République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université Mohammed Seddik Benyahai - Jijel

Faculté des Sciences Exactes et Informatique
Département de Mathématiques

Mémoire de fin d'études

Présenté pour l'obtention du diplôme de

Master

Spécialité : Mathématiques

Option : Équations aux dérivées partielles et applications

Thème

Méthode de Résolution d'un Problème Quadratique Convexe à Variables Bornées

---

Pésenté par :

M$^{\text{elle}}$ Chihab Ikram

Devant le jury :

Présidente	: Maarouf Sarra	M.C.A Université de Jijel
Encadreur	: Djemai Samia	M.C.B Université de Jijel
Examinatrice	: Mesdoui Fatiha	M.C.B Université de Jijel

Promotion **2022/2023**

Soutenu le **25/06/2023**

Remerciements

*Tout d'abord et avant tout, je remercie **ALLAH** tout puissant pour la volonté, la santé, et la patience qu'il m'a données durant ces longues années d'étude et le courage pour terminer ce mémoire.*

*J'exprime mon profond remerciement à mon encadreur **Mme Djemai Samia** pour la pertinence de ses remarques et ses conseils durant la réalisation de ce travail. J'admire beaucoup son sérieux et sa manière de diriger qui furent pour moi une grande source d'inspiration et de motivation. J'ai pris un grand plaisir à travailler avec lui.*

*Je remercie vivement **Mme Maarouf Sarra** pour avoir accepté de présider le jury de soutenance.*

*J'adresse également mes vifs remerciements à l'examinatrice **Mme Mesdoui Fatiha** pour avoir accepté d'être membre de jury.*

Je voudrais aussi remercier tous mes enseignants du département de Mathématiques à l'université de Jijel.

Enfin, je n'oubliera pas de remercier profondément ma famille, mes amis, mes collègues de promotion pour leur soutien et leurs encouragements et tous les personnes qui ont participé de près ou de loin, à terminer ce travail.

IKRAM

Dédicace

Je dédie ce modeste travail à :

♣ À mes très chers parents pour ses motivations et ses encouragements.
♣ À mes roses de ma vie : *kawther, Loubna, Asma, Rouya, Sydra.*
♣ À mon frère : *Oussama.*
♣ À mon fiancé *Rabah* et sa famille.
♣ À ma meilleure amie *Sana* et tous mes amies pour leurs aide.

IKRAM

Table des matières

Introduction Générale

La programmation mathématique est une branche de l'optimisation qui s'occupe de la minimisation ou de la maximisation sous contraintes d'une fonction à plusieurs variables, ainsi que la conception et la mise en œuvre des algorithmes de résolution.

La programmation quadratique est le plus simple problème d'optimisation non linéaire sous contraintes. Théoriquement, elle présente un aspect très intéressant, car elle est intermédiaire entre l'optimisation linéaire et non linéaire. En effet, les méthodes développées pour l'optimisation quadratique sont une extension directe de celles conçues pour le cas linéaire. De plus, les modèles quadratiques sont utilisés comme sous-problèmes à résoudre à chaque itération pour le calcul d'une direction admissible dans les méthodes de programmation non linéaire, dites méthodes de programmation séquentielle [26].

La résolution numérique d'un problème quadratique nécessite l'utilisation des méthodes spécifiques. La plupart de ces méthodes sont itératives et génèrent une suite de vecteurs x_i. Les itérations successives sont construites de façon à faire converger cette suite vers la solution optimale du problème x^*.

Plusieurs méthodes ont été développées pour la résolution de ce type de problèmes. Barankin et Dorfman [3] furent historiquement les premiers à remarquer qu'en combinant les conditions d'optimalité de Lagrange avec celle du système original, la solution optimale était une solution de base d'un système élargi ayant la propriété que seuls certains couples de variable figuraient dans l'ensemble de base. De son côté, Markowitz [22] montra qu'il était possible de modifier le système élargi et d'engendrer paramétriquement une classe de solutions de base ayant la propriété particulière ci-dessus et convergeant vers l'optimum en un nombre fini d'itérations. Enfin, Wolfe [33] montra qu'en modifiant légèrement l'algorithme du simplexe [9] de façon à ne pas autoriser l'introduction d'une variable dans l'ensemble de base si sa variable complémentaire s'y trouvait déjà, on parvenait aisément à l'optimum recherché. Ainsi, en modifiant seulement quelques instructions de la méthode du simplexe, il a été possible de résoudre un programme quadratique convexe.

