lambda_i=1$ *et* $0\leq\lambda_i\leq 1$,

en particulier

$$\lambda_i=\frac{1}{n},\ i=\overline{1,n},$$

alors

$$-\log\left(\sum_{i=1}^{n}\frac{1}{n}x_i^k\right)\leq-\sum_{i=1}^{n}\frac{1}{n}\log\left(x_i^k\right),$$

donc

$$\log\left(\frac{1}{n}\sum_{i=1}^{n}x_i^k\right)\geq\frac{1}{n}\sum_{i=1}^{n}\log\left(x_i^k\right),$$

ce qui implique

$$\log\frac{1}{n}\geq\frac{1}{n}\log\left(\prod_{i=0}^{n}x_i^k\right),$$

alors

$$\frac{1}{n} \geq \exp\log\left(\prod_{i=0}^{n} x_i^k\right)^{\frac{1}{n}},$$

d'où

$$\left(\prod_{i=0}^{n} x_i^k\right)^{\frac{1}{n}} \leq \frac{1}{n},$$

ce qui donne

$$\left(\prod_{i=0}^{n} n x_i^k\right)^{\frac{1}{n}} \leq 1,$$

finalement

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

Ce qui complet la preuve. ■

Lemme 2.2.3. *[20] Si $x \in D_x$, alors*

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

Lemme 2.2.4. *[20] À chaque itération k, la fonction potentiel diminue de la quantité suivante*

$$\forall k \in \mathbb{N}, f(x^{k+1}) - f(x^k) = -\Delta,$$

telles que

$$x^{k+1} = T_k^{-1}\left(y^k\right) \text{ et } \Delta = n\log\frac{c^t D_k e_n}{c^t D_k y^k} + \sum_{i=1}^{n} \log y_i^k.$$

2.2.1 Première amélioration du pas de déplacement

Cette partie est consacré à l'étude de la complexité de l'algorithme de Karmarkar avec le nouveau pas du déplacement de Bouafia et al. [1, 2].

Théorème 2.2.5. *[1] Si*

$$\alpha_M = \frac{5 + nr^2}{5 + (5 + nr^2)\, nr},$$

alors l'algorithme de Karmarkar est converge après $\left(\dfrac{n}{1 - \log 2 + \dfrac{nr^2}{10}} \log\left(\dfrac{c^t e_n}{\varepsilon}\right)\right)$ *itérations.*

Pour prouver ce théorème, on a besoin des lemmes suivants.

Lemme 2.2.6. *[20] Si* $0 < \alpha < \dfrac{1}{nr}$, *alors*

$$\Delta \geq \alpha n^2 r^2 + n\psi_1\left(-\alpha\frac{r}{R}\right) - \psi_1\left(-\alpha nr\right),$$

où $R = \sqrt{\dfrac{n-1}{n}}$.

Lemme 2.2.7.
Pour $\alpha_M = \dfrac{5+nr^2}{5+\left(5+nr^2\right)nr}$ *et* $\alpha_S = \dfrac{1}{1+nr}$, *on a*

a) $\alpha_M > \alpha_S$.

b) $\psi_1\left(-\alpha_M nr^2\right) > \frac{1}{5}\alpha_M n^2 r^4$.

c) $\Delta \geq \psi_1\left(\frac{1}{5}nr\left(5+nr^2\right)\right)$.

Démonstration.

a) On a $r = \sqrt{\dfrac{1}{n(n-1)}} > 0$ et $n > 0$, alors

$$5 + nr^2 > 5,$$

donc

$$\left(5+nr^2\right) + \left(5+nr^2\right)nr > 5 + \left(5+nr^2\right)nr,$$

ce qui donne

$$\left(5+nr^2\right)(1+nr) > 5 + \left(5+nr^2\right)nr,$$

alors

$$\frac{5+nr^2}{5+\left(5+nr^2\right)nr} > \frac{1}{1+nr},$$

d'où

$$\alpha_M > \alpha_S.$$

b) On a d'après le Lemme 2.2.1

$$\psi_1(x) = e_1^t x - \sum_{i=1}^{n}\log(1+x_i) = x - \log(1+x),$$

donc

$$\psi_1(-\alpha_M nr^2) = -\alpha_M nr^2 - \log(1 - \alpha_M nr^2),$$

on a

$$0 < \alpha_M nr^2 < 1,$$

alors

$\log(1 - \alpha_M nr^2) < -\alpha_M nr^2 - \dfrac{(\alpha_M nr^2)^2}{2}$ (d'après le développement de Taylor),

d'où

$$\psi_1(-\alpha_M nr^2) > -\alpha_M nr^2 - (-\alpha_M nr^2) + \frac{(\alpha_M nr^2)^2}{2} = \frac{\alpha_M^2 n^2 r^4}{2},$$

comme $\alpha_M > \alpha_S$,

$$\psi_1(-\alpha_M nr^2) > \frac{\alpha_M n^2 r^4}{2}\alpha_S,$$

on a

$$\alpha_s = \frac{1}{1+nr} = \frac{1}{1+n\sqrt{\dfrac{1}{n(n-1)}}} = \frac{1}{1+\sqrt{\dfrac{n}{n-1}}},$$

