REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI DE TIZI OUZOU
FACULTE DES SCIENCES
DEPARTEMENT DES MATHEMATIQUES

Memoire de Master

Présenté pour l'obtention du diplôme de :

MASTER EN MATHEMATIQUES

Option

Mathématique appliquer à la gestion

Thème

Méthode branch and bound en optimisation non linéaire non convexe

Présenté par :

BENKACIMI Kahina

GUESSAB Rym

Devant le jury :

M^r. M. CHEBBAH	Président
M^r. M. OUANES	Rapporteur
M^r. M. GOUBI	Examinatreur

TIZI-OUZOU, OCTOBRE 2019

Remerciements

Nous tenons à remercier :

Le bon Dieu de nous avoir donné la patience et la volonté pour accomplir ce travail.

Nos remerciements s'adressent également à :

Notre promoteur M^r^ OUANES pour ses conseils, ses orientations pour nous avoir transmis les renseignements nécessaires à la réalisation de ce travail, et son aide durant l'encadrement.

Nous remercions également :

Les membres de jury, pour l'honneur qu'ils nous font en acceptant de juger de lire et d'évaluer ce mémoire.

Nous tenons également à remercier :

Tous les enseignants de notre département qui nous ont accompagnés au cours de notre formation et à tout le personnel de la bibliothèque de l'université.

Enfin, nous remercions toute personne ayant contribué de près ou de loin à la réalisation de ce travail.

DEDICACES KAHINA

Je dédie ce modeste travail

A

Mes chers parents

Mon frère

Mes sœurs

Et à

Toute ma famille

Et

Mes ami(e)s

DEDICACES RYM

Je dédie ce modeste travail

A

Mes chers parents

Mon cher mari

Mes chers beaux parents

Mes frères, mon beau frère

Mes soeurs, mes belles soeurs

Et à

Toute ma famille

Et

Mes ami(e)s

Table des matières

Introduction générale

Les mathématiques appliquèes sont une branche des mathématiques qui s'intéressent à l'application du savoir mathématique aux autres domaines. L'analyse numèrique, les mathématiques de l'ingénierie, l'optimisation linéaire, la programmation dynamique, l'optimisation et la recherche opérationnelle, la thèorie de l'information, les probabilités et les statistiques et jusqu'à un certain point, la combinatoire et la géométrie fnie ..., ainsi qu'une bonne partie de l'informatique sont autant de domaines d'application des mathématiques. La classifcation logique des mathématiques appliquées repose davantage sur la sociologie des professionnels qui se servent des mathématiques que sur la question d'en déterminer la nature exacte. Habituellement, les méthodes mathématiques sont appliquèes au domaine d'un problème particulier à l'aide d'un modèle mathématique.
L'optimisation est une branche des mathématiques appliquées, cherchant à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à déterminer le meilleur élément d'un ensemble, au sens d'un critère quantitatif donné. Ce mot vient du latin optimum qui signife le meilleur.
L'optimisation joue un rôle important en recherche opérationnelle (donc en économie et microéconomie), dans les mathématiques appliquées (fondamentales pour l'industrie et l'ingénierie), en analyse et en analyse numérique, en statistique pour l'estimation du maximum de vraisemblance d'une distribution, pour la recherche de stratégies dans le cadre de la théorie des jeux, ou encore en théorie du contrôle et de la commande.

Pendant des décennies, les chercheurs ont travaillé sur les méthodes déterministes d'optimisation locale, et le terme local s'est perdu... Ainsi, dans le cas continu, quand on parle de méthode d'optimisation, on a plutôt tendance à penser optimisation locale. L'existence de minima locaux impose l'utilisation de méthodes d'exploration efficaces, pour éviter de

rester bloqué aux alentours de ces minima. Plusieurs méthodes ont été proposées et ont souvent été inspirées par des phénomènes naturels :

- Les algorithmes génétiques font référence à la sélection, la mutation et le croisement des individus au sein d'une même espèce biologique ;
- Le recuit simulé est basé sur des principes d'équilibre énergétique, lors de la cristallisation des métaux ;
- La méthode tabou introduit la notion d'histoire (mémoire) dans la stratégie d'exploration des solutions.

Depuis ces vingt dernières années, les ordinateurs ont vu leur puissance intrinsèque crôitre de manière quasi-exponentielle année après année. D'es lors, il est devenu possible d'effectuer un nombre très important de calculs. Ceux-ci sont n´ecessaires pour la d´etermination de solutions meilleures que celles produites avec des algorithmes d´eterministes locaux. Ainsi l'optimisation dite globale par opposition à locale a vu le jour.

L'une des trois grandes idées directrices en optimisation globale est le d´eveloppement des heuristiques (i.e. des règles) afin de sortir des optima locaux.

Le second principe est issu des statistiques. Ces méthodes dites stochastiques sont basées sur des générations aléatoires de points à l'intérieur d'un domaine de recherche. Toutes ces techniques différent par la manière dont elles procèdent pour l'élaboration de la génération suivante de points ; les méthodes de recuit simulé prennent pour heuristique un principe de thermodynamique, les algorithmes génétiques, des lois de la biologie cellulaire, etc.

Le troisième principe de méthode est lui dit exact, ou global et déterministe, car l'algorithme ne s'arrêtera que lorsque la preuve numérique complète de la globalité de la solution sera effectuée. Ces algorithmes sont tous basés sur des techniques de d´ecompositions successives du domaine initial en pavés de plus en plus petits, de telle sorte que l'on puisse en éliminer certains au fil des itérations, en montrant que la solution globale ne peut être dans ces pavés exclus.

Ce type d'algorithme dit de séparation-évaluation et qui est plus connu sous le nom anglais de Branch and Bound (BδB) a d'abord été introduit pour résoudre des programmes linéaires en nombres entiers. Le principe de résolution est simple. Tout d'abord, on calcule les solutions exactes du programme linéaire relaxé au cas continu ; si la solution est par hasard entière c'est gagné, sinon une variable est choisie et l'on sépare le problème initial

en sous-problèmes dans lesquels cette variable de séparation sera fixée à l'une de ses valeurs possibles. On étudie ensuite un par un tous ces sous-problèmes en les relaxant au cas continu, et en utilisant par exemple, l'algorithme du simplexe pour calculer des bornes, et ainsi de suite jusqu'à ce que l'arborescence de recherche soit close. Dans le cas continu, ce type d'algorithme de BδB sera utilisé lorsque les fonctions considérées ne posséderont plus des hypothèses simplificatrices telles que la linéarité ou la convexité.
Une façon d'étendre la théorie sur l'optimisation dans le cas convexe au cas non-convexe est de s'intéresser aux fonctions qui peuvent se décomposer comme différence de deux fonctions convexes et plus récemment comme différence de deux fonctions monotones (tout comme dans notre cas étudié). Certes, de trè nombreux résultats théoriques ont été trouvés, et une très grande variété de fonctions peut se décomposer ainsi mais cependant, les théorèmes ne sont pas constructifs dans le sens où l'on ne dispose pas de techniques pour réaliser effectivement ces décompositions (qui de plus, ne sont pas généralement uniques).

Le chapitre 1 est dédié à la présentation des notions essentielles à la bonne compréhension de ce mémoire. Nous introduisons tout d'abord des notions propres à l'optimisation non linéaire, ainsi les différents ensembles convexes et leurs fonctions.

Le deuxième chapitre est consacré à l'optimisation non linéaire convexe dont on cite plusieurs méthodes de calcule on déffinera par la suite la méthode branch and bound.

Nous nous intéressons dans le chapitre 3 à exposer les résultats trouvés on utilisant l'algorithme branch and bound, après avoir donné cette algorithme et expliquer toutes les étapes de ce dernier.

Chapitre 1

Généralités

1.1 Quelques définitions

Définition 1.1. [6] soient les vecteurs $x_1, x_2, x_3, ..., x_m$ dans un espace $\mathbb{R}^n$ et soient $\lambda_i \geq 0$, $i = 1, ..., m$ *avec* $\sum_{i=1}^{m} \lambda_i = 1, \lambda_1 x_1 + \lambda_2 x_2 + ... + \lambda_m x_m$ est dite combinaison convexe de ces points. par exemple, l'ensemble des combinaisons convexes de deux points est le segment joignant ces deux points et l'ensemble des combinaisons convexes de trois points est un triangle.[6]

Définition 1.2. Soit $C \subseteq \mathbb{R}^n$ un ensemble. On appelle enveloppe convexe de C et on note conv(C) l'ensemble convexe le plus petit contenant C. En dimension finie, c'est aussi l'ensemble des combinaisons convexes d'éléments de C [1] :

$$conv(C) = \{x \in \mathbb{R}^n \mid x = \sum_{i=1}^{m} \lambda_i x_i \ ou \ x_i \in C, \ m \in \mathbb{N} \ et \ \sum_{i=1}^{m} \lambda_i = 1, \ \lambda_i \geq 0\}.$$

Définition 1.3. Soit C une partie de $\mathbb{R}^n$. Un point x est appellé point intérieur à C s'il éxiste $r > 0$ tel que $B_r(x) \subset C$. L'ensemble des points intérieurs à C est appellé intérieur de C, on le note int(C).[6]

Définition 1.4. Un point $x \in C$ est appellé point extrême ou bien un sommet de C, s'ils n'existent pas deux points distincts [6] $y, z \in C$ tel que $x = \lambda y + (1 - \lambda)z$ pour tout $\lambda \in]0, 1[$.

Définition 1.5. Soit la fonction $f : C \subset \mathbb{R}^n \to \mathbb{R}^n, x_0 \in C$
$f$est continue en x_0 si [6]

$$\forall \epsilon > 0, \exists \delta \text{ tel que } \|x - x_0\| < \delta \Rightarrow |f(x) - f(x_0)| < \epsilon$$

Définition 1.6. On définit la dérivée partielle de $f : \mathbb{R}^n \to \mathbb{R}^n$ par rapport la limite [6]

$$\lim_{h \mapsto 0} \frac{f(x_1, ..., x_i + h, ..., x_n) - f(x_1, ..., x_n)}{h}$$

Quand elle existe, on la note

$$\frac{\partial f(x)}{\partial x_i}$$

Définition 1.7. On définit le gradient de $f : \mathbb{R}^n \to \mathbb{R}$ [6] :par

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix}$$

Définition 1.8. On appelle épigraphe d'une fonction $f : C \subset \mathbb{R}^n \to \mathbb{R}$ l'ensemble [6]

$$epi(f) : \{(x, r) \in C \times \mathbb{R} \, tel \, que \, f(x) \leq r\}$$

avec C ensemble convexe.

Définition 1.9. L'envloppe convexe d'une fonction est la plus grande fonction convexe de $f(x)$ qui soit inférieure ou égale à f sur l'ensemble de définition.

Définition 1.10. Une fonction f est dite affine sur C si $f(x)$ est finie, convexe et concave. Une fonction affine sur $\mathbb{R}^n$ a la forme [6]

$$f(x) = < a, x > + \alpha \text{ avec } a \in \mathbb{R}^n, \alpha \in \mathbb{R}.$$

Définition 1.11. Soit $x_1, x_2, ..., x_{m+1}$ un nombre fini des points dans $\mathbb{R}^n$, si $x_2 - x_1, x_3 - x_1, ..., x_{m+1} - x_1$ sont linéairement indépendants alors l'enveloppe convexe de $x_1, x_2, ..., x_{m+1}$,c'est à dire l'ensemble de toutes les combinaisons convexes de $x_1, x_2, ..., x_{m+1}$ sont appelées un m-simplexe dans $\mathbb{R}^n$ avec les sommets $x_1, x_2, ..., x_{m+1}$,.Un 0-simplexe est un point, un 1-simplexe est un segment, un 2-simplexe est un triangle.[6]

Définition 1.12. Un ensemble convexe $P \subset \mathbb{R}^n$ est un polyèdre s'il est l'intersection d'une famille finie ou infinie de demi-espase fermé. En d'autres termes, un polyèdre est un ensemble de solution d'un systéme fini d'inégalité linéaire de la forme [6]

$$< a^i, x > \leq b, i = 1, ..., m.$$

où sous forme matricielle

$$Ax \leq b.$$

où A est une matrice $m \times n, b \in \mathbb{R}^m$, m et n deux entiers positifs.

1.2 Qu'est-ce qu'un probléme d'optimisation

L'optimisation vise à résoudre des problèmes où l'on chercher à déterminer parmi un grand nombre des solutions candidates celle qui donne le meillieur rendement. Plus précisément, on cherche à trouver une solution satisfaisant un ensemble de contraintes qui minimise ou maximise une fonction donnée. L'application de l'optimisation est en expansion croissante et se retrouve dans plusieurs domaines. Les problémes considérés dans ce document s'écrivent sous la forme standard

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

$$s.c. \quad x \in S$$

où $x = (x_1, x_2, ..., x_n)^T$ est un vecteur de $\mathbb{R}^n$, $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ est la fonction que l'on désire minimiser (appelée fonction objectif), $S \subseteq \mathbb{R}^n$ est l'ensemble dans lequel les points doivent appartenir, et $s.c$ est l'abréviation de sous la ou les contraintes.
La formulation (1.1) signifie que l'on cherche à trouver une solution du domaine réalisable $x^* \in S$ dont la valeur de la fonction objectif est la plus petite.

Définition 1.13. Une solution $x^* \in S$ est un minimum global de la fonction f sur le domaine S si

$$f(x^*) \leq f(x) \quad \forall x \in S$$

La valeur optimale est $f(x^*)$.

