République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et

De la recherche scientifique

Université de Blida

Mémoire de Fin d'Etudes

en vue de l'obtention du diplôme de master

Spécialité : Recherche Opérationnelle

Thème

Méthodes ellipsoïdales et projectives de résolution des problèmes linéaires

(Khachiyan et Karmarkar)

Proposé et dirigé par :

- **MR. M.BLIDIA**

Préparé par :

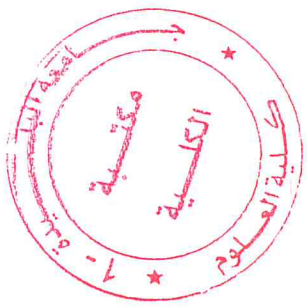
- Daoudi Djaouida
- Bachsaiss Imene

Membres de jury :

M.R. F.Hannane

M.R. M.Chellali

M.R. M.Bougerrara



Promotion 2015\2016

DIDICACE

C'est avec toute l'ardeur de mes sentiments que je dédie ce modeste travail qui est le fruit de ma profonde reconnaissance à : Mes parents, que Dieu les garde et les protège.

Mes chers frères et Ma chère sœur Samia.

Mes Familles : Bachsaiss et Serir.

Mon binôme : Djaouida et mes amis des études. Tous ceux que j'aime dans le monde.

Imene

DIDICACE

Je dédie le fruit de ce modeste travail comme un geste de gratitude à :

La mémoire de ma chère sœur Noudjoud

Mes très chers parents, qui m'ont soutenue, encouragée pour que je puisse mener à bien mes études, et qui ont attendu ce jour avec impatience.

Mes chers frères et mes adorables sœurs

. A toute la famille : Daoudi.

Mon binôme : Imene

Mes enseignants et mes amies des études.

A tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail

Djaouida

REMERCIMENT

A Mr M.BLIDIA

Nous avons eu l'honneur d'être parmi vos élèves et de bénéficier de votre riche enseignement.

Vos qualités pédagogiques et humaines sont pour nous un modèle.

Votre compétence, votre encadrement ont toujours suscité notre profond respect.

Votre gentillesse, et votre disponibilité permanente ont toujours suscité notre admiration. Veuillez bien monsieur recevoir notre remerciement pour le grand honneur que vous nous avez fait d'accepter l'encadrement de ce travail.

Djaouida et Imene

REMERCIMENT

Nous désirons aussi remercier tous nos enseignants qui au cours de nos études, ne nous ont jamais épargné leur enseignement, conseils et encouragements.

Aux membres de jury, d'avoir bien voulu présider et examiner ce travail,

A Asma et Youcef qui nous ont aidées à réaliser l'interface graphique

A tous ceux qui, de près ou de loin, ont permis à ce travail d'aboutir

Djaouida et Imene

Résumé

Dans cette thèse, nous avons fait une étude adaptative et comparative de trois méthodes qui permettent la résolution d'un programme linéaire à savoir :

- *La méthode de Dantzig (simplexe).*
- *La méthode de Khachiyan (ellipsoïdale).*
- *La méthode de Karmarkar (projective).*

La méthode du simplexe, en générale n'est pas polynomiale comme ont signalé Klee et Minty en fournissant un exemple de programme linéaire pour lequel la méthode du simplexe croit en exponentiel.

Ceci a motivé des chercheurs pour l'établissement de méthodes polynomiales pour résoudre des programmes linéaires.

Ainsi Khachiyan et Karmarkar ont fourni dans les années 80 des algorithmes théoriquement polynomiaux.

Dans ce mémoire, nous implémentons ces méthodes et nous comparons leur efficacité sur des exemples concrets. La méthode du simplexe reste en pratique la plus utilisée, vu sa simplicité dans son exécution et la complexité de la mise en œuvre des méthodes ellipsoïdales et projectives.

Mots clés :

La programmation linéaire, algorithme polynomial, méthode du simplexe, Méthode des ellipsoïdes, transformation affine, transformation projective.

Abstract

We propose an adaptive and comparative study based on three known approaches to resolve the linear program such us:

- *Dantzig method (Simplexe).*
- *Khachiyan method (Ellipsoidal).*
- *Karmarkar method (Projective).*

Simplex approach results are usually not polynomial, for solving linear programs, as reported by Klee and Minty. We provide an example for which the Simplex method increases exponentially. This fact motivate researcher to look for polynomial method for solving linear programs.

Therefore Khachiyan and Karmarkar provided in the 80^{th} theoretically polynomial algorithms.

In this work, we implemented Khachiyan and Karmarkar methods, and we compare their efficiency on well-known examples. In Practice, the Simplex method is still the most used, due to its simplicity and the complexity of the implementation of the ellipsoidal and projective methods.

Keywords:

Linear programming, polynomial algorithm, Simplex method, ellipsoidal method, affine transformation, projective transformation.

ملخص

تناولنا في أطروحتنا دراسة مقارنة واستنباطية لثلاث طرق مهمة لحل مسائل البرمجة الخطية وهي:

طريقة" دان تزيغ" (البسيطة)

طريقة "كاشيان"(الإهليجية)

طريقة" كارمركار"(الإسقاطية)

قدم كل من" كلي "و"مينتي" مثال تزايد فيه وقت الحل بالطريقة البسيطة بشكل أسي و بالتالي هي ليست كثيرة حدود بصفة عامة.

حفزت هذه النتائج تطور البحث في اتجاه اخر لإيجاد دوال كثيرات الحدود لحل مسائل البرمجة الخطية.

وفي هذا السياق توصل "كاشيان"و"كارمركار" إلى إيجاد خوارزميات كثيرة الحدود فرضيا.

في الأطروحة المقدمة، تم برمجة ومقارنة فاعلية هذه الطرق من خلال مجموعة من الأمثلة.

رغم ما توصلا إليه الطريقة البسيطة تظل الأكثر تطبيقا نظرا لتعقيد الطريقتين الإهليجية والإسقاطية.

كلمات المفتاح:

البرمجة الخطية، خوارزمية كثيرة حدود، الطريقة البسيطة، طريقة كاشيان، التحويل الإهليجي، طريقة كارمركار، التحويل الإسقاطي.

Sommaire

Introduction générale :

Beaucoup de problèmes concrets, de notre monde réel peuvent être modélisés mathématiquement. En effet, une fois un problème modélisé sous la forme d'équations linéaires, des méthodes assurent la résolution du problème de manière exacte.

En 1947, G.B. Dantzig [4] a proposé la méthode du simplexe : une technique de résolution très efficace pour des problèmes de programmation linéaire.

Depuis ce temps, la programmation linéaire a suscité un grand intérêt chez les chercheurs qui ont écrit des centaines de livres et ont publié des milliers d'articles sur le sujet.

Bien que la complexité de cette méthode soit exponentielle [16], dans la pratique elle s'est montrée très efficace et d'une grande utilité.

En 1979, le mathématicien soviétique L.G. Khachiyan [7] a mis en œuvre le premier algorithme polynomial pour la programmation linéaire : l'algorithme des ellipsoïdes. Toutefois, l'algorithme proposé s'est révélé complètement inefficace dans la pratique.

En 1984, N. Karmarkar [13] a révolutionné le domaine de la programmation linéaire en mettant en œuvre un algorithme polynomial basé sur les méthodes de points intérieures.

Celui-ci s'est avéré un compétiteur sérieux de l'algorithme du simplexe. Depuis, une recherche intense s'est déclenchée dans ce domaine et a donné comme résultat une grande variété d'algorithmes de ce type.

Ce travail se veut une synthèse de ces différentes méthodes: nous définissons leurs principes, nous proposons leurs algorithmes et nous faisons une étude comparative entre eux.

Ce mémoire est structuré comme suit :

Dans le premier chapitre, nous rappelons les résultats fondamentaux en programmation linéaire et plus particulièrement ceux concernant la dualité.

Dans le deuxième chapitre on trouve la présentation générale de la méthode du simplexe et son algorithme avec sa complexité (contre-exemple de Klee et Minty).

Le troisième chapitre est consacré à la présentation de la méthode ellipsoïdale (Khachiyan).

Dans le quatrième chapitre, nous exposons la méthode projective de Karmarkar. Et enfin le dernier chapitre est destiné à l'implémentation de ces méthodes suivi d'une étude comparative entre les différentes méthodes étudiées.

Chapitre I

NOTIONS FONDAMENTALES DE LA PROGRAMMATION LINEAIRE

I.1. Introduction :

La programmation linéaire peut être considérée comme faisant partie d'un grand développement révolutionnaire qui a donné à l'humanité la possibilité d'énoncer des objectifs généraux et d'établir un chemin de décisions détaillées à prendre afin de "mieux" atteindre ses objectifs lorsqu'ils sont confrontés à des situations pratiques d'une grande complexité.

Nos outils pour ce faire sont façons de formuler des problèmes du monde réel en termes mathématiques détaillées (modèles), les techniques de résolution des modèles (algorithmes) et les moteurs pour exécuter les étapes d'algorithmes (ordinateurs et logiciels) [5].

I.2. Modélisation :

Modéliser un problème consiste à identifier:

- Les variables intrinsèques (inconnues)
- Les différentes contraintes aux quelles sont soumises ces variables
- L'objectif visé (optimisation).

Les étapes de modélisation :

La détermination des variables de décision :

Les variables $x_1, x_2, \dots, x_n$ sont appelées des variables de décision ou variables réelles du problème.

La détermination des contraintes :

La contrainte peut être assimilée à un obstacle tel que les limitations techniques scientifiques, économiques, les lois de la nature, les délais, etc.

La détermination de la fonction objectif (économique) :

La fonction objectif est une fonction qui permet de déterminer l'optimum (max de profit /min de Coût).

Le but du problème de programmation linéaire est alors de minimiser ou de maximiser cette fonction jusqu'à l'optimum, par différents procédés comme par exemple la méthode graphique dans le cas de deux variables ou trois variables à la limite.

La fonction objectif est une forme linéaire en fonction des variables de décision de type:

$\text{Max(ou Min)} Z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$ où les coefficients $c_1, c_2, \dots, c_n$ doivent avoir une valeur bien déterminée et peuvent être positifs, négatifs ou nuls.

Exemple d'un problème de production :

Voici un petit exemple traitable par la programmation linéaire.

Une usine fabrique 2 produits P_1et P_2 nécessitant des ressources d'équipement, de main d'œuvre et de matières premières disponibles en quantité limitée.

	P_1	P_2	Disponibilité
Equipement	3	9	81
Main d'œuvré	4	5	55
Matière première	2	1	20

P_1 et P_2 rapportent à la vente 6 euros et 4 euros par unité.

Quelles quantités (non entières) de produits P_1et P_2 doit produire l'usine pour maximiser le bénéfice total venant de la vente des 2 produits ?

Variables : x_1 et x_2 sont les quantités de produits P_1 et P_2 fabriqué $(x_1, x_2) \in \mathbb{R}^2$

Fonction objectif à maximiser : la fonction objectif Z correspond au bénéfice total

$\text{Max } Z = 6x_1 + 4x_2$

Contraintes :

Disponibilité de chacune des ressources :

$3x_1 + 9x_2 \leq 81$

$4x_1 + 5x_2 \leq 55$

$2x_1 + x_2 \leq 20$

Positivités des variables : $x_1, x_2 \geq 0$

En résumé, le problème de production se modélise sous la forme d'un programme linéaire :

$\text{Max } Z = 6x_1 + 4x_2$

$3x_1 + 9x_2 \leq 81$

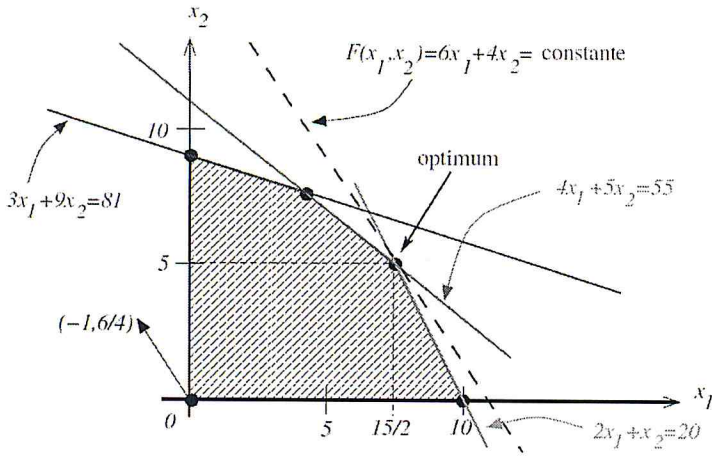
$4x_1 + 5x_2 \leq 55$

$2x_1 + x_2 \leq 20$

$x_1, x_2 \geq 0$

Résolution graphique :

Les contraintes où apparaissent des inégalités correspondent géométriquement à des demi-plans. Intersection de ces demi-plans = ensemble des variables satisfaisant à toutes les contraintes. L'ensemble des contraintes est un polygône convexe.



Fonction objectif $\text{Max } Z = 6x_1 + 4x_2$ peut être vue comme une droite de vecteur directeur $(-1, 6/4)$.

Pour déterminer Max Z, on fait "glisser" la droite (translation parallèle à la direction de la droite) du haut vers le bas jusqu'à rencontrer l'ensemble des variables satisfaisant les contraintes (solution optimale).

$(x_1, x_2) = (15/2,5)$ avec $\text{Max}(Z) = 65$

On remarque que le maximum de Z est atteint en un sommet du polygône convexe des contraintes.

I.3. Formes générales d'un programme linéaire :

Forme canonique mixte :

$$\text{Max}(Z) = c_1x_1 + c_2x_2 + \cdots + c_nx_n = \sum_{j=1}^{n} c_j\, x_j$$

Contraintes d'inégalités

$$\forall\, i \in I_1, \sum_{j=1}^{n} a_{ij}x_j = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$$

Contraintes d'égalités :

$$\forall\, i \in I_2, \sum_{j=1}^{n} a_{ij}x_j = b_i$$

Contraintes des signes :

$\forall j \in J_1, x_j \geq 0$

$\forall j \in J_2, x_j$ de signe qulconque

$I = I_1 \cup I_2$ Ensemble des indices de contraintes,

$\text{card}(I) = m \Rightarrow m$ contraintes

$J = J_1 \cup J_2$ Ensemble des indices des variables,

$\text{card}(J) = n \Rightarrow n$ variables

Notations :

Vecteurs :

$x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ (Les inconnues)

$c = (c_1, c_2, \dots, c_n)^T \in \mathbb{R}^n$

$b = (b_1, b_2, \dots, b_m)^T \in \mathbb{R}^m$

Matrice A de taille $m \times n$

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{1m} & \cdots & a_{mn} \end{pmatrix}$$

Forme canonique :

Sous cette forme, pas de contraintes d'égalités $I_2 = \emptyset$ et $J_2 = \emptyset$

$$\text{Max}(Z) = C^T x = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$$

Sous les contraintes :

$$Ax \leq b$$

$$x \geq 0$$

Forme standard :

Sous cette forme, $I_1 = \emptyset$ et $J_2 = \emptyset$

$$\text{Max}(Z) = C^T x = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$$

Sous les contraintes :

$$Ax = b$$

$$x \geq 0$$

I.3.1. Remarque sur la positivité des variables.

Sous la forme canonique ou standard, on impose toujours la positivité des variables $x \geq 0$. En fait, on peut toujours se ramener au cas $x \geq 0$:

- Si la variable x a une borne inférieure non nulle $x \geq l$, il suffit de considérer la nouvelle variable $y = x - l$ à la place de la variable x et alors on a $y \geq 0$.
- S'il n'y a pas de borne inférieure sur x (variable libre), on peut toujours poser $x = y - z$ avec les nouvelles variables $y \geq 0; z \geq 0$.

I.3.2. Variables d'écarts :

Proposition 1 [9]:

Tout PL sous forme standard s'écrit de façon équivalente en un PL sous forme canonique et inversement.

Démonstration :

Soit un PL sous forme canonique on a :

$$Ax \leq b \Leftrightarrow Ax + e = b\,, e \geq 0$$

Où $e^T = (e_1, e_2, \dots, e_m)^T$ sont appelées variables d'écarts

Ainsi $\begin{cases} Ax \leq b \\ x \geq 0 \end{cases} \Leftrightarrow \begin{cases} (A|I_m)\binom{x}{e} = b \\ \binom{x}{e} \geq 0 \end{cases} \Leftrightarrow \begin{cases} \tilde{A}\,\check{x} = b \\ \check{x} \geq 0 \end{cases}$

Avec $\tilde{A} = (A|I_m)$ est une matrice de taille $m \times (n + m)$

(Réciproquement) soit un PL sous forme standard on a :

$$Ax = b \Leftrightarrow \begin{cases} Ax \leq b \\ Ax \geq b \end{cases} \Leftrightarrow \begin{cases} Ax \leq b \\ -Ax \leq -b \end{cases}$$

$$\Leftrightarrow \binom{A}{-A} x \leq \binom{b}{-b} \Leftrightarrow \tilde{\tilde{A}}x \leq \check{b}$$

Ou $\tilde{\tilde{A}}$ est une matrice de taille $2m \times n$ et $\check{b} \in \mathbb{R}^{2m}$

I.3.3. Solution de base réalisable :

PL sous forme standard $Ax = b$

Hypothèse de rang plein :

On suppose que la matrice A est de taille $m \times n$ avec $rang(A) = m \leq n$

Rappel :

rang(A)= nombre maximal de lignes de A linéairement indépendantes (=nombre max. de colonnes linéairement indépendantes).

Remarque:

Sous l'hypothèse de rang plein :

le système $Ax = b$ admet toujours des solutions.

Si $m < n$, le système $Ax = b$ admet une infinité de solution.

Si $m = n$, la solution est unique et vaut $x = A^{-1}b$, dans ce cas, il n'y a rien à maximiser...

Hypothèse non restrictive : si rang(A) <m le système $Ax = b$ n'a pas de solution en général.

I.3.4. Quelques définitions :

Définition 2 (solution de base réalisable) :

On appelle solution réalisable tout vecteur x qui satisfait les contraintes du PL i.e. tel que $Ax = b$ et $x \geq 0$

Définition 3 (variable de base) :

Soit $B \subset \{1, \dots \dots, n\}$ un ensemble d'indices avec $card(B) = m$ tel que les colonnes $A^j, j \in B$, de A sont linéairement indépendantes. Autrement dit, la matrice carrée $A_B$formée des colonnes $A^j, j \in B$, est inversible. On dit que l'ensemble B des indices est une base.

Les variables $x_B = (x_j, j \in B)$ sont appelées variables de base.

Les variables $x_{HB} = (x_j, j \notin B)$ sont appelées variables hors-base.

Remarque :

Sous l'hypothèse de rang plein, il existe toujours une base non vide. Quitte à renuméroter les indices, on peut toujours écrire les décompositions par blocs :

$A = (A_B | A_{HB})$

$x = \begin{pmatrix} x_B \\ x_{HB} \end{pmatrix}$

Le système $Ax = b$ est équivalant à :

$A_B x_B + A_{HB} x_{HB} = b$

Définition 4: (Solution de base)

On dit que $x = \begin{pmatrix} x_B \\ x_{HB} \end{pmatrix}$ est solution de base associée à la base B si $x_{HB} = 0$

I.3.5. Propriétés des solutions de base réalisables :

Si $x = \begin{pmatrix} x_B \\ x_{HB} \end{pmatrix}$ est une solution de base réalisable, alors $x_{HB} = 0$ et $x_B = A^{-1}b$

I.3.6. Propriétés géométriques des solutions de base réalisables :

On note :$D_R = \{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$

L'ensemble des solutions réalisables d'un PL sous forme standard

Définition 5 :

Un polyèdre Q de $\mathbb{R}^n$ est défini par $Q = \{x \in \mathbb{R}^n \mid Mx \leq b\}$ où M est une matrice $m \times n$.

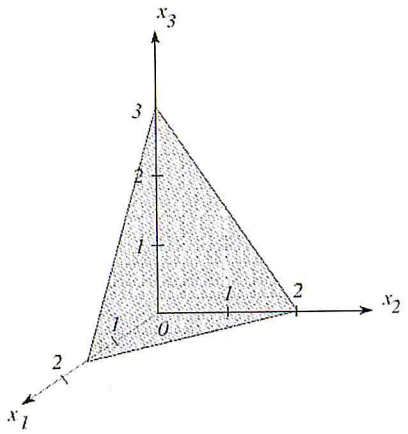
Un ensemble E est dit convexe si $\forall x, y \in E, \lambda x + (1-\lambda)y \in E$ Pour tout $0 \leq \lambda \leq 1$.

Proposition 2 [9]:

L'ensemble D_R des solutions réalisables est un polyèdre convexe fermé.

Exemple :

$D_R = \{x \in \mathbb{R}^3 \mid 2x_1 + 3/2\, x_2 + x_3 = 3, \quad x_1, x_2, x_3 \geq 0\}$



I.3.7. Caractérisation de l'optimum :

Définition 6:

Un point $x \in D_R$ est un sommet (ou point extrême) si et seulement s'il n'existe pas $y, z \in D_R, y \neq z$ telque $x = \lambda y + (1-\lambda)z$ avec $0 < \lambda < 1$

Théorème 1 [9] :

x est une solution de base réalisable si et seulement si x est un sommet de D_R.

L'optimum de la fonction objectif Z sur D_R, s'il existe, est atteint en au moins un sommet de D_R.