pour tout $n \geq 2$,

$$2n \geq 2 + n,$$

alors

$$2 \geq \frac{n}{n-1},$$

donc

$$\alpha_S = \frac{1}{1+\sqrt{\dfrac{n}{n-1}}} \geq \frac{1}{1+\sqrt{2}} > \frac{2}{5},$$

d'où

$$\psi_1(-\alpha_M nr^2) > \frac{\alpha_M n^2 r^4}{2}\frac{2}{5} = \frac{\alpha_M n^2 r^4}{5}.$$

c) Du Lemme 2.2.6 $\Delta \geq \alpha n^2 r^2 + n\psi_1(-\alpha\dfrac{r}{R}) - \psi_1(-\alpha nr)$,

tel que $0 < \alpha < \dfrac{1}{nr}$ et $R = \sqrt{\dfrac{n-1}{n}}$,

on a

$$\frac{R}{r} = n - 1 \Longrightarrow R = \frac{1}{nr},$$

alors

$$\Delta > \alpha n^2 r^2 + n\psi_1(-\alpha n r^2) - \psi_1(-\alpha n r),$$

en particulier $\alpha = \alpha_M$ et d'après b),

$$\Delta > \alpha_M n^2 r^2 + n(\frac{1}{5}\alpha_M n^2 r^4) - \psi_1(-\alpha_M n r),$$

on a

$$\begin{aligned}
&\alpha_M n^2 r^2 + n(\frac{1}{5}\alpha_M n^2 r^4) - \psi_1(-\alpha n r) \\
&\quad = \alpha_M n^2 r^2 + \frac{1}{5} n r^2 (\alpha_M n^2 r^2) + \alpha_M n r + \log(1 - \alpha_M n r) \\
&\quad = (n^2 r^2 + \frac{1}{5} n r^2 (n^2 r^2) + n r)\alpha_M + \log(1 - \alpha_M n r) \\
&\quad = \frac{1}{5}(5n^2 r^2 + n r^2 (n^2 r^2) + n r)\frac{5 + n r^2}{5 + (5 + n r^2)\, n r} + \log(1 - \frac{5 + n r^2}{5 + (5 + n r^2)\, n r} n r) \\
&\quad = \frac{1}{5} n r (5 n r + n r^2 (n r) + 5)\frac{5 + n r^2}{5 + (5 + n r^2)\, n r} + \log\left(\frac{5}{5 + (5 + n r^2 n r)}\right) \\
&\quad = \frac{1}{5} n r (n r (5 + n r^2) + 5)\frac{5 + n r^2}{5 + (5 + n r^2)\, n r} + \log\left(\frac{5}{5 + (5 + n r^2 n r)}\right) \\
&\quad = \frac{1}{5} n r (5 + n r^2) + \log\left(\frac{5}{5 + (5 + n r^2 \frac{n r}{5})}\right) \\
&\quad = \frac{1}{5} n r (5 + n r^2) - \log(1 + (5 + n r^2 \frac{n r}{5})) = \psi_1(\frac{1}{5} n r (5 + n r^2)),
\end{aligned}$$

d'où $\Delta > \psi_1(\frac{1}{5} n r (5 + n r^2))$.

■

Démonstration du Théorème 2.2.5.

D'après le Lemme 2.2.7 c), nous avons

$$\Delta > \psi_1\left(\frac{1}{5} n r (5 + n r^2)\right) = \psi_1\left(n r + \frac{n r}{5} n r^2\right) > \psi_1\left(1 + \frac{1}{5} n r^2\right),$$

car $nr > 1$ et ψ_1 est croissante,

d'autre part

$$\begin{aligned}\psi_1\left(1+\frac{1}{5}nr^2\right)-\psi_1(1) &= 1+\frac{1}{5}nr^2-\log\left(1+1+\frac{1}{5}nr^2\right)-1+\log 2\\ &= \frac{1}{5}nr^2+\log\left(\frac{2}{2+\frac{1}{5}nr^2}\right)\\ &= \frac{1}{5}nr^2-\log\left(1+\frac{1}{10}nr^2\right),\end{aligned}$$

on a

$$\log\left(1+\frac{1}{10}nr^2\right)\leq\frac{1}{10}nr^2,\ car\ \log(1+x)\leq x\ ,\ \forall x\in\,]0,1[\,,$$

donc

$$\psi_1\left(1+\frac{1}{5}nr^2\right)-\psi_1(1)>\frac{1}{5}nr^2-\frac{1}{10}nr^2,$$

d'où

$$\Delta>\psi_1(1)+\frac{1}{10}nr^2, \tag{2.2}$$

d'autre part, du Lemme 2.2.4

$$f(x^k)-f(x^{k-1})=-\Delta,\forall k\in\mathbb{N}^*,$$

de (2.2)

$$f(x^k)-f(x^{k-1})<-\left(\psi_1(1)+\frac{1}{10}nr^2\right),$$

d'ou

$$\sum_{i=1}^{k}\left(f(x^i)-f(x^{i-1})\right)<-\sum_{i=1}^{k}\left(\psi_1(1)+\frac{1}{10}nr^2\right),$$