Notez que le minimum global n'est pas nécessairement unique, mais la valeur optimale l'est. par exemple, le probléme d'optimisation suivant

$$\min_{x\in\mathbb{R}} \sin x$$

possède une infinité de minima globaux, soient $\{\frac{3\pi}{2} + 2k\pi : k \in \mathbb{Z}\}$ mais une seule valeur optimale : -1.

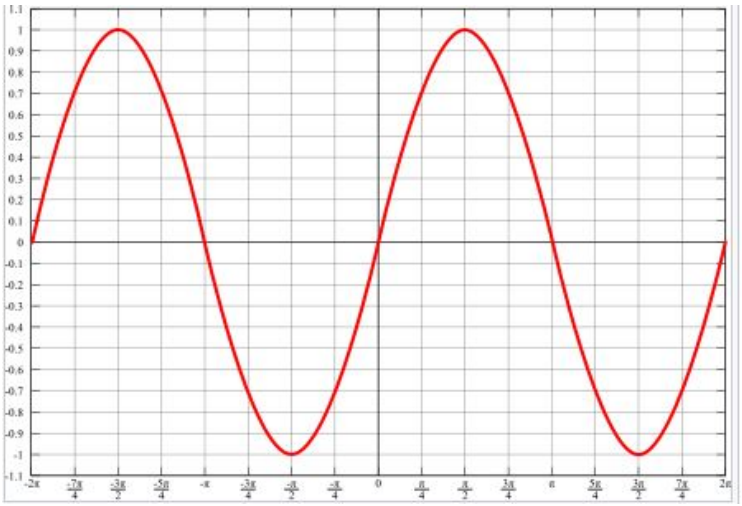

FIG. 1.1: Représentation graphique de la fonction sinus.

Définissons maintenant la notion d'optimalité dans un voisinage restreint. Pour introduire cette idée, on va définir $B_\varepsilon(x^*)$ comme étant l'ensemble des points de $\mathbb{R}^n$ dont la distance à x^* est inférieure à ε, un scalaire positif donné. Cet ensemble est communément appleé une boule de rayon ε centrée en x^*, et s'écrit formellement :

$$B_\varepsilon(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| < \varepsilon\}.$$

Définition 1.14. Une solution $x^* \in S$ est un minimum local de la fonction f sur le domaine S si

$$f(x^*) \leq f(x) \quad \forall x \in S \cap B_\varepsilon(x^*).$$

Les maxima sont définis de façon similaire, il suffit de remplacer les inégalités ($\leq$) aux définitions 1.15 et 1.16 par ($\geq$). La figure illustre le cas d'une fonction d'une seule variable possédant trois minima locaux, dont un minimum global. [3]

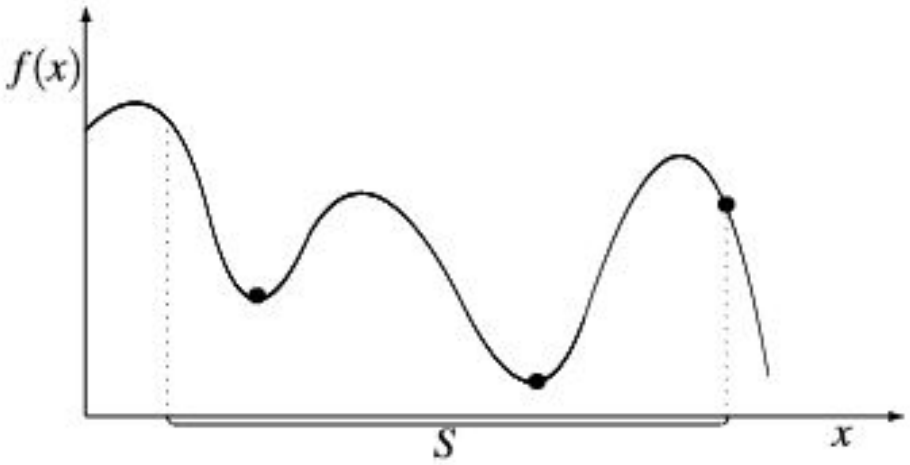

FIG. 1.2: Trois minima locaux, dont un global.

On peut facilement démontrer que les problémes (avec ou sans contraintes) $\min_x f(x)$ et $\max_x -f(x)$ sont équivalents dans le sens où ils ont même ensemble de solution est :

$$\min_x f(x) = -\max_x -f(x) \quad \textit{ou encore} \quad \max_x f(x) = -\min_x -f(x)[1]$$

- L'optimisation globale est une branche des mathématiques, le damaine de sa recherche est considéré beaucoup plus riche ces derniéres années grace aux ordinateurs développés par la nouvelle technologie qui peuvent résoudre des problémes d'optimisation de taille plus grande.
- Les méthodes d'optimisation globae s'intéressent à la recherche d'un optimum (maximuou minimum) globale, non à la recherche d'un optimum local que les méthodes classiques permettent de le trouver dans un voisinage réalisable, ce qui fait la différence entre les deux méthodes.

1.3 convexité

La convexité est à la base une propriété géométrique, assez intuitive d'alleurs, qui permet de caractériser certains objets. On voit assez bien ce qu'est un objet convexe dans un espace à deux ou trois dimensions, nous allons maintenant montrer comment cette propriété peut aussi s'appliquer aux fonctions de $\mathbb{R}^n$ dans $\mathbb{R}$.

1.3.1 Ensembles convexes

Définition 1.15. *Un ensemble $C \subset \mathbb{R}^n$ est dit convexe si pour tout couple $(x, y) \in C^2$ et $\lambda \in [0, 1]$, on a :*

$$\lambda x + (1 - \lambda)y \in C.$$

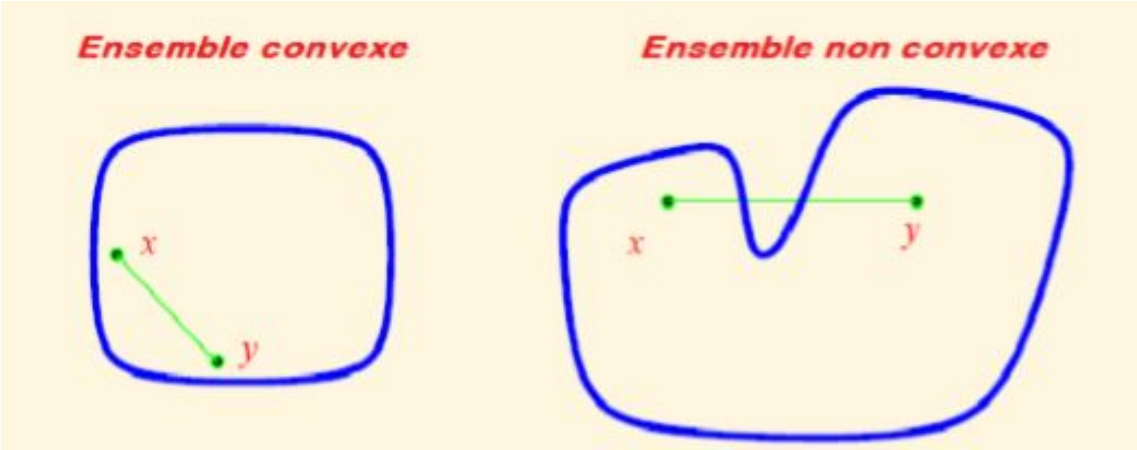

FIG. 1.3: Ensemble convexe et non convexe.

Géométriquement, cette notion s'interpréte comme suite :
"Pour tout segment reliant deux points quelconques x et y de C, le segment $[x, y]$ doit être aussi inclus dans C."
Elle se généralise de la façon suivante : on dira qu'un vecteur y est une combinaison convexe des points $\{x^1, ..., x^p\}$ si on a

$$y = \sum_{i=1}^{p} \lambda_i x^i$$

avec $\lambda_i \geq 0$ *et* $\sum_{i=1}^{p} \lambda_i = 1$
On peut citer quelques cas particuliers : $\mathbb{R}^n$ *tout entier est un ensemble convexe. de même qu'un singleton* $\{a\}$.

1.3.2 proprités des ensembles convexes

Propriété 1.1. Soit une famille $\{K_i\}_{i=1\ldots p}$ d'ensemble convexes et $S = \cap_p^{i=1} K_i$. Alors S est convexe.

1.3.3 Fonctions convexes

Définition 1.16. On dit qu'une fonction $f : C \to \mathbb{R}$, définie sur un ensemble convexe C, est convexe si elle vérifie

$$\forall (x, y) \in C^2, \forall \lambda \in [0, 1], f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y). \tag{1.2}$$

On dira que f est <u>strictement</u> convexe si

$$\forall (x, y) \in C^2, \forall \lambda \in [0, 1], f(\lambda x + (1 - \lambda)y) < \lambda f(x) + (1 - \lambda) f(y).$$

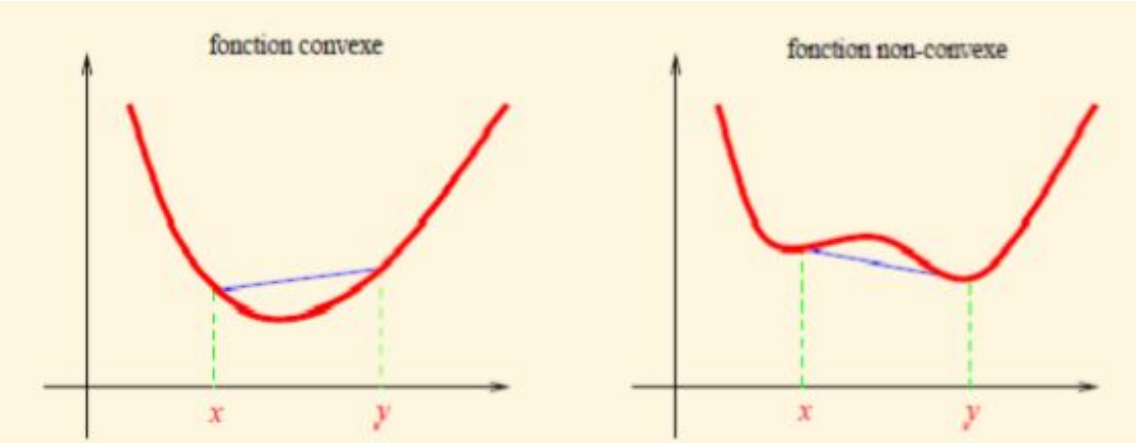

FIG. 1.4: Exemple de fonction convexe et non convexe.

Lorsque $n = 1$ cette définition s'interpréte bien géometriquement : le graphe de la fonction est toujours en dessous de segment reliant les points $(x, f(x))$ et $(y, f(y))$.

1.3.4 propriétés des fonctions convexes

Propriété 1.2. Soit une fonction réelle définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors f est convexe si seulement si son épigraphe

$$epi(f) = \{(x, r) : x \in C, r \geq f(x)\} \subset \mathbb{R}^n \times \mathbb{R}$$

est un ensemble convexe.

Propriété 1.3. soit f une fonction réelle définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors f est convexe si seulement si :

$$f(\sum_{k=1}^{n} \lambda_i x_i) \leq \sum_{k=1}^{n} \lambda_i f(x_i) \tag{1.3}$$

ou $x_i \in C, i = 1, 2, ..., n, \lambda_i \geq 0, \sum_{i=1}^{n} \lambda_i = 1$

Propriété 1.4. Soit f une fonction réelle de classe C^1, définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors f est convexe, si et seulement si :

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

Propriété 1.5. Soit f une fonction réelle de classe C^2, définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors f est convexe, si et seulement si [9],[10] :

$$(y - x)^T H(x)(y - x) \geq 0, \forall x, y \in C. \tag{1.5}$$

où $H(x) = \nabla^2 f(x)$.

1.3.5 Enveloppe convexe

Étant donnée un sous-ensemble A de $\mathbb{R}^n$, l'espace $\mathbb{R}^n$ est un convexe contenant A. De plus, l'intersection de convexe contenant A étant convexe, on peut poser la définition suivante.

Définition 1.17. L'enveloppe convexe de $A \subset \mathbb{R}^n$ est le plus petit convexe (au sens de l'inclusion) qui contient A.
Elle est notée $conv(A)$. [2]

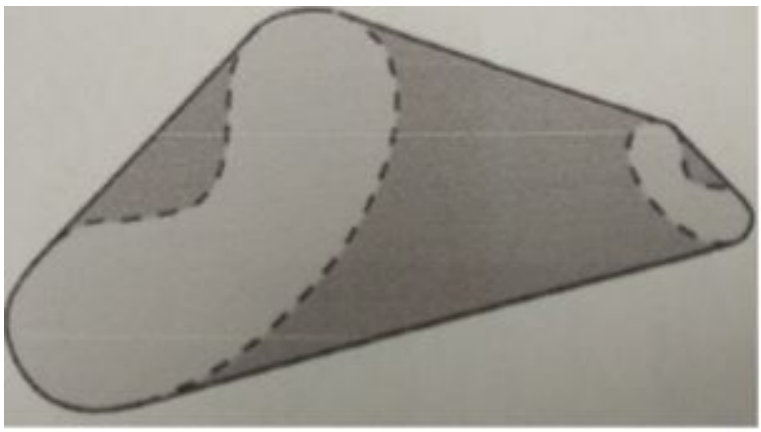

FIG. 1.5: Enveloppe convexe (en clair et en sombre)d'un sous-ensemble (en clair)

Chapitre 2

Optimisation non linéaire convexe

2.1 Introduction

Dans ce chapitre, nous présentons les notions de base et quelques notions sur l'optimisation non linéaire convexe qui seront utiles toute on long de ce mémoire.
Nous donnons quelques méthode numérique pour la minimisation d'une fonction différentiable dont on va cité les avantages et les inconvénients, et aprés on vas parlé sur l'optimisation non linéaire avec contrainte, nous terminons par optimisation quadratique convexe et la notion du dual.