Remarque :

Tout se passe donc avec les solutions de base : pour résoudre un PL sous forme standard, il suffit de se restreindre aux solutions de base. Trois situations possibles :

$D_R = \emptyset$: Le PL n'a pas de solution.

$D_R \neq \emptyset$ mais la fonction objectif n'est pas majorée sur D_R : Le maximum de Z vaut $+\infty$

$D_R \neq \emptyset$ mais la fonction objectif est majorée sur D_R : le PL admet une solution optimale (non nécessairement unique).

I.4. Dualité en programmation linéaire :

Exemple :

Deux produits P_1 et P_2 fabriqués en quantité x_1 et x_2, nécessitant trois ressources disponibles en quantités données. L'entreprise cherche à maximiser le bénéfice total provenant de la vente des 2 produits.

$$\text{Max } Z = 6x_1 + 4x_2$$

$$3x_1 + 9x_2 \leq 81$$

$$4x_1 + 5x_2 \leq 55$$

$$2x_1 + x_2 \leq 20$$

$$x_1, x_2 \geq 0$$

Supposons à présent qu'un acheteur se présente pour acheter toutes les ressources de l'entreprise. Il propose à l'entreprise les prix unitaires y_1, y_2, y_3 pour chacune des ressources. L'entreprise acceptera de lui vendre toutes ses ressources uniquement si elle obtient pour chaque produit un prix de vente au moins égal au profit qu'elle ferait en vendant ses produits.

De son côté, l'acheteur cherche à minimiser ses dépenses. Quels prix unitaires y_1, y_2, y_3 l'acheteur doit-il proposer à l'entreprise en question pour qu'elle accepte de vendre toutes ses ressources ?

Programme linéaire :

$\text{Min } w = 81y_1 + 55y_2 + 20y_3$

$3y_1 + 4y_2 + 2y_3 \geq 6$

$9y_1 + 5y_2 + y_3 \geq 4$

$y_1, y_2, y_3 \geq 0$

matrice A de taille $m \times n$

vecteurs $c \in \mathbb{R}^n$ et $b \in \mathbb{R}^m$

Définition 7 (problème dual) :

Au problème linéaire primal :

$\text{Max}(Z = C^T x)$

$Ax \leq b$

$x \geq 0$

On associe le programme linéaire dual :

$\text{Min}(w) = b^T y$

$A^T y \geq c$

$y \geq 0$

I.4.1 Définition générale de la dualité :

primal		dual
$\text{Max } (Z = c^T x)$	$\Leftrightarrow$	$\text{Min}(Z = b^T y)$
$\forall i \in I_1, \sum_{j=1}^{n} a_{ij} x_j \leq b_i$	$\Leftrightarrow$	$\forall i \in I_1, y_i \geq 0$
$\forall i \in I_2, \sum_{j=1}^{n} a_{ij} x_j = b_i$	$\Leftrightarrow$	$\forall i \in I_2$, y_i de signe qelconque

$$\forall j \in J_1, x_j \geq 0 \quad \Leftrightarrow \quad \forall j \in J_1, \sum_{i=1}^{m} a_{ij}\, y_i \geq c_j$$

$$\forall j \in J_2, x_j \text{ de signe quelconque} \quad \Leftrightarrow \quad \forall j \in J_2, \sum_{i=1}^{m} a_{ij} y_j = c_j$$

I.4.2.Propriétés (Théorèmes de dualité) :

Proposition 3 [10] :

Le dual du dual est le primal

Preuve :

Le dual d'un PL sous forme canonique :

$\text{Min } (Z = b^T y)$ $\quad$ $\text{Max}(-Z = (-b^T)y)$

$A^T y \geq c$ $\quad$ $-A^T y \leq -c$

$y \geq 0$ $\quad$ $y \geq 0$

On prend le dual du dual :

$\text{Min } (-c^T)x$ $\quad$ $\text{Max } c^T x$

$(-A^T)^T x \geq -b$ $\quad$ $Ax \leq b$

$x \geq 0$ $\quad$ $x \geq 0$

I.4.3. Théorèmes de dualité:

Théorème 2 [10]:

Soit x une solution réalisable d'un (PL) sous forme canonique mixte et y une solution réalisable du problème dual (PLD). Alors : $Z \leq w$.

Si $Z = w$, alors x et y sont des solutions optimales de primal (PL) et dual (PLD) respectivement

Preuve :

(PL) sous forme canonique

1. On a $Ax \leq b$, $x \geq 0$ et $A^T y \geq c$, $y \geq 0$

$Z = c^T x \leq (A^T y)^T x = y^T Ax \leq y^T b = w$

2. Soit x^* , y^* des solutions réalisables de (PL) et (PLD) telles que $Z^* = w^*$

D'après 1 pour x solution réalisable de (PL), on a $Z \leq w = Z^*$ donc x^*est une solution réalisable optimale.

Théorème 3 [10] :

Si le problème primal (PL) admet une solution réalisable optimale x^*, alors le problème dual (PLD) admet lui aussi une solution réalisable optimale y^* et on a : $Z^* = w^*$

Preuve :

On suppose (PL) mis sous forme standard.

S'il existe une solution réalisable optimale, alors il existe une solution de base réalisable optimale $x_{B^*} = A_{B^*}^{-1^T} c_{B^*}$

On montre que y^* est une solution réalisable optimale pour le dual avec $y^* = (A_{B^*}^{-1})^T c_{B^*}$, on a :

$A_{H^*}^T y^* = A_{B^*}^T (A_{B^*}^{-1})^T c_{B^*} = (A_{B^*}^{-1} A_{H^*})^T c_{B^*} = c_{H^*} - L_{H^*}$

Or à l'optimum $L_{H^*} \leq 0$ donc $A_{B^*}^T y^* \geq c_{H^*}$ puisque $A_{B^*}^T y^* = c_{B^*}$

On a $A^T y \geq c$, y^* de signe quelconque .

i.e. y^* est une solution réalisable du dual (PLD) (pas de contrainte de positivité sur les variables y du dual).

$Z^* = c^T x^* = c_{B^*}^T A_{B^*}^{-1} b = ((A_{B^*}^{-1})^T c_{B^*})^T b = y^* b = w^*$

Théorème de dualité ⇛ y^*est optimal pour (PLD)

Théorème 4 [10] :

Etant donnés un problème primal (PL) et son dual (PLD), une et une seule des trois situations suivantes a lieu :

(a) les deux problèmes possèdent chacun des solutions optimales (à l'optimum, les coûts sont égaux).

(b) un des problèmes possède une solution réalisable avec un optimum infini, l'autre n'a pas de solution.

(c) aucun des deux problèmes ne possède de solution réalisable.

I.4.4. Conditions d'optimalité primal-dual (COPD) :

Théorème 5 [10] :

Soient x et y des solutions réalisables respectivement du problème primal (PL) et du problème dual (PLD). Alors x et y sont des solutions réalisables optimales si et seulement si les conditions d'optimalité primal-dual (COPD) suivantes sont vérifiées:

Si une contrainte est satisfaite en tant qu'inégalité stricte dans (PL) (resp. (PLD)), alors la variable correspondante de (PLD) (resp. (PL)) est nulle.

Si la valeur d'une variable dans (PL) ou (PLD) est strictement positive, alors la contrainte correspondante de l'autre programme est une égalité.

Remarque :

(COPD) permettent de vérifier si une solution réalisable d'un (PL) est optimale ou non, à partir de la connaissance d'une solution optimale du problème dual.

Chapitre II

METHODES DE RESOLUTION D'UN PROGRAMME LINEAIRE, METHODE DE DANTZING (SIMPLEXE)

II.1. Méthodes de résolution des problèmes de PL:

II.1.1.Historique :

Les premiers mathématiciens qui se sont occupés de problèmes, que l'on ne nommait pas encore à l'époque « programmes linéaires » (P.L.), sont : LAPLACE (1749-1827) et le baron FOURIER qui, vers 1825, a proposé une méthode d'élimination pour traiter des systèmes d'inéquations

linéaires et qui a défini une méthode géométrique pour atteindre le point le plus bas d'un solide délimité par des facettes planes (c'est-à-dire d'un « polyèdre »), idée très voisine de celle qui précède à l'algorithme dit « du simplexe » (le plus courant actuellement pour résoudre le P.L.) ; FOURIER est certes plus connu pour ses séries trigonométriques ou encore en physique pour sa résolution de l'équation de la chaleur. Mais l'importance économique de la P.L. n'apparaissait pas à l'époque et ses travaux sont tombés dans l'oubli.

De même vers 1911, Charles DE LA VALLEE-POUSSIN, un mathématicien belge, s'est vu confier par des astronomes un problème d'approximation minimale qui revenait en fait à résoudre un P.L. ; il l'a résolu par une méthode de changement de base (tout comme dans le simplexe) ; mais il était trop tôt pour qu'elle soit utilisée en économie et sa méthode n'avait été diffusée que dans le cercle des astronomes. Elle est restée inconnue dans la recherche opérationnelle jusqu'à une date bien postérieure de l'invention de l'algorithme du simplexe.

Il a fallu attendre l'époque de la seconde guerre mondiale pour que l'on modélise et résolve des problèmes de logistique qui se formulaient en tant que P.L. : ainsi le russe KANTOROVITCH en 1939 a imaginé une méthode inspirée des multiplicateurs de LAGRANGE, classiques en mécanique, pour résoudre des « programmes de transport ». Il faut être patient dans la vie : ce n'est qu'en 1975 qu'il fut récompensé pour l'ensemble de ses travaux par le prix Nobel d'économie, conjointement avec l'américain KOOPMANS...

La contribution décisive a été l'invention de l'algorithme du SIMPLEXE, développé à partir de 1947 notamment par G.B. DANTZIG [4] et le mathématicien VON NEUMANN. Cet algorithme a ensuite été implémenté sur les premiers ordinateurs et perfectionné (« méthode révisée du simplexe ») pour accroître la précision des résultats et diminuer le volume de mémoire nécessaire à la résolution.

En 1955 K. R. Frisch invente une méthode de point intérieur pour résoudre un problème non-linéaire (méthode de potentiel logarithmique, utilisation d'une fonction barrière, cf. plus loin) [11]

A. V. Fiacco et G. P. Mc Cormick développent l'utilisation des méthodes de point intérieur en programmation convexe non-linéaire [2]

Vers 1975-1979, les mathématiciens soviétiques SHORR [16] puis KHACHIYAN [12] ont apporté une avancée théorique concernant la complexité de la programmation linéaire, (en créant un algorithme polynômial) mais sans qu'elle débouche sur un algorithme plus efficace en pratique que le SIMPLEXE.

Au milieu des années 80, l'indien KARMARKAR [14] a proposé une nouvelle méthode créée aux Bell Laboratoires qui permettait de résoudre de très gros problèmes linéaires, par une démarche « intérieure » au polyèdre des solutions admissibles.

II.1.2. Classification :

Parmi les méthodes que les mathématiciens ont eues à utiliser on va présenter :

Méthode du simplexe [4] :

La Méthode de simplexe est un procédé itératif permettant d'effectuer une exploration dirigée de l'ensemble des solutions de base réalisables sur l'ensemble des points extrémaux. L'algorithme de simplexe, réputé très efficace dans ce domaine résout dans la majorité des cas un programme linéaire est de m contrainte et n variables en un temps polynomial alors que sa complexité théorique en général est de l'ordre de 2^n itération (exponentiel).

Méthode ellipsoïdale [12] :

En 1979 Khachiyan propose un algorithme polynomial pour la programmation linéaire.

Ce fut un grand succès théorique, malheureusement est beaucoup moins efficace que l'algorithme de simplexe pour résoudre la plupart des programmes linéaires.

Méthode des points intérieurs [14] :

En 1984 Karmarkar propose un algorithme polynomial plus attractif : c'est une approche originale en pratique pour résoudre des programmes linéaires. il est (aux dires de son auteur) 50 à 100 fois plus rapide que l'algorithme du simplexe, ce qui a provoqué des controverses dans la communauté de la programmation mathématique et incité plusieurs recherches dans le domaine de la programmation linéaire.

II.2. Méthode du simplexe :

L'algorithme de simplexe est une méthode de résolution des problèmes de PL qui se concentre sur les points et rayons extrêmes du polyèdre considéré. Grossièrement exprimé, elle passe d'un point extrême du polyèdre à un point extrême voisin tout en améliorant la valeur linéaire à maximiser.

Remarque :

On appelle n-simplexe ou simplement simplexe, l'enveloppe convexe d'un ensemble de n + 1 points (n = 1 : un segment, n = 2 : un triangle, n = 3 : un tétraèdre).

II .2.1. Présentation de la méthode de simplexe :

Pour introduire la méthode du simplexe, nous considérons le problème linéaire PL sous forme standard :

$Min\ Z = c^T x$ (1)

$Ax = b$ (2)

$x \geq 0$ (3)

Où $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$

$c = (c_1, c_2, \dots, c_n) \in \mathbb{R}^n$

$b = (b_1, b_2, \dots, b_m) \in \mathbb{R}^m$

$A\ matrice\ m \times n$

Le système (2) peut être ramené à la forme diagonale, par rapport aux " m " variables $(x_1, x_2, \dots, x_m)$ et devient :

$$x_1 + a_{1m+1}x_{m+1} + a_{1m+2}x_{m+1} + \cdots + a_{1n}x_n = b_1$$

$$x_2 + a_{2m+1}x_{m+1} + a_{2m+2}x_{m+2} + \cdots + a_{2n}x_n = b_2$$

$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$

$$x_i + a_{im+1}x_{m+1} + a_{im+2}x_{m+2} + \cdots + a_{in}x_n = b_i$$

$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$

$$x_m + a_{mm+1}x_{m+1} + a_{mm+2}x_{m+2} + \cdots + a_{mn}x_n = b_m$$

Considérons l'équation $Z = cx$ sous la forme $-Z + cx = 0$ (4)

Remplaçons dans (4), les x_i par leur valeur de l'expression :

$x_i = -a_{im+1}x_{m+1} - a_{im+2}x_{m+2} - \cdots - a_{in}x_n + b_i i = 1, \dots, m;$

L'équation (1) devient :

$-Z + c_{m+1}x_{m+1} + c_{m+2}x_{m+2} + \cdots + c_n x_n + Z_0 = 0$

Le problème (1), (2) et (3) devient sous la forme standard :

$$-Z + c_{m+1}x_{m+1} + c_{m+2}x_{m+2} + \cdots + c_n x_n + Z_0 = 0$$

$$x_1 + a_{1m+1}x_{m+1} + a_{1m+2}x_{m+1} + \cdots + a_{1n}x_n = b_1$$

$$x_2 \quad + a_{2m+1}x_{m+1} + a_{2m+2}x_{m+2} + \cdots + a_{2n}x_n = b_2$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$x_i \quad + a_{im+1}x_{m+1} + a_{im+2}x_{m+2} + \cdots + a_{in}x_n = b_i$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$x_m \quad + a_{mm+1}x_{m+1} + a_{mm+2}x_{m+2} + \cdots + a_{mn}x_n = b_m$$

Cette forme diagonale donne une solution du problème $\overline{x} = (b_1, b_2, \ldots, b_m, 0, \ldots, 0)$

$\overline{x}$ est réalisable si les $b_i \geq 0, i = 1, \ldots, m$ et $Z(\overline{x}) = Z_0$

Considérons la relation suivante :

$$-Z + c_{m+1}x_{m+1} + c_{m+2}x_{m+2} + \cdots + c_n x_n = -Z_0$$

Deux cas sont possibles :

Cas 1 : $c_{m+1} \geq 0, c_{m+2} \geq 0, \ldots, c_n \geq 0$. Alors la solution $\overline{x} = (b_1, b_2, \ldots, b_m, 0, \ldots, 0)$ et $Min\,(Z) = Z(\overline{x}) = Z_0$

Preuve :

Soit $K = \{x \in \mathbb{R}^n | \, Ax = b, x \geq 0\}$ des contraintes

Et $\overline{y} \in K$, $y = (x'_1, x'_2, \ldots, x'_m, \ldots, x'_n)$

$Z(\overline{y}) = c_{m+1}x'_{m+1} + c_{m+2}x'_{m+2} + \cdots + c_n x'_n + Z_0 \geq 0 = Z(\overline{x})$

$\forall \overline{y} \in K, Z(\overline{x}) \leq Z(\overline{y})$ $\overline{x}$ est une solution optimale.

Cas 2 : Au moins un coefficient c_j $(j = m+1, \ldots, n)$ est négatif.

$\overline{x}$ n'est pas une solution optimale. On peut trouver une autre solution de base réalisable $\overline{y}$ telle que $Z(\overline{y}) \leq Z(\overline{x})$.

Pour déterminer une autre solution $\overline{y}$, nous devons faire rentrer un nouveau vecteur hors base dans la base et faire sortir un vecteur de la base et obtenir une nouvelle base.

Il faut donc déterminer quelle est la variable hors base qui va rentrer dans la base $(a_1, a_2, \ldots, a_m)$

On choisira de telle sorte que la valeur de la fonction objective diminue le plus possible et que la solution de base correspondante soit encore réalisable.

Si l'on veut faire rentrer dans la base la variable hors base x_{j_0}, $m+1 \leq x_{j_0} \leq n$, il faut prendre une variable de base (qui sort de la base) x_{i_0} telle que i_0 vérifie :

$\frac{b_{i_0}}{a_{i_0 j_0}} = min\{\frac{b_{i_0}}{a_{i_0 j_0}}, a_{i_0 j_0} > 0\}$ (7)

Car on peut augmenter x_{j_0} de 0 à $\frac{b_{i_0}}{a_{i_0 j_0}}$

Chaque composante de y sera positive ou nulle et la valeur de la fonction objective diminuera de $c_{j_0} \frac{b_{i_0}}{a_{i_0 j_0}}$ $Z(\overline{y}) = Z(\overline{x}) + c_{j_0} \frac{b_{i_0}}{a_{i_0 j_0}}$, $c_{j_0} < 0$. Si l'on veut que cette quantité soit la plus petite possible on peut prendre $c_{j_0} = min\{c_j, c_j < 0\}$ (8)

Variable de base	1	X_1	X_2	X_3	.....	X_m	X_{m+1}	X_{m+2}	....	X_n
X_0	a_{00}	0	0	0		0	a_{0m+1}	a_{0m+2}		a_{0n}
X_1	a_{10}	1	0	0		0				
X_2	a_{20}	0	1	0		0	a_{1m+1}	a_{1m+2}		a_{1n}
.										
.										
X_{m-k}	a_{m-k0}									
X_{m-1}	a_{m-10}									
X_m	a_{m0}									
		0	0	1		0	.	.		.
		.	.	.		.	.	.		.
		.	.	.		.	.	.		.
		.	.	.		.	.	.		.
		.	0	0		0	.	.		.
						1				
							$a_{m\,m+1}$	$a_{m\,m+2}$		$a_{m\,n}$

Ecrivons le système des contraintes (5) sous forme de tableau (9) avec $a_{00} = -Z_0$ *et* $x_0 = Z$

Avec $a_{0j} = c_j \quad j = m+1, \dots, n;\ a_{i0} = b_i, i = 1, \dots, m.$

II .2.2. Algorithme du Simplexe :

1. On a le tableau (9) avec $a_{i0} \geq 0, i = 1, \dots, m$

2. Si toutes les $a_{j0} \geq 0, \quad j = m+1, \dots, n.$

La solution $\overline{x} = (a_{10}, a_{20}, \dots, a_{m0}, 0, \dots, 0)$ est optimale $\quad minZ = Z(\overline{x}) = Z_0$

Sinon déterminons $a_{0r} = min\{a_{0j}, a_{0j} < 0, j = m+1, \dots, n\}$ (le choix de la colonne pivot ou de la variable rentrante).

3. La variable x_r rentre dans la base et déterminons,

$\frac{a_{s0}}{a_{sr}} = min\{\frac{a_{i0}}{a_{ir}}, a_{ir} > 0\}$ (La ligne pivot est S et a_{sr} est le pivot).

4 . On applique la méthode de Gauss pour obtenir une autre solution en faisant un pivotage des variables.

Les trois pas sont répétées jusqu'à l'obtention d'un tableau avec des

$a'_{0j} \geq 0, j = m+1, \dots, n$

Remarque :

1. Lorsqu'il y a dégénérescence (au moins un a_{i0}), alors la fonction objectif peut prendre à l'itération suivante la même valeur et la nouvelle solution de base réalisable est encore dégénérée. Si un tel cas se produit dans plusieurs itérations successives, il est possible de retrouver une base déjà obtenue précédemment et à partir de là, cycler indéfiniment.

Il y a une méthode permettant d'éviter le cycle appelée " Méthode lexicographique ".

2. Si un $a_{0j_0} < 0$ *et* $a_{ij_0} \leq 0, \forall\, i = 1, \dots, m$ la variable x_{j_0} peut prendre une valeur arbitraire aussi grande que l'on veut et obtenir chaque fois une solution de base réalisable.

On peut aussi donner à Z des valeurs négatives que l'on veut, alors le problème ne possède pas de solutions optimale finie et $Min\ Z = -\infty$

3. Si on considère le problème de maximisation (P)

$Max\ Z = cx$

$Ax = b$

$x \geq 0$

L'algorithme du simplexe s'énonce par : si $c_j < 0$ alors la solution de base de départ est optimale.

Sinon s'il existe un j un indice hors base, tel que $c_j > 0$, alors la solution de départ n'est pas optimale.