D'autres approches ont été proposées pour résoudre ce genre du problème telles que :

les méthodes d'activation des contraintes[15], les méthodes de points intérieurs [25], la méthode du gradient avec projection[20], ainsi que la méthode adaptée de support de R.Gabasov et F.M.Kirolova [17, 16], qui font l'objet de notre travail.

La méthode adaptée de support [19, 17, 18] est une méthode intermédiaire entre les méthodes d'activation des contraintes et celles de points intérieurs. Elle appartient à la classe des méthodes primales-duales, basées sur des informations du problème primal et et d'autres du problème dual. Elle traitent les problèmes linéaires et quadratiques tels qu'ils se présentent et utilisent un critère d'arrêt de suboptimalité.
Le principe de cette méthode est simple : partant d'une solution réalisable de support initiale, chaque itération consiste à trouver une direction d'amélioration et un pas maximal le long de cette direction en améliorant la valeur de la fonction objectif tout en veillant à ne pas sortir du domaine réalisable déterminé par les contraintes du problème.

L'objectif de ce mémoire consiste à résoudre un problème quadratique convexe à variables bornées par la méthode adaptée. L'avantage de cette méthode par rapport à la méthode de simplexe réside dans l'utilisation d'une métrique adaptée qui change tous les indices non-optimaux à la fois, généralisant ainsi la méthode du simplexe. En effet, la méthode adaptée est basée sur la notion de support qui est un concept plus général que le concept de base en programmation linéaire. Contrairement à une solution réalisable de base, la solution réalisable et le support dans la méthode adaptée peuvent être construits indépendamment l'un de l'autre. Il est à noter qu'une solution réalisable de support peut être un point extrême, un point frontière ou un point intérieur, alors qu'une solution réalisable de base est toujours un point extrême.

Ce mémoire est organisé comme suit :

Le premier chapitre présente quelques rappels mathématiques sur l'algèbre matricielle, des propriétés des formes quadratiques, et des éléments d'analyse convexe.

Le deuxième chapitre traite la programmation quadratique convexe (P.Q.C.) sous contraintes, ainsi que quelques définitions et des théorèmes sur les conditions d'optimalité pour un P.Q.C. sous contraintes d'égalité, et enfin la notion de la dualité.

Le troisième chapitre présente la méthode adaptée pour la résolution d'un problème de la programmation quadratique convexe à variables bornées, et un application numérique sur cette méthode.

Le quatrième chapitre consacre l'implémentation de la méthode adaptée sous le langage MATLAB, exécutée sur des exemples générés aléatoirement, ainsi qu'une application simple de la méthode adaptée sur le problème des machines à vecteurs de support (SVM).

Enfin, ce mémoire se termine par une conclusion générale et des perspectives.

Chapitre 1

Rappels sur l'algèbre linéaire et l'analyse convexe

1.1 Introduction

Dans ce chapitre, nous allons mentionner quelques définitions et propriétés de l'algèbre matricielle, des formes quadratiques et des notions importantes sur les ensembles et les fonctions convexes.

1.2 Algèbre matricielle

1.2.1 Vecteurs et matrices

Définition 1.1. *Soit $n, m \in \mathbb{N}^*$. Une matrice d'ordre $m \times n$ à coefficients dans $\mathbb{R}$ est un tableau à deux dimensions, ayant m lignes et n colonnes, écrit sous la forme suivante :*

$$A = A(I, J) = (a_{ij}, i \in I, j \in J) = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix},$$

avec $I = \{1, 2, ..., m\}$ et $J = \{1, 2, ..., n\}$ représentent respectivement l'ensemble des indices des lignes et colonnes de A. La matrice A, pour des calculs pratiques, se note aussi :

$$A = (a_1, a_2, ..., a_j, ..., a_n) = \begin{pmatrix} A_1' \\ A_2' \\ \vdots \\ A_j' \\ \vdots \\ A_m' \end{pmatrix},$$

où $a_j = A(I,j) = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix}$ *est un vecteur-colonne de dimension* m, $A_i^{'} = A(i,J) = (a_{i1}, a_{i2}, ..., a_{in})$ *est un vecteur-ligne de dimension* n.
Le symbole $(')$ *est celui de la transposition.*
Chaque vecteur, noté $x = x(J) = (x_j, j \in J)$*, sera ainsi considéré comme un vecteur colonne, tandis que le vecteur ligne sera noté* $x^{'}$.