2.2 Position du probléme et définition

soit f une fonction non linéaire, définie de $\mathbb{R}^n$ dans $\mathbb{R}$ et de classe C^1. Le probléme de programmation non linéaire consiste à trouver $x^* \in S \subset \mathbb{R}^n$ tel que

$$f(x^*) = \min f(x), x \in S \tag{2.1}$$

L'ensemble des contraintes S est donné en général par des équations et des inéquations linéaires ou non. Il peut être représenté de la maniére suivante :

$$S = \{x \in \mathbb{R}^n / g(x) \leq 0\} \tag{2.2}$$

où la fonction g est une fonction vectorielle définie de $\mathbb{R}^n$ dans $\mathbb{R}^m$ et de classe C^1, avec

$$g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix}$$

$$Jg(x) = \begin{pmatrix} (\nabla g_1(x))^T \\ (\nabla g_2(x))^T \\ \vdots \\ (\nabla g_m(x))^T \end{pmatrix} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_2}{\partial x_1} & \cdots & \frac{\partial g_m}{\partial x_1} \\ \frac{\partial g_1}{\partial x_2} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_2} \\ \vdots & \vdots & \vdots & \\ \frac{\partial g_1}{\partial x_n} & \frac{\partial g_2}{\partial x_n} & \cdots & \frac{\partial g_m}{\partial x_n} \end{bmatrix}$$

c'est la matrice jacobienne.

Les résultats fondamentaux de l'optimisation non linéaire sont obtenus dans le cas ou la fonction objectif f est non linéaire et où les contraintes qui définissent l'ensemble S sont linéaire.

Définition 2.1. Soit f une fonction réelle définie sur un ensemble ouvert S de $\mathbb{R}^n$. La fonction f admet un minimum local (strict) en $x^* \in S$ si $\exists B(x^*, \epsilon) = \{x / \|x - x^*\| < \epsilon\} \subset S$ tel que :

$$f(x) \geq f(x^*), \forall x \in B(x^*, \epsilon) \quad (f(x) > f(x^*)) \tag{2.3}$$

Définition 2.2. Soit f une fonction réelle, définie sur un ensemble de S de $\mathbb{R}^n$. La fonction f admet un minimum global en $x^* \in S$ si : [8]

$$f(x) \geq f(x^*), \forall x \in S \tag{2.4}$$

2.2.1 conditions nécessaires d'optimalité locale

Supposons que f est continue et a des dérivées partielles premiéres $\partial f/\partial x_i ; i = 1, ..., n$ et secondes $\partial^2 f/\partial x_i \partial x_j ; i, j = 1, ..., n$ continues pour tout $x \in \mathbb{R}^n$. Alors :

Théorème 2.1. Une condition nécessaire pour que x^* soit un minimum local de f est :
(a) $\nabla f(x^*) = 0$ (stationnarité).
(b) le hessien $\nabla^2 f(x^*) = [\partial^2 f/\partial x_i \partial x_j(x^*)]$ est une matrice semi-définit positive.

Démonstration : Soit x^* un minimum local de f.
-Comme f est deux fois continûment différentiable, le développement de Taylor à l'ordre 2 au voisinage de x^* donne :

$$f(x) = f(x^*) + \nabla f^T(x^*)(x - x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(x^*)(x - x^*) + \|x - x^*\|^2 \epsilon(x - x^*)$$

avec $\epsilon(x - x^*) \to 0$ quand $x \to x^*$.
(a) Si $\nabla f(x^*) \neq 0$ alors en choisissant $x = x^* - \theta \nabla f(x^*)$ on aurait, pour $\theta > 0$ sufisamment petit : $f(x) < f(x^*)$ ce qui contredirait le fait que x^* est un minimum local. Donc la condition (a) est bien nécessaire, et on peut écrire :

$$f(x) = f(x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(x^*)(x - x^*) + \|x - x^*\|^2 \epsilon(x - x^*)$$

.
(b) Si la matrice $\nabla^2 f(x^*)$ n'est pas semi-définie positive, c'est qu'il existe un vecteur $d \in \mathbb{R}^n (d \neq 0)$ tel que : $d^T \nabla^2 f(x^*) d < 0$.
En choisissant alors $x = x^* + \theta d$, pour $\theta > 0$ suffisamment petit on aurait $f(x) < f(x^*)$ ce qui contredirait encore l'optimalité locale de x^*.
La condition (b) est donc bien nécessaire aussi

Remarque 2.1. Un point x^* qui vérifié la condition (a) est appelé un point stationnaire (critique).
Ces conditions sont nécessaires mais elles ne sont pas suffisantes pour garantir un minimum local.[7]

2.2.2 conditions suffisantes d'optimalité locale

Théorème 2.2. Sous les mêmes hypothéses qu'avant, une condition suffisante pour que x^* soit un optimum locale de f sur $\mathbb{R}^n$ est :
(a) $\nabla f(x^*) = 0$ (stationnarité)
(b) le hessien $\nabla^2 f(x^*)$ est une matrice définie positive

Démonstration : Considérons un point x^* satisfaisant les deux conditions (a) et (b) du théoréme. Le développement de Taylor de f au voisinage de x^* s'écrit alors :

$$f(x) = f(x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(x^*)(x - x^*) + \|x - x^*\|^2 \epsilon(x - x^*)$$

avec $\epsilon(x - x^*) \to 0$ quand $x \to x^*$.
Pour toute direction de déplacement $d \in \mathbb{R}^n (\|d\| = 1)$ on a alors :

$$f(x^* + \theta d) = f(x^*) + \frac{\theta^2}{2} d^T \nabla^2 f(x^*) d + \theta^2 . \epsilon(\theta)$$

où $\epsilon(\theta) \to 0$ quand $\theta \to 0$.
En vertu de la condition (b) on a : $d^T . \nabla^2 f(x^*) . d > 0$ et par suite, pour θ suffisamment petit, on aura : $f(x^* + \theta d) > f(x^*)$.
Ce qui montre que x^* est bien un minimum locale de f.
la condition (b) revient à supposer que f est strictement convexe dans un voisinage de x^* [7]

2.2.3 Cas des fonctions convexes : condition nécessaire et suffisante d'optimalité globale

Dans le cas d'une fonction convexe propre f définie sur $\mathbb{R}^n$, la condition necessaire et suffisante pour que x^* soit un minimum global de f est que 0 soit un sous-gradiente de f en x^*.
Pour une fonction continûment différentiable, on obtient donc :

Théorème 2.3. Si f est une fonction convexe continument différentiable, une condition nécessaire et suffisante pour que x^* soit un optimum global de f sur $\mathbb{R}^n$ est que : $\nabla f(x^*) = 0$.
Dans le cas convexe, la stationnarité à elle seule constitue une condition nécessaire et suffisante d'optimalité globale. [7]

2.3 Méthodes numériques pour la minimisation d'une fonction différentiable

Les méthodes numériques de minimisation d'une fonction sur $\mathbb{R}^n$ consistent toutes à rechercher un point stationnaire $\bar{x}$ à partir d'une approximation initiale donnée x^0. L'algoritheme générale se construit sur la forme suivante :

$$x^{k+1} = x^k + \theta_k l^k,\ k = 0, 1, ... \tag{2.5}$$

où l^kest k^{ime} direction d'amélioration ou de descente et θ_k est k^{ime} pas le long de cette direction. Ces méthodes numériques se différencient seulement par le choix de l^k et θ_k .
Les méthodes du gradient sont les plus couramment utilisées parmi les algorithmes de minimisation des fonctions différentiables.

2.3.1 Les méthodes du gradient

Toute les méthodes du gradient sont basées sur le schéma suivant :

$$x^{k+1} = x^k + \theta_k l^k,\ k = 0, 1, ...$$

les différents algorithmes existants ne se différencent que par le choix de θ_k le long de la direction de descente qui est l'anti-gradient.
Parmi ces dernières on distingue notamment :
• L'algorithme du gradient à pas fixe ($\theta_k = \theta = constant$)
• L'algorithme du gradient à pas optimal(ou l'algorithme de la descente la plus rapide)
• L'algorithme du gradient à pas prédéterminé (ou méthode de la série divergente)

Ces algorithmes en général ne sont pas finis ; on doit alors définir un test d'arrêt. Ce dernier est basé sur l'un des critéres suivants :
i) $\|\nabla f(x^k)\| < \epsilon, \epsilon > 0$ donnée $\epsilon = 10^{-6}\|\nabla f(x^0)\|$
ii) $\max |\frac{\partial f}{\partial x_i}| < \epsilon, i = 1, 2, ..., n, \epsilon$ donné.

iii) $\|x^k - x^{k-1}\| < \epsilon, \epsilon$ donné.

Une fois le point stationnaire $\bar{x} \simeq x^k$ est trouvé, il restera alors à vérifier que le vecteur x^k correspond bien à un minimum, car ça peut être un point selle.
Dans ce travail nous allons citer que l'algorithme du gradient a pas optimal.

Algorithme du gradient à pas optimal

Pour cette méthode, les itérations sont construites selon la formule itérative :

$$x^{k+1} = x^k - \theta_k \nabla f(x^k),\ k = 0, 1, 2, ... \tag{2.6}$$

où le pas θ_k vérifie la condition

$$f(x^k - \theta_k \nabla f(x^k)) = min_{\theta \geq 0} f(x^k - \theta \nabla f(x^k))$$

Cette condition signifie que le mouvement le long de la direction de l'anti-gradient se poursuit tant que la fonction $f(x)$ décroit.
Si $\lim_{\|x\| \to \infty} f(x) = \infty$; alors la suite $\{x^k\}$ construite selon la forme itérative (2.6) vérifie la relation suivante :

$$\lim_{k \to \infty} \nabla f(x^k) = 0$$

Propriété 2.1. L'algorithme du gradient à pas optimal possède la propriété suivante : deux direction de déplacement successives sont orthogonales.

En effet, le pas θ_h est calculer de façon à minimiser la fonction

$$\varphi(\theta) = f(x^k + \theta l^k);\ l^k = -\nabla f(x^k).$$

On doit avoir

$$\varphi'(\theta_k) = \nabla f'(x^k + \theta_k l^k).l^k = \nabla f'(x^{k+1}) l^k = 0$$

D'où l'on déduit $< l^{k+1}, l^k >= 0$

Cette propriétés des direction successives orthogonales a des effets néfastes pour des fonctions dual conditionnées. En effets, le nombre d'itération peut être considérable pour minimiser ce genre de fonction.[8]

2.3.2 Méthode de Newton

La méthode de Newton n'est pas une méthode d'optimisation à proprement parler. C'est en réalité une méthode utilisée pour resoudre des équations non linéaires de la forme $F(x) = 0$ où F est une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^n$. Nous allons d'abord la décrire puis montrer comment on peut l'appliquer à la recherche de minimum.

2.3.2.1 Description de la méthode

Considérons le problème d'optimisation sans contraintes (P)

$$(P) : \min_{x \in \mathbb{R}^n} f(x) \tag{2.7}$$

oú $f : \mathbb{R}^n \to \mathbb{R}$

Le principe de la méthode de Newton consiste à minimiser successivement les approximation du second ordre de f, plus précisément si

$$f(x) = f(x) + \nabla f(x^k)'(x - x^k) + (x - x^k)' H(x^k)(x - x^k) + o\|x - x^k\|^2,$$

posons

$$q(x) = f(x) + \nabla f(x^k)'(x - x^k) + (x - x^k)' H(x^k)(x - x^k)$$

Soit x^{k+1} l'optimisation de q, alors il vérifie $\nabla q(x^{k+1}) = 0$, soit en remplaçant :

$$\nabla f(x^k) + H(x^k)(x^{k+1} - x^k) = 0,$$

ou encore

$$H(x^k)(x^{k+1} - x^k) = -\nabla f(x^k)$$

donc

$$x^{k+1} = x^k - [H(x^k)]^{-1}\nabla f(x^k).$$

Algorithme

Etape initiale : Soit $\epsilon > 0$, critère d'arrêt. Choisir x^1 point initial, poser $k = 1$ et aller a l'etape principale.

Etape principale : Si $\|\nabla f(x)\| \leq \epsilon$ stop, sinon poser
$x^{k+1} = x^k - [H(x^k)]^{-1}\nabla f(x^k)$ remplacer k par $k + 1$ est aller a l'etape principale.

2.3.2.2 Avantage et inconvénients

Avantages

Si le point x^1 et assez proche de la solution optimale locale x^* telle que $H(x^*)$ soit définie positive, alors l'algorithme de Newton converge de façon quadratique vers la solution x^*. c'est à dire que l'on a,

$$\|x^{k+1} - x^k\| \leq \gamma\|x^k - x^*\|^2 \qquad \gamma \geq 0.$$

Inconvénients

1-Cette méthode fonctionne trés bien pour les problémes de petite dimension ($1 \leq n \leq 10$), lorsque ou peut calculer facilement la matrice Hessienne H et sont inverse. Ce calcul nécessite des itérations plus nombreuses et couteuses dans les problémes de grandes tailles.
2- Comme $x^{k+1} = x^k - [H(x^k]^{-1}\nabla f(x^k)$ On voit bien que le successeur x^{x^k+1} de x^k n'est pas toujours bien défini.
3- Méme si $H(x^k)^{-1}$ existe la direction $d^k = -[H(x^k)]^{-1}\nabla f(x^k)$ n'est pas toujours une direction de descente.