donc

$$f(x^1)-f(x^0)+f(x^2)-f(x^1)...+f(x^k)-f(x^{k-1})<-k\left(\psi_1(1)+\frac{1}{10}nr^2\right),$$

nous avons

$$f(x^k)-f(x^0)<-k\left(\psi_1(1)+\frac{1}{10}nr^2\right),$$

alors

$$f(x^k) < -k\left(\psi_1(1) + \frac{1}{10}nr^2\right) + f(\frac{e_n}{n}),$$

donc

$$f(x^k) < -k\left(\psi_1(1) + \frac{1}{10}nr^2\right) + n\log\left(c^t\frac{e_n}{n}\right) - \sum_{i=1}^{n}\log\frac{1}{n},$$

ce qui donne

$$\frac{f(x^k)}{n} < \frac{-k\left(\psi_1(1) + \frac{1}{10}nr^2\right) + n\log\left(c^te_n\right)}{n},$$

d'après le Lemme 2.2.3

$$c^tx^k < \exp\left(\frac{-k(\psi_1(1) + \frac{1}{10}nr^2) + n\log\left(c^te_n\right)}{n}\right).$$

L'algorithme converge si $c^tx^k < \varepsilon$ ($\varepsilon > 0$, plus petite), donc on cherche k le nombre des itérations qui vérifie

$$\exp\left(\frac{-k\left(\psi_1(1) + \frac{1}{10}nr^2\right) + n\log\left(c^te_n\right)}{n}\right) < \varepsilon,$$

qui ce implique

$$\log\exp\left(\frac{-k\left(\psi_1(1) + \frac{1}{10}nr^2\right) + n\log\left(c^te_n\right)}{n}\right) < \log\varepsilon,$$

alors

$$-k\left(\psi_1(1) + \frac{1}{10}nr^2\right) < n\log\varepsilon - n\log\left(c^te_n\right),$$

donc

$$k\left(\psi_1(1) + \frac{1}{10}nr^2\right) > n\log\left(\frac{c^te_n}{\varepsilon}\right),$$

comme $\psi_1(1) = 1 - \log 2$, alors

$$k > \frac{n}{1 - \log 2 + \frac{nr^2}{10}}\log\left(\frac{c^te_n}{\varepsilon}\right). \tag{2.3}$$

Donc $c^tx^k \leq \varepsilon$ si k vérifie (2.3). ■

Remarque 2.2.1. *D'après la preuve du Lemme 2.2.7 b), on a* $\Delta > \alpha_M n^2 r^2 + \frac{1}{5}\alpha_M n^3 r^4 - \psi_1(-\alpha_M nr)$, *en effet*
on pose

$$\varphi(\alpha) = \alpha n^2 r^2 + \frac{1}{5}\alpha n^3 r^4 - \psi_1(-\alpha nr),$$

la fonction φ *définie sur* $\left[0, \frac{1}{nr}\right]$ *atteint son maximum en un point unique* $\alpha_M = \dfrac{5+nr^2}{nr(5+nr^2)+5}$, *d'où*

$$\alpha_M = \arg\max_{\left[0,\frac{1}{nr}\right]} \varphi(\alpha) = \frac{5+nr^2}{nr(5+nr^2)+5}$$

$$\begin{aligned}\varphi(\alpha_M) = \max_{\left[0,\frac{1}{nr}\right]} \varphi(\alpha) =& \varphi\left(\frac{5+nr^2}{nr(5+nr^2)+5}\right)\\ =& \psi_1\left(\frac{1}{5}nr(5+nr^2)\right),\end{aligned}$$

d'après le lemme 2.2.7 b)

$$\Delta > \psi_1\left(\frac{1}{5}nr(5+nr^2)\right) = \varphi(\alpha_M) > \varphi(\alpha),$$

i.e.,

$$\Delta > \varphi(\alpha),\ \forall \alpha \in \left]0, \frac{1}{nr}\right[,$$

on conclut que pour $\alpha = \alpha_M$ *on a une réduction maximale de* $f(x^k)$,
$(car\ f(x^k) - f(x^{k-1}) = -\Delta)$.

2.2.2 Deuxième amélioration du pas de déplacement

Dans cette partie, on donne dans le théorème suivant un nouveau pas de déplacement $\frac{1}{2} < \alpha_B < 1$ (Bouafia et all [2, 3]). Qui permet de réduire le nombre d'itérations nécessaires pour la convergence de l'algorithme de Karmarkar.

Théorème 2.2.8. *[3] Pour*

$$\alpha_B = \frac{n}{2n-1},$$

l'algorithme de Karmarkar converge après $\dfrac{n}{1 - \log 2 + \dfrac{nr^2}{2}} \log\left(\dfrac{c^t e_n}{\varepsilon}\right)$ *itérations.*

Remarque 2.2.2.