Exemple :

$Min\ z = 24x_1 + 9x_2 - 6x_3$

$x_1 \quad + x_3 + e_1 = 3$

$x_1 + x_2 - 2x_3 + e_2 = 2$

$x_1, x_2, x_3, e_1, e_2 \geq 0$

Variables de base	1	x_1	x_2	x_3	e_1	e_2
$-Z$	0	24	9	-6	0	0
e_1	3	1	0	1	1	0
e_2	2	1	1	-2	0	1
$-Z$	18	30	9	0	6	0
x_3	3	1	0	1	1	0
e_2	8	3	1	0	2	1

La solution optimale est $x^* = (0; 0; 3)$.

II.2.3.Variantes de la méthode :

L'application de la méthode du simplexe à la résolution des PL suppose la connaissance préalable d'une solution de base réalisable de départ.

Lorsque celle-ci ne peut s'obtenir de manière immédiate, on peut avoir recours à deux méthodes qui permettent de construire de manière systématique cette solution base réalisable de départ, ces deux méthode _ la méthode big M et la méthode des deux phases reposent sur l'introduction des variables artificielles.

II.3. Complexité et efficacité de la méthode du simplexe :

Dans la section précédente, nous avons étudié la méthode du simplexe, et sa variante, pour résoudre des problèmes de programmation linéaire. Le procédé reste largement utilisé dans la pratique pour résoudre les problèmes de société. Cependant, la quantité de temps requis pour calculer une solution en utilisant la méthode du simplexe croît rapidement car le nombre de n composantes de la variable $x \in \mathbb{R}^n$ augmente. Plus précisément, il apparaît que la relation entre la quantité de temps requise de l'algorithme pour trouver une solution et la taille n est exponentielle dans le pire des cas. Un exemple d'un problème de LP pour lequel cette relation est évidente a été conçu par Klee et Minty en 1972 [17].

Ci-dessous, nous donnons une version de l'exemple Klee-Minty

Soit n un entier donné.

Exemple de Klee et Minty (PL_n) :

$\text{Max}(z)= \sum_{j=1}^{n} 10^{n-j} x_j$

$2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \leq 100^{i-1}$ i=1,…,n.

$x_j \geq 0$ j=1,…,n.

On fait deux observations

Première observation :

(a) $x_1 = x_2 = \ldots = x_{n-1}=0$, $x_n=10^{n-1}$ est la seule solution optimale de (PL_n). Donc dans le tableau final de la méthode du simplexe de (PL_n), cela correspond à une solution de base $y_1, y_2,\ldots, y_{n-1}, y_n$.

(b) Dans chaque tableau et pour chaque i correspond une des deux variables x_i , y_i une variable de base, sinon pour certaines i, les deux variables ne sont pas des variables de base, alors $y_i= 0$ et donc la i éme contrainte de $(PL_n)'$ est une égalité, cependant $x_j= 0$, mais l'égalité de la i éme contrainte contredit la (i-1) iéme (comme il y'a n variables de base et n variables hors base, quand les deux

variable x_i et y_i sont des variables de base, alors il existe au moins un j x_i et y_i sont des variables hors base et retombe sur le cas étudié.

Deuxième observation :

PL_n demande 2^n tableaux de simplexe et dans chaque tableau la ligne correspondant à la fonction objective est entière.

L'application de la méthode du simplexe pour la réduction de quelques exemples de cette classe confirme les deux observations.

Pour n=1 :

$$\begin{cases} \text{Max } x_1 \\ x_1 \leq 1 \\ x_1 \geq 0 \end{cases}$$

Sous forme standard

$$\begin{cases} \text{Max } x_1 \\ x_1 + y_1 = 1 \\ x_1 \geq 0 \\ y_1 \geq 0 \end{cases}$$

Variable de base	1	X_1	Y_1
Z	0	1	0
Y_1	1	1	1
Z	-1	0	-1
X_1	1	1	1

Pour n=2

$$\begin{cases} \text{Max } 10x_1 + x_2 \\ x_1 \leq 1 \\ 20x_1 + x_2 \leq 100 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

Sous forme standard

$$\begin{cases} \text{Max } 10x_1 + x_2 \\ x_1 + y_1 = 1 \\ 20x_1 + x_2 + y_2 = 100 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

Variable de base	1	X_1	X_2	Y_1	Y_2
-Z	0	10	1	0	0
Y_1	1	1	0	1	0
Y_2	100	20	1	0	1
-Z	-10	0	1	-10	0
X_1	1	1	0	1	0
Y_2	80	0	1	-20	1
-Z	-90	0	0	10	-1
X_1	1	1	0	1	0
X_2	80	0	1	-20	1

-Z	-100	-10	0	0	-1
Y_1	1	1	0	1	0
X_2	100	20	1	0	1

Pour n=3 :

$$\begin{cases} \text{Max } 100\ x_1 + 10\ x_2 + x_3 \\ x_1 \leq 1 \\ 20\ x_1 + x_2 \leq 100 \\ 200\ x_1 + 20\ x_2 + x_3 \leq 10000 \\ x_1 \geq 0\ ,\ x_1 \geq 0\ ,\ x_1 \geq 0 \end{cases}$$

Sous forme standard

$$\begin{cases} \text{Max } 100\ x_1 + 10\ x_2 + x_3 \\ x_1 + y_1 = 1 \\ 20\ x_1 + x_2 + y_2 = 100 \\ 200\ x_1 + 20\ x_2 + x_3 + y_3 = 10000 \\ x_1 \geq 0\ ,\ x_2 \geq 0\ ,\ x_3 \geq 0 \\ y_1 \geq 0\ ,\ y_2 \geq 0\ ,\ y_3 \geq 0 \end{cases}$$

Variable de base	1	X_1	X_2	X_3	Y_1	Y_2	Y_3
-Z	0	100	10	1	0	0	0
Y_1	1	1	0	0	1	0	0
Y_2	100	20	0	0	0	1	0
Y_3	1000	200	20	1	0	0	1

-Z	-100	0	10	1	-100	0	0
X_1	1	1	0	0	1	0	0
Y_2	80	0	1	0	-20	1	0
Y_3	9800	0	20	1	-200	0	1
-Z	-900	0	0	1	100	-10	0
X_1	1	1	0	0	1	0	0
X_2	80	0	1	0	-20	1	0
Y_3	8200	0	0	1	200	-20	1
-Z	-1000	-100	0	1	0	-10	0
Y_1	1	1	0	0	1	0	0
X_2	100	20	1	0	0	1	0
Y_3	8000	-200	0	1	0	-20	1
-Z	-9000	100	0	0	0	10	-1
Y_1	1	1	0	0	1	0	0
X_2	100	20	1	0	0	1	0
X_3	8000	-200	0	1	0	-20	1
-Z	-9100	0	0	0	-100	10	-1
X_1	1	1	0	0	1	0	0
X_2	80	0	1	0	-20	1	0
X_3	8200	0	0	1	200	-20	1

-Z	-9900	-10	0	0	100	0	-1
X_1	1	1	0	0	1	0	0
Y_2	80	0	0	0	-20	1	0
X_3	9800	0	1	1	-200	0	1
-Z	-10000	-100	-10	0	0	0	-1
Y_1	1	1	0	0	1	0	0
Y_2	100	20	1	0	0	1	0
X_3	10000	200	20	1	0	0	1

On remarque que pour (n=1, n=2, n=3), l'application de la méthode du simplexe nécessite pour résoudre (PL_n) : 2^n tableaux de simplexe.

D'où le théorème suivant :

Théorème 1 [3] :

L'application de la méthode du simplexe pour les (PL_n), nécessite 2^n -1 tableaux.

Preuve :

On démontre le théorème par récurrence sur n.

Pour n=1.

Le programme linéaire associé s'écrit (PL_1) :

$$\begin{cases} \text{Max } x_1 \\ x_1 \leq 1 \\ x_1 \geq 0 \end{cases}$$

Sous forme standard

$$\begin{cases} \text{Max } x_1 \\ x_1 + y_1 = 1 \\ x_1 \geq 0 \\ y_1 \geq 0 \end{cases}$$

On applique la méthode du simplexe sur (PL_1) on trouve la solution optimale, l'algorithme nécessite deux tableaux. Donc pour n=1 est démontrée, on suppose que le théorème est vérifié pour n, on le montre pour (n+1).

Posons $X^T=(x_1, x_2, \dots, x_{n-1})$, $Y^T=(y_1, y_2, \dots, y_{n-1})$

On écrit alors le système sous la forme suivante :

$$(PL_n)' \quad \begin{cases} \text{Max } (CX + x_n) \\ AX + Y = b \\ 2CX + x_n + y_n = 10^{n-1} \\ X, x_n, y_n \geq 0 \end{cases}$$

On considère le (1^{ier}) et (2^n) tableaux de (PL_n) :

Tableau 1 :

Variable de base	1	X	x_n	Y	y_n
-Z	0	C	1	0	0
Y	B	A	0	1	0
y_n	10^{n-1}	2C	1	0	1

Tableaux 2^n :

Variable de base	1	X	x_n	Y	y_n
-Z	-10^{n-1}	-C	0	0	-1
Y	B	A	0	1	0
x_n	10^{n-1}	2C	1	0	1

$$(PL_{n+1})' \begin{cases} \text{Max } (10CX + 10\,x_n + x_{n+1}) \\ AX + Y = b \\ 2CX + x_n + y_n = 10^{n-1} \\ 20CX + 20x_n + x_{n+1} + y_{n+1} = 10^n \\ x_1, x_2, \dots, x_n, x_{n+1} \geq 0 \;,\; y_1, y_2, \dots, y_n, y_{n+1} \geq 0 \end{cases}$$

On représente le (PL_{n+1}) dans le tableau 1, on remarque que si on élimine la partie colorée on obtient le tableau 1 de (PL_n) où la première ligne est multipliée par 10, par conséquent les (2^n) premiers tableaux de (PL_{n+1}) sont simulés par ceux de (PL_n) donc la dernière ligne de (2^n) tableaux (pour la résolution (PL_{n+1}) est par la multiplication de la dernière ligne de (2^n) par 10.

Tableau 1 :

Variable de base	1	X	x_n	x_{n+1}	Y	y_n	y_{n+1}
-Z	0	C	10	1	0	0	0
Y	B	A	0	0	1	0	0
y_n	10^{n-1}	2C	1	0	0	1	0
y_{n+1}	10^n	20C	20	1	0	0	1

Tableau 2^n :

Variable de base	1	X	x_n	x_{n+1}	Y	y_n	y_{n+1}
-Z	-10^{n-1}	-10C	0	1	0	-10	0
Y	B	A	0	0	1	0	0
x_n	10^{n-1}	2C	1	0	0		0
y_{n+1}	-10^n	-20C	0	1	0		1

On applique une autre fois la méthode du simplexe, on obtient :

Variable de base	1	X	x_n	x_{n+1}	Y	y_n	y_{n+1}
-Z	0	10C	0	0	0	10	-1
Y	B	A	0	0	1	0	0
x_n	10^{n-1}	2C	1	0	0	1	0
x_{n+1}	-10^n	-20C	0	1	0	-20	1

On remarque qu'on a obtenu pour la deuxième fois un tableau qui est le même que le premier, on applique pour la deuxième fois 2^n -1 itérations.

Tableau 2^{n+1}

Variable de base	1	X	x_n	x_{n+1}	Y	y_n	y_{n+1}
-Z	-10^n	-10C	-10	0	0	0	-1
Y	B	A	0	0	1	0	0

y_n	10^{n-1}	2C	1	0	0	1	0
x_{n+1}	10^n	20C	20	1	0	0	1

Le passage du tableau 1 au tableau 2^n demande 2^n -1 tableaux et le passage du tableau 2^n +1

Au tableau 2^{n+1} demande de même 2^n tableaux ce qui donne en tous 2^n tableaux pour (PL_{n+1})

II.4. Conclusion :

L'algorithme du simplexe permet de résoudre les problèmes de PL. Bien que cet algorithme soit efficace en pratique et qu'il soit assuré de trouver l'optimum, son comportement dans le pire cas peut être mauvais. Il est ainsi possible de construire un PL pour lequel la méthode du simplexe requiert un nombre d'étapes exponentiel à la taille du problème [17]. Ainsi, pendant plusieurs années, savoir si la PL était un problème NP-complet ou polynomial est resté une question ouverte.

Chapitre III

METHODE ELLIPSOIDALE DE KHACHIYAN

III.1. Méthode ellipsoïdale :

III.1.1. Introduction

L'algorithme ellipsoïdal a été introduit par le mathématicien soviétique L. G.Kachiyan en 1979 [4]. L'algorithme a comblé un vide théorique important, à savoir l'existence d'une méthode polynomiale de résolution pour les problèmes de PL .L'idée maitresse des méthodes ellipsoïdales et de construire une séquence d'ellipsoïdes contenant une solution optimale au problème et dont le volume décroit selon une progression géométrique.

Notation :

Inégalités linéaires (LI): Soit une matrice $A\ (m\times n)$ et un vecteur $b\in\mathbb{R}^m$, trouver un vecteur $x\in\mathbb{R}^n$ tel que $A\,x \leq b$.

Inégalités Linéaires strictes (LSI): Étant donné une matrice $A\ (m\times n)$ et un vecteur entier b, trouver un $x\in\mathbb{R}^n$ tel que $A\,x < b$.

Nous allons voir que l'algorithme ellipsoïdal est un algorithme en temps polynomial pour(LSI) et peut être utilisé pour construire un algorithme en temps polynomial pour résoudre (PL).

Définition 1 :

La taille d'un (PL) $\begin{cases} \min c^T x \\ A\,x=b \\ x \geq 0 \end{cases}$

est $L=m\times n + \lceil \log_2|P| \rceil + n \lceil \log_2 n \rceil$, où P est le produit de tous les nombres non nuls dans le problème (en A, b et c).

III.1.2. Présentation du Méthode Ellipsoïdale (Khachiyan 1979) :

Résoudre un PL $\begin{cases} \mathrm{Min}(c^T x) \\ A\,x \geq b \\ x \geq 0 \end{cases}$

Revient à résoudre

$$P\,z \leq q \quad \text{où} \quad P = \begin{pmatrix} c^T & -b^T \\ -c^T & b^T \\ -A & 0 \\ -I_n & 0 \\ 0 & A^T \\ 0 & -I_m \end{pmatrix} \qquad z = \begin{pmatrix} x \\ y \end{pmatrix} \qquad q = \begin{pmatrix} 0 \\ 0 \\ -b \\ 0 \\ c \\ 0 \end{pmatrix}$$

En effet PL $\begin{cases} \text{Min } (c^T x) \\ A\,x \geq b \\ x \geq 0 \end{cases}$ Dual de PL $\begin{cases} \text{Max } b^T y \\ A^T y \leq c \\ y \geq 0 \end{cases}$

$\text{Min } c^T x = \text{Max } b^T y$ d'après [3]

x S R de PL y S R du Dual

$$\Leftrightarrow \begin{cases} c^T x - b^T x = 0 \\ -A\,x + 0\,y \leq -b \\ -I_n\,x + 0\,y \leq 0 \\ 0\,x + A^T y \leq c \\ 0\,x - I_m\,y \leq 0 \end{cases} \Leftrightarrow \begin{pmatrix} c^T & -b^T \\ -c^T & b^T \\ -A & 0 \\ -I_n & 0 \\ 0 & A^T \\ 0 & -I_m \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ -b \\ 0 \\ c \\ 0 \end{pmatrix}$$

Soient LI $\begin{cases} A\,x \leq b \\ A\ (m\times n) \\ b \in \mathbb{R}^m \end{cases}$ LSI $\begin{cases} A\,x < b \\ A\ (m\times n) \\ b \in \mathbb{R}^m \end{cases}$

L=La taille de PL, $L=m\times n + \lceil \log_2 |P| \rceil + n \lceil \log_2 n \rceil$, où P est le produit de tous les termes de A, b et c.

On peut supposer que les éléments sont tous des entiers relatifs sinon multiplier par un nombre de telle manière à les rendre entiers.

Lemme 1 :

Si x est une S.R.B de LPS de taille L, alors il existe des entiers $D \neq 0$, $D_1, D_2, \ldots, D_n$ tels que $x_j = \frac{D_j}{D}$ où $|D| < 2^L$ et $x_j < 2^L$

Preuve :

Soit B (m×n) une base $D = \det B \neq 0$

Voir la formule de déterminant $|D| \leq m!|P| \leq n!|P| \leq 2^{log_2(n!)+\log_2|P|}$

$\leq 2^{n\lceil log_2 n\rceil + log_2|P|} < 2^L$. $|D| \leq m!|P| \leq 2^{m\, log_2 m + log_2|P|} < 2^{m\times n + \lceil log_2 P\rceil} = 2^L$

Si j∉ Base $x_j = \frac{0}{D}$ donc $x_j < 2^L$

Si j∈ Base $x_j = \frac{|b_j|}{D} \leq \frac{m!|P|}{D}$ comme $D \neq 0$ $|D| \geq 1$ (entier) $\Rightarrow x_j < 2^L$

Lemme 2 :

Supposons que deux S.R w, v de LPS satisfont $|c^T w - c^T v| \leq 2^{-2L}$ alors $c^T w = c^T v$

Preuve :

Supposons que le lemme est faux c- à -d $c^T w \neq c^T v$

Alors $c^T w = \frac{N}{D}$ et $c^T v = \frac{N'}{D'}$ $D'N - N'D \neq 0 \Rightarrow |c^T w - c^T v| = |\frac{N}{D} - \frac{N'}{D'}| = |\frac{D'N - N'D}{DD'}|$

$\geq \frac{1}{|D||D'|} > \frac{1}{2^{2L}} \Rightarrow |c^T w - c^T v| > 2^{-2L}$.

Théorème 1 [7] :

Il existe un algorithme polynomial pour LP si et seulement s'il existe un algorithme polynomial pour LI

Preuve :

⇒) Facile, il suffit de prendre la fonction objectif =0 et éliminer les lignes redondantes et ensuite résoudre LP pour obtenir une solution de LI

⇐) Soit LP $\begin{cases} \text{Min } c^T x \\ A x \geq b \\ x \geq 0 \end{cases}$

1/ Utiliser LI algorithme pour tester si LP admet une S.R sinon il n'y a pas de solution

2/ considérer LI système $P z \leq q$, l'algorithme de LI donne une solution en même temps pour le primal et le dual, sinon il n'y a pas de solution

LP a une solution si et seulement si LI $(P z \leq q)$ a une solution

Au fait l'algorithme de Khachiyan donne une solution au problème LSI = $\{Ax < b\}$

Pour cela le théorème suivant est important

Théorème 2 [7] :

Le système $a_i x \leq b_i \quad i=1,\ldots,m$ a une solution si et seulement si $a_i x < b_i + \mathcal{E} \quad i = \overline{1,m}$ avec $\mathcal{E} = 2^{-2L}$ a une solution.

Preuve :

Si $a_i x \leq b_i \quad i=1,\ldots,m$ a une solution alors

$a_i x < b_i + \mathcal{E} \quad i=1,\ldots,m$ avec $\mathcal{E} = 2^{-2L}$ a une solution c'est évident

Pour l'inverse : A partir d'une solution de $a_i x < b_i + \mathcal{E} \quad i=1,\ldots,m$ on donne une solution x de $a_i x \leq b_i \quad i=1,\ldots,m$ (analogue à la construction d'une S.R.B à partir d'une S.R)

Soit x_0 une solution de ($a_i\, x < b_i + \mathcal{E}$ $i=1,\ldots, m$ avec $\mathcal{E} = 2^{-2L}$)

Soit $I = \{a_i \ / b_i \leq a_i\, x_0 < b_i + \mathcal{E}\}$ On peut supposer que pour tout j $a_j = \sum_{a_i \in I} B_{ji}\, a_i$

Sinon on peut trouver une autre solution x_1 avec I plus grand.

En effet si $\exists\, j$ / a_j est indépendant des $a_i \in I$ alors le système $\begin{cases} a_i\, z = 0 & a_i \in I \\ a_j\, z = 1 \end{cases}$

Admet une solution z_0 (c-à-d ; il existe une base qui contient a_j et a_i $a_i \in I' \subset I$ qui donne la S.B)

En prenant $x_1 = x_0 + \lambda\, z_0$ avec λ suffisamment petit

On crée une autre solution x_1 de ($a_i\, x < b_i + \mathcal{E}$ $i=1,\ldots, m$ avec $\mathcal{E} = 2^{-2L}$), qui a au moins un vecteur en plus dans I et après m étapes on s'arrête.