Théorème 2.4. Soit F est une fonction de classe C^2 de $\mathbb{R}^n$ dans $\mathbb{R}^n$ et x_* un zéro de F (c'est à dire $F(x_*) = 0$). On suppose en autre que ce zéro est isolé et que $DF(x_*)$ est inversible (DF désigne la dérivée premiére de F).
Alors il existe une boule fermée ß centrée en x_* telle que, pour tout point $x^0 \in$ ß, la suite (x^k) définie par la méthode de Newton est entièrement contenue dans ß et converge vers

x_* qui est le seul zéro de F dans ß.
Enfin la convergence est géometrique : il existe $\beta \in]0,1[$ tel que

$$\forall k \geq 0 \|x^k - x_*\| \leq \beta_k \|x^0 - x_*\|$$

En d'autre termes, si on choisit le point de départ x^0 "assez prés" de x_*, alors l'algorithme converge vers x_*[4]

2.4 optimisation non linéaire avec contrainte

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

$$minf(x), (x \in \mathbf{S}) \tag{2.8}$$

Où $\mathbf{S} = \{x \in \mathbb{R}^n / g_i(x) = 0, i = 1, 2, ..., k, g_i(x) \leq 0, i = k+1, ..., m\}$ désigne l'ensemble des solution réalisable. On suppose que les fonction $\mathbf{f}$ et $g_i (i = 1, ..., m)$ sont de classe C^1 ; c'est à dire continument différeniable sur $\mathbb{R}^n$

Définition 2.3. • Un vecteur $x \in \mathbb{R}^n$ est appelé solution réalisable ou plan de problème (2.7) s'il vérifie toutes les contrainte de problème i.e $x \in S$
• Une solution réalisable x^* est appelé solution optimale du problème (2.7)si

$$f(x^*) \leq f(x), \forall x \in S$$

et on note $\min_{x \in S} f(x) = f(x^*)$
• $x^* \in S$ est appelé minimum local de problème (2.7) s'il existe un réel $\epsilon > 0$, telque :

$$f(x^*) \leq f(x), \forall x \in S \cap \mathbb{B}(x^*, \epsilon)$$

,où $\mathbb{B}(x^*, \epsilon)$est la boule de centre x^* et de rayon ϵ

2.4.1 Conditions d'optimalité pour le cas des contraintes linéaires de type égalité

Le problème se formule sous la forme suivante :

$$\begin{cases} minf(x), \\ g_i(x) = A_i'x - b_i = 0, \forall i \in I = \{1, 2, ..., m\} \end{cases} \tag{2.9}$$

où f :$\mathbb{R}^n \to \mathbb{R}$ est une fonction continument différentiable, b est un vecteur de $\mathbb{R}^m$ et A une matrice d'ordre $m\times$ n, formée des vecteurs colonnes et ligne suivant :

$$A = (a_1, a_2, ..., a_n); \quad A = \begin{pmatrix} A\prime_1 \\ A\prime_2 \\ \vdots \\ A\prime_n \end{pmatrix}; \quad g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix},$$

est une fonction vectorielle définie de $\mathbb{R}^n \longrightarrow \mathbb{R}$ pour que l'ensemble des solution réalisable S ne soit pas vide ou ne soit pas reduit à un point isolé, on considérera que *rang* $A = m < n$.

Définition 2.4. La fonction $L(x, \lambda) = f(x) + \Sigma_{i=1}^{m} \lambda_i g_i(x)$ est appelée fonction de Lagrange associée au problème de (2.8)où le vecteur $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m) \in \mathbb{R}^m$, formé des multiplicateurs de Lagrange λ_i.

Théorème 2.5. soit x^* un minimum du problème (2.8), alors il existe nécessairement un vecteur $\lambda \in \mathbb{R}^m$ vérifiant :

$$\nabla f(x^*) + A'\lambda = 0 \tag{2.10}$$

Si de plus A est de rang complet en ligne, alors λest unique.
La condition (2.9) peut être donné autrement, en utilisant la fonction de Lagrange

$$\nabla_x L(x, \lambda) = \nabla f(x) + A'\lambda = 0 \tag{2.11}$$

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

$$Ax = b \Rightarrow \nabla_\lambda(x, \lambda) = Ax - b = 0 \tag{2.12}$$

En combinant les relation (2.10) et (2.11) on obtient alors la condition nécessaire d'optimalité de premier ordre pour le problème (2.8)

Théorème 2.6. *(théorème de Lagrange)*
soit x^* un minimum local (ou global) pour le problème (2.8), Alors il existe un vecteur multiplicateur de Lagrange $\lambda^* \in \mathbb{R}^m$, tel que :

$$\nabla(x^*, \lambda^*) = 0 \Longleftrightarrow \begin{cases} \nabla_{\lambda^*} L(x^*, \lambda^*) = 0 \\ \nabla_{x^*} L(x^*, \lambda^*) = 0 \end{cases} \tag{2.13}$$

Le couple (x^*, λ^*)est appelé point stationaire de la fonction de lagrange. La condition nécessaire du second ordre pour le problème (2.8) est la suivante.

Théorème 2.7. *soit x^* un minimum local pour le problème (2.8) et λ^* un vecteur multiplicateur de Lagrange vérifiant (2.12), alors la matrice $\nabla^2 f(x^*)$ est semi-définie positive sur l'ensemble des points de la variété linéaire $Ay = 0$. Autrement dit :*

$$y' \nabla^2 f(x^*) u \geq 0 \ , \forall y \in \nu = \{y \in \mathbb{R}^n \quad : \quad Ay = 0\} \tag{2.14}$$

La condition suffisante de second ordre est la suivante :

Théorème 2.8. soit (x^*, λ^*) un couple de vecteurs vérifiant la condition nécessaire d'optimalité de premier ordre du problème (2.8) i.e ;

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

pour que x^0 soit un minimum local du problème (2.8), il est suffisant que la matrice $\nabla^2 f(x^*)$ soit définie positive sur le sous espace vectoriel ν.

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

Considérons maintenant un problème non linéaire avec contraintes linéaires de type inégalité

$$\begin{cases} minf(x), \\ g_i(x) = A_i'x - b_i \leq 0, \forall i \in I = \{1, 2, ..., m\} \end{cases} \tag{2.15}$$

Définition 2.5. soit x une solution réalisable du problème (2.14). l'ensemble des contraintes actives (saturées) au point x est l'ensemble d'indices suivant :

$$I_0 = I_0(x) = \{i \in I \quad : \quad A_i'x = b_i\}.$$

2.5 Optimisation quadratique convexe

l'hypothése de la convexité apporte élégance et simplicité à la théorie de l'optimisation. En particulier, les conditions nécessaires d'optimalité deviennent également suffisantes, et tout le résultat acquiert un caractére global.

Définition 2.6. *On dit qu'un problème de programmation mathématique est convexe (res.strictement convexe) s'il consiste à minimiser une fonction convexe (res.strictement convexe) sur un domaine convexe.*
L'étude des problèmes convexes et des algorithmes de résolution correspondants est l'objet de la programmation convexe. L'hypothése de convexité est cruciale en optimisation. Notons que

- *Les problème convexes sont synonymes de minimisation*
- *Les problèmes convexes sont les* $\prec$ bons $\succ$ *problèmes de la théorie : ceux pour les quels il existe des algorithmes de résolutions efficaces.*
- *l'hpothése de convexité ne garantit cependant ni l'existence ni l'unicité d'une éventuelle solution.*

Voici quelques propriétés pour tout problème de programmation convexe :

Propriété 2.2. soit f une fonction définie sur un convexe $\mathbb{R}^n$. Alors l'ensemble $\mathbb{M}$ des point où f atteint son minimum est convexe.

Propriété 2.3. Tout problème strictement convexe admet au plus une solution.

Propriété 2.4. Soit $f : C \subset \mathbb{R}^n \to \mathbb{R}$. Alors tout minimum local est un minimum global.

Propriété 2.5. Si la fonction f est strictement convexe, alors son minimum global est atteint en un seul point x^*.

2.5.1 Position du problème quadratique convexe(PQC)

Il s'agit d'une classe de problème d'optimisation où la fonction objectif est quadratique, s'écrivant sous la forme $F(x) = \frac{1}{2}x'Dx + C'x$, avec D symétrique, que l'on minimise sur un polyèdre convexe fermé. Ce genre de programmes est convexe dés que la matrice D est semi-définie positive.
On remarquera qu'un problème linéaire n'est autre qu'un problème

quadratique dégénéré (D=0), et c'est toujours un problème convexe.
L'étude des problèmes quadratiques convexes constitue un domaine propre de la théorie de la programmation quadratique ; le résultat le plus remarquable est le suivant :

Propriété 2.6. Tout problème quadratique convexe dont la valeur est finie admet au moins une solution.

2.6 La notion de Dualité

La notion de dualité est un concept fondamental en programmation linéaire et conduit à un résultat important d'un point point vue théorique et pratique : *Le théorème de la dualité.*

2.6.1 Dual d'un programmation linéaire sous forme standard

Considérons le programme linéaire (sous forme standard)

$$(P)\begin{cases} \textbf{Min } z = c^t.x \\ A.x = b \\ x \geq 0. \end{cases}$$

Associons à chaque contrainte, c'est-à-dire à chaque ligne i de la matrice, une variable u_i positive, négative ou nulle (appelée : variable duale) et considérons le programme linéaire :

$$(D)\begin{cases} \textbf{Max } \omega = u^t.b \\ \textit{sous les contraintes} : \\ u^t.A \leq c \end{cases}$$

où u est le m-vecteur ligne : $(u_1, u_2, ..., u_m)(m = \textit{nombre de ligne de } A)$
Le programme linéaire (D) est trés lié au programme linéaire (P). On remarque en effet :
• que la matrice des contraintes de (D) est la transposée de la matrice des contraintes de (P) ;
• que le vecteur des coûts de (P) est le seconds membres de (D) et vice-versa.
(D) est appelé le *dual* du programme linéaire (P).
Par oppostion au *dual*, le programme (P) est appelé le primal.

2.6.2 Définition du dual dans le cas générale

Évidement, on peut définir le dual d'un programme linéaire quelconque (pas nécessairement sous forme standard). Le tableau suivant résume les correspondances entre primale et dual et permet d'écrire directement le dual d'un programme linéaire quelconque. Le symbole $\gtrless 0$ est utulisé poue indiquer qu'une variable est non contrainte en signe.

PRIMAL	**DUAL**
Fonction économique (Min)	Seconde membre
Second membre	Fonction économique (Max)
A matrics des contraintes	A^T matrice des Contraintes
Contrainte i :$\geq$	Variable $u_i \geq 0$
Contrainte i :=	Variable $u_i \gtrless 0$
Variable $x_j \geq 0$	Contrainte j :$\leq$
Variable $x_j \gtrless 0$	Contrainte j :=

On obeservera en particulier, que le dual du dual est le primale. On peut donc, indifférmment désigner l'un des problèmes comme le primal et l'autre comme le dual.

2.6.3 Le théorème de la dualité

Il n'est aucunement restrictif de supposer que le primal (P) est mis sous forme standard. Le dual(D) prend alors la forme indiquée au 2.6.1

Lemme 2.1. Si $\bar{x}$ et $\bar{u}$ sont respictivement deux solutions quelconques du primal et du dual, alors :

$$\bar{z} = \text{c. } \bar{x} \geq \bar{w} = \bar{u}.\ b$$

Démonstration

$$\text{A . } \bar{x} = \text{b} \Longrightarrow \bar{u}\ \text{ . A . } \bar{x} = \bar{u}\ .b$$

et comme $\bar{x} \geq 0,\ \bar{u}$. A $.\bar{x}\ = \bar{u}\ .\ b \leq$ c$\bar{x}$ d'où le résultat découle.
On en déduit immédiatement le :

Corollaire 2.1. Si x^* et u^* sont respictivement des solutions du primal et du dual vérifiant $c.x^* = u^*.b$ alors x^* est solution optimale du primal et u^* solution optimale de dual.

Lemme 2.2. Supposons que (P) ait un optimum de valeur finie.
Soient π^* les multiplicateurs de la grange associés à une solution optimale x^* de (P).
Alors π^* est solution du dual et vérifie de plus $c.x^* = \pi^*.b$ (ce qui montre que π^* est solution optimale du dual).

Démonstration :

Si π^* sont les multiplicateurs du simplexe correspondant à une solution optimale x^* de(P), alors on a :

$$\bar{c_j} = c_j - \pi^*.A^j \geq 0 \quad \text{pour } x_j \text{ hors-base}$$
$$\text{et :} \quad \bar{c_j} = c_j - \pi^*.A^j = 0 \quad \text{pour } x_j \text{ en base.}$$

Donc $\pi^*.A \leq c$ et π^* est solution du dual (D).

D'autre part, soit B la base optimale de (P) correspondant à la solution x^*. On a :

$$\pi^*{=}c_B.B^{-1}$$

et par suite :

$$\pi^*.b = c_B.B^{-1}.b = c.x^*$$

D'après le corollaire, π^* est solution optimale du dual

Le lemme corespond au cas où les deux programme (P) et (D) ont des solution. Le théorème suivant englobe l'ensemble des cas possibles.

Théorème 2.9. (Théorème de la dualité [7])

Étant donné un programme linéaire (P) et le programme dual (D) associé :

(**a**) Si (P)et (D) ont des solutions ,alors chacun d'eux a une solution optimale et :

$$z^* = Min(P) = Max(D) = w^*$$

(**b**) Si l'un d'eux a un optimum non borné l'autre n'a pas de solution.