- $\alpha_B > \dfrac{1}{2}$.
- $\psi_1\left(1+\dfrac{1}{n-1}\right) \geq 1 - \log 2 + \dfrac{nr^2}{2}$, *en effet*

$\psi_1\left(1+\dfrac{1}{n-1}\right) = 1 + \dfrac{1}{n-1} - \log\left(2+\dfrac{1}{n-1}\right)$,
on pose $x = \dfrac{1}{n-1}$,
et comme

$$\log(2+x) - \log 2 = \log\left(\frac{2+x}{2}\right) = \log\left(1+\frac{x}{2}\right) \leq \frac{x}{2}, \text{ pour } 0 < x < 1,$$

alors

$$-\log(2+x) \geq -\frac{x}{2} - \log 2,$$

d'où

$$\psi_1\left(1+\frac{1}{n-1}\right) \geq 1 + \frac{1}{n-1} - \frac{1}{2(n-1)} - \log 2,$$

d'autre part

$$r = \frac{1}{\sqrt{n(n-1)}} \Rightarrow r^2 \frac{1}{n(n-1)} \Rightarrow nr^2 = \frac{1}{n-1},$$

d'où

$$\psi_1\left(1+\frac{1}{n-1}\right) \geq 1 + nr^2 - \frac{nr^2}{2} - \log 2,$$

alors

$$\psi_1\left(1+\frac{1}{n-1}\right) \geq 1 + \frac{nr^2}{2} - \log 2.$$

Pour prouver le Théorème 2.2.8, on a besoin du lemme suivant.

Lemme 2.2.9. *[3] Pour* $0 < \alpha < 1$ *et* y^k *défini par l'algorithme de Karmarkar, nous avons*

a) $\sum_{i=1}^{n} \log(1 - \alpha n r d_i^k) \geq (n-1)\log(1 + \dfrac{\alpha}{n-1}) + \log(1-\alpha)$.

b) *Pour* $\alpha = \alpha_B = \dfrac{n}{2n-1}$, *on a* $\Delta \geq 1 - \log 2 + \dfrac{nr^2}{2}$.

Démonstration.

a) D'après le Lemme 2.1.5, on a

$$\frac{1}{n\left(\prod_{i=1}^{n} y_i^k\right)^{\frac{1}{n}}} \leq \frac{1}{1+\dfrac{\alpha}{n-1}} \left(\frac{1+\dfrac{\alpha}{n-1}}{1-\alpha}\right)^{\frac{1}{n}}, \quad n \geq 3$$

alors

$$n\left(\prod_{i=1}^{n} y_i^k\right)^{\frac{1}{n}} \geq \left(1+\frac{\alpha}{n-1}\right) \frac{(1-\alpha)^{\frac{1}{n}}}{\left(1+\dfrac{\alpha}{n-1}\right)^{\frac{1}{n}}},$$

d'où

$$\left(\prod_{i=1}^{n} y_i^k\right)^{\frac{1}{n}} \geq \frac{1}{n} \frac{1}{\left(1+\dfrac{\alpha}{n-1}\right)^{\frac{1-n}{n}}} (1-\alpha)^{\frac{1}{n}},$$

donc

$$\prod_{i=1}^{n} y_i^k \geq \frac{1}{n^n}\left(1+\frac{\alpha}{n-1}\right)^{n-1}(1-\alpha),$$

alors

$$\log\left(\prod_{i=1}^{n} y_i^k\right) \geq \log\left[\frac{1}{n^n}\left(1+\frac{\alpha}{n-1}\right)^{n-1}(1-\alpha)\right],$$

ce qui donne

$$\sum_{i=1}^{n} \log\left(y_i^k\right) \geq -n\log n + (n-1)\log\left(1+\frac{\alpha}{n-1}\right) + \log(1-\alpha), \tag{2.4}$$

on a

$$y^k = \frac{e_n}{n} - \alpha r d^k,$$

alors

$$y_i^k = \frac{1}{n} - \alpha r d_i^k,$$

ce qui donne

$$\log(y_i^k) = \log\left(\frac{1-\alpha n r d_i^k}{n}\right),$$

d'où

$$\sum_{i=1}^{n} \log(y_i^k) = -n\log n + \sum_{i=1}^{n} \log(1-\alpha n r d_i^k),$$

alors (2.4) devient

$$\sum_{i=1}^{n} \log(1 - \alpha n r d_i^k) \geq (n-1) \log\left(1 + \frac{\alpha}{n-1}\right) + \log(1-\alpha).$$

D'où le résultat.

b) D'après Lemme 2.2.4, on a

$$\Delta = n \log \frac{c^t D_k e_n}{c^t D_k y^k} + \sum_{i=1}^{n} \log y_i^k$$

et du Lemme 2.1.3, on a

$$\frac{c^t D_k y^k}{c^t D_k \dfrac{e_n}{n}} \leq \left(1 - \alpha \frac{1}{n-1}\right) = \left(1 - \alpha \frac{r}{R}\right) = \left(1 - \alpha n r^2\right),$$

alors

$$\frac{c^t D_k y^k}{c^t D_k e_n} = \frac{1}{n} \left(\frac{c^t D_k y^k}{c^t D_k \dfrac{e_n}{n}}\right) \leq \frac{1}{n} \left(1 - \alpha \frac{1}{n-1}\right) = \frac{1}{n} \left(1 - \alpha \frac{r}{R}\right),$$

d'où

$$\frac{c^t D_k e_n}{c^t D_k y^k} \geq n \left(\frac{1}{1 - \alpha \dfrac{1}{n-1}}\right),$$