Pour le choix de $\lambda \geq 0$

On sait que $i \in I$ $b_i \leq a_i\, x_0 < b_i + \mathcal{E}$

$i \notin I$ $a_i\, x_0 < b_i$

Et z_0 vérifie $\begin{cases} a_i\, z = 0 & i \in I \\ a_j\, z = 1 \end{cases}$

Si $i \in I$ $b_i \leq a_i\,(x_0 + \lambda\, z_0) = a_i\, x_0 < b_i + \mathcal{E}$ car $a_i\, z_0 = 0$

Si $i \notin I$ $a_i\, x_1 - b_i = a_i\, x_0 - b_i + \lambda\, a_i\, z_0 \leq 0$

$b_i - a_i\, x_0 > 0$ et $a_i\, x_0 - b_i < 0$

Si $a_i\, z_0 < 0$ alors $a_i\, x_1 - b_i < 0$

Si $a_i\, z_0 > 0$ $\lambda \leq \frac{b_i - a_i\, x_0}{a_i\, z_0}$ il existe au moins $a_j\, z = 1$

Donc il suffit de choisir $\lambda = Min_{i \notin I} \left[\frac{b_i - a_i\, x_0}{a_i\, z_0}\right]$ $a_i\, z_0 > 0$

Donc avec ce choix il existe au moins a_i tel que $b_i \leq a_i\, x_1 < b_i + \mathcal{E}$

Alors on augmente I avec au moins un vecteur

Alors pour tout j $a_j = \sum_{a_i \in I'} B_{ji}\, a_i$ où I' vecteurs linéairement indépendants de I et

$B_{ji} = \frac{D_{ij}}{|D|}$

Soit $\hat{x}$ la solution des équations $a_i\ x = b_i \quad a_i \in I'$, on montre que x est une solution de

$(a_i\ x \leq b_i \quad i = 1, \ldots, m)$

Pour tout j $|D|\ (a_j\ \hat{x} - b_j) = \sum_{a_i \in I'} D_{ji}\, a_i\ \hat{x} - |D|\ b_j = \sum_{a_i \in I'} D_{ji}\, b_i - |D|\ b_j$ par définition de $\hat{x}$

$= -\sum_{a_i \in I'} D_{ji}(a_i\ x_0 - b_i) + |D|\ (a_j\ x_0 - b_i)$ (en ajoutant et en retranchant $|D|\ a_j\ x_0$)

$< \varepsilon(\sum_{a_i \in I'} |D_{ji}| + |D|)$ Car $|\ a_i\ x_0 - b_i\ | < \mathcal{E} \quad i \in I' \quad a_j\ x_0 - b_j < \mathcal{E} \ \forall j$

$|D|\ (a_j\ \hat{x} - b_j) < \varepsilon(\sum_{a_i \in I'} |D_{ji}| + |D|) \qquad |D| < 2^L$

$$|D_{ji}| < 2^L$$

$$< 2^{-2L}(m+1)2^L = 2^{-L}(m+1) < 1$$

Donc pour tout j $|D|\ (a_j\ \hat{x} - b_j) < 1$, on conclut que $a_j\ \hat{x} - b_j \leq 0$ pour tout j car $|D|\ (a_j\ \hat{x} - b_j)$ est un entier c –à –d $\hat{x}$ est la solution désirée.

Corollaire 1 :

Si ∃ un algorithme polynomial pour LSI, alors il existe un pour LI et donc pour LP.

Preuve :

Pour trouver une solution qui satisfait $A\ x \leq b$

On construit b' par l'algorithme de LSI avec $b'_i = 2^{2L}\, b_i + 1$ et $A' = 2^{2L} A \qquad (A'x < b')$

L'algorithme de LSI donne une solution v qui satisfait $A'x < b'$. Ensuite en temps polynomial on construit une solution de $A\ x \leq b$ à partir de v comme s'est fait dans le théorème précédent.

III.2. Transformations affines et Ellipsoïdes :

Définition 2 :

$T(x) = t + Q\,x$ est une transformation affine où $t \in \mathbb{R}^n$ et Q est une matrice non singulière $(m \times n)$ c – à – d $\det Q \neq 0$.

Remarques:

1. Si T est une transformation affine, alors $T' = T^{-1}(x) = Q^{-1}(x - t)$ est aussi une transformation affine.
2. Si T est une transformation affine et $A, B \subseteq \mathbb{R}^n$, $T(A \cap B) = T(A) \cap T(B)$.
3. Si $\| x\| = \sqrt{x^T x}$ alors $\| Ux\| = \| x\|$ si U est une matrice orthogonale unitaire (Preuve : les distances)
4. Pour $t \in \mathbb{R}^n$ et $r > 0$, $S(t,r) = \{ x \in \mathbb{R}^n / \| x - t\| \leq r\}$, est la sphère de rayon r et de centre t .
5. Si T est une transformation affine, alors l'image de $S(0, 1)$ est une ellipsoïde. De façon équivalente, si $T(x) = t + Q\,x$, $T(S(0, 1)) = \{t + Q\,x \ / \ x^T x \leq 1\}$.
6. Si $y = t + Q\,x$, $x = Q^{-1}(y - t)$. Ainsi $T(S(0, 1)) = \{y / (Q^{-1}(y - t))^T Q^{-1}(y - t) \leq 1\}$
   $= \{y / (y - t)^T Q^{-1\,T} Q^{-1}(y - t) \leq 1\} = \{y / (y - t)^{T(QQ^T)^{-1}}(y - t) \leq 1\}$
   $= \{y / (y - t)^T B^{-1}(y - t) \leq 1\}$ Où $B = QQ^T$. Puisque Q est non singulière, B est définie positive.

Soit $E = \{x \in \mathbb{R}^n / x^T B^{-1} x \leq 1\}$ avec B définie positive

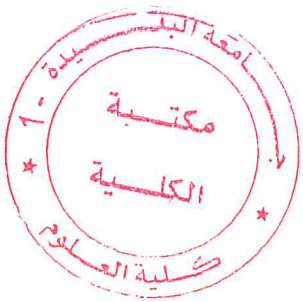
$$B = U^{-1} \beta U$$

Où β matrice diagonale et U est obtenue à partir des vecteurs propres.

Soit $\sqrt{D} = \begin{bmatrix} \sqrt{d_j} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sqrt{d_j} \end{bmatrix}$ $D = \sqrt{D}\sqrt{D}$

$U^{-1} = U^T$

$$B = (\sqrt{\beta} U)^T (\sqrt{\beta} U) = U^T \sqrt{\beta}^T \sqrt{\beta}\, U$$



$B^{-1}=U^T(\sqrt{\beta})^{-1}(\sqrt{\beta})^{-1T}U=(\sqrt{\beta^{-1}}\,U)^T(\sqrt{\beta^{-1}}U)$.

En d'autres termes,

$E=\{x\in\mathbb{R}^n \quad / \quad x^T(\sqrt{\beta^{-1}}U)^T(\sqrt{\beta^{-1}}U)\,x\leq 1\}$

$=\{ \quad x\in\mathbb{R}^n \quad / \quad [(\sqrt{\beta^{-1}}U)\,x]^T[(\sqrt{\beta^{-1}}U)\,x] \leq 1\}$.

Maintenant, nous allons $y=(\sqrt{\beta^{-1}}U)\,x$, de sorte que $x=(U^T\sqrt{\beta})\,y$. Alors

$$E=\{(U^T\sqrt{\beta})\,y \quad / \quad y^Ty\leq 1\}.$$

Alors l'ellipsoïde $E=\{x \quad / x^TB^{-1}x \leq 1\}$ peut être obtenu à partir de la sphère $S(0,1)$ par une translation et par une rotation.

Propriétés :

Supposons que $P\subseteq\mathbb{R}^n$ a un volume μ

1. Soit $T(x)=t+Q\,x$ transformations affine $\det Q\neq 0$.

   Alors $T(P)$ a un volume = $|\det Q|\,\mu$ c – à – d $\iint_{T(P)} dy = \iint_P (detQ)\,dx = |\det Q|\,\mu$
2. Soit $\Gamma_n=\{x\in\mathbb{R}^n \quad / \; x\geq 0 \quad \sum_{j=1}^{n} x_j \leq 1\}$ n-simplexe le volume de $\Gamma_n=\frac{1}{n!}$
3. Soit P un polytôpe dans $\mathbb{R}^n$ $Int(P)=\{x \; / \quad \exists \quad \varepsilon>0 \quad S(x,\varepsilon)\subseteq P\}$
4. Si $T\subseteq\mathbb{R}^n$ l'enveloppe convexe $c(T)$ de T est l'intersection de tous les ensembles convexes contenant T.
5. Si $Int(P)\neq 0$ alors P a $(x+1)$ sommets alors l'enveloppe convexe a un volume > 0.
6. Le volume d'une enveloppe convexe de point $v_0, v_1, \ldots, v_n\in\mathbb{R}^n$ est égale à

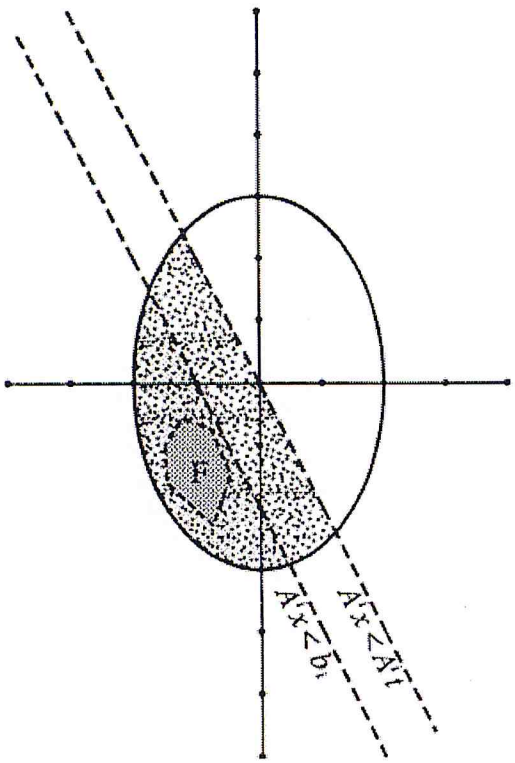
   $$\frac{1}{n!}\det\begin{bmatrix} 1 & 1 & \cdots & 1 \\ | & | & \cdots & | \\ v_0 & v_1 & \cdots & v_n \end{bmatrix}$$

7. Soit $a\in\mathbb{R}^n$, alors il existe une matrice orthogonale U telle que :

   $$U_a=(\|a\|, 0, \ldots, 0).$$

III.3. Algorithme de Khachiyan [7] :

L'algorithme résout LSI en temps polynomial, dans chaque itération, il produit un ellipsoïde E qui contient la région réalisable $F=\{ x \ / \ A x < b\}$ et teste le centre t de E pour la faisabilité. Si $t \in F$ il retourne t et il s'arrête, ayant résolu le problème avec succès. Sinon, il trouve une contrainte LSI qui n'est pas vérifiée par t (c'est-à-dire $A^i t \geq b_i$).



Car $A^i t \geq b_i$, pas de points réalisables pouvant se situer dans la moitié ellipsoïde 'au-dessus' de l'hyperplan passant par t qui est parallèle à la contrainte qui n'est pas vérifiée par t . (la ligne passant par l'origine dans l'image) . l'autre moitié de E (ombrée dans l'image) contient toute F.

L'idée est de construire un nouvel ellipsoïde E' qui contient la moitié de l'ellipsoïde E et dont le volume est nettement inférieur au volume de E.

E' sera l'ellipsoïde pour l'itération suivante.

De cette manière, l'algorithme génère une séquence d'ellipsoïdes qui contiennent tous F, et qui se rétrécissent en volume. Si un des ellipsoïdes possède un centre réalisable, l'algorithme s'arrête.

Mais si F est vide. Aucun rétrécissement des ellipsoïdes n'est possible,

et l'algorithme pourrait fonctionner pour toujours.

Nous le verrons, cependant, que dans ce cas, le volume de la région réalisable peut-être minoré par un v positif. Si tant d'itérations ont été réalisées et que l'ellipsoïde actuel a un volume inférieur à v, l'algorithme peut s'arrêter en toute sécurité « F=∅ ».

Dans le cas où l'ellipsoïde ne peut pas contenir tout F, si F est énorme l'ellipsoïde contient juste une fraction "assez grande" de celui-ci.

- **L'Algorithme :**

1. Soit K=0,et N=16n(n+1)L.

Soit t_0 0 n-vecteur, et soit $B_0 = n^2 2^{2L} I$.

(l'ellipsoïde K est définie par $E_K=\{ x \ / (x - t_K)^T B_K^{-1}(x - t_K) \leq 1$

Où B_K est définie positive, et l'ellipsoïde initial est $S(0 , n 2^L)$).

2. Pour K=0 jusqu'à K faire

Si t_K est la solution de $A x < b$, alors stop

Sinon construire un nouvel ellipsoïde dont le volume est nettement inférieur au volume

« Trouvé i telle que $A^i x \geq b_i$ »

Soit $a=(A^i)^T$.

$T_{K+1}= t_K - \frac{1}{n+1} \frac{B_K a}{\sqrt{a^T B_K a}}$ Soit $B_{K+1}=\frac{n^2}{n^2-1} (B_K - \frac{2}{n+1} \frac{(B_K a)(B_K a)^T}{a^T B_K a})$.

3. Après K itérations, l'ellipsoïde est trop petit pour que l'instance LSI soit réalisable affiché « irréalisable» et arrêtée.

Fin.

Exemple :

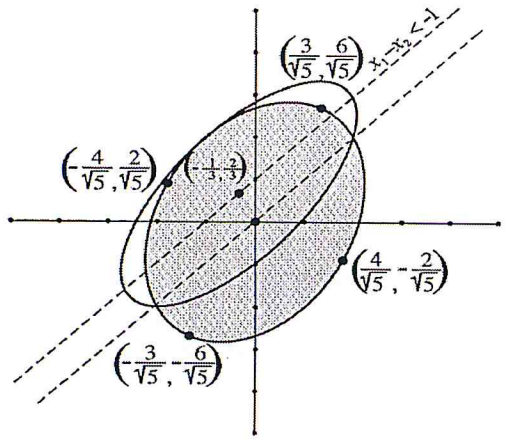
$B_K=\begin{bmatrix}5 & 2\\ 2 & 8\end{bmatrix}$, $t_K=\begin{bmatrix}0\\ 0\end{bmatrix}$

Parmi les contraintes de LSI $x_1 - x_2 < -1$

$$B_K\, a=\begin{bmatrix}5 & 2\\ 2 & 8\end{bmatrix}\begin{bmatrix}1\\ -1\end{bmatrix}=\begin{bmatrix}3\\ -6\end{bmatrix}$$

Et $a^T B_K\, a = 9$ nous en déduisons que

$$T_{K+1}=\begin{bmatrix}-\frac{1}{3}\\ \frac{2}{3}\end{bmatrix} \qquad \text{et} \qquad B_{K+1}=\begin{bmatrix}\frac{52}{9} & \frac{40}{9}\\ \frac{40}{9} & \frac{64}{9}\end{bmatrix}$$



Pour le nouveau ellipsoïde, nous pourrions utiliser tout ellipsoïde qui contient la partie de la région ombrée qui se trouve au-dessus de la ligne $x_1 - x_2 < -1$. Les points à l'extérieur de cette région ne sont pas nécessairement réalisables. Nous choisissons d'inclure la moitié de

l'ellipsoïde au dessus de la ligne $x_1 - x_2 = 0$ (Qui est parallèle à la contrainte violet et passe par le centre d'ellipsoïde) puisque la formule décrivant est relativement simple.

Chaque itération prend un temps borné par un polynôme en L et donc K est borné par un polynôme.

Est-ce que l'algorithme est Correct?

Pour prouver la validité de l'algorithme nous prouvons :

Théorème 3 [7]:

Soit B_K une matrice définie positive, $t_K \in \mathbb{R}^n$ et $a \in \mathbb{R}^n$. Construire d'après l'algorithme B_{K+1} et t_{K+1} donc:

1. B_{K+1} est définie positive tell que $E_{K+1}=\{ x \ / \ (x-t_{K+1})^T B_{K+1}^{-1}(x-t_{K+1}) \leq 1\}$ est un ellipsoïde
2. Soit $HE_K[a]$ Le demi-ellipsoïde contenant F

   $HE_K[a]=E_K \cap \{ x \in \mathbb{R}^n \ / \ a^T(x-t_K) \leq 0\}$

   $=E_{K+1}=\{ x \ /(x-t_{K+1})^T B_{K+1}^{-1}(x-t_{K+1}) \leq 1$ et $a^T(x-t_K) \leq 0\}$

   Alors $HE_K[a] \subseteq E_{K+1}$
3. $\frac{volume\ (E_{K+1})}{volume\ (E_K)} < 2^{\frac{-1}{2(n+1)}}$

   Laissez-nous d'abord prouver notre résultat dans un cas particulier simple.

   Lemme 3 :

   Soit $n>2$

   $$E=\{ x \in \mathbb{R}^n \ / \ (x-t)^T B^{-1}(x-t) \leq 1\}$$

   Où

   $$t=\begin{bmatrix} \frac{-1}{n+1} \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \quad \text{Et} \quad B=\begin{bmatrix} \frac{n^2}{n^2+1} & 0 & \cdots & 0 \\ 0 & \frac{n^2}{n^2-1} & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \frac{n^2}{n^2-1} \end{bmatrix}$$

Alors :

- B est définie positive (donc E est un ellipsoïde)
- l'hémisphère. $HS=\{ x \in \mathbb{R}^n \ / \ x^T x \leq 1$ et $x_1 \leq 0\} \subseteq E$.
- $\frac{volume\ (E_{K+1})}{volume\ (E_K)} < 2^{\frac{-1}{2(n+1)}}$

Preuve :

$B=QQ^T$

Où

$$Q=\begin{bmatrix} \frac{n}{n+1} & 0 & \cdots & 0 \\ 0 & \frac{n}{\sqrt{n^2-1}} & 0\cdots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \frac{n}{\sqrt{n^2-1}} \end{bmatrix}$$

- Soit $x \in HS$

$$(x-t)^T B^{-1}(x-t)=\frac{(n+1)^2}{n^2}(x_1+\frac{1}{n+1})^2+\frac{n^2-1}{n^2}\sum_{i=2}^{n} x_i^2 2$$

$$=\left[\frac{n^2-1}{n^2}\right] x^T x+\frac{2n+2}{n^2}x_1^2+\frac{2n+2}{n^2}x_1+\frac{1}{n^2} \leq 1+\frac{2n+2}{n^2}(x_1^2+x_1)$$

Comme $x \in S(0,1)$

Car $x \in HS$ $x_1^2+x_1=x_1(x_1+1) \leq 0$ par conséquent $(x-t)^T B^{-1}(x-t) \leq 1$ donc $HS \subseteq E$.

- $E=T(S(0,1))$ où $T(x)=t+Q\,x$ d'après propriété 1

volume(E)=|detQ| $[volume(S(0,1))]$.

$$\det Q = \frac{n}{n+1}\left[\frac{n^2}{n^2-1}\right]^{\frac{n-1}{2}}$$

On a ; $\forall y \in \mathbb{R}$ $1+y \leq e^y$

$$\frac{n}{n+1}=1-\frac{1}{n+1} \leq e^{-\frac{1}{n+1}}$$

$$\frac{n^2}{n^2-1}=1+\frac{1}{n^2-1} \leq e^{\frac{1}{n^2-1}}$$

Alors :

$$\det|Q| \leq e^{-\frac{1}{n+1}+\frac{n-1}{2(n^2-1)}} = e^{-\frac{1}{2(n+1)}} < 2^{-\frac{1}{2(n+1)}} .$$

Le lemme suivant concerne le cas particulier du lemme 3 au cas général du théorème 3.

Lemme 4 :

Soit B_K matrice définie positive et t_K et soit $a \in \mathbb{R}^n$

$$E=\{ x\in\mathbb{R}^n \quad / \quad (x-t)^T B^{-1}(x-t)\leq 1 \}$$

$HE_K[a]=E_K \cap \{ x\in\mathbb{R}^n \ / \ a^T(x-t_K) \leq 0\}$ comme dans le théorème 3 t_{K+1} et B_{K+1} sont obtenu par l'étape 2 de l'algorithme et soit HS, E définie comme le théorème précédant

Alors il existe une transformation affine T tel que :

(a) Le sphère unité sur E_K : $T(S(0,1))=E_K=\{ x\in\mathbb{R}^n \quad / \quad (x-t_K)^T B_K^{-1}(x-t_K)\leq 1 \}$.

(b) $T(E)=E_{K+1}=\{ x\in\mathbb{R}^n \quad / \quad (x-t_{K+1})^T B_{K+1}^{-1}(x-t_{K+1})\leq 1 \}$.

(c) L'hémisphère sur le demi –ellipsoïde $T(HS)=HE_K[a]$.

Preuve :

Puisque B est définie positive $B_K=QQ^T$; Q non-singulière d'après lemme 3 il existe une matrice orthogonale U^T tel que :

$$U^T(Q^T a)=\begin{bmatrix} \| Q^T a \| \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

Et soit $T(x)=t_K+(QU)\,x$, une transformation.