Démonstration :

Le point (a) résulte du lemme 2 précédent (en effet on peut toujours se ramener au cas où le programme (P) est mis sous forme standard). Pour démontrer le point (b) supposons que (D) par exemple soit non borné. Cela signifie qu'il existe toujours une solution duale u telleque ub> M pour M aussi grand que l'on veut. Alors, si (P) avait une solution $\bar{x}$, en vertu de lemme 1 on devrait avoir $u.b \leq c.\bar{x}$, $\forall u$ solution duale, et il en résulterait une contradiction

Remarque que le théorème précédent n'exclut pas le cas où aucun des deux problème (P) et (D) n'a de solution

Théorème 2.10. (Complémentarité)
Deux solutions $(\bar{x}, \bar{u})$ du primale et du dual respictivement sont optimales si et seulement si :

$$(\bar{u}\ .A^j - c_j).\bar{x}_j = 0 \qquad \forall j = 1, ..., n$$

(A^j=j-iéme colonne de A).

Démonstration :
On : $A.\bar{x} = b \Longrightarrow \bar{u}.A.\bar{x} - c.\bar{x} = \bar{u}.b - c.\bar{x}$.
Alors si $(\bar{x}, \bar{u})$ est une paire de solutions optimles , on a : $c.\bar{x} = \bar{u}.b$ d'où : $(\bar{u}.A - c).\bar{x} = 0$
.
Comme $\bar{x} \geq 0$ et $\bar{u}.A - c \leq 0$, la nullité du produit scalaire implique pour chaque j :

$$(\bar{u}.A^j - c_j)\bar{x}_j = 0$$

Inversement, si $(\bar{u}.A - c).\bar{x} = 0$ alors $c.\bar{x} = \bar{u}.b$ et en vertu du corollaire, le couple $(\bar{x}, \bar{u})$ est optimale.
Le théorème de complimentarité s'exprime encore sous la forme suivante :
Si un programme linéaire a des contraintes d'inégalités (par exemple, le dual (D) du problème (P) alors, à l'optimum :
- une variable duale correspondant à une contrainte non saturé est nécessairement nulle ;
- à une variable duale strictement positive correspond nécessairement une contrainte saturée.[7]

Chapitre 3

la méthode Branch and Bound en optimisation non linéaire non convexe

Branch and Bound est un algorithme assez général qui joue un rôle trés important dans la théorie d'optimisation globale. L'idée générale de la méthode est de décomposer le problème primaire en sous problèmes parallèlles (Branching) qui peut être graduellement plud facile à résoudre, puis évaluer les bornes inférieures et supérieures (Bounding) des valeurs des solutions optimales sur ces problémes secondaires. Par conséquent, le procédé Branch and Bound peut être représenté sous forme d'un arbre : le probléme initial est situé comme racine de cet arbre et les branches sont les sous problémes hiérarchiquement construits par l'algorithme. Cette construction est régie par la stratégie de la recherche qui déterminera la suite de solution des problème secondaires. La séquence de la décomposition et la recherche de la solution continue jusqu'à ce qu'il puisse vérifier que l'un ou l'autre sur la branche indiquée ne peut pas apporter une meilleure solution que la solution du condidat sortant (trouvé déja par le procédé Branch and Bound).
Notons que Branch and Bound a été à l'origine développée pour résoudre des problème de la programation entiére. Plus tard, cet algorithme a ètè appliqué avec succées dans des problème trés difficiles en optimisation globale comme : la minimisation des fonctions lipchitziennes, la minimisation de la différence de deux fonctions convexes.

3.1 Le procédé de l'algorithme Branch and Bound

Définition 3.1. *Considérons le problème d'optimisation globale :*

$$\min_{x \in D} f(x)$$

où $f : A \longrightarrow \mathbb{R},\ D \subset A \subset \mathbb{R}^n$

Définition 3.2. *Soit* P *un polyédre dans* $\mathbb{R}^n$*, tel que* $\text{int}(P) \neq \emptyset$*, et* I *un ensemble fini d'indice, la famille* $P_i : i \in I$ *de sous- polyédres* $(\text{int}(P_i) \neq \emptyset)$ *est dite partition de* P*, si*

$$P = \cup_{i \in I} P_i \text{ et } P_i \cap P_j = \partial P_i \cap \partial P_j,\ \forall i, j \in I, i \neq j$$

où ∂P_i *est la frontière de* P_i

Définition 3.3. *Soit* M *un élément de la partition d'un polyédre* $P(D \subset P)$

- *Si* $M \cap D = \emptyset$ M *est dit infaisable*
- *Si* $M \cap D \neq \emptyset$ M *est dit faisable*
- *Sinon* M *est dit incertain*

3.1.1 L'algorithme Branch and Bound de base

- **Etape 0** :Initialisation :

Chisir

$$P_0 \supseteq D$$
$$S_{P_0} \subset D$$
$$-\infty < \beta_0 \leq \min f(D)$$

P_0 est le plus petit polyèdre qui contient D si ce dernier n'est pas un polyèdre.

S_{P_0} est un ensemble de point choisis dans l'ensemble faisable.

Poser

$$\rho_0 = P_0$$
$$\alpha_0 = \min f(S_{P_0})$$
$$\beta_0 = \beta(P_0)$$

- Si $\alpha_0 < +\infty$, alors choisir

$$x^0 \in arg\min f(S_{P_0})\ i.e : f(x^0) = \alpha_0.$$

- Si $\alpha_0 - \beta_0 \leq \epsilon$, alors arrêter et poser $\alpha_0 = \beta_0 = minf(D)$

-Sinon, aller à l'étape 1

- Etape k :(k=1,2,...)

Au début de chaque étape k, on a la partition actuelle ρ_{k-1} de sous-ensemble de P_0 qui reste aprés élimination. De plus, pour tout $P \in \rho_{k-1}$, on a un ensemble de points $S_p \subseteq D \cap P$ et des bornes $\beta(P)$, $\alpha(P)$ vérifiant :

$$\begin{cases} \beta(P) \leq inf f(P \cap D) \leq \alpha(D),\ si\ P\ est\ faisable \\ \beta(P) \leq inf f(D),\ si\ P\ est\ incertain \end{cases}$$

On a aussi, les bornes β_{k-1},α_{k-1} qui vérifient :

$$\beta_{k-1} \leq inf\, f(D) \leq \alpha_{k-1}$$

Si $\alpha_{k-1} < +\infty$, alors on aura un point $x^{k-1} \in D$ tel que :

$$f(x^k - 1) = \alpha_{k-1}$$

- *k.1.*Eliminer tout sous ensemble $P \in \rho_{k-1}$ qui vérifie :

$$\beta(P) \geq \alpha_{k-1}$$

Considérer ℓ_k la classe des sous ensembles restants de ρ_{k-1}

- *k.2.* Sélectionner une classe d'ensembles non vide $\Im_k \subset \ell_k$ est construire une nouvelle partition avec chaque élément de $\Im_k$.

soit $\Im'_k$ la classe de tous les nouveaux éléments de la nouvelle partition.

- *k.3.*Eliminer chaque $P \in \Im'_k$ qui vérifie :

$$P \cap X = 0$$

Poser ρ'_k la classe de tous les ensembles restants se $\Im'_k$.

-*k.4.*Déterminer pour chaque $P \in \rho'_k$ un ensemble $S_p \subseteq P \cap D$ et un nombre $\beta(P)$ tels que :

$$\begin{cases} \beta(P) \leq inf\, f(P \cap D) \leq \alpha(P);\ si\ Pest faisable \\ \beta(P) \leq inf\, f(P),\ si\ P \cap D\ est\ un\ incertain \end{cases}$$

De plus il faut avoir :

$$S_p \supset S_{p'} \cap P \text{ et } \beta(P) \geq \beta(P')$$

pour tout ensemble $P' \supset P$ de ρ_{k-1}.
Poser

$$\alpha(P) = \min f(S_p)$$

-*k.5.* Soit

$$\rho_k = \{\ell_k \setminus \Im_k\} \cup \rho'_k$$

Calculer

$$\alpha_k = \min\{\alpha(P) : P \in \rho_k\}$$
$$\beta_k = \min\{\beta(P) : P \in \rho_k\}$$

Si $\alpha_k < +\infty$, alors déterminer $x^k \in D$ tel que :

$$f(x^k) = \alpha_k$$

- *k.6.*
- Si $\alpha_k - \beta_k \leq \epsilon$ alors arrêter et poser $\alpha_k = \beta_k = \min f(D)$ (x^k est la solution optimale)
- Sinon, poser $k = k + 1$

Aller à l'Etape k.

Remarque

1. Pour qu'un élément P de la partition ρ_k, sera supprimé à l'étape k il faut qu'il soit un élément "sondé", c-à-d il doit vérifier la condition $\beta(p) \geq \alpha_k - 1$. Alors le critére d'arrêt $\alpha_k = \beta_k$ signifie que tous les éléments de la partition sont sondés.
2. L'étape k est exécutée seulement s'il reste des éléments de la partition l_k ; alors il suffit

d'exiger que $l_k \subset \Im$, c-à-d ; chaque élément sondé de la partition peut être encore divié. En général, l'operation de la subdivition est définie seulement sur une certaine famille $\Im$ de sous-ensemble de $\mathbb{R}^n$ (par exemple, rectangle, cônes, simplexes).

3. Dans l'étape $k.4$ on peut évidemment remplacer n'importe quel $P \in \rho'_k$ par un plus petit ensemble $\widetilde{P} \subset P$ tel que $\widetilde{P} \in \Im, \widetilde{P} \cap D = M \cap D$.

4. Pour chaque partition P, S_P est une collection de points faisables incluse dans P, elle peut être rénovée au long de l'exécution de l'algorithme.

Au cour de l'algorithme, l'ensemble S_P est toujours fini. Alors le procédé ci-dessus est également défini dans le cas où les ensembles S_P sont vides et on peut avoir $\alpha_k = \infty$ pour tous k.

D'autre part il faut imposer des conditions sur S_P et $\beta(P)$ pour que $\{\alpha_k\} = \{f(x^k)\}$ soit une suite décroissante, $\{\beta_k\}$ une suite croissante, et $\alpha_k \geq \min f(D) \geq \beta_k$, afin que la défférence $\alpha_k - \beta_k$ peut mesurer approximativement la meilleure solution optimale x^k à l'étape k.

Pour une tolérance donnée $\epsilon > 0$, l'algorithme va s'arrêter dés que $\alpha_k - \beta_k < \epsilon$. Et puisque $\{\alpha_k\}$ a une monotonie décroissante et $\{\beta_k\}$ une monotonie croissante, alors les limites $\alpha = \lim_{k\to\infty} \alpha_k$ et $\beta = \lim_{k\to\infty} \beta_k$ vont exister, et par récurrence, il vont satisfaire $\alpha_k \geq \min f(D) \geq \beta_k$.

L'algorithme est dit fini si $\alpha_k = \beta_k$ à une certaine étape, tandis qu'il converge si $\alpha_k - \beta_k \to 0$, c-à-d $\alpha = \lim_{k\to\infty} f(x^k) = \beta = \min f(D)$.

3.2 L'interêt de L'optimisation

L'optimisation globale est une branche des mathématiques, le domaine de sa recherche est considéré beaucoup plus riche ces dernière années grâce aux ordinateurs dévelopés par la nouvelle technologie qui peuvent résoudre des problèmes d'optimisation de taille plus grand.

Les méthodes d'optimisation globale s'intéressent à la recherche d'un optimum(maximum ou minimum) globale d'une fonction, et non à la recherche d'un optimum local que les

méthodes classiques permettent de le trouver dans un voisinage réalisable, et c'est ça qui fait ladifférence entre les deux méthodes.
L'intérêt de l'étude de cette classede problème d'optimisation est qu'elle englobe un domaine très vaste d'applications réelles.
Un problème d'optimisation globale d'une fonction f est données est donné sous la forme suivante :

$$(P)\begin{cases} \min f(x) \\ x \in S \end{cases}$$

On cherche une solution optimale globale : $x_0 \in S$ et$\forall x \in S$,$f(x) \geq f(x_0)$
Et une solution optimale locale :$x_0 \in S$ et $\exists \nu_0$ un voisinage de x_0 telleque $\forall x \in \nu_0 \cap S$,$f(x_0) \geq f(x)$
Définition :(Minimum global)
$x_0 \in R^n$ est un minimum globale si et seulement si $x_0 \in R^n$ est un minimum globale de f sur S si et seulement si $x_0 \in S$ et$\forall x \in S$,$f(x) \geq f(x_0)$