alors

$$n \log \left(\frac{c^t D_k e_n}{c^t D_k y^k}\right) \geq n \log n + n \log \left(\frac{1}{1 - \alpha \dfrac{1}{n-1}}\right),$$

ce qui donne

$$\Delta = n \log \frac{c^t D_k e_n}{c^t D_k y^k} + \sum_{i=1}^{n} \log y_i^k \geq n \log n + n \log \left(\frac{1}{1 - \alpha \dfrac{1}{n-1}}\right) + \sum_{i=1}^{n} \log y_i^k,$$

donc

$$\Delta \geq n \log n + n \log \left(\frac{1}{1 - \alpha \dfrac{1}{n-1}}\right) + \left[-n \log n + \sum_{i=1}^{n} \log(1 - \alpha n r d_i^k)\right],$$

d'après le Lemme 2.2.9 a) on a

$$\Delta \geq n \log\left(\frac{1}{1-\alpha\frac{1}{n-1}}\right) + (n-1)\log\left(1+\frac{\alpha}{n-1}\right) + \log(1-\alpha) + \log\left(\frac{1}{1-\alpha\frac{1}{n-1}}\right) - \log\left(\frac{1}{1-\alpha\frac{1}{n-1}}\right),$$

donc

$$\Delta \geq (n-1)\log\left(\frac{1+\frac{\alpha}{n-1}}{1-\frac{\alpha}{n-1}}\right) - \log\left(1-\frac{\alpha}{n-1}\right) + \log(1-\alpha), \tag{2.5}$$

pour $0 \leq x < 1$, on a

$$\log\left(\frac{1+x}{1-x}\right) \geq 2x \; et \; -\log(1-x) \geq x,$$

alors

$$\log\left(\frac{1+\frac{\alpha}{n-1}}{1-\frac{\alpha}{n-1}}\right) \geq 2\left(\frac{\alpha}{n-1}\right) \; et \; -\log\left(1-\frac{\alpha}{n-1}\right) \geq \frac{\alpha}{n-1},$$

l'inégalité (2.5) devient

$$\Delta \geq 2(n-1)\left(\frac{\alpha}{n-1}\right) + \frac{\alpha}{n-1} + \log(1-\alpha),$$

d'où

$$\Delta \geq \left(2+\frac{1}{n-1}\right)\alpha + \log(1-\alpha), \tag{2.6}$$

pour $\alpha = \alpha_B = \dfrac{n}{2n-1}$, (2.6) devient

$$\Delta \geq \left(2+\frac{1}{n-1}\right)\frac{n}{2n-1} + \log\left(1-\frac{n}{2n-1}\right),$$

on a

$$\begin{aligned}\left(2+\frac{1}{n-1}\right)\frac{n}{2n-1}+\log\left(1-\frac{n}{2n-1}\right) &= \frac{2n-1}{n-1}\frac{n}{2n-1}+\log\left(\frac{n-1}{2n-1}\right)\\ &= \frac{n}{n-1}-\log\left(\frac{2n-1}{n-1}\right)\\ &= 1+\frac{1}{n-1}-\log\left(2+\frac{1}{n-1}\right)\\ &= \psi_1\left(1+\frac{1}{n-1}\right),\end{aligned}$$

alors

$$\Delta \geq \psi_1\left(1+\frac{1}{n-1}\right),$$

d'après la Remarque 2.2.2, on a

$$\Delta \geq 1-\log 2+\frac{nr^2}{2}.$$

■

Démonstration du Théorème 2.2.8.

D'après le Lemme 2.2.4

$$f(x^{k-1})-f(x^k)=\Delta, \forall k\in\mathbb{N}^*,$$

alors

$$f(x^{k-1})-f(x^k)\geq 1-\log 2+\frac{nr^2}{2}, \forall k\in\mathbb{N}^*,$$

par conséquent

$$\sum_{i=1}^{k} f(x^i)-f(x^{i-1}) \leq -\sum_{i=1}^{k}\left(1-\log 2+\frac{nr^2}{2}\right),$$

ce qui donne

$$f(x^k)-f(x^0)\leq -k\left(1-\log 2+\frac{nr^2}{2}\right),$$

donc

$$f(x^k)\leq -k\left(1-\log 2+\frac{nr^2}{2}\right)+f(x^0),$$

comme $x^0=\dfrac{e_n}{n}$ et $f(x)=n\log\left(c^t x\right)-\sum_{i=1}^{n}\log x_i$, on a

$$f(x^0)=n\log\left(c^t e_n\right),$$

alors

$$f(x^k) \leq -k\left(1 - \log 2 + \frac{nr^2}{2}\right) + n\log\left(c^t e_n\right),$$

$$\frac{f(x^k)}{n} \leq \frac{-k\left(1 - \log 2 + \frac{nr^2}{2}\right) + n\log\left(c^t e_n\right)}{n},$$

d'après le Lemme 2.2.3, on a

$$c^t x \leq \exp\left(\frac{f(x)}{n}\right),$$

d'où

$$c^t x \leq \exp\left(\frac{-k\left(1 - \log 2 + \frac{nr^2}{2}\right) + n\log\left(c^t e_n\right)}{n}\right).$$