(a)

$$T(S(0,1)) =\{ T(x) \quad / \ x^T x\leq 1\}$$
$$=\{ y \ / \ (T^{-1}(y))^T(T^{-1}(y)) \leq 1\}$$
$$=\{ y \ /(y-t_K)^T(Q^{-1})^T UU^T Q^{-1}(y-t_K) \leq 1\}$$

$(Q^{-1})^T Q^{-1}=B_K^{-1}$ implique que

$$T(S(0,1)) =\{ y \ /(y-t_K)^T B_K^{-1}(y-t_K) \leq 1\}=E_K$$

(b)

$$B_{K+1}=\frac{n^2}{n^2-1}\left[B_K-\frac{2}{n+1}\frac{(B_Ka)(B_Ka)^T}{a^TB_Ka}\right]$$

$$=\frac{n^2}{n^2-1}\left[B_K-\frac{2}{n+1}\frac{(QUU^TQ^Ta)(a^TQUU^TQ^T)}{a^TQQ^Ta}\right]$$

Mais $U^T(Q^Ta)=\begin{bmatrix}\| Q^Ta \|\\0\\\cdot\\\cdot\\\cdot\\0\end{bmatrix}$ implique que :

$$(U^TQ^Ta)(a^TQU)=\begin{bmatrix}\| Q^Ta \|^2 & 0 & \cdots & 0\\0 & 0 & \cdots & 0\\\vdots & \cdots & \ddots & \vdots\\0 & 0 & \cdots & 0\end{bmatrix}$$

Donc $\frac{n^2}{n^2-1}B_{K+1}$

$$=B_K-\frac{1}{\|Q^Ta\|^2}\left[\frac{2}{n+1}QU\begin{bmatrix}\| Q^Ta \|^2 & 0 & \cdots & 0\\0 & 0 & \cdots & 0\\\vdots & \cdots & \ddots & \vdots\\0 & 0 & \cdots & 0\end{bmatrix}U^TQ^T\right]$$

$$=QQ^T-\frac{2}{n+1}QU\begin{bmatrix}1 & 0 & \cdots & 0\\0 & 0 & \cdots & 0\\\vdots & \vdots & \ddots & \vdots\\0 & 0 & \cdots & 0\end{bmatrix}U^TQ^T$$

$$=QU\left[I-\frac{2}{n+1}\begin{bmatrix}1 & 0 & \cdots & 0\\0 & 0 & \cdots & 0\\\vdots & \vdots & \ddots & \vdots\\0 & 0 & \cdots & 0\end{bmatrix}\right]U^TQ^T$$

$$=QU\begin{bmatrix}\frac{n-1}{n+1} & 0 & \cdots & 0\\0 & 1 & 0 & 0\\\vdots & 0 & \ddots & \vdots\\0 & 0 & 0 & 1\end{bmatrix}U^TQ^T$$

Par conséquent $B_{K+1}=QUBU^TQ^T$ où B est comme dans le lemme 3 aussi

$$x-t_{K+1}=x-t_K+\frac{QUU^TQ^Ta}{(n+1)\sqrt{a^TQQ^Ta}}$$

Mais

$$U^T(Q^Ta)=\begin{bmatrix} \| Q^T a \| \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

Alors :

$$x - t_{K+1} = x - t_K + \frac{QU\begin{bmatrix} \|Q^T a\| \\ 0 \\ \vdots \\ 0 \end{bmatrix}}{(n+1)\|Q^T a\|}$$

$$= x - t_K + QU\begin{bmatrix} \frac{1}{n+1} \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

$$=QU(U^{-1}Q^{-1}(x - t_K) - t)$$

$$=QU(T^{-1}(x) - t)$$

Par conséquent :

$$T(E)=\{ T(x) \quad / \ (x - t)^T B^{-1}(x - t) \leq 1\}$$
$$=\{ x \quad / \ (T^{-1}(x) - t)^T B^{-1}(T^{-1}(x) - t) \leq 1\}$$
$$=\{ x \quad / \ (x - t_{K+1})^T (Q^{-1})^T U B^{-1} U^T Q^{-1}(x - t_{K+1}) \leq 1\}$$

Mais comme nous montrons ci-dessus :

$$B_{K+1}^{-1}=(Q^{-1})^T U B^{-1} U^T Q^{-1} \quad \text{donc}$$

$$T(E)=\{ x \quad / \ (x - t_{K+1})^T B_{K+1}^{-1} (x - t_{K+1}) \leq 1\}$$

ceux-ci complètent la preuve de (b).

(c) le point y est dans $T(\{ x \in \mathbb{R}^n \ / \ x_1 \leq 0\})$ si et seulement si $y=t_K + QU\, x$

Pour un certain $x \in \mathbb{R}^n$ avec $x_1 \leq 0$

$$x_1 \leq 0 \Leftrightarrow \|Q^T a\| x_1 \leq 0 \Leftrightarrow x^T \begin{bmatrix} \| Q^T a \| \\ 0 \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \leq 0$$

$$\Leftrightarrow x^T U^T Q^T a \leq 0$$

Donc

$$y \in T(\{ x \ / \ x_1 \leq 0\}) \Leftrightarrow [U^{-1}Q^{-1}(y - t_K)]^T U^T Q^T a \leq 0$$

$$\Leftrightarrow (y - t_K)^T a \leq 0$$

$$\Leftrightarrow a^T (y - t_K) \leq 0$$

$T(\{ x \ / \ x_1 \leq 0\}) = \{ y \ / \ a^T(y - t_K) \leq 0\})$ d'après

$$T(HS) = T(S(0,1)) \cap \{ x \ / \ x_1 \leq 0\}$$

$$= T(S(0,1)) \cap \{ x \ / \ a^T(x - t_K) \leq 0\}$$

$$= E_K \cap \{ x \ / \ a^T(x - t_K) \leq 0\}$$

$$= HE_K[a]$$

Preuve de théorème 4 :

(a) D'après lemme (b) on a :

$$B_{K+1} = QUBU^T Q^T$$

$$= (QU\sqrt{B})(QU\sqrt{B})$$

Où $QU\sqrt{B}$ est une matrice non singulière, donc B_{K+1} est Définie positive et

$E_{K+1} = \{ x \ / \ (x - t_{K+1})^T B_{K+1}^{-1} (x - t_{K+1}) \leq 1\}$ est un ellipsoïde.

(b) $HE_K[a] = T(HS)$ D'après (c) de lemme 4 et (b) lemme 3on a : $HS \subset E$

Donc $HE_K[a] \subseteq T(E) = E_{K+1}$

(c) D'après propriété 1 on :

$$\frac{volume\,(E_{K+1})}{volume(E_K)} = \frac{volume(T(E))}{volume(T(S(0,1))}$$

$$= \frac{|detQU|(volume\,(E))}{|detQU|(volume(S(0,1))}$$

$$= \frac{volume(E)}{volume(S(0,1))} \leq 2^{-\frac{1}{2(n+1)}}$$

D'après lemme 3(c).

Lemme 5:

Si un système LSI de taille L a une solution, l'ensemble des solutions contenu dans la sphère $S(0,n2^L)=E_0$ a au moins $2^{-(n+2)L}$ volume.

Preuve :

Soit L la taille du système , si $A\,x< b$ a une solution z alors

$$A\,x< b$$
$$x_j < 2^L \quad \text{pour tout } j$$
$$x_j > 2^{-2L} \quad \text{pour tout } j$$

a une solution.

Preuve : Si $A\,x< b$ a une solution alors le système $A\,x \leq b$ a une solution.

Soit v solution du système $A\,x \leq b$ tel que $|v_j|< 2^L$ pour tout j

On analyse le système $A\,x + I\,y=b$, $y\geq 0$ Pour $\mathcal{E}>0$, $v + \mathcal{E}\,(z - v)$ satisfait les inégalités du système strict.

Donc le polytope $A\,x < b$ a un point intérieur d'après propriété 4

$$x_j < 2^L \quad \text{pour tout } j$$
$$x_j > 2^{-2L} \quad \text{pour tout } j$$

Il a n+1 sommets v_0 , v_1 , . . . , v_n . où C (convexe) a un volume positif. tout les points intérieurs de C sont des solutions du système $A\,x \leq b$ dans la sphère $S(0,n2^L)$

Lemme dit que C de dimension n a au moins

$$\frac{1}{n!}\left|\det\begin{bmatrix} 1 & 1 & \cdots & 1 \\ | & | & \cdots & | \\ v_0 & v_1 & \cdots & v_n \\ | & | & \cdots & | \end{bmatrix}\right| > 0$$

Chaque v_K peut s'écrire comme $\frac{u_K}{D_K}$ où u_K est un vecteur entier et le déterminant

$|D_K| < 2^L$

$$\text{donc} \quad \frac{1}{n!}\left|det\begin{bmatrix} D_0 & D_1 & \cdots & D_n \\ | & | & \cdots & | \\ u_0 & u_1 & \cdots & u_n \\ | & | & \cdots & | \end{bmatrix}\right| \frac{1}{|D_0||D_1|\ldots|D_n|}$$

$\det\begin{bmatrix} D_0 & D_1 & \cdots & D_n \\ | & | & \cdots & | \\ u_0 & u_0 & \cdots & u_n \\ | & | & \cdots & | \end{bmatrix}$ est différent de zéro.

Alors le volume de C est au moins $(n! \prod_{K=0}^{n} |D_K|)^{-1}$ puisque $|D_K| < 2^L$ pour tout K

$\prod_{K=0}^{n} |D_K| < 2^{(n+1)L}$ et $n! < 2^L$ implique que le volume de C est $< 2^{-L(n+2)}$

Théorème 5 [7]:

L'algorithme ellipsoïde résout LSI [7].

Preuve :

Soit $E_0 = S(0,n2^L)$

$\forall r \geq 0$, $S(0,r) \subseteq \{ x \quad / \quad |x_j| \leq r \quad \forall j \}$

Volume $(S(0,r)) < (2r)^n$

Si l'algorithme renvoie que t_K est réalisable, il est évidemment correct, donc supposons que l'algorithme se termine par irréalisable t_K mais le problème est réalisable d'après lemme

volume $(F \cap E_0) > 2^{-L(n+2)}$

$$\text{volume } (E_K) < (\text{volume}(E_0))2^{-\frac{K}{2(n+1)}}$$

$$< \text{volume } (E_0)2^{-8nL}$$

$$\text{volume } (E_0) < (2n\,2^L)^n \quad \text{car} \quad E_0 = \subseteq \{ x \quad / \quad |x_j| \leq n\,2^L \}$$

volume $(E_K) < (2n\,2^L)^n(2^{-8nL}) < 2^{2nL}2^{-8nL} = 2^{-6nL} < 2^{-(n+1)L}$, contradiction.

III.4. Conclusion :

Depuis les travaux de Khachiyan [12], on sait que les programmes linéaires peuvent être résolus en temps polynomial en fonction de la taille du problème.

L'algorithme polynomial étudié par Khachiyan fut un grand succès théorique malheureusement cet algorithme est moins efficace en pratique que l'algorithme du simplexe pour résoudre la plupart des programmes linéaires.

Chapitre IV

METHODE PROJECTIVE DE KARMARKAR

IV.1. Introduction :

En 1984 N.Karmarkar a proposé un algorithme polynomial qui produit une suite de solutions réalisables (dans l'intérieur relatif de la région admissible) convergeant vers une solution optimale du problème traité.

IV.2. Définition de problème :

Soit A un élément de $\mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ et $c \in \mathbb{R}^n$ et posons par définition :

$\Omega = \{x \in \mathbb{R}^n : Ax = b\}$

$P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\} = \Omega \cap \mathbb{R}^n_+$ Où $\mathbb{R}^n_+$ désigne l'orthant positif.

Considérons alors le programme linéaire général :

$$(p.l.g)\begin{cases}\text{Min } c^T x \\ x \in P\end{cases}$$

Si l'on remplace $\mathbb{R}^n_+$, par une sphère (ou un ellipsoïde) $E_1 \subset \mathbb{R}^n_+$

On obtient le problème :

$$(1)\begin{cases}\text{Min } c^T x \\ x \in \Omega \cap E_1\end{cases}$$

Or l'intersection d'une sphère et d'un espace affine est une sphère de dimension plus petite dans cet espace affine, donc$(\Omega \cap E_1 = E_2)$.

Le problème devient :

$$(2)\begin{cases}\text{Min } c'^T x \\ x \in E_2\end{cases}$$

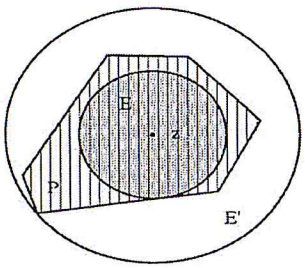
C' désigne la projection orthogonale de c sur Ω.

D'après ce qui précède, le problème(2) est trivial : il suffit de se déplacer à partir de centre de E_2 dans la direction –c' d'une distance égale au rayon deE_2.Finalement tout revient à remplacer P par un ellipsoïde (ou une sphère)E de centre $a^0(a^0 \in P: a_i^0 > 0, i = 1,..,n)$et minimiser$c^T x$ sur E au lieu de P.

Des bornes sur la fonction objectif:

Soit a^0 point intérieur dans le polytope P. Supposons que nous tirons un ellipsoïde E avec le centre a^0 qui est contenu dans le polytôpe et résoudre le problème d'optimisation sur la région E restreint au lieu de P. La question donc de savoir quelle sera la qualité de la solution obtenue comparée à la solution du problème optimale? Pour tirer un bond, nous créons un autre E'ellipsoïde en agrandissant E par un nombre suffisamment important facteur v de telle sorte que E' contient P.

$$E \subset P \subset E', E' = vE$$



Notons par $f_E, f_p, f_{e'}$ les minima de $f(x) = c^T x$ sur E,P,E'.

Lemme 1 :

$$0 \leq \frac{f_E - f_p}{f(a^0) - f_p} \leq (1 - \frac{1}{v})$$

Preuve :

On a:$f(a^0) - f_E \leq f(a^0) - f_P \leq f(a^0) - f_{E'} = v[f(a^0) - f_E]$Car f est linéaire

$$\frac{f_E - f_p}{f(a^0) - f_p} \leq (1 - \frac{1}{v})$$

En partant donc de a^0 on arrive à point a_1 minimisant f sur E et f_P est approcher par un facteur de $(1 - \frac{1}{v})$ .On recommence l'opération en choisissant a^ncomme centre d'ellipsoïdeE_1 contenu dans et d'un ellipsoïde $E'_1 = v_1 E_1$ contenant P on peut encore réitérer un point $a_2$et centre d'un ellipsoïde E_2 et ainsi de suite .Le but étant de borner $f_{E_i} - f_p$ par une valeur de plus en plus petit jusqu' à atteindre une précision voulue.

L'idée serait bonne si seulement si on pouvait facilement construire les ellipsoïdes E_i et E_i'

Malheureusement ce n'est pas le cas pour un polyèdre P quelconque, c'est pourquoi Karmarkar introduit une transformation projective qui envoie P dans un n-simplexe de$\mathbb{R}^{n+1}$.

IV.3.Transformation projective :

Soit $a = (a_1, a_2, \dots, a_n)$ un point strictement intérieur à P (i.e. tel que $a \in \Omega$ et $a_i > 0$)

Et posons $D = \mathrm{diag}(a)$. Définissons alors le simplexe S_{n+1} de dimension n contenu dans $\mathbb{R}^{n+1}$

$$S^{n+1} = \{x \in \mathbb{R}^{n+1}, x \geq 0 \text{ et } e_{n+1}^T x = 1\}$$

La transformation projective (notée T) est une fonction :$\mathbb{R}_+^n \rightarrow \mathbb{R}_+^{n+1}$

Définie par :

$$y = T(x) \text{avec} \begin{cases} y_i = \dfrac{x_i/a_i}{1 + \sum_{i=1}^n x_i/a_i} \\ y_{n+1} = 1 - \sum_{i=1}^n y_i \end{cases} \quad i = 1, \dots, n$$

Ou encore $y_{[n]} = (D^{-1}x)y_{n+1}$ ou $y_{[n]}$ désigne les n première composantes de y

Ceci montre que Test bijective : En effet $T^{-1}(y) = x = \dfrac{Dy_{[n]}}{y_{n+1}}$

Evidemment :$T(\mathbb{R}_+^n) = S_{n+1} \subset \mathbb{R}^{n+1}$

$T(a) = a^0 = (1/n+1)e_{n+1}$ centre de S_{n+1}

Quelques propriétés :

1. L'image d'un espace affine Ω par T est un espace affine Ω', donc puisque $a \in \Omega$ son image $a^0 \in \Omega'$
2. L'image d'une face $x_i = 0$ de $\mathbb{R}_+^n$ est la face correspondante ($y_i = 0$ $i = 1, \dots, n$)

Du simplexe S_{n+1}.Quant à la face $y_{n+1} = 0$ de S_{n+1} , elle est l'image des points à l'infini.

Il est facile de voir que la plus grand sphère inscrit dans S_{n+1} et la plus petite sphère contenant S_{n+1} centrées en a^0 sont respectivement :

$B(a^0, r) = \{x \in \mathbb{R}^{n+1} : e_{n+1}^T x = 1, \|x - a^0\| \leq r\}$

$B(a^0, R) = \{x \in \mathbb{R}^{n+1} : e_{n+1}^T x = 1, \|x - a^0\| \leq R\}$

Où $r = 1/\sqrt{n}(n+1)$ et $R = \sqrt{n}/(n+1)$ D'où $R/r = n$

La situation est donc la suivante : $B(a^0, r) \subset S_{n+1} \subset B(a^0, R)$ ce qui entraine

$B(a^0, r) \cap \Omega' \subset S_{n+1} \cap \Omega' \subset B(a^0, R) \cap \Omega'$.

Or $S_{n+1} \cap \Omega' = P' = T(P = \Omega \cap \mathbb{R}_+^n)$.De plus l'intersection d'une sphère B avec un espace affine Ω est une sphère B' de dimension plus petit et de même rayon que B si Ω contiennent

le centre de B, C'est le cas ici puisque $a^0 \in \Omega'$.Ainsi : $B'(a^0, r) \subset P' \subset B'(a^0, R)$

où : $B'(a^0, r) = B(a^0, r) \cap \Omega'$ et $B'(a^0, R) = B(a^0, R) \cap \Omega'$.

Ceci prouve que la minimisation sur la sphère inscrite dans la région admissible réduit la valeur de l'objectif d'au moins $1 - 1/n$.

IV.4. Description de la méthode :

IV.4.1. Le problème traité par Karmarkar et les hypothèses de travail :

On se place désormais dans $\mathbb{R}^n$: le simplexe $S_n = \{x \in \mathbb{R}^n : e_n^T x = 1, x \geq 0\} \subset \mathbb{R}^n$ est donc de dimension $(n-1)$.

La méthode projective de Karmarkar résoud directement le programme linéaire réduit suivant :

$$(p.l.r)\begin{cases} \text{Min } c^T x \\ Ax = 0 \\ x \in S_n \end{cases}$$

En suppose que :

1. La valeur optimale est nulle, i.e. : si x^* est une solution de (p.l.r) alors :

$c^T x^* = Z^* = 0$.

2. Le point $a^0 = \frac{1}{n} e_n$(centre de simplexe S_n)est une solution réalisable de (p.l.r).
3. La matrice A est de plein rang : rg(A)=m.

Ces trois hypothèse seront appelées les conditions de Karmarkar.

On suppose également que $c^T a^0 > 0$.puisque si $c^T a^0 = 0$on s'arrête immédiatement avec a^0optimal .Ceci implique que $c^T x$ n'est pas constant sur la région admissible et par conséquent il est strictement positif pour tout x réalisable.

Remarques :

(a) I la valeur optimal est connu à priori mais non nulle l'égalité $e_n^T x = 1$permet de se ramener à un objectif nul. En effet, soit x une solution optimale au problème et Z^* la valeur optimale de l'objectif Alors$c^T x = Z^* = Z^* e_n^T x \Rightarrow (c - Z^* e_n)^T x = c'^T x = 0$. On minimise alors l'objectif $c'^T x$ au lieu de $c^T x$, ou$c'_i = c_i - Z^*\ i = 1, \dots, n$.

(b) Si le système de contrainte est de la forme $Ax = b, b \neq 0$, on se ramène facilement à un système homogène .il suffit d'écrire : $Ax = be_n^T x \Rightarrow (A - be_n^T)x = 0$. Autrement dit on obtient un système de la forme $A'x = 0$ ou les élémentsde A'sont $a'_{ij} = a_{ij} - b_i\ , i = 1, \dots, m, j = 1, \dots, n$.

(c) Le problème $(p.l.r)$ ainsi défini peut être vue comme un problème de faisabilité (Voir[6] et [8]). En effet puisque 'on suppose la valeur optimale Z^*nulle ou connue à priori, il s'agit alors de trouver $x \geq 0$ tel que $\begin{cases} c^T x = Z^* \\ Ax = 0 \\ e_n^T x = 1 \end{cases}$

IV.4.2. L'algorithme de base :

Nous présentons dans ce paragraphe l'algorithme de base de KARMAKAR pour résoudre un programme linéaire du type $(p.l.r)$ pour cela on se donne une précision ε(par exemple $\varepsilon = 2^{-q}$ Où q est un entier,$q \geq 1$).

Partant de la solution initiale$x^0 = a^0$, l'algorithme produit une suite de pointsintérieurs qui converge vers une solution optimale du problème en un temps polynomial.

Dans le but de ramener l'objectif $c^T x$ à zéro on le minimise localement sur une sphère inscrite dans la région admissible. A chaque itération (k) l'itéré $(x^k > 0)$ est ramené au centre de S_n par la transformation projectiveT_k définie par:

$T_k: x \in S_n \quad \dashrightarrow \quad T_k(x) = y \in S_n$ avec

$$T_k(x) = \frac{D_k^{-1}\, x}{e_n^T D_k^{-1} x} = y \quad : \quad T_k^{-1}(y) = \frac{D_k x}{e_n^T D_k x} \quad . D_k = \mathrm{diag}(x^k)$$

Et ainsi de suite jusqu'à ce que le test d'optimalité ($c^T x^k \leq \varepsilon$) soit vérifié.