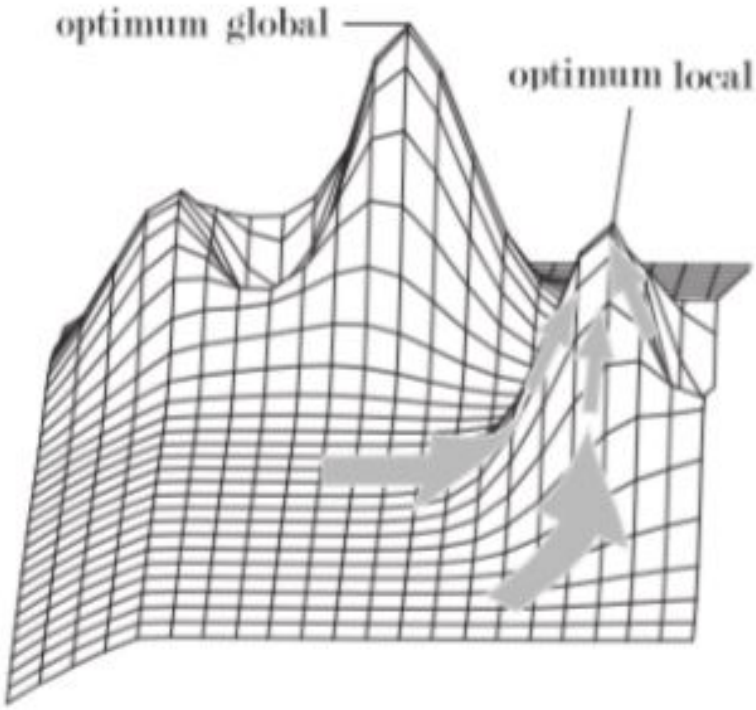

FIG. 3.1: - Optimum Global et Optimum Local

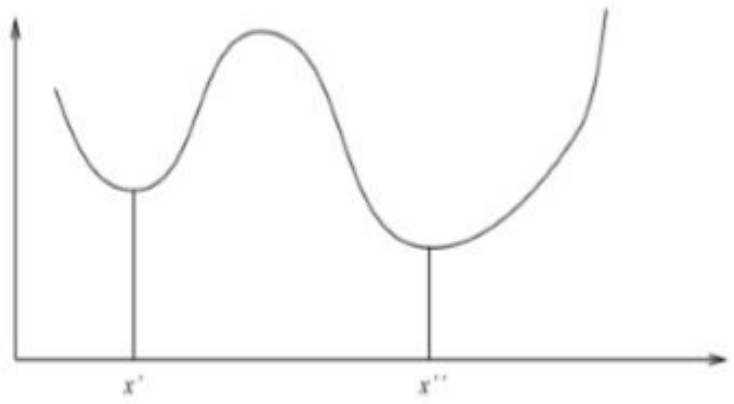

FIG. 3.2: - x' Minimum local et x" Minimum global

Définition :(Minimum Local)

$x_0 \in R^n$ est un minimum local de f sur $S \subset R^n$ si et seulement si $x_0 \in S$ et $\exists \nu_0$ un voisinage de x_0 telleque $\forall x \in \nu_0 \cap S, f(x_0) \geq f(x)$ C'est clair q'un minimum globale est un minimum local, et le contraire n'est pas vrai que pour le problème convexes

L'objectif dans la résolution d'un probléme d'optimisation globale (P) donné.

Dans le cas d'un programme linéaire, la solution est obtenu à un sommet en appliqueant la méthode du simplexe

Les problème d'optimisation convexe possédent des propriétés exeptionnelles citons deux fondamentales :

$1*$ La première est que x_0 est une solution optimale locale elle est aussi solution optimale globale.

$2 *$ La seconde est la suivante : si S est convexe et si f est differentiable alors $\nabla f(x) = 0$ est une condition nécessaire et suffisante d'optimalité, alors dans les problèmes non convexes

Le problème (P) présente plusieurs minimums locaux et en générale un minimum local est différent d'un minimum globale. Les méthodes classiques ne donnent pas la solution globale, mais seulement des solutions locales. Pour cela, plusieurs chercheurs dans le domaine ont proposé les méthodes d'optimisation non convexe.

Il existe deux catégories de méthodes utilisées en optimisation globale :

1. **Les méthodes stochastiques** qui sont appliquées pour des problèmes sans contraintes et sans structure particulière,elles sont plus rapides mais le résultat est obtenu avec une certaine probabilité proche de 1.

2.**Les méthodes déterministes**qui sont appliquées pour des problème avec contraintes et ayant une structure particulière, elles sont moins plus rapides mais l'optimum global existe toujours.

FIG. 3.3: - Méthodes d'optimisation globale.

Nous allons nous intéresser aux méthodes déterministes, parmi elles, la méthode d branche and bound qui repose sur la structure du problème d'optimisation étudié, telle la convexité ou la non convexité de la fonction objectif ou du domaine réalisable, la taille du problème, etc. Donc elle sera utulisée pour l'optimisation globale

La méthode branch and bound consiste à remplacer la fonction objectif par la fonction borne inférieure, qu'on va minimiser.

Cette méthode se dote d'une fonction qui permet de mettre une borne sur certaines

solutions pour soit les exclure, soit les maintenir comme des solutions potentielles.
L'idée a été déjà proposée pour les fonctions objectis satisfaisantes la propriété lipchitzienne.
En effet, la propriété lipchtizienne a fourni un chemain pour construire la fonction borne inférieure.

3.3 Condition d'arrêt et de convergence

Si le procédé s'achève à l'itération k, alors évidemment, x^k est la solution optimale et α_k est la valeur optimale de la fonction objectif. Cependant, on ne peut pas garantir l'arrêt de l'algorithme en un nombre fini d'itérations, on doit donc établir des conditions qui assurent que tous point d'accumulation de la suite $\{x^k\}$ est une solution optimale du problème (2.1). Notions d'abord que si l'algorithme est infini, alors il doit générer au moins une suite "extraite" $\{P_{k_q}\}$ de l'ensemble P_{k_q} issus des partitions successivement raffinées et vérifiant $P_{k_q} \supset P_{k_{q+1}}$. Ceci est une conséquence immédiate du fait que, à chaque itération k, la classe ρ_k contient seulement un nombre fini d'ensembles issus de la partition. Dans l'arbre qui représente Branch and Bound la suite infinie extraite correspond à une branche dont les noeuds sont P_{k_q} (q = 1,2,...).

Théorème 3.1. *Si pour toute suite infinie $\{P_{k_q}\}$, $P_{k_q} \supset P_{k_{q+1}} (q = 1, 2, ...)$ d'ensemble de partitions successivement raffinées, et toutes les bornes à l'itération k_q on a :*

$$\lim_{q\to\infty} (\alpha_{k_q} - \beta_{k_q}) = \lim_{q\to\infty} (\alpha_{k_q} - \beta(P_{k_q})) = 0, \tag{3.1}$$

alors

$$\beta = \lim_{k\to\infty} \beta_k = \lim_{k\to\infty} f(x^k) = \lim_{k\to\infty} \alpha_k = \alpha, \tag{3.2}$$

et chaque point d'accumulation x^* de la suite $\{x^k\}$ est une solution optimale de $\min\{f(x) : x \in D\}$

Preuve

Supposons que $\{x^k\}$ une suite infinie, puisque D est un compact, et $\{x^k\} \subset D$, alors la suite $\{x^k\}$ posséde un point d'accumulation.

D'autre part soit x^* un point d'acumulation de $\{x^k\}$, donc il existe une sous-suite appartenant à $\{x^k\}$ qui converge vers x^* . Par construction des ensembles P_k, cette sous-suite doit contenir une sous-suite infinie $\{x^{k_q}\}$ telle que $P_{k_q} \supset P_{k_{q+1}}$ (q = 1,2,...).

On a alors d'aprés les hypothése du théorème, et par la continuité de f sur le compact D on a :

$$\lim_{q\to\infty} f(x^{k_q}) = f(x^*).$$

Notons $f^* = \{f(x) : x \in D\}$, la suite des bornes inférieures $\{\beta_k\}$ satisfait les conditions $\beta_{k+1} > \beta_k$ et $\beta_k < f^*$ alors sa limite existe, posons $\beta = \lim_{k\to\infty} \beta_k$. En plus, la suite des bornes supérieurs $\{\alpha_k\}$ satisfait $\alpha_{k+1} < \alpha_k$ et $\alpha_k > f^*$ alors elle converge vers une limite finie $\alpha(\alpha = \lim_{k\to\infty} \alpha_k)$. On déduit donc que :

$$\beta \leq f^* \leq \lim_{k\to\infty} f(x^k) = f(x^*) = \alpha,$$

mais avec la condition (2.3) qui s'ecrit aussi sous forme

$$\lim_{q\to\infty} \alpha_{k_q} = \lim_{q\to\infty} \beta_{k_q},$$

et

$$\begin{cases} \lim_{q\to\infty} \alpha_{k_q} = \lim_{k\to\infty} \alpha_k \ car\ \{\alpha_{k_q}\}\ est\ une\ sous-suite\ de\ \{\alpha_k\} \\ et \\ \lim_{q\to\infty} \beta_{k_q} = \lim_{k\to\infty} \beta_k \ car\ \{\beta_{k_q}\}\ est\ une\ sous-suite\ de\ \{\beta_k\} \end{cases}$$

on déduit que :

$$\lim_{k\to\infty} \alpha_k = \lim_{k\to\infty} \beta_k.$$

Remarque 3.1. *En regardant les differentes étapes de l'algorithme, on voit qu'il se base sur deux étapes trés importantes :*

- *La subdivision de l'esembles faissable et les ensembles de partitions .*
- *Le calcul des bornes inférieurs et supérieurs de la fonction f sur les sous ensembles de partitions.*

3.4 Exemples numériques

3.4.1 Application de la méthode Branch and Bound à l'optimisation non linéaire non convexe

3.4.1.1 l'algorithme de Branch and Bound

Initialisation

a) soit ε un nombre suffisament petit et que K constant vérifier $K \geq \mid f''(x) \mid$ pour tout x dans $[a, b]$

b) posons $K := 0$,$T^0 = [a, b]$, $M := T^0$,$h_0 = a - b$,

$x^* = \frac{1}{2}(a + b) - \frac{1}{k(b-a)}(f(b) - f(a))$.

c) soit UB_0 :=$\min f(a), f(b), f(x_0^*)$.

d) soit $LB_0 := LB(T^0) := q_0(x_0^*) = \frac{K}{2}x_0^{*^2} + [\frac{f(b)-f(a)}{b-a} - \frac{K}{2}(b + a)]x_0^* + \frac{K}{2}ab + \frac{f(a)b-f(b)a}{b-a}$.

2) alors que : $UB_K - LB_k > \varepsilon$.

3) soit l'intervalle $T^k = [a_k, b_k] \in M$ tel que : $LB_k = LB(T^k)$

soit K_k une constante vérifier $K_k \geq \mid f''(x) \mid$ pour tout x dans $[a_k, b_k]$.

4) Subdiviser T^k en deux sous intervalles : $T_1^K = [a_k, x_k^*] := [a_k^1, b_k^1]$ et $T_2^k = [x_k^*, b_k] := [a_k^2, b_k^2]$.

5) pour i=1,2 posons :

5.1) posons $x_{k,i}^*$=$\frac{1}{2}(a_k^i + b_k^i)$-$\frac{f(b_k^i)-f(a_k^i)}{K_k(b_k^i-a_k^i)}$.

5.2) Si $x_{k,i}^* \notin]a_k^i, b_k^i[$ Alors $LB(T_i^k) = UB(T_i^k) = \min\{f(a_k^i), f(b_k^i)\}$.

sinon

$LB(T_i^k) := \frac{K_k}{2}x_{k,i}^{*^2} + [\frac{f(b_k^i)-f(a_k^i)}{b_k^i-a_k^i} - \frac{k}{2}(b_k^i + a_k^i)]x_{k,i}^* + \frac{K_k}{2}a_k^i b_k^i + \frac{f(a_k^i)b_k^i-f(b_k^i)a_k^i}{b_k^i-a_k^i}$.

5.3) $T_i^k : M \leftarrow M \cup \{T_i^k : UB^k - LB(T_i^k) \geq \varepsilon, i = 1, 2\} \setminus \{T_i^k\}$.

5.4) $UB_k = \min\{UB_k, f(a_k^i), f(b_k^i), f(x_{k,i}^*)\}$.

6) Mettre $LB_k = \min\{UB(T) : T \in M\}$

7) Supprimer de M tout intervalle T tel que : $LB(T) > UB_k - \varepsilon$.

8) posons $k \longleftarrow k + 1$

9) Fin tant que

10) Stop :x^k est une solution optimale de (P). [5]

3.4.1.2 Application Numériques de l'algorithme Branch and Bound

Probléme d'une fonction non convexe. Ètant donnée le probléme de minimisation suivante :

$$(P)\begin{cases} \alpha := \min f(x) \\ x \in S, \end{cases}$$

Exemple 1 :

Soit à résoudre le Probléme de minimisation suivant (S) :

$$(P)\begin{cases} \alpha := \min f(x) \\ x \in S \end{cases} \Rightarrow (P)\begin{cases} \min\{\sin x + sin(\frac{10x}{3}) + \ln x - 0.84x\} \\ x \in [2.7, 7.5] \end{cases}$$

Avec $\varepsilon = 2 \times 10^{-3}$

0 Itération

a) $K \geq \mid f''(x) \mid$ donc $K = 12.5$ et $\varepsilon = 2 \times 10^{-3}$ pour tout x dans $[a^0, b^0]$.

b) posons $K := 12.5, T^0 = [a^0, b^0] = [2.7, 7.5], M := \{T^0\}$.

- calculer $f(a^0)$, $f(b^0)$, x_0^*

$$\begin{cases} f(a^0) = -0.4352. \\ f(b^0) = -3.4794. \\ x_0^* = 5.1507 \end{cases}$$

c) calculer de $f(x_0^*), UB_0, LB_0$

$$\begin{cases} f(x_0^*) = -4.5869. \\ UB_0 = -4.5869. \\ LB_0 = -38.5389. \end{cases}$$

d) $UB_0 - LB_0 = -4.5869 - (-38.5389) = 33.952 > \varepsilon$ On passe à la 1^{iere} itération, c.à.d : subdiviser l'intervalles $I_0 = [2.7, 7.5]$ en deux sous intervalles I_{1_1} et I_{1_2}.