Rappelons que l'algorithme converge lorsque $c^t x^k$ est inférieur à ε ($\varepsilon > 0$ assez petit). Il suffit de chercher k vérifie

$$\exp\left(\frac{-k\left(1 - \log 2 + \frac{nr^2}{2}\right) + n\log\left(c^t e_n\right)}{n}\right) < \varepsilon,$$

ce qui donne

$$\left(\frac{-k\left(1 - \log 2 + \frac{nr^2}{2}\right) + n\log\left(c^t e_n\right)}{n}\right) < \log\varepsilon,$$

alors

$$-k\left(1 - \log 2 + \frac{nr^2}{2}\right) < n\log\varepsilon - n\log\left(c^t e_n\right),$$

ce qui conduit

$$-k\left(1 - \log 2 + \frac{nr^2}{2}\right) < n\log\left(\frac{\varepsilon}{c^t e_n}\right),$$

alors

$$k > \frac{n}{1 - \log 2 + \frac{nr^2}{2}}\log\left(\frac{c^t e_n}{\varepsilon}\right). \tag{2.7}$$

Donc $c^t x^k \leq \varepsilon$, si k vérifie (2.7). ■

Chapitre 3

Expérimentations numériques

Dans ce chapitre, nous présentons des tests numériques dans un cadre comparatif entre les trois pas de déplacement $\alpha_S,\ \alpha_M$ *et* α_B en utilisant l'algorithme de Ye-Lustig.
Les exemples testés sont pris de la littérature [2, 6] et sont réalisés en langage MATLAB R2008b sur Intel(R) Core(TM) i3-3110M CPU @ 2.40 GHz avec RAM 4.00 Go, avec une précision $\varepsilon \in [10^{-6}, 10^{-4}]$.
On désigne par

- α_S le pas de déplacement de Schrijver.
- α_M le pas de déplacement de (Bouafia et Benterki).
- α_B le pas de déplacement de (Bouafia, Benterki et Yassine).
- k : Le nombre d'itérations.
- t : Le temps de calcul en secondes.
- (m, n) : la taille des exemples testés.

3.1 Exemples à taille fixe

Voici le programme de l'algorithme de Ye-Lustig.

```
clear all;
clc;
[ A,b,c,x0 ] = donne( );
[m n]=size(A);
eps=0.000001;
r=1/(sqrt(n*(n-1)));
alpha=n/(2*n-1);
nP0=(1+eps)*abs(c'*x0);
nPk=nP0;
xk=x0;
k=0;
tic
while nPk/abs(c'*xk)>eps
    Dk=diag(xk);
    Bk=[A*Dk -b];
    u=Bk*Bk';
    j=eye(n+1)-(Bk'*inv(u))*Bk;
    g=[Dk*c;-c'*xk];
    Pk=(j*g);
    dk=Pk/norm(Pk);
    yk=(ones(n+1,1)/(n+1))-alpha*r*dk;
    Y=yk(n+1);
    yn=yk(1:n);
    xk=(Dk*yn)/Y;
    z=c'*xk;
    nPk=norm(Pk);
    k=k+1;
end
toc
disp(k);
disp(z);
```

Exemple 01

$A = \begin{pmatrix} -1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 1, 2 \end{pmatrix}^t$, $c = \begin{pmatrix} -2, -4, 0 \end{pmatrix}^t$.

- $\varepsilon = 10^{-6}$.

- La solution strictement réalisable initiale

$x^0 = (0.193812, 1.193810, 0.612381)$.

- La solution optimale trouver pour les trois pas de déplacement

$x^* = (0.5000, 1.5000, 0.0000)$.

- Valeur optimale trouvée pour les trois pas de déplacement

$z^* = -7.0000$.

Exemple 02

$$A = \begin{pmatrix} 1 & 1.2 & 1 & 1.8 & 0 \\ 3 & -1 & 1.5 & -2 & 1 \\ -1 & 2 & -3 & 4 & 2 \end{pmatrix}, b = \begin{pmatrix} 9.31, 5.45, 6.60 \end{pmatrix}^t, c = \begin{pmatrix} 1, -1.5, 2, 1.5, 3 \end{pmatrix}^t.$$

- $\varepsilon = 10^{-6}$.
- Solution strictement réalisable initiale

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

- La solution optimale trouver pour les trois pas de déplacement

$x^* = (3.4194, 4.9089, 0.0000, 0.0000, 0.1008)$.

- Valeur optimale trouvée pour les trois pas de déplacement

$z^* = -3.6415$.

Exemple 03

$$A = \begin{pmatrix} 1 & -1 & 1.9 & 1.25 & 1.2 & 0.4 & -0.7 & 1.06 & 1.5 & 1.05 \\ 1.3 & 1.2 & 0.15 & 2.15 & 1.25 & 1.5 & 0.4 & 1.52 & 1.3 & 1 \\ 1.5 & -1.1 & 3.5 & 1.25 & 1.8 & 2 & 1.95 & 1.2 & 1 & -1 \end{pmatrix}, b = \begin{pmatrix} 11.651, 16.672, 21.295 \end{pmatrix}^t,$$

$c = \begin{pmatrix} -0.5, -1, 0, 0, -0.5, 0, 0, -1, -0.5, -1 \end{pmatrix}^t$.