- <u>Début d'algorithme :</u>

Initialisation $x^0 = a^0 = (1/n)e_n k = 0$

Tant que $c^T x^k > \varepsilon$ **faire**

Pas 0 $\begin{cases} D_k = \mathrm{diag}(x^k) \\ B_k = \begin{bmatrix} A_k \\ \cdots \\ e_n^T \end{bmatrix} \end{cases} \quad A_k = AD_k$

Pas 1 $p_k = \left\{ I - B_k^T \left(B_k B_k^T \right)^{-1} B_k \right\} D_k c$

Pas 2 $d_k = \frac{p_k}{\|p_k\|}$

Pas 3 $y^k = a^0 - \alpha r d_k \quad .r = \frac{1}{\sqrt{n(n-1)}} \quad .0 < \alpha < 1$

Pas 4 $x^{k+1} = \frac{D_k x}{e_n^T D_k x} = T_k^{-1}(y) \quad .k = k + 1$

Fin tant que

Fin d'algorithme.

Dans le pas 0, on ne fait que construire la matrice des contraintes (i.e. B_k)

Le pas 1 consiste à projeter $c_k = D_k c$ sur le noyau B_k.la formule donnant p_k est obtenue par un calcul élémentaire en utilisant le fait que$A_k e_n = 0$.

Au pas 2 on calcule le vecteur normé d_k correspondant àp_k .

Au pas 3 on choisit y^kà une distance αr de a^0 dans la direction $-d_k$.

Karmarkar choisit $\alpha = \frac{1}{4}$.

Et en fin au pas 4 on effectue la transformation inverse T_k^{-1}pour calculer le nouvel itéré x^{k+1}.

Exemple :

$$\begin{cases} \text{Min } 2x_2 - x_3 \\ x_1 - 2x_2 + x_3 = 0 \\ x_1 + x_2 + x_3 = 1 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

n=3, $c = \begin{bmatrix} 0 \\ 2 \\ -1 \end{bmatrix}$, A=[1 -2 1], $x^0 = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}$, $r = \frac{1}{\sqrt{6}}$, α=0.22.

Itération k=0 :

$$D_0 = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$$

$$AD_0 = [1 \; -2 \; 1] \times \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix}$$

$$B = \begin{bmatrix} AD_0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & -\frac{2}{3} & \frac{1}{3} \\ 1 & 1 & 1 \end{bmatrix}$$

$$B_0^T(B_0B_0^T)^{-1}B_0 = \begin{bmatrix} 0.5 & 0 & 0.5 \\ 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \end{bmatrix}$$

$$p_0 = \{I - B_0^T(B_0B_0^T)^{-1}B_0\}D_0c = \begin{bmatrix} \frac{1}{6} \\ 0 \\ -\frac{1}{6} \end{bmatrix}$$

$$\|p_0\| = \sqrt{(1/6)^2 + 0 + (1/6)^2} = \frac{\sqrt{2}}{6}$$

$$d_0 = \frac{p_0}{\|p_0\|} = \begin{bmatrix} \sqrt{2} \\ 0 \\ -\sqrt{2} \end{bmatrix}$$

$$y^0 = a^0 - \alpha r d_0 = \begin{bmatrix} 0.2692 \\ 0.3333 \\ 0.3974 \end{bmatrix} \text{ Et } x^1 = \begin{bmatrix} 0.2692 \\ 0.3333 \\ 0.3974 \end{bmatrix}.$$

IV.4.3. Dérivation et analyse de l'algorithme :

On a vu que la transformation T_k applique le simplexe S_n dans lui-même, en même temps l'itéré $x^k > 0$(dont les composantes forment la matrice diagonaleD_k) est envoyé au centre de S_n.Cependant le transformé du programme linéaire $(p.l.r)$ est le programme suivant :

$$(p.n.l)\begin{cases} \text{Min } \dfrac{c^T D_k y}{e_n^T D_k y} \\ \dfrac{AD_k y}{e_n^T D_k y} = 0 \\ e_n^T y = 1 \ . y \geq 0 \end{cases}$$

Mais on a pour tout $y \in S_n e_n^T D_k y = \sum_{i=1}^n x_i^k y_k \geq \text{Min}\{x_i^k : i = 1, \dots, n\} > 0$

Les égalités $\frac{c^T D_k y}{e_n^T D_k y} = 0$ et $\frac{AD_k y}{e_n^T D_k y} = 0$ sont donc satisfaites si et seulement si :

$c^T D_k y = 0$ et $AD_k y = 0$

(p.n.l) est alors équivalent au programme linéaire :

$$(p.l)\begin{cases} \text{Min } c^T D_k y \\ AD_k y = 0 \\ e_n^T y = 1 \ , y \geq 0 \end{cases}$$

Qui est de la forme (p.l.r) et vérifie les conditions de Karmarkar.

Une remarque fondamentale :

Si on ajoute au problème $(p.l)$la contrainte $\{y \in \mathbb{R}^n : \|a^0 - y\| \leq \alpha r$Où $0 < \alpha < 1$ et

$r = \frac{1}{\sqrt{n(n-1)}}$.

i.e. la sphère $B(a^0, \alpha r)$ de centre a^0et de rayon αr,la contrainte de positivité $(y \geq 0)$

devient redondante. Ce résultat est une conséquence du lemme général suivant :

Lemme 2 :

Si pour un programme linéaire donné on connait une solution réalisable y telle que

$(y_i > 0\,, i = 1, \dots, n)$ alors l'ellipsoïde $E = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{y_i^2} \leq \beta^2 \;, 0 < \beta < 1\}$

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

Preuve :

Supposant au contraire qu'il existe $j \in \{1, \dots, n\}$ tel que $x_j \leq 0$ alors

$\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{y_i^2} \geq \frac{(x_i - y_i)^2}{y_i^2} \geq 1 > \beta^2$ (c.q.f.d)

Pour retrouver le resultat précédent il suffit de prendre $y = a^0$ et $\beta = \alpha r$

Finalement, le problème (p.l) devient :

$$(\text{paux}) \begin{cases} \text{Min } c^T D_k y \\ AD_k y = 0 \\ e_n^T y = 1 \\ \|y - a^0\| \leq \alpha r \end{cases}$$

On vient de remplacer l'orthant positif par la sphère $B(a^0, \alpha r)$ qui est beaucoup plus simple à manier.

Théorème 1[1] :

Le point y^k (du pas 3 de l'algorithme) est une solution optimale de (paux).

Preuve :

Posons $x = y - a^0$, alors $B_k x = 0$, où B_k est la matrice des contraintes de (paux) définie au pas 0 de l'algorithme.(Paux) est alors équivalent (à une translation prés) au problème suivant :

$$(p*)\begin{cases} \text{Min } c^T D_k x \\ B_k x = 0 \\ \|x\|^2 = \sum_{i=1}^{n} x_i^2 \leq \alpha r \end{cases}$$

Or x* est une solution optimale de (p*) si et seulement si $\exists\, \lambda \in \mathbb{R}^{m+1}$ et un scalaire $\mu \geq 0$ tels que :

$$D_k c + B_k^T \lambda + \mu x^* = 0$$

En multipliant les deux membres de (i) 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 $(B_k x = 0) \mapsto \lambda = -\left(B_k B_k^T\right)^{-1}(B_k D_k c)$

D'où en substituant dans(i) :

$$x^* = (-1/\mu)\left[I - B_k^T\left(B_k B_k^T\right)^{-1} B_k\right] D_k c = (-1/\mu) p_k$$ (Ce qui justifié le pas 1).

Mais x^* vérifier $\|x^*\| = \frac{1}{\mu}\|p_k\| = \alpha r \;\mapsto \frac{1}{\mu} = \frac{\alpha r}{\|p_k\|} \mapsto x^* = -\alpha r \frac{p_k}{\|p_k\|} = \alpha r d_k$

Soit finalement :

$$y^k = y^* = a^0 + x^* = a^0 - \alpha r d_k$$ (c.q.f.d)

IV.4.4. Propriété fondamentale de y^k:

Le point y^k est tel que : $\frac{c^T D_k y}{c^T D_k a^0} \leq 1 - \frac{\alpha}{n-1}$

Preuve :

Cette propriété est démontrée dans ([13] lemme 4) et même dans ([15] théorème 3). Cependant, on peut la démontrer facilement en appliquant le lemme 1 : il suffit de considérer le problème (p.l) et prendre :

$$P = \{y \in \mathbb{R}^n : AD_k y = 0,\ e_n^T y = 1, y \geq 0\}$$

$$E = B(a^0, \alpha r)$$

Dans ce cas on a :

$$f_E = c^T D_k y^k f_P = 0f(a^0) = c^T D_k a^0 Etv = \frac{R}{r} = \frac{\sqrt{(n-1)/n}}{\frac{1}{\sqrt{n(n-1)}}} = n - 1.$$

IV.4.5. Fonction potentiel (Etude de la convergence):

Pour établir la convergence de l'algorithme, il faut montrer que :

$\frac{c^T x^{k+1}}{c^T x^k} < q_0$ Où $0 < q_0 < 1$ est indépendant de k

Or il est difficile de trouver directementq_0. Pour surmonter cette difficulté, Karmarkar associe à l'objectif linéaire $c^T x$ la fonction potentiel $f(x) = \sum_{i=1}^{n} \log \frac{c^T x}{x_i}$ définie sur

$F = \{x \in \mathbb{R}^n : x > 0 , Ax = 0 , e_n^T x = 1\}$

Le lemme suivant montre que la minimisation (réduction) de f(x) conduit droit à celle de $c^T x$.

Lemme 3 :

Soit x^k le $K^{ième}$ itéré de l'algorithme, alors :$\frac{c^T x^k}{c^T x^0} \leq (\exp[f(x^k) - f(x^0)])^{\frac{1}{n}}$ où $x^0 = \frac{e_n}{n}$

Preuve :

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

Or $\left(\prod_{i=1}^{n} \frac{x_i^0}{x_i^k}\right)^{\frac{1}{n}} = n(\prod_{i=1}^{n} x_i^k)^{\frac{1}{n}} \leq \frac{n\sum_{i=1}^{n} x_i^k}{n} = 1$ (puisque$x^k \in S_n$), d'où le resultat.

Conséquence immédiate :

Si la suite $f(x^k)$ tend vers -∞ alors la suite $c^T x^k$ tend vers zéro, autrement dit pour diminuer suffisamment f(x).

IV.4.6. Propriétés de f(x) :

1. Une propriété crucial de f(x) est qu'elle conserve toutes ses caractéristique par T_k i.e. :

Elle est transformée en une fonction de la même forme :

$g(y) = \sum_{i=1}^{n} \log \frac{c^T D_k y}{y_i} - \sum_{i=1}^{n} \log(x_i^k)$ Où $y = T_k(x)$.

g(y) est bien définie sur $F = \{y \in \mathbb{R}^n : y > 0 , AD_k y = 0 , e_n^T y = 1\}$

2. Soit y=T(x) et $y^0 = T(x^0)$ alors on a :

$f(x) \leq f(x^0) - d \Leftrightarrow g(y) \leq g(y^0) - d$, d étant un réel positif. En d'autres termes tout réduction de f(x) entraîne la même réduction de g(y) et réciproquement.

3. La fonction potentielle g (y) est associée à la fonction linéaire $c^T D_k y$ (objectif du problème paux).
4. g(y) est une fonction monotone croissante de $c^T D_k y$.

Théorème de convergence de Karmarkar [1] :

Si $0 < \alpha < \frac{1}{4}$, alors en partant de $x^0 = e_n/n$ après O (nq+nlog) itération l'algorithme trouve un point réalisable x tel que :

1. $c^T x = 0$

   Ou

2. $\frac{c^T x}{c^T x^0} \leq 2^{-q}$ où q est une précision fixée.

Un résultat plus précis est donné juste après par [11] qui montrent en utilisant la fonction potentiel: $h(x) = \frac{c^T x}{\prod_{i=1}^{n} x_i^{\frac{1}{n}}}$

Que la convergence et réalisée en O (nq) itérations pour $0 < \alpha < 0.7968 \ldots$

Nous allons nous attarder peu sur le choix du paramètre de convergence α dont dépend en partie la vitesse de convergence.

Choix de Karmarkar :

Karmarkar montre en fait que pour n grand ($n \longrightarrow +\infty$) la fonction potentiel f(x) diminue à chaque itération d'une quantité :

$\delta(\alpha) = \alpha - \alpha^2/2 - \alpha^2/(1-\alpha)$ i.e. $f(x^k) \leq f(x^{k-1}) - \delta(\alpha)$. Il suggère alors de prendre α=1/4,

valeur qui correspond approximativement à la valeur maximale de $\delta(\alpha)$:

$$\text{Max}\{\delta(\alpha), 0 < \alpha < 1\} = 0.245122 \ldots$$

Autre choix de α :

Nous allons montrer en s'inspirant de la démonstration de [13] que l'o peut très bien choisir (α>1/4)

Considérons ainsi le théorème suivant :

Théorème 2[1] :

Soit x^k le $K^{ième}$ itéré de l'algorithme, alors on a : $\frac{c^T x^k}{c^T x^0} \leq (\frac{\exp(-2\alpha)}{1-\alpha})^{\frac{k}{n}}$

Preuve :

Montrons d'abord que $\frac{h(x^k)}{h(x^0)} \leq (\frac{(\exp(-2\alpha)}{1-\alpha})^{\frac{k}{n}} \longrightarrow (1)$

D'après [11] : $\frac{h(x^k)}{h(x^0)} \leq [g(\alpha, n)]^k$ où $g(\alpha, n)$ est définie par :

$g(\alpha, n) = \frac{1-\alpha(n-1)}{1+\alpha(n-1)} [\frac{1+\alpha/(n-1)}{1-\alpha}]^{\frac{1}{n}}$ Or $g(\alpha, n)$peut etre sécrit sous la forme suivante:

$$g(\alpha, n) = \left(\frac{[1-\alpha/(n-1)]^{n-1}\,[1-\alpha/(n-1)]\,[1+\alpha/(n-1)]}{[1-\alpha/(n-1)]^{n-1}\,[1+\alpha/(n-1)[1-\alpha]}\right)^{1/n}$$

De plus$[1-\alpha/(n-1)] \leq 1$, il vient que :

$$g(\alpha, n) \leq \left(\frac{[1-\alpha/(n-1)]^{n-1}}{[1-\alpha/(n-1)]^{n-1}\,[1-\alpha]}\right)^{1/n}$$

Or pour |x|<1 on a : $\log\frac{1-x}{1+x} \leq -2x$ d'où pour $x = \frac{\alpha}{n-1}$

$$\log\frac{[1-\alpha/(n-1)]}{[1+\alpha/(n-1)]} \leq -\frac{2\alpha}{n-1} \Rightarrow (\frac{1-\alpha/(n-1)}{1+\alpha/(n-1)})^{n-1} \leq \exp(-2\alpha)$$

Soit finalement : $g(\alpha, n) \leq (\frac{\exp(-2\alpha)}{1-\alpha})^{\frac{1}{n}}$

Cette dernière inégalité entraine (1)

Il reste à montrer que $\frac{c^T x^k}{c^T x^0} \leq \frac{h(x^k)}{h(x^0)} \longrightarrow (2)$

En effet $\frac{h(x^k)}{h(x^0)} = \frac{c^T x^k}{c^T x^0} (\prod_{i=1}^n \frac{x_i^0}{x_i^k})^{\frac{1}{n}} = \frac{c^T x^k}{c^T x^0} (\prod_{i=1}^n \frac{1}{nx_i^k})^{\frac{1}{n}} \geq \frac{c^T x^k}{c^T x^0}$

Puisque $(\prod_{i=1}^n x_i^k)^{\frac{1}{n}} \leq \frac{1}{n}\sum_{i=1}^n x_i^k = \frac{1}{n} \Rightarrow (\prod_{i=1}^n \frac{1}{nx_i^k})^{\frac{1}{n}} \geq 1$ c.q.f.d

Considérons alors la fonction : $t(\alpha) = \frac{\exp(-2\alpha)}{1-\alpha}$ où 0<α<1

On peut montrer que :Min{t(α) : 0<α<1=t (1/2)=0.735<1}.

I.e. : (α=1/2) est un choix théoriquement plus convenable que celui de Karmarkar.

Remarques :

1. Du point de vue pratique pour (α=1/2) l'algorithme est deux fois plus rapide que pour (α=1/4).
2. Le fait que t (1/2) soit la plus petite valeur de t(α) pour 0<α<1 ne signifie pas que c'est pour cette valeur l'algorithme est plus rapide, d'ailleurs il converge plus vite pour α>1/2

IV.4.7. Résolution de (p.l.g) :

Remarque :

La mise en œuvre de l'algorithme de Karmarkar, son implémentation effective l'étude de ces deux problèmes :

1. La transformation du problème (p.l.g) (ci-dessous) à la forme réduite

$$(\text{p.l.g})\begin{cases}\text{Min } c^T x = Z^* \\ Ax = b \\ x \geq 0\end{cases}$$

2. La modification de l'algorithme de Karmarkar en vue de son extension au cas où la valeur optimale Z^* n'est pas connue.

Pour résoudre (p.l.g) Karmarkar suggère indirectement une méthode à deux phases qui consiste à ramener (p.l.g) à la forme (p.l.r) en appliquant la transformation projective T ce qui nécessite la connaissance d'une solution réalisable ou la résolution du problème de faisabilité (phase 1) et passe en suite à la minimisation (phase 2).

IV.4.7.1.résolution du problème de faisabilité (phase 1) :

Un point réalisable de (p.l.g) est une solution du problème :

$$p(1)\begin{cases}Ax = b \\ x \geq 0\end{cases}$$

Or d'après Karmarkar (P1) est équivalent au problème de minimisation

$$p(2)\begin{cases}\text{Min } \lambda \\ Ax + \lambda(b - Ax^0) = 0 \\ x \geq 0, \lambda \geq 0\end{cases}$$

Où $x^0 > 0$ choisit arbitrairement dans l'orthant positif et λ une variable artificielle.

Notons que (P2) est toujours réalisable il suffit de prendre $x=x^0$ et $\lambda=1$.

Soit λ_m la valeur minimale de l'objectif. Alors $\lambda_m = 0$ correspond à une solution réalisable de (P1), cependant on peut se contenter d'une solution presque nulle. Plus précisément Karmarkar démontre le théorème suivant :

Théorème3 [1] :

$\exists \varepsilon_0 > 0$ Tel que : les deux propositions suivant son équivalentes :

1. P(1) est réalisable.
2. (P2) admet une solution (x, λ) telle que$\lambda \leq \varepsilon_0$.

Remarques :

1. La résolution de(P1) se ramène donc à celle de (P2), problème auquel on appliquera l'algorithme de base, puisque connaissant la valeur optimale et une solution réalisable de ce dernier il est facile de le mettre sous la forme (p.l.r).
2. Le problème de faisabilité (P1) peut être résolu par une quelconque méthode à condition que la solution trouvée soit dans l'intérieur relatif du domaine de faisabilité$\{x \geq 0, Ax = b\}$. Par exemple la méthode de simplexe ne convient pas puisqu'elle donne une solution sur la frontière.

IV.4.7.2. Problème de (Phase2) :

Soit a $(a_i > 0, i = 1, \dots, n)$ une solution réalisable de (p.l.g) obtenue (par exemple par la méthode précédente). En appliquant la transformation de système Ax=b devient A'y=0 où A' est une $m \times (n+1)$matrice définie par :

$A' = [AD_a - b]$, D_a=diag (a).

Quant à l'objectif il est transformé en une fonction non linéaire : $\frac{c^T D_a y[n]}{y_{n+1}}$

Où y[n] est un vecteur de $\mathbb{R}^n$ formé des n premières composantes de $y = T(x) \in \mathbb{R}^{n+1}$

On se retrouve alors avec le problème :

$$(\text{p.l.f})\begin{cases}\text{Min}\dfrac{c^{T}D_{a}y[n]}{y_{n+1}}\\ A'y=0\\ e_{n+1}^{T}y=1\\ y\geq 0\end{cases}$$

Or si la valeur optimale Z^{*} est connue, on peut écrire

$$\frac{c^{T}D_{a}y[n]}{y_{n+1}}=Z^{*}\mapsto c^{T}D_{a}y[n]-Z^{*}y_{n+1}=c'^{T}y=0$$

$$\text{Où}\begin{cases}c'_{i}=c_{i}a_{i} & i=1,\dots,n\\ c'_{n+1}=-Z^{*}\end{cases}$$

On obtient le programme linéaire réduite :

$$(\text{p.l.f})\begin{cases}\text{Min}c'^{T}y\\ A'y=0\\ e_{n+1}^{T}y=1\\ y\geq 0\end{cases}$$

IV.5. Transformation de (p.l.g) à la forme (p.l.r) :

Nous présentant ici deux formulations différentes supposons alors qu'il existe un nombre réel σ> 0 assez grand pour que l'on ait $\sum_{n}x_{i}\leq\sigma,\forall\, x\in P=\{x\geq 0,Ax=b\}$ ce qui revient à supposer P borné. Ajoutons une variable d'écart $(x_{n+1}\geq 0)$ telle que $\sum_{n+1}x_{i}=\sigma$ ou encore

$\sum_{n+1}\frac{x_{i}}{\sigma}=1$ Et posons $y_{i}=\frac{x_{i}}{\sigma}$ d'où $x_{i}=\sigma y_{i}\ i=1,\dots,n.$

L'objectif s'écrit : $c^{T}x=(\sigma c^{T}\ 0)y$

Le système Ax=b s'écrit :

$$[A\quad 0]\begin{bmatrix}x\\ x_{n+1}\end{bmatrix}=\sigma[A\quad 0]y=b$$

$$\mapsto[A\quad 0]y=\frac{b}{\sigma}$$

Première formulation :

Dans cette formulation (due essentiellement à Karmarkar).