1 Itération

1) soit $x \in [a^1, b^1]$

2) posons $I_1 = [a^1, b^1] = [2.7, 7.5]$

$1^{er} Cas$ **:** $x \in [a^1, x_0^*] = [2.7, 5.1507] = I_{1_1}$

- calculer $f(a_1^1)$, $f(b_1^1)$, $x_{1_1}^*$

$$\begin{cases} f(a_1^1) = -0.4352. \\ f(b_1^1) = -4.5869. \\ x_{1_1}^* = 4.0609 \end{cases}$$

c) calculer de $f(x_{1_1}^*)$ LB_{1_1} et UB_{1_1} :

$$\begin{cases} f(x_{1_1}^*) = -1.9801 \\ UB_{1_1} = -4.5869 \\ LB_{1_1} = -12.0101. \end{cases}$$

$2^{eme} Cas$ **:** $x \in [a^1, b^1] = [5.1507, 7.5] = I_{1_1}$

- calculer $f(a_2^1)$, $f(b_2^1)$, $x_{1_2}^*$

$$\begin{cases} f(a_2^1) = -4.5869. \\ f(b_2^1) = -3.4794. \\ x_{1_2}^* = 6.3254 \end{cases}$$

c) calculer de $f(x_{1_2}^*)$ LB_{1_2} et UB_{1_2} :

$$\begin{cases} f(x_{1_2}^*) = -2.6392. \\ UB_{1_2} = -4.5869. \\ LB_{1_2} = 13.2107. \end{cases}$$

d) calculer LB_1, UB_1

$$\begin{cases} UB_1 = \min\{UB_{1_1}, UB_{1_2}, UB_0\} = -4.5869. \\ LB_1 = \min\{LB_{1_1}, LB_{1_2}\} = -13.2107. \end{cases}$$

$UB_1 - LB_1 = -4.5869 - (-13.2107) = 8.6238 > \varepsilon$ On passe à la 2^{eme} itération, c.à.d : subdiviser l'intervalles $I_{1_2} = [5.1507, 7.5]$ en deux sous intervalles I_{2_1} et I_{2_2}. car $LB_{1_1} > LB_{1_2}$

- Pas d'élimination d'intervalles car les $LB_{k_j} \not\geq UB_{k+1}$

2 Itération

1) soit $x \in [a^2, b^2]$

2) posons $I_1 = [a \cdot b^2] = [5.1507, 7.5]$

$1^{er} Cas$ **:** $x \in [a^1, x_0^*] = [5.1507, 6.3254] = I_{2_1}$

- calculer $f(a_1^2)$, $f(b_1^2)$, $x_{2_1}^*$

$$\begin{cases} f(a_1^2) = -4.5869. \\ f(b_1^2) = -2.6390. \\ x_{2_1}^* = 5.6054 \end{cases}$$

c) calculer de $f(x_{2_1}^*)$ LB_{2_1} et UB_{2_1} :

$$\begin{cases} f(x_{2_1}^*) = -3.7760 \\ UB_{2_1} = -4.5869 \\ LB_{2_1} = -5.8789. \end{cases}$$

$2^{eme} Cas$ **:** $x \in [a^2, b^2] = [6.3254, 7.5] = I_{2_2}$

- calculer $f(a_2^2)$, $f(b_2^2)$, $x_{2_2}^*$

$$\begin{cases} f(a_2^2) = -2.6390. \\ f(b_2^2) = -3.4794. \\ x_{2_2}^* = 7.0453 \end{cases}$$

c) calculer de $f(x_{2_2}^*)$ LB_{2_2} et UB_{2_2} :

$$\begin{cases} f(x_{2_2}^*) = -4.2722. \\ UB_{2_2} = -4.2722. \\ LB_{2_2} = -5.8789. \end{cases}$$

d) calculer LB_2, UB_2

$$\begin{cases} UB_2 = \min\{UB_{2_1}, UB_{2_2}, UB_1\} = -4.5869. \\ LB_2 = \min\{LB_{2_1}, LB_{2_2}, LB_{1_2}\} = -13.2107. \end{cases}$$

$UB_2 - LB_2 = -4.5869 - (-13.2107) = 8.6238 > \varepsilon$ On passe à la ^{eme} itération, c.à.d : subdiviser l'intervalles $I_{1_1} = [2.7, 5.1507]$ en deux sous intervalles I_{3_1} et I_{3_2}.

- Pas d'élimination d'intervalles car les $LB_{k_j} \ngeq UB_{k+1}$

Remarque :

On à écrit les deux itérations dans le but d'expliquer la procedure de l'algorithme branch and bound pour un problème d'une fonction à une seule variable d'optimisation globale.

Conclusion

Au beau de la 7^{ieme} itérations on aura la solution suivante du probléme (P) :

Donc la solution optimale exacte est $x^* = 5.199778$ et la valeur optimale est -4.601308.

Exemple 2 :

où : $f(x) = \frac{3}{4}\sin x + \frac{1}{4}\cos x$ et $x \in [0, 1]$.

Avec $\varepsilon = 2 \times 10^{-3}$

0 Itération :

a) $K \geq \mid f''(x) \mid$ donc $K = 1$ et $\varepsilon = 2 \times 10^{-3}$ pour tout x dans $[a^0, b^0]$.

b) posons $K := 1, T^0 = [a^0, b^0] = [0, 1], M := \{T^0\}, h_0 = a - b = -1$.

- calculer $f(a^0)$, $f(b^0)$, x_0^* :

$f(a^0) = 0.25$.

$f(b^0) = 0.7662$.

$x_0^* := \frac{1}{2}(a^0 + b^0) - \frac{1}{K(b^0 - a^0)}(f(b^0) - f(a^0)) = \frac{1}{2} - (f(1) - f(0)) = -0.0162$.

c) calculer de LB_0 et UB_0.

d) $x_0^* \notin [0, 1]$ Alors $LB_0 = UB_0 = \min\{f(0), f(1)\} = f(0) = 0.25$. Donc la solution optimale exacte est $x_0^* = 0$ et la valeur optimale est 0.25.

Chapitre 4

Verification des Résultats sur LINGO

4.1 Introduction

Ce chapitre est basé sur la programmation des problémes d'optimisation globale des fonctions non linéaires non convexes.
La programmation des problémes d'OG est faite sur le ligiciel LINGO 13 avec la méthode branch and bound.
Aprés l'exécution du MODEL de chaque probléme (P) donnée, nous aurons directement la solution qui est un minimum global, est non local, en un nombre fixé d'itérations, et en un temps bien compter en seconde.
La programmation des problèmes de fonction à une seule variable définie sur un intervalle, nous avons choisies 2 exemples.

LINDO : Logiciel pour la résolution des programmes linéaires et non linéaires.

I. Introduction

Lindo est un logicial utilisé pour résoudre les modèles d'optimisation linéaires, entiers et quadratiques.Il est aussi utilisé pour résoudre les modéles d'optimisation globale non linéaires. Une des caractéristiques de Lindo c'est qu'il offre dans outils qui peuvent aider à l'analyse dand modéles en utilisant la méthode Simplexe.

II. Installation du Logiciel

Les étapes de l'instalation sont :

1. Démarrer Windows.

2. Insérer le CD-ROM.

3. Cliquer sur l'icone Setup (install) dans votre explorateur de Windows.

4. Suivre les instructions sur l'écran.

Pour plus d'nformation sur ce logiciel visiter l'adresse Web : WWW.lindo.com

Remarque : Une fois le logiciel est installé, vous cliquez sur la commande Help.

Puis cliquer sur about LINGO, et vous la figure suivante :

FIG. 4.1: -

4.2 Programmation d'exemples de problémes de fonction avec une seule variable sur LINGO avec la Méthode Branch and Bound

Dans cette section, et sur la base l'exemple d'une fonction à seule variable, on va focaliser notre attention sur les opérations suivantes : introduire les données, résoudre le probléme, et présenter les résultats que donne LINDO.

(a)-Le probléme (P) à résoudre :

Exemple 1 Le programme non linéaire qui modélise le probléme (P) à résoudre est le suivant :

$$(P)\begin{cases} Min\{\sin x + sin(\frac{10x}{3}) + \ln x - 0.84x\} \\ x \in [2.7, 7.5] \end{cases}$$

(b)-Introduction le modèle :

Double clique "13.0 for Windows" de votre menu démmarrer/programmes.

Le logiciel va s'exécuter on aura cette fenêtre qui s'affiche sur notre écran après avoir saisi les données de la fonction f.

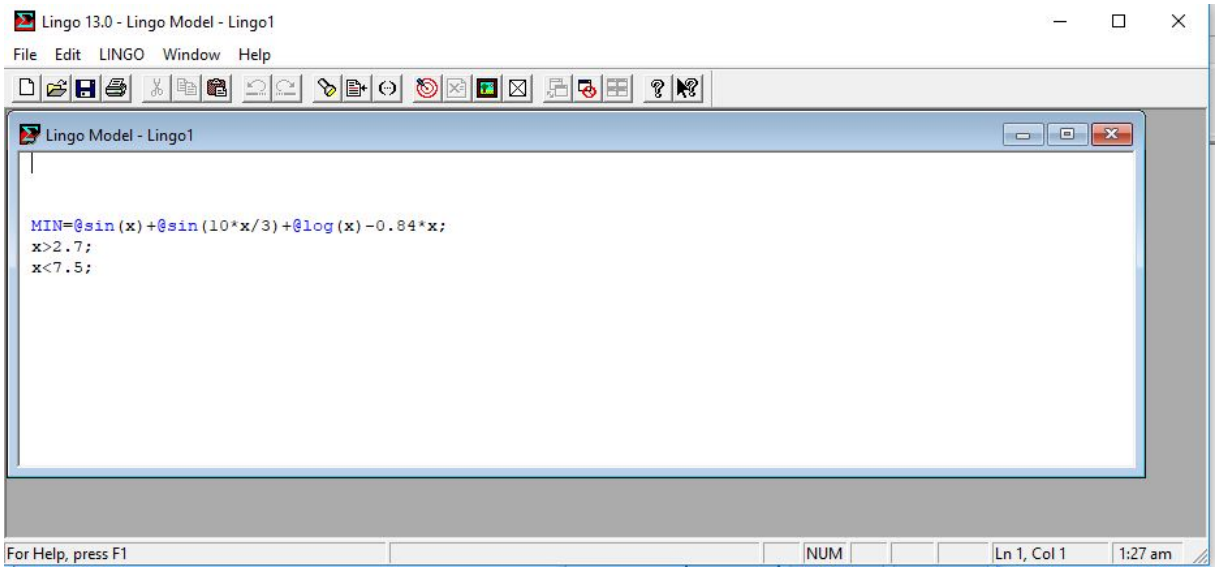

FIG. 4.2: -

Dans tous les modèles de Lindo la fonction objectif est définie en première ligne. Dans notre exemple la fonction objectif est : Min

$$Min \quad \sin x + sin(\frac{10x}{3}) + \ln x - 0.84x.$$

(c)-Résolution du probléme :

Aprés avoir écrit convenablement le programme non lineaire, en passe maintenant à la résolution. Pour résoudre notre programme il faut cliquer sur le bouton (qui représenter par la figure suivante) dans la barre d'outils.

FIG. 4.3: -

Lindo va commencer ainsi à compiler le modéle. Lors de la compilation, on voit les deux figures suivantes :