- $\varepsilon = 10^{-6}$.

• Solution strictement réalisable initiale

$$x^0 = (1.475426, 0.868316, 1.752137, 1.637813, 1.560784,$$
$$1.926482, 1.538978, 1.416004, 1.223422, 0.728767).$$

• Solution optimale trouvée pour les trois pas de déplacement

$x^* = (0.0000, 10.3911, 10.1172, 0.0000, 0.0000, 0.000, 0.0000, 0.0000, 0.0000, 2.6851)$.

• Valeur optimale trouvée pour les trois pas de déplacement

$z^* = -13.0762$.

Exemple 04

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

$$b = \Big(5, 4, -4.1779, 0, 0, 0, 0\Big)^t,\ c = \Big(-1, -3, 1, -1, 0, 0, 0, 0, 0, 0, 0\Big)^t.$$

• $\varepsilon = 10^{-6}$.

• Solution strictement réalisable initiale

$$x^0 = (0.130437, 1.421849, 0.709742, 0.000010, 1.316122, 0.055277,$$
$$0.082906, 0.130437, 1.421849, 0.709742, 0.000010).$$

• Solution optimale trouvée pour les trois pas de déplacement

$x^* = (0.0000, 1.6889, 0.6223, 0.0000, 1.0000, 0.0000, 0.0000, 0.0000, 1.6889, 0.6223, 0.0000)$.

• Valeur optimale trouvée pour les trois pas de déplacement

$z^* = -4.4444$.

Le tableau suivant résume les résultats obtenus pour les différentes pas de déplacement.

Exemple	Taille(m,n)	Alternative utilisant α_S	Alternative utilisant α_M	Alternative utilisant α_B
01	(2,3)	$k = 13$ $t = 0.007550s$	$k = 12$ $t = 0.007514s$	$k = 8$ $t = 0.007362s$
02	(3,5)	$k = 27$ $t = 0.007983s$	$k = 26$ $t = 0.007848s$	$k = 23$ $t = 0.007776s$
03	(3,10)	$k = 45$ $t = 0.009151s$	$k = 45$ $t = 0.009292s$	$k = 42$ $t = 0.008974s$
04	(7,11)	$k = 32$ $t = 0.008642s$	$k = 31$ $t = 0.008504s$	$k = 29$ $t = 0.008315s$

3.2 Exemples à taille variable

Exemple cube

$n = 2m$,

$$A(i,j) = \begin{cases} 1 \ si \ i = j \ ou \ j = i + m, \\ 0 \ sinon. \end{cases}$$

$c(i) = -1$, $c(i + m) = 0$ et $b(i) = 2$ pour $i = \overline{1, m}$.

• Solution strictement réalisable initiale $x^0 = (1, 1, \ldots, 1)^t$.

Programme de la méthode de Ye-Lustig pour l'exemple de cube.

```
clear all;clc;
m=2;n=2*m;
 for i=1:m
     for j=1:n
         if(( i==j) | (j==i+m))
             A(i,j)=1;
         else
             A(i,j)=0;
         end
     end
end
for i=1:m
     c(i)=-1;c(i+m)=0;b(i)=2;
end
r=1/(sqrt(n*(n-1)));alpha=n/(2*n-1);eps=0.00001;x0=ones(n,1);
nP0=(1+eps)*abs(c*x0);nPk=nP0;xk=x0;k=0;
tic
while nPk/abs(c*xk)>eps
     Dk=diag(xk);Bk=[A*Dk -b'];
     u=Bk*Bk';
     j=eye(n+1)-(Bk'*inv(u))*Bk;
     g=[Dk*c';-c*xk];
     Pk=(j*g);dk=Pk/norm(Pk);
     yk=(ones(n+1,1)/(n+1))-alpha*r*dk;
     Y=yk(n+1);yn=yk(1:n);
     xk=(Dk*yn)/Y;
     z=c*xk;
     nPk=norm(Pk);
     k=k+1;
end
toc
disp(k)
disp(z)
```

- La valeur optimale $z^* = -2m$.

- Solution optimale trouvée $x_i^* = \begin{cases} 2 & si \quad i = \overline{1,m}, \\ 0 & si \quad i = \overline{m+1,n}. \end{cases}$

Le tableau suivant résume les résultats obtenus pour les différentes pas de déplacement.