L'équation$e_{n+1}^{T}y = 1$, permet d'écrire$[A \quad 0]y = \frac{b}{\sigma}e_{n+1}^{T}y$, ce qui donne

$\left[A - \frac{b}{\sigma}e_{n+1}^{T} \quad -\frac{b}{\sigma}\right]y = 0$ D'où le programme linéaire :

$$(1)\begin{cases} \text{Min } (\sigma c^{T}\ 0)y \\ By = \left[A - \dfrac{b}{\sigma}e_{n+1}^{T} \quad -\dfrac{b}{\sigma}\right]y = 0 \\ e_{n+1}^{T}y = 1\ , y \geq 0 \end{cases}$$

Pour obtenir une solution réalisable de (1) on peut soir résoudre le problème de faisabilité (By=0, $y \geq 0$), Soit utiliser la technique suivant connue sous le nom de Méthode de big M.

On ajoute une colonne d à la matrice B et donc une variable ($y_{n+2} \geq 0$) au vecteur y telles que le point $e_{n+2}/n+2$ soit solution du système $By + dy_{n+2} = 0$ et$e_{n+1}^{T}y + y_{n+2} = 1$

Il est facile de voir que d= (n+1) b/(σ-Ae_n).

Le problème (1) s'écrit finalement :(1.1)
$$\begin{cases} \text{Min } ((\sigma c^{T}c_{n+2})\begin{bmatrix} y \\ y_{n+2} \end{bmatrix} \\ [B \qquad (n+1)\,b/(\sigma - Ae_n)]\begin{bmatrix} y \\ y_{n+2} \end{bmatrix} \\ e_{n+1}^{T}y + y_{n+2} = 1\ , y \geq 0, y_{n+2} \geq 0 \end{cases}$$

c_{n+2} Étant le coût associe à $(y\ , y_{n+2})^{T}$ est une solution optimale de (1.1) avec y_{n+2} alors y est une solution optimale de (1). Cette possibilité peut être réalisée en imposant à la composante c_{n+2} d'être assez grande. Dans le cas où y_{n+2} reste loin de zéro, le problème (1) est non réalisable.

Deuxième formulation :

Pour rendre le système $[A \quad 0]y = \frac{b}{\sigma}$ homogène, on introduit une variable artificielle (notéey_{n+2}) en lui attribuant la valeur 1.

Les contraintes s'écrivent

$$[A \quad 0]y - \frac{b}{\sigma} y_{n+2} = 0$$
$$y_{n+2} = 1$$
$$e_{n+1}^T y = 1$$

En remplaçant les deux dernière contraintes par

$$e_{n+1}^T y - y_{n+2} = 0$$

$$e_{n+1}^T y + y_{n+2} = 2$$

Et en multipliant toutes les variables par (1/2) on obtient :

$$A'y' = \begin{bmatrix} A & 0 & -\frac{b}{\sigma} \\ e_n^T & 1 & -1 \end{bmatrix} y' = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \; y' \geq 0$$

$$e_{n+2}^T y' = 1$$

Avec $y_i = 2y'_i \; i = 1, \dots, n+2 \nRightarrow x_i = 2\sigma y'_i \; i = 1, \dots, n$

Comme précédemment, on ajoute une colonne d' à A' et une variable y'_{n+3} à y' telles que :

$e_{n+3}/(n+3)$ Soit solution de système $A'y' + d'y_{n+3} = 0$et

$$e_{n+2}^T y' + y'_{n+3} = 1$$

On obtient le programme linéaire suivant :

$$(1.2) \begin{cases} \text{Min } (2\sigma c^T, 0, 0, c_{n+3}) \begin{bmatrix} y' \\ y'_{n+3} \end{bmatrix} \\ \begin{bmatrix} A & 0 & -b/\sigma & -b/\sigma - Ae_n \\ e_n^T & 1 & -1 & -n \end{bmatrix} \begin{bmatrix} y' \\ y'_{n+3} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ e_{n+2}^T y' + y'_{n+3} = 1 \quad , y' \geq 0, y'_{n+3} \geq 0 \end{cases}$$

Le coût c_{n+3} associe à y'_{n+3} est assez grand pour que y'_{n+3} soit proche de zéro.

IV.6. Modification de l'algorithme de base :

On s'intéresse toujours à la résolution du problème linéaire général :

$$(p.l.g) \begin{cases} \text{Min } c^T x = Z^* \\ Ax = b \\ x \geq 0 \end{cases}$$

En supposant cette fois la valeur optimale Z^* inconnue, Karmarkar propose deux variantes :

1. La méthode « primal-dual » qui n'est autre que l'application de la phase 2 de l'algorithme de base à un problème de faisabilité obtenu par la combinaison de (p.l.g) par son dual.
2. La « sliding objective function method » qui consiste à localiser la valeur optimale dans un intervalle suffisamment petit.
3. dans un intervalle suffisamment petit.

IV.7. Méthode primal-dual :

Dans ce paragramme, on développe une manière de voir comment on peut transformer, un programme linéaire en un programme linéaire de Karmarkar (P.l.r), en utilisant les problèmes linéaires primal et dual (voir théorème des écarts complémentaires).

Rappelons que la même idée est utilisée dans le cas de l'application de l'algorithme de Khachyan.

Considérons le problème linéaire et son dual :

$$\begin{array}{lll} \text{Min } (c^T x) & \text{Dual de PL} & \text{Max} b^T y \\ A x \geq b & & A^T y \leq c \\ x \geq 0 & & y \geq 0 \end{array}$$

$$\text{Min } c^T x = \text{Max } b^T y \quad \text{d'aprés} \quad [3]$$

x S R de PL y S R du Dual

Dans une première étape dans le processus de reformulation, nous combinons primal et dual

Problème pour obtenir l'écart de la dualité.

$$\begin{array}{l} c^T x - b^T y = 0 \\ A x \geq b \\ A^T y \leq c \\ x \geq 0 \quad y \geq 0 \end{array}$$

Ajouter des variables d'écart pour se débarrasser des inégalités, cela donne :

$$
\begin{aligned}
c^T x - b^T y &= 0 \\
A x - v &= b \\
A^T y + u &= c \\
x \geq 0 \quad y \geq 0 \quad u \geq 0 \quad v \geq 0 &
\end{aligned}
\qquad (I)
$$

Maintenant, le problème principal est d'obtenir un point de départ possible. Cela peut être fait en introduisant une nouvelle variable λ. Soient x_0, y_0, u_0, v_0 des points strictement intérieurs dans l'orthant positif. Cela se traduit par :

$$
\begin{aligned}
&\text{Min } \lambda \\
&c^T x - b^T y + (-c^T x_0 - b^T y_0)\lambda = 0 \\
&A x - v + (b - A x_0 + v_0)\lambda = b \\
&A^T y + u + (c - A^T y_0) = c \\
&x \geq 0 \quad y \geq 0 \quad u \geq 0 \quad v \geq 0 \quad \lambda \geq 0
\end{aligned}
$$

Ce problème (dit Pb artificiel de Karmarkar) peut être écrit sous la forme en notation matricielle :

$$
\begin{aligned}
&\text{Min } \check{c}^T \check{x} \\
&\check{A}\check{x} = \check{b} \\
&\check{x} \geq 0
\end{aligned}
$$

Où $\quad \check{x} = [x^T \; y^T \; u^T \; v^T \; \lambda]^T$

$$\check{c} = [0_{2m+2n}^T \; 1]^T$$

$$\check{A} = \begin{bmatrix} c^T & -b^T & 0_n^T & 0_m^T & (c^T x_0 + b^T y_0 \\ A & 0_{m\times m} & 0_{m\times n} & -I_m & (b - A x_0 + v_0) \\ 0_{n\times n} & A^T & I_n & 0_{n\times m} & (c - A^T y_0) \end{bmatrix}$$

$$\check{b} = [0 \; b^T \; c^T]^T$$

Observons que le point suivant est un point réalisable de l'intérieur du domaine du problème :

$$\begin{bmatrix} x \\ y \\ u \\ v \\ \lambda \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ u_0 \\ v_0 \\ 1 \end{bmatrix}$$

En outre, la valeur minimale de la fonction objective du Pb artificiel de Karmarkar est nul si et seulement si le problème précédent admet une solution réalisable, c-à-d, il existe x, y, u et v vérifiant (I). Par conséquent, le problème artificiel de Karmarkar est équivalent au problème LP initial.

Chapitre V

Mise en œuvre et testes des résultats

I. Mise en œuvre :

Introduction :

Contrairement à la méthode du simplexe qui explore les sommets c.-à-d une itération consiste à un passage d'un sommet (point extrême) à un autre sommet du polyèdre qui définit le domaine des solutions réalisables. Khachiyan et la Karmarkar cherchent une solution optimale à un problème de PL en se déplaçant à travers l'intérieur de la région réalisable.

Khachiyan se rapproche de la solution optimale d'un problème de PL en créant une séquence d'ellipsoïdes qui converge vers la solution optimale et Karmarkar part d'une région (suffisamment grande) contenant l'ensemble des solutions, il s'agit de rétrécir ce domaine pour cibler la solution. Ainsi, pas après pas, le domaine réalisable se restreint jusqu'à saréduction à un voisinage arbitrairement petit de la solution.

Dans ce chapitre on va faire une étude comparative entre les trois méthodes (Simplexe, Khachiyan, Karmarkar) pour voire qui elle est la plus rapide entre eux.

1. Elément de base :

1.1. Programme linéaire :

Un Programme Linéaire (PL) est un problème d'optimisation consistant à maximiser (ou minimiser) une fonction objectif linéaire de variables soumises à un ensemble de contraintes exprimées sous forme d'équations ou d'inéquations linéaires.

1.2. Forme général :

La forme générale d'un programme linéaire est la suivante :

$\text{Max ou Min}(Z) = c_1x_1 + c_2x_2 + \cdots + c_nx_n = \sum_{j=1}^{n} c_j x_j$ (Fonction objectif)

Sous les contraintes :

$\sum_{j=1}^{n} a_{ij}x_j = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$ (Contraintes d'inégalité)

$$\sum_{j=1}^{n} a_{ij}x_j = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$$

$\sum_{j=1}^{n} a_{ij}x_j = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i$ (Contraintes d'égalité)

Ou i=1,...,m et j=1,...,n.

Notation :

- n nombre de variables $x=(x_1,....,x_n)^T$
- m nombre de contraintes ($m \leq n$)

- A = (a_{ij}):Matrice des contraintes (m × n, rang(A) = m).
- c = (c_1,...,c_n) : vecteur ligne des profits (ou gains).
- b = (b_1,...,b_m): vecteur colonne des seconds membres.

Exemple :

$Minz = 24x_1 + 9x_2 - 6x_3$

$$x_1 \quad + x_3 \leq 3$$

$$x_1 + x_2 - 2x_3 \leq 2$$

$$x_1, x_2, x_3 \geq 0$$

Avec c= (24, 9,-6) $b=(3,2)^T$ $A=\begin{pmatrix} 101 \\ 11\ -2 \end{pmatrix}$

2. Méthodes de résolution d'un PL :

Les problèmes typiques de programmation linéaire que nous avonsprésentés dans le paragraphe précédent se résolvent à l'aide des méthodes mathématiques particulières. Les méthodes que nous avons implémenter sont:

2.1. Méthode du simplexe :

Procédure :

Pas 0. Initialisation

Etablir la forme standard du programme linéaire (cas minimum)

Pas 1. Test d'optimalité

Exprimer la fonction économique à partir des variables hors base

- Si tous les coûts marginaux sont positifs ou nuls Alors STOP, la solution obtenue est la solution optimale

- Sinon

Pas 2. Choix de la variable entrante

- Choisir la variable hors base dont le coût marginal est négatif et le plus petit possible. Soit x_r la variable entrante.

Pas 3. Choix de la variable sortante

- La variable sortante est la première à s'annuler : c'est celle pour la quelle le b_i/a_{ir} est le plus petit avec $a_{ir} > 0$. Soit x_s la variable sortante.

Pas 4. Déterminer la nouvelle solution de base : opération du pivot

- Remplacer les a_{ij} par $a_{ij} - a_{jr}\ a_{si}/a_{sr}$

- Remplacer les b_j par $b_j - a_{jr}\ b_s/a_{sr}$ (Elimination de Gauss)

Retour au Pas 1.

Exemple :

$$Minz = 24x_1 + 9x_2 - 6x_3$$

$$x_1 + x_3 + e_1 = 3$$

$$x_1 + x_2 - 2x_3 + e_2 = 2$$

$$x_1, x_2, x_3, e_1, e_2 \geq 0$$

Variables de base	1	x_1	x_2	x_3	e_1	e_2
$-Z$	0	24	9	-6	0	0
e_1	3	1	0	1	1	0
e_2	2	1	1	-2	0	1
$-Z$	18	30	9	0	6	0
x_3	3	1	0	1	1	0
e_2	8	3	1	0	2	1

La solution optimale est $x^* = (0; 0; 3)$.

2.2. Méthode de Khachiyan :

Procédure :

Pas 0. Initialisation K=0, N=16n (n+1) L, t_0 0 n-vecteur, et soit $B_0 = n^2 2^{2L} I$.

Pas 1. Pour K=0 jusqu'à K faire

Si t_K est la solution de A x <b, alors stop

Sinon $t_{K+1} = t_K - \frac{1}{n+1}\frac{B_K a}{\sqrt{a^T B_K a}}$ et $B_{K+1} = \frac{n^2}{n^2-1}\left(B_K - \frac{2}{n+1}\frac{(B_K a)(B_K a)^T}{a^T B_K a}\right)$.

Pas 2. Après K itérations, l'ellipsoïde est trop petit pour que l'instance LSI soit réalisable

affiché « irréalisable» et arrêtée.

Fin.

Exemple :

K=0

$B_0 = \begin{bmatrix} 5 & 2 \\ 2 & 8 \end{bmatrix}$, $t_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Parmi les contraintes de LSI $x_1 - x_2 < -1$

$$B_0\, a=\begin{bmatrix}5 & 2\\ 2 & 8\end{bmatrix}\begin{bmatrix}1\\ -1\end{bmatrix}=\begin{bmatrix}3\\ -6\end{bmatrix}$$

Et $a^T B_0\, a = 9$ nous en déduisons que

$$t_1=\begin{bmatrix}\frac{-1}{3}\\ \frac{2}{3}\end{bmatrix} \qquad \text{et} \qquad B_1=\begin{bmatrix}\frac{52}{9} & \frac{40}{9}\\ \frac{40}{9} & \frac{64}{9}\end{bmatrix}$$

t ne vérifié pas la contrainte

K=1

$$B_1\, a=\begin{bmatrix}52/9 & 40/9\\ 40/9 & 64/9\end{bmatrix}\begin{bmatrix}1\\ -1\end{bmatrix}=\begin{bmatrix}12/9\\ -24/9\end{bmatrix}$$

Et $a^T B_0\, a = 4$ nous en déduisons que

$$t_2=\begin{bmatrix}\frac{-5}{9}\\ \frac{10}{9}\end{bmatrix}\text{et} \qquad B_2=\begin{bmatrix}\frac{592}{81} & \frac{1088}{162}\\ \frac{1088}{162} & \frac{640}{81}\end{bmatrix}$$

2.3. Méthode de Karmarkar :

Procédure :

Initialisation $x^0 = a^0 = (1/n)e_n k = 0$

Tant que $c^T x^k > \varepsilon$ **faire**

Pas 0
$$\begin{cases} D_k = \mathrm{diag}(x^k) \\ B_k = \begin{bmatrix} A_k \\ \cdots \\ e_n^T \end{bmatrix} \quad A_k = AD_k \end{cases}$$

Pas 1 $p_k = \left\{ I - B_k^T \left(B_k B_k^T\right)^{-1} B_k \right\} D_k c$

Pas 2 $d_k = \frac{p_k}{\|p_k\|}$

Pas 3 $y^k = a^0 - \alpha r d_k \quad .r = \frac{1}{\sqrt{n(n-1)}} \quad .0 < \alpha < 1$

Pas 4 $x^{k+1} = \frac{D_k x}{e_n^T D_k x} = T_k^{-1}(y) \quad .k = k+1$

Fin tant que

Fin d'algorithme.

$$\begin{cases} \text{Min } x_1 + x_2 \\ x_1 - x_2 = 0 \\ x_1 + x_2 + x_3 = 1 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

n=3, $c = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$, A=[1 -1 0], $x^0 = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}$, r = 0,40 , α=0.25.

Ce problème satisfait toutes les conditions de Karmarkar en particulier sa valeur optimal est nulle

Sa solution optimale exact est x^*=(0,0,1)

$$D_0 = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix} \quad AD_0 = [1 \ -1 \ 0] \times \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 \end{bmatrix}$$

$$B_0 = \begin{bmatrix} AD_0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$B_0^T(B_0B_0^T)^{-1}B_0 = \begin{bmatrix} 0.83 & -0,16 & 0.33 \\ -0,16 & 0,83 & 0,33 \\ 0.33 & 0,33 & 0.33 \end{bmatrix} \quad p_0 = \{I - B_0^T(B_0B_0^T)^{-1}B_0\}D_0c = \begin{bmatrix} 0,11 \\ 0,11 \\ -0,22 \end{bmatrix}$$

$$\|p_0\| = 0,27 \qquad d_0 = \frac{p_0}{\|p_0\|} = \begin{bmatrix} 0,41 \\ 0,41 \\ -0,82 \end{bmatrix}$$

$$y^0 = a^0 - \alpha r d_0 = \begin{bmatrix} 0.29 \\ 0.29 \\ 0.41 \end{bmatrix} \text{ Et } x^1 = \begin{bmatrix} 0.29 \\ 0.29 \\ 0.41 \end{bmatrix}$$

Pour ε =0,0001 on trouve la solution après 29 itérations

$$x^{29} = \begin{bmatrix} 0 \\ 0 \\ 0.99 \end{bmatrix}$$

3. Implémentation des algorithmes :

3.1. Environnement utilisé :

MATLAB « *matrix laboratory* » est un langage de programmation de quatrième génération émulé par un environnement de développement du même nom ; il est utilisé à des fins de calcul numérique. Développé par la société The MathWorks, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes, de créer des interfaces utilisateurs, et peut s'interfacer avec d'autres langages comme le C, C++, Java, et Fortran. Les utilisateurs de MATLAB (environ un million en 2004) sont de milieux très différents comme l'ingénierie, les sciences et l'économie dans un contexte aussi bien industriel que pour la recherche.