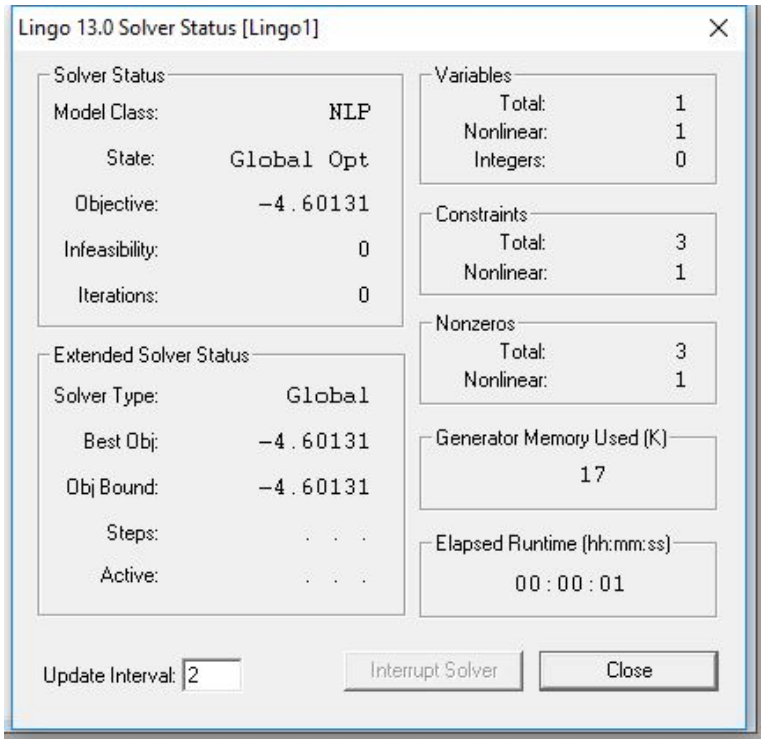

FIG. 4.4: -

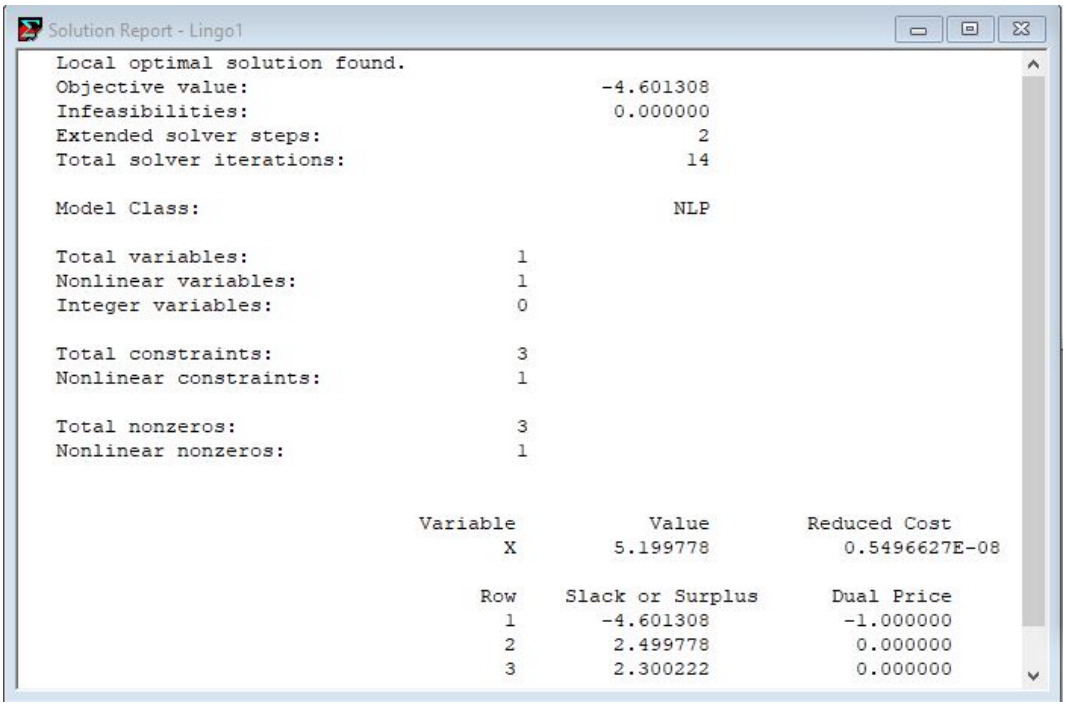

FIG. 4.5: -

Exemple 2 Le programme non linéaire qui modélise le probléme (P) à résoudre est le suivant :

$$\begin{cases} Min \frac{3}{4}\sin x + \frac{1}{4}\cos x \\ x \in [0,1] \end{cases}$$

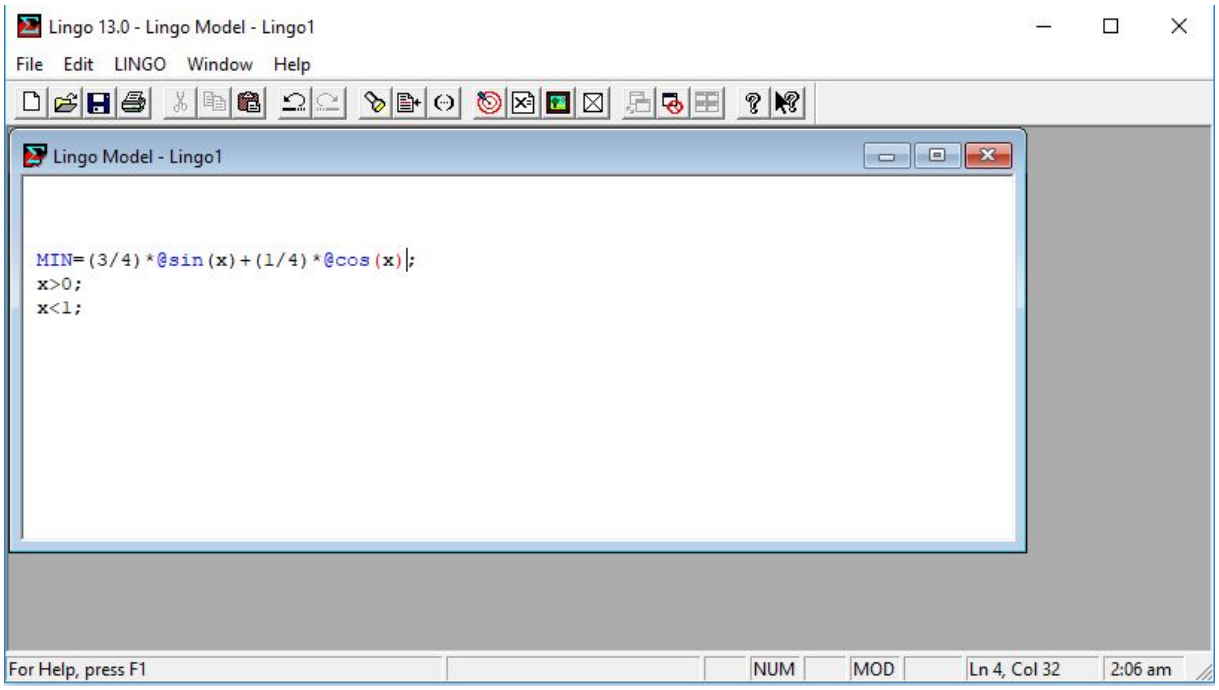

FIG. 4.6: -

Lindo va commencer ainsi à compiler le modéle. Lors de la compilation, on voit les deux figures suivantes :

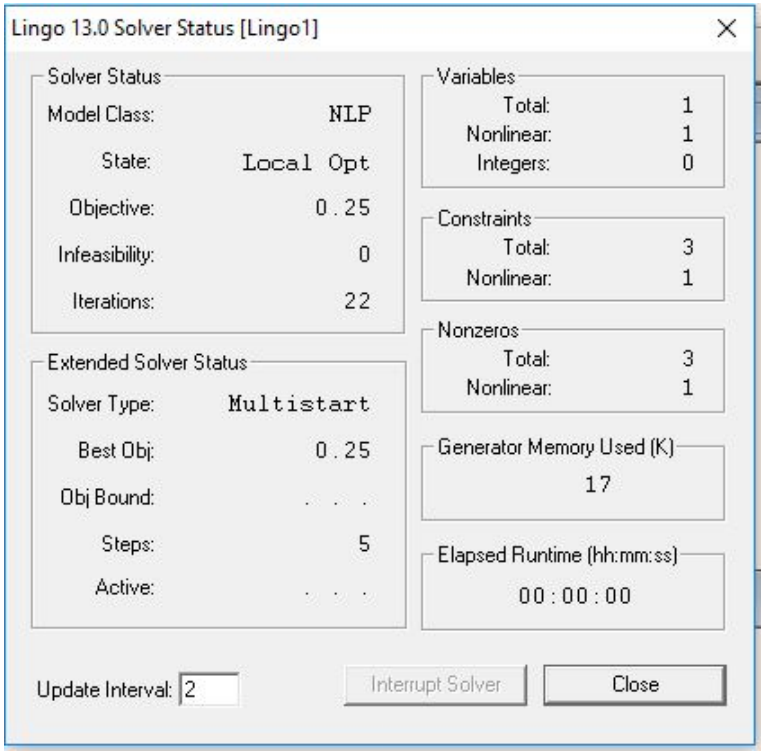
Lingo 13.0 Solver Status [Lingo1]
Solver Status
Model Class: NLP
State: Local Opt
Objective: 0.25
Infeasibility: 0
Iterations: 22
Extended Solver Status
Solver Type: Multistart
Best Obj: 0.25
Obj Bound: . . .
Steps: 5
Active: . . .
Variables
Total: 1
Nonlinear: 1
Integers: 0
Constraints
Total: 3
Nonlinear: 1
Nonzeros
Total: 3
Nonlinear: 1
Generator Memory Used (K)
17
Elapsed Runtime (hh:mm:ss)
00:00:00
Update Interval: 2
Interrupt Solver
Close

FIG. 4.7: -1

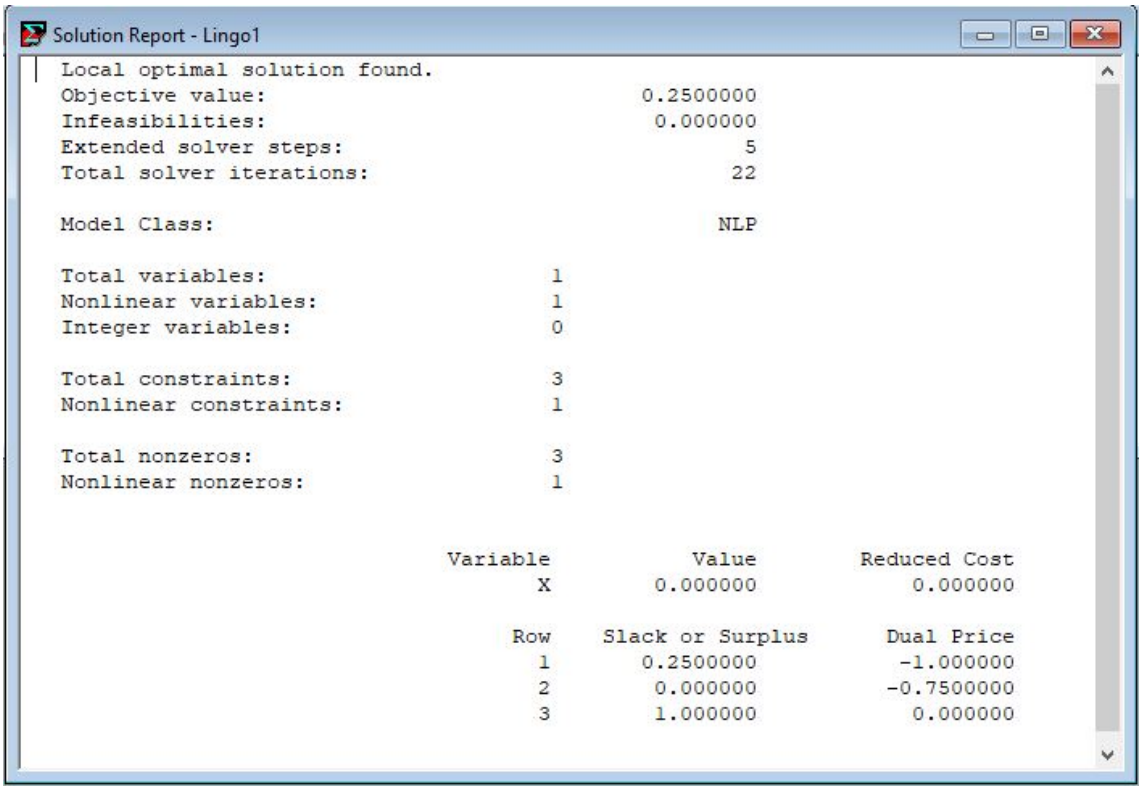

FIG. 4.8: -

4.3 Conclusion

On a programmé 02 exemple de problème d'optimisation globale à une seule variable sur le logiciel LINGO, et on a trouvé pour chacun de ces problèmes le minimum globale et sa valeur minimale en ce point.

Conclusion générale

Ce travail présente une vision globale sur l'optimisation non linéaire qui englobe un sujet très intéressant qui est la résolution d'un problème d'optimisation non linéaire non convexe on utilisant la méthode branch and bound.

La méthode branch and bound a été utilisée dans plusieurs domaines d'optimisation, comme l'optimisation combinatoire, l'optimisation semi-infini et l'optimisation quadratique ainsi que l'optimisation globale.

Le travail réalisé dans le cadre de ce mémoire est la résulution d'un problème d'optimisation non linéaire on utilisant la méthode banch and bound, on expliquant en détail cette méthode.

Nous avons donc, dans le premier chapitre, dressé un aperçu général sur les notions d'optimisation non linéaire, dont on avait besoin tout au long de ce mémoire. Le second chapitre est consacré à l'optimisation non linéaire convexe, et les différentes méthodes qu'on utilises pour la résolution des problèmes d'optimisation. Dans le dernier chapitre, qui représente l'essentiel du travail élaboré, nous avons exposer l'algorithme général de banch and bound, et avec ce dernier on pu avoir la solution de l'exemple donné.

Bibliographie

[1] Aude Rondepierre. Méthodes Numériques pour l'optimisation Génie Mathématique et Modélisation. Toulouse, (2017-2018).

[2] B.Aizel, A.Ouazib et M. O. Bibi. Minimisation d'une forme quadratique concave sous contraintes linéaires d'inégalité. Mémoire de Master, Université de Béjaia, 2014.

[3] Charles Audet. Èlément D'optimisation. École polytechnique de montérial, 2011.

[4] Dr. Belloufi. m. Cours d'optimisation Sans Contraintes. Université Mohamed Chérif Messaadia de Souk-Ahras. 2015.

[5] Le Thi Hoai An And M. Ounes. Convex Quadratic Underestimation And Branch And Bound for Univariate Global Optimization With One Nonconveex Contraint. Rairo Operation research. 2006.

[6] M. Mansour . F.Zaidi, Algorithme simpliciale Branch and Bound pour la minimisation concave. Mémoire de Master. Tizi Ouzou,(2012).

[7] M. Minaux. Programmation mathématique . Dunod, (1983).

[8] M. SACI F. Sur la programmation Quadratique convexe. Mémoire de Master, Université de béjaia, 2014/2015.

[9] M. Stéphane, M. Elbagdaoui. Optimisation non linéaire. Université de Technologie de compiègne. (2000).

[10] N. Ilkhaneche. Méthode de support pour la minimisation d'une fonctionnelle quadratique convexe. Mémoire de Magister. Département de Recherche Opérationnelle, Université de Béjaia, 2004.