Taille(m,n)	Alternative utilisant α_S	Alternative utilisant α_M	Alternative utilisant α_B
(2,4)	$k = 18$ $t = 0.007462s$	$k = 18$ $t = 0.007374s$	$k = 15$ $t = 0.007265s$
(10,20)	$k = 47$ $t = 0.010340s$	$k = 47$ $t = 0.010418s$	$k = 45$ $t = 0.009743s$
(35,70)	$k = 85$ $t = 0.038125s$	$k = 84$ $t = 0.040212s$	$k = 84$ $t = 0.029355s$
(100,200)	$k = 136$ $t = 0.473646s$	$k = 136$ $t = 0.455569s$	$k = 135$ $t = 0.50846s$
(150,300)	$k = 163$ $t = 1.690317s$	$k = 163$ $t = 1.700121s$	$k = 163$ $t = 1.801860s$
(200,400)	$k = 185$ $t = 4.251249s$	$k = 185$ $t = 4.221053s$	$k = 185$ $t = 4.246701s$
(400,800)	$k = 252$ $t = 40.359043s$	$k = 252$ $t = 40.019618s$	$k = 252$ $t = 40.315206s$

Commentaires

• Les tests numériques qu'on a effectués montrent que la contribution de Bouafia et all a permis une amélioration légère du nombre d'itération dans les exemples à faibles tailles. On a pu réduire de 1 et 2 le nombre des itérations, par contre, le temps d'exécution étant négligeable dans les trois alternatives.

• Dans l'exemple à grande taille, plus la taille augmente plus les trois alternatives coïncident en nombre d'itérations. Cela revient à l'égalité des trois pas α_S, α_M et α_B lorsque n tends vers l'infini.

• On montre que le nouveau pas de déplacement (α_B)a introduit une amélioration des résultats de convergence polynomiale, est une petite réduction du nombre d'itération.

• Ce travail est consolidé par des tests numériques comparative réalisés sur l'algorithme de Ye-Lustig s'avère que le pas de déplacement (α_B) est le meilleur.

Bibliographie

[1] **D. Benterki, M. Bouafia**, Improving complexity of Karmarkar's approach for linear programming, Journal of Numerical Analysis and Approximation Theory, 43(2), 159-167, (2015).

[2] **M. Bouafia**, Étude asymptotique des méthodes de points intérieurs pour la programmation linéaire, Université du Havre, Thèse de doctorat de mathématiques appliquées, Français, (2016).

[3] **M. Bouafia, D. Benterki, A. Yassine**, A new efficient short-step projective interior point method for linear programming, Soumise Ukrainian Mathematical Journal, (2014).

[4] **J.P. Crouzeix, B. Merikhi**, A logarithm barrier method for semidefinite programming, RAIRO-Operations Research, 42 (2008) 123–139.

[5] **G. B. Dantzig**, Linear programming and extensions, Princeton University Press, Princeton, N. J, (1963).

[6] **E. Djeffel**, Etude de quelques algorithme de points intérieurs pour la programmation convexe, Thèse de doctorat de mathématiques appliquées de l'université Hadj Lakhdar de Batna (2013).

[7] **D. den Hertog**, Interior point approach to linear quadratic and convex programming, Kluwer Academic Publishers, Dordrecht, 1994.

[8] **N. K. Karmarkar**, A new polynomial-time algorithm for linear programming, in : Proceedings of the 16^{th} Annual ACM Symposium on Theory of Computing, 4, 373-395, (1984).

[9] **Z. Kebbiche**, Mise en oeuvre d'une méthode de trajectoire centrale pour les problèmes complémentaires linéaires monotones, Thèse de magistère, Département de Mathématiques, Université Ferhat Abbas, Sétif-1, 1997.

[10] **Z. Kebbiche**, Étude et extentions d'algorithmes de points intérieurs pour la programmation non linéaire, Thèse doctorat d'état, Université Farhat Abbas - Setif, (2007).

[11] **A. Keraghel**, Etude adaptative et comparative des principales variantes dans l'algorithme de Karmarkar, Thèse de doctorat, Université Joseph Fourier Grenoble I, France, (1989).

[12] **A. Keraghel, D. Benterki**, Sur les performances de l'algorithme de Karmarkar pour la programmation linéaire, Revue Roumaine des Sciences Techniques-Mécanique Appliquées, 46(1), 87-96, (2001).

[13] **I. J. Lustig**, A pratical approach to Karmarkar's algorithm, Technical report sol 85-5 Systems optimizatioin laboratory, Dept of operations research Stanford univ, Stanford California 94305, (1985).

[14] **L. Menniche**, Etude théorique et numérique d'une classe de méthodes de points intérieurs pour la programmation linéaire, Thése de doctorat, Université de Setif, (2017).

[15] **M. Minoux**, Programmation mathématique : Théorie et algorithmes, Tome 1 Dunod, Paris, (1983).

[16] **R. Naseri, A. Valinejad**, An extended variant of Karmarkar's interior point algorithm, Applied Mathimatics and Computation 184, 737-742, (2007).

[17] **Y.E. Nestrov, A. Nemiroveski**, Interior-point polynomial algorithms in convex programming, SIAM, 1994.

[18] **M. Padberg**, A different convergence proof of the projective method for linear programming, New York University, (1985).

[19] **C. Roos**. A full-Newton step O(n) infeasible interior-point method for linear optimization. SIAM Journal on optimization, 16 (2006) 1110-1136.

[20] **C. Roos, T. Terlaki, J. Vial**, Optimization theory and algorithm for linear programming optimization, Princton university, (2001).

[21] **A. Schrijver**, Theory of linear and integer programming, John Wiely & Sons, New York, (1986).

[22] **Y. Ye, E. Tse**, An extension of Karmarkar's projective algorithm for convex quadratic programming, Mathematical programming, 44 (1989) 157–179.