3.2. Codes sur MATLAB :

Entrée :

Matrice A

Vecteur c et b

3.2.1. Algorithme du simplexe :

```
%lire les entrées

n=input('n=');

m=input('m=');

b=input('b=');
disp('donner le vecteur C');
for j=1:m,
C(j)=input('');
end
disp('donner le vecteur D');
fori=1:n,
b(i)=input('');
end
```

```
disp('donner la matrice A');
fori=1:n,
for j=1:m,
A(i,j)=input('');
end
end
% Construire la matrice de systeme
M(1,1)=-d;
for j=1:m,
M(1,j+1)=C(j);
end
fori=1:n,
M(i+1,1)=b(i);
end
fori=1:n,
for j=1:m,
M(i+1,j+1)=A(i,j);
end
end
dim=m-n;
[n,m]=size(M);
fori=1:dim,
ind1(i)=0;
ind2(i)=0;
sol(i)=0;
end
z=optima(M,m); % Optima fonction teste l'optimalité de systeme
it=1;
while z~=0 && it<1000
x=calmin1(M,m) %  calmin1 fonction determine le minimum de la colonne
ind1(it)=x-1
y=calmin2(M,n,x) % calmin2 fonction determine le minimum de la ligne
```

```
ind2(it)=y
M=gauss(M,n,m,x,y) % gauss fonction faire l'elimination de gauss
z=optima(M,m);
it=it+1;  % nombre d'ittération
end
%  extraire  la solution
fori=1:dim-1,
for j=i:dim-1,
if ind2(i)==ind2(j+1)
ind2(i)=0;
end
end
end
fori=1:dim,
if ind1(i)~=0 && ind2(i)~=0
sol(ind1(i))=M((ind2(i)),1);
end
end
disp(it);
disp(sol);
%fin de programme
```

Les fonctions utilisées :

- function z=optima(M,m)

```
z=0;
fori=2:m,
if M(1,i)>0
    z=z+1;
end
end
end
```

- function x=calmin1(M,m)

```
s=-10000;
for j=2:m,

 if M(1,j)>s
    s=M(1,j);
    x=j;
end
end
```

- ```
  function y=calmin2(M,n,x)
  t=10000;
  fori=2:n,
  if M(i,x)>0
  if (M(i,1)/M(i,x))<t
        t=M(i,1)/M(i,x);
        y=i;
  end
  end
  end
  end
  ```

- ```
  function M=gauss(M,n,m,x,y)
  fori=1:n,
  for j=1:m,
  ifi==y
  N(i,j)=M(i,j)/M(y,x);
  else
  N(i,j)=M(i,j)-(M(y,j)*M(i,x))/M(y,x);
  end
  end
  end
  fori=1:n,
  for j=1:m,
  M(i,j)=N(i,j);
  ```

```
end
end
end
```

Sortie:

Vecteur de taille n représente la solution de problème primal

3.2.2. Algorithme de Khachiyan:

```
function [t,K]= LK1(A,b,Z)
[m,n]=size(A);
% calcule de N
a=produit(b,m);  % fonction calcule le produit des éléments de vecteur b
a1=produit1(A,n,m); %  fonction calcule
P=m*n+floor((log(a1)/log(2)))+m+floor((log(a)/log(2)));
L=m*n+floor((log(P)/log(2)))+n*floor((log(n)/log(2)));
N=16*n*(n+1)*L;
I=eye(n);
B=(n.^2*2.^(2*L)*I);
eps=2.^(-2*L);
t=zeros(n,1);
if Z==0
b=b+eps;
end
l=0;
K=0;
while K<N
C=A*t;
D=t;
z=testKach(C,b,m); % function test l'optimalité
if z~=0
t=t-(1/(n+1)*(B*A(z,:)'/round(sqrt(A(z,:)*B*A(z,:)'))));
B=(B-(2/(n+1)*((B*A(z,:)')*(B*A(z,:)')'/(A(z,:)*B*A(z,:)'))))*(n.^2/(n.^2-1));
l=tist(t,n); % fonction test la positivité de matice B
if l==1
```

```
t=D;

fprintf(' la solution est \n ');
disp(t);
break;
end
else
fprintf(' la solution est \n');
disp(t);
break
end
K=K+1;
end
end
% fin de programme
```

Les fonctions utilisées :

- function a=produit(c,m)

```
a=1;
fori=1:m;
   a=a*(1+abs(c(i)));
end
end
```

- function a1=produit1(A,n,m)

```
a1=1;
fori=1:m;
for j=1:n;
   a1=a1*(1+abs(A(i,j)));
end

end
end
```

- function z=testKach(C,b,m)

```
z=0;
fori=1:m;
if C(i)>=b(i)
    z=i;
end
end
end
```

- function l=tist(t,n)

```
l=0;
fori=1:n,
if t(i)==inf || t(i)==-inf
l=1;
end
end
end
```

Sortie :

Un vecteur de taille m+n représente la solution de problème primal et dual

3.2.3. Algorithme de Karmarkar :

```
epsi=0.0001; Alpha= 0.25;  %Entrées supplémentaire
z=0;
e1=ones(1,n);
A=A-b*e1;
C=c'-z*e1;
e=ones(1,n);
I=eye(n);
x0=(1/n)*e';
x=x0;
```

```
    r=1/(sqrt(n*(n-1)));
k=1;
    while (C*x>epsi&& k<100)
     D=diag(x)
      A=A*D
B=[A
     e]
      p=(I-B'*inv(B*B')*B)*D*C'
    d=p/norm(p)
    y=x0-Alpha*r*d
      x=(D*y)/(e*D*y)
       k=k+1
    end
    % fin de programme
```

Sortie :

Un vecteur de taille m+n représente la solution de problème primal et dual

II. Testes et Résultats :

Dans ce chapitre, on s'intéresse à l'implémentation de ces trois algorithmes et la comparaison numérique entre ces trois algorithmes.

Nous avons choisis le temps d'exécutions comme objective de comparaison.

Exemple 1 :

$$\begin{aligned} Min\ z &= 4x_1 + 3x_2 \\ x_1 + 2x_2 &\geq 5 \\ 2x_1 + 3x_2 &\geq 1 \\ x_1,\ x_2 &\geq 0 \end{aligned}$$

Exemple 2 :

$$\begin{aligned} Min\ z &= 4x_1 + 12x_2 - 4x_3 \\ x_1 + x_2 - x_3 &\geq 3 \\ 2x_2 - x_3 &\geq 2 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

Exemple 3 :

$$Min\ z = 24x_1 + 9x_2 - 6x_3$$
$$-x_1 - x_3 \geq -3$$
$$-x_1 - x_2 + 2x_3 \geq -2$$
$$x_1, x_2, x_3 \geq 0$$

Exemple 4 :

$$Min\ z = 120x_1 + 25x_2 + 40x_3 + 60x_4$$
$$3x_1 + x_2 - 3x_3 \geq 7$$
$$2x_1 - x_3 + x_4 \geq 3$$
$$x_1, x_2, x_3, x_4 \geq 0$$

Exemple 5 :

$$Min\ z = 1000x_1 + 1000x_2$$
$$x_1 + 2x_2 \geq 90$$
$$x_1 + 4x_2 \geq 120$$
$$6x_1 + 3x_2 \geq 180$$
$$x_1, x_2 \geq 0$$

Exemple 6 :

$$Min\ z = 2x_1 + 3x_2$$
$$-2x_1 - 3x_2 \geq -30$$
$$x_1 + 2x_2 \geq 10$$
$$x_1 - x_2 \geq 0$$
$$x_1, x_2 \geq 0$$

Exemple7 :

$$Min\ z = 2x_1 + 3x_2 + 4x_3 + 5x_4$$
$$x_1 - x_2 + x_3 - x_4 \geq 10$$
$$x_1 - 2x_2 + 3x_3 - 4x_4 \geq 6$$
$$3x_1 - 4x_2 - 5x_3 - 6x_4 \geq 15$$
$$x_1, x_2, x_3, x_4 \geq 0$$

Exemple 8 :

$$Min\ z = 5x_1 + 15x_2 + 30x_3 + 20x_4 + 50x_5$$
$$x_2 + x_3 \geq 30$$
$$x_1 + x_2 + x_5 \geq 40$$
$$2x_1 + x_3 + 3x_4 \geq 0$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Exemple 9 :

$$Min\ z = 20x_1 + 10x_2 + 20x_3 + 20x_4 + 10x_5$$
$$x_1 + x_2 + 2x_3 + x_4 + x_5 \geq 2$$
$$x_1 + 2x_2 + x_3 + 2x_4 + 2x_5 \geq 5$$
$$2x_1 + x_2 + x_3 + 2x_5 \geq 7$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Exemple 10 :

$$Min\ z = 4500x_1 + 4000x_2 + 3000x_3$$
$$x_1 + 2x_2 \geq 900$$
$$3x_1 + x_2 + x_3 \geq 1800$$
$$x_1 + 2x_2 + 4x_3 \geq 1400$$
$$x_1 + x_2 + x_3 \geq 450$$
$$x_1, x_2, x_3 \geq 0$$

TABLEAU I

Le Tableau I, donne les solutions fournis par la M. Simplexe, la M. de Khachiyan et la M. de Karmarkar pour les 10 exemples de P.L précédents

Méthode	**Simplexe**		**Khachiyan**		**Karmarkar**	
Solution	**Primal**	**Dual**	**Primal**	**Dual**	**Primal**	**Dual**
Exemple 1	$X_1=0$ $X_2=2.5$	$Y_1=1.5$ $Y_2=0$	$X_1=0.2532$ $X_2=2.4175$	$Y_1=1.5514$ $Y_2=0.0253$	$X_1=0$ $X_2=2.5$	$Y_1=1.5$ $Y_2=0$
Exemple 2	$X_1=2$ $X_2=1$ $X_3=0$	$Y_1=4$ $Y_2=4$	$X_1=1.9679$ $X_2=0.9861$ $X_3=0.0006$	$Y_1=3.8645$ $Y_2=3.8720$	$X_1=2$ $X_2=1$ $X_3=0$	$Y_1=4$ $Y_2=4$
Exemple 3	$X_1=0$ $X_2=0$ $X_3=3$	$Y_1=6$ $Y_2=0$	$X_1=0.0137$ $X_2=0.1139$ $X_3=2.8121$	$Y_1=6.8690$ $Y_2=0.2889$	$X_1=0$ $X_2=0$ $X_3=3$	$Y_1=6$ $Y_2=0$
Exemple 4	$X_1=2.5$ $X_2=1.5$ $X_3=0$ $X_4=0$	$Y_1=25$ $Y_2=22.5$	$X_1=1.3335$ $X_2=3.1419$ $X_3=0.0163$ $X_4=0.3249$	$Y_1=25.0324$ $Y_2=22.3187$	$X_1=1.5$ $X_2=2.5$ $X_3=0$ $X_4=0$	$Y_1=25$ $Y_2=22.5$
Exemple 5	$X_1=10$	$Y_1=333,33$	$X_1=10.0498$	$Y_1=332.865$	$X_1=10$	$Y_1=333.333$

	$X_2=40$	$Y_2=0$ $Y_3=111.11$	$X_2=39.9584$	$Y_2=0.1960$ $Y_3=111.115$	$X_2=40$	$Y_2=0$ $Y_3=111.111$
Exemple 6	$X_1=3.33$ $X_2=3.33$	$Y_1=0$ $Y_2=1.67$ $Y_3=0.33$	$X_1=4.2190$ $X_2=2.8509$	$Y_1=0.0164$ $Y_2=1.5230$ $Y_3=0.1431$	$X_1=3.3334$ $X_2=3.3333$	$Y_1=0$ $Y_2=1.6666$ $Y_3=0.3333$
Exemple 7	$X_1=10$ $X_2=0$ $X_3=0$ $X_4=0$	$Y_1=2$ $Y_2=0$ $Y_3=0$	$X_1=9.7922$ $X_2=0.0841$ $X_3=0.4379$ $X_4=0.1017$	$Y_1=1.7202$ $Y_2=0.2801$ $Y_3=0.0064$	$X_1=10$ $X_2=0$ $X_3=0$ $X_4=0$	$Y_1=2$ $Y_2=0$ $Y_3=0$
Exemple 8	$X_1=10$ $X_2=30$ $X_3=0$ $X_4=0$ $X_5=0$	$Y_1=10$ $Y_2=5$ $Y_3=0$	$X_1=10.0365$ $X_2=30.1726$ $X_3=0.1342$ $X_4=0.1644$ $X_5=0.0043$	$Y_1=10.3805$ $Y_2=4.5764$ $Y_3=0.1553$	$X_1=10$ $X_2=30$ $X_3=0$ $X_4=0$ $X_5=0$	$Y_1=10$ $Y_2=5$ $Y_3=0$
Exemple 9	$X_1=0$ $X_2=0$ $X_3=0$ $X_4=0$ $X_5=3.5$	$Y_1=0$ $Y_2=0$ $Y_3=5$	$X_1=0.0438$ $X_2=0.2192$ $X_3=0.0471$ $X_4=0.0034$ $X_5=3.3478$	$Y_1=0.3389$ $Y_2=0.5561$ $Y_3=4.2592$	$X_1=0$ $X_2=0$ $X_3=0$ $X_4=0$ $X_5=3.5$	$Y_1=0$ $Y_2=0$ $Y_3=5$

Exemple 10	X_1=490 X_2=205 X_3=125	Y_1=1000 Y_2=1000 Y_3=500 Y_4=0	X_1=489.9995 X_2=204.9976 X_3=125.0095	Y_1=999.8619 Y_2=999.9951 Y_3=499.9951 Y_4=0.1704	X_1=489.9998 X_2=204.9999 X_3=124.9999	Y_1=999.9996 Y_2=999.9997 Y_3=499.9998 Y_4=0

On remarque que les résultats obtenue par la méthode du simplexe et Karmarkar sont exact par rapport les résultats obtenue par Khachiyan qui sont approchées.

TABLEAU II

Le tableau II donne le nombre d'itération et le temps d'exécution(Seconde) de chaque méthode

Méthode	Simplexe		Khachiyan		Karmarkar	
	Nbr° d'itération	Temps D'exécution	Nbr° d'itération	Temps D'exécution	Nbr° d'itération	Temps D'exécution
Exemple 1	1	5.9	1242	0.14	17	0.0012
Exemple 2	2	5.6	2729	0.31	16	0.0015
Exemple 3	1	3.48	2702	0.30	15	0.061
Exemple 4	1	4.72	5155	0.63	15	0,011
Exemple 5	2	6.9	2728	0.34	14	0.010
Exemple 6	2	5.23	2709	0.32	10	0.001
Exemple 7	1	3.82	9158	1.13	15	0.065
Exemple 8	2	4.26	15783	1.97	15	0.03
Exemple 9	1	4.9	15787	1.973	15	0.02
Exemple 10	1	4.34	9326	1.11	14	0.30

D'après le deuxième tableau on remarque que Le nombre d'itération du simplexe est meilleur que celui de Khachiyan et de Karmarkar par contre comme le temps d'exécution est notre objective de comparaison on voie que la méthode de Karmarkar est la meilleur (voir les résultats colorée en bleu dans le tableau).

Conclusion :

Notre objectif est de choisir la meilleure méthode pour résoudre un problème en peu de temps.

Nous pouvons dire qu'à travers nos expérimentations numériques relatives à des programmes linéaires de faible taille, l'algorithme de Karmarkar est plus rapide que la méthode de Khachiyan et du simplexe.

CONCLUSION GENERALE

En effet, l'algorithme du simplexe réputé très efficace en pratique pour la résolution d'un programme linéaire de m contraintes et n variable. Son temps d'exécution est dans la plupart des cas un plynôme, alors que sa complexité théorique est d'ordre 2^n -1 itérations.

La méthode de Khachiyan a fait un grand succès théorique, malheureusement cette algorithme est beaucoup moins efficace en pratique que l'algorithme de simplexe pour résoudre la plupart des programmes linéaires à cause des formules itératives complexes.

Ce qui montre en partie que l'efficacité pratique d'un algorithme ne découle pas nécessairement de sa polynomialité théorique.

Le succès théorique de la méthode de Karmarkar est indiscutable car grâce à cet algorithme on peut résoudre des problèmes linéaires et non linéaires en temps polynômial
C'est alors que cette méthode devienne compétitive face à l'algorithme du simplexe.
Pour terminer, nous pouvons affirmer que la méthode du simplexe reste la méthode de premier choix, pour la résolution des programmes linéaires à cause de sa mise en œuvre facile et simple et ses avantages calculatoires.

Bibliographie.

[1]A.KERAGHAL. Etude Adaptative et Comparative des Principales Variantes dans L'algorithme de Karmarkar. Thèse de Doctorat Mathématique appliqué. L'université de Joseph Fourier-Grenoble I. 4 juillet 1989.

[2]A.V.FIACOO Et G.P.MCCORMICK Nonlinear Programming, Séquentiel Unconstrained Minimization Technique.wily.Newyork, 1968 Remprimé par SIAM publication 1990.

[3] EDWIN K.P. CHONG ET STANISLAWH.ZAK. An Introduction to Optimization. Second Edition John Wiley & sons Inc. Canada 2001pp.

[4] G.B DANTZIG. Linear Programming and Extensions. Princeton University press

Princeton N.J 1963.

[5]G.B DANTZIG. Linear Programming. Department of management science and engineering. Stanford University. Stanford California 94 305-4023.

[6]G.DE GHELLINCK ET J-PH VIAL. An Extension of Karmarkar's Algorithm for Solving Linear Homogeneous Equation on the Simplexe. Mathematical Programming 39(1987) 79-92.
[7] Howard KARLOFF. Linear Programming. Bir Kariser Boston Springer 2009.pp.149.

[8] I.S.DUFF, J NOCEDAL ET J.K REID. The Use of Linear Programming for the Solution of Spares sets of Nonlinear Equation. Siam J.SC. Stat. Comput. Vol.8.N°2, March 1987.

[9]J-F-SHIED Programmation Linéaire Graphes ET R.O – Tellecom Nancy 2A

www.iecl.univ-lorain.Fr/Jean Francois.Shied /Enseignement /PL 1.Pdf.

[10]J-F-SHIED Dualité en Programmation Linéaire Graphes et R.O – Tellecom Nancy 2A

www.iecl.univ-lorain.Fr/Jean Francois.Shied /Enseignement /PL 4.Pdf.

[11] K. R. FRISCH. The logarithmic Potential Method of Convex Programming, Technical Report University Institute of economies, Oslo. Norway 1955.

[12] L. G. KHACHIYAN, A polynomial Algorithm in Linear programming, Soviet Mathematics Doklady, 20 (1979), pp. 191-194.

[13]M.PADBEGR. A different Convergence Proof of the Projective Method for Linear Programming. Newyork University, February 1985.

[14] N.KARMARKAR.A new Polynomial Time Algorithm for Linear Programming, Combinatorica 4(1984) pp.373-395.

[15]N.KARMARKAR, M.G.C.RESENDE, K.GRAMAKRISHNAN. An Interior Point Algorithm for Zero-One Integer Programming. August 1988.

[16] NZ. SHOR. Utilization of the Operation of Space Dilatation in the Minimization of Convex Functions.Kibernetika 1(1970) pp 6-12 Traduction anglaise: cybernetics 6pp 7-15.

[17] V. Klee Et G.J. Minty. How good is the Simplexe Algorithm? In inequalities,

O. Shisha Ed.Acadimie Press New York 1972.pp.155-157.

Annexe

Manuel d'utilisation

Annexe
Manuel d'utilisation :

Objectif :

Ce logiciel permet de résoudre des problèmes linéaire (I) avec des méthodes polynomiales (Méthodes Ellipsoïdales 'Khachiyan' et Projectives 'Karmarkar').

$$\text{Min}(w) = c^T x$$

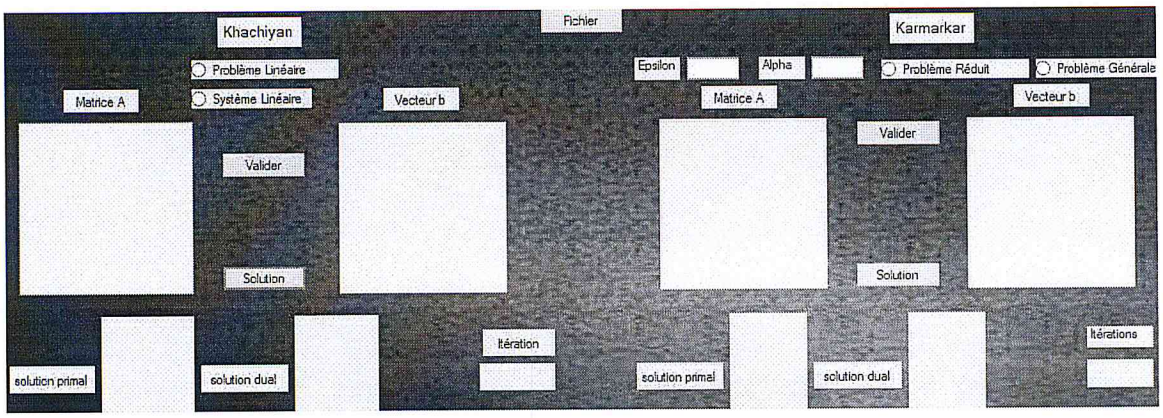
$$Ax \geq b \qquad (I)$$

$$x \geq 0$$

Le logiciel se compose de deux interfaces comme illustrés la figure suivante :

Interface 1 :

Interface 2 :



Exécution :

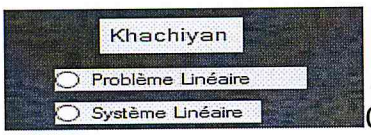
1. Créer un document texte.

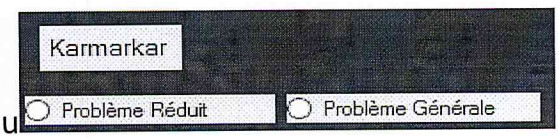
Contient les éléments suivants par ordre :

- Nombre de ligne (m).
- Nombre de colonne (n).
- pour le problème linéaire et générale { Les éléments de la matrice (A) ; Les éléments de vecteur (c) ; Les éléments de vecteur (b)
- pour le système linéaire { Les éléments de la matrice (A) ; Les éléments de vecteur (b)
- pour le problème réduit { Les éléments de la matrice (A) ; Les éléments de vecteur (c)

2. Cliquer sur le bouton commencer. 

3. Choisir la méthode et le critère souhaités.



Ou


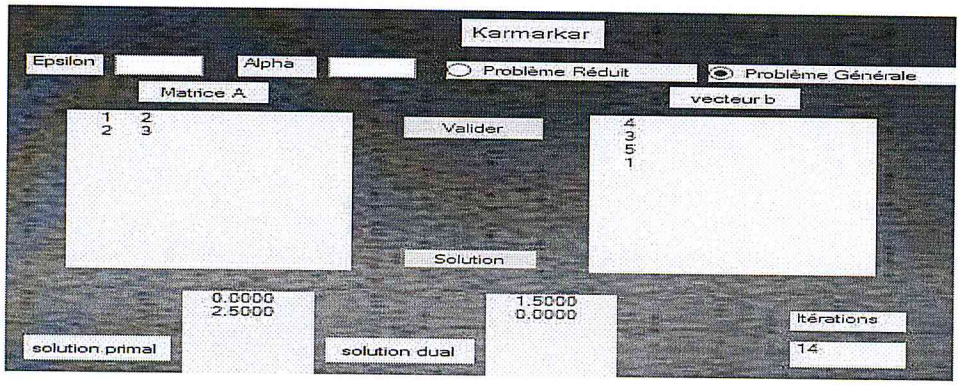
4. Cliquer sur le bouton fichier et choisir un exemple et puis sur valider.

Finalement on obtient la solution.

Traitement d'un exemple :

Exemple :

$$Min\ z = 4x_1 + 3x_2$$
$$x_1 + 2x_2 \geq 5$$
$$2x_1 + 3x_2 \geq 1$$
$$x_1,\ x_2 \geq 0$$



Le même exemple est traité par Khachiyan