- *La matrice transposée de* A *sera notée :*

$$A^{'} = A^{'}(J,I) = (a_{ji}, j \in J, i \in I).$$

- *Un vecteur colonne de dimension* n *peut être considéré comme une matrice d'ordre* $(n \times 1)$*, tandis qu'un vecteur ligne de dimension* n *peut être considéré comme une matrice d'ordre* $(1 \times n)$.

- *On dit que la matrice* A *est carrée si* $n = m$ *; de plus, si* $A = A^{'}$*, la matrice est dite symétrique.*

- *La matrice d'identité d'ordre* n *sera notée* I_n.

1.2.2 Matrices et vecteurs partitionnés

On peut effectuer le produit d'une matrice A et d'un vecteur x après les avoir partitionnés. Alors, on dit qu'on a effectué le produit par blocs. En effet, si l'on a :

$$A = [A_1|A_2],\ x = \left[\tfrac{x_1}{x_2}\right].$$

Alors, on peut écrire :

$$Ax = [A_1|A_2]\left[\tfrac{x_1}{x_2}\right] = A_1x_1 + A_2x_2.$$

De même, pour

$$\text{A}=\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},\ x = \left[\tfrac{x_1}{x_2}\right],\ b = \left[\tfrac{b_1}{b_2}\right],$$

l'équation $Ax = b$ peut alors s'écrire :

$$\begin{cases} A_{11}x_1 + A_{12}x_2 = b_1, \\ A_{21}x_1 + A_{22}x_2 = b_2. \end{cases}$$

Vous pouvez diviser la matrice au hasard. Par exemple, si $A = A(I,J)$ est une matrice d'ordre ($m \times n$), J_1 et J_2 sont deux sous-ensembles quelconques de J, tels que :

$$|J_1| = m,\ J_1 \cup J_2 = J,\ J_1 \cap J_2 = \emptyset.$$

Alors, on peut diviser A comme suit :

$$A = (a_1, a_2, ..., a_j, ..., a_n) = [A_1|A_2],$$

où $A_1 = A(I, J_1)$, et $A_2 = A(I, J_2)$.
Si $x = x(J) = \left[\frac{x_1}{x_2}\right]$, avec $x_1 = x(J_1)$, $x_2 = x(J_2)$. Donc, on obtient :

$$\begin{aligned} Ax = \sum_{j=1}^{n} a_j x_j = \sum_{j \in J_1} a_j x_j + \sum_{j \in J_2} a_j x_j &= A(I, J_1)x(J_1) + A(I, J_2)x(J_2) \\ &= A_1 x_1 + A_2 x_2. \end{aligned}$$

1.2.3 Noyau et image d'une matrice

Définition 1.2. *Soit A une matrice d'ordre $m \times n$.*
Le noyau de la matrice A est définie par l'ensemble :

$$N(A) = \{x \in \mathbb{R}^n, Ax = 0\}.$$

L'image (l'espace image) de la matrice A est définie par l'ensemble :

$$Im(A) = \{y \in \mathbb{R}^m : \exists x \in \mathbb{R}^n, Ax = y\}.$$

1.2.4 Rang d'une matrice

Définition 1.3. *On appelle rang d'une matrice A, noté $rg(A)$, la dimension de l'image de la matrice A :*

$$rg(A) = dim(Im(A)).$$

1.2.5 L'inverse d'une matrice

Définition 1.4. *Soit A une matrice d'ordre n. On dit que A est inversible ou régulière s'il existe une unique matrice notée A^{-1}, tel que $AA^{-1} = A^{-1}A = I_n$. A^{-1} est alors appelée la matrice inverse de A.*

Définition 1.5. *Une matrice carrée A est inversible si son déterminant est non nul, sinon elle est dite singulière.*

1.3 Les formes quadratiques

Définition 1.6. *Une fonction $F(x) : \mathbb{R}^n \to \mathbb{R}$ est dite forme quadratique de n variables $x_1, x_2, ..., x_n$ si elle s'écrit de la manière suivante :*

$$F(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j = x' A x, \tag{1.1}$$

avec $x = (x_1, x_2, ..., x_n)$*' est n-vecteur colonne ;* $A = (a_{ij}, 1 \leqslant i, j \leqslant n)$ *est une matrice carrée d'ordre n.*
Pour $i \neq j$*, le coefficient du terme* $x_i x_j$ *s'écrit* $a_{ij} + a_{ji}$*. En vertu de cela, la matrice A peut être supposée symétrique. En effet, en définissant de nouveaux coefficients :*

$$d_{ij} = d_{ji} = \frac{a_{ij} + a_{ji}}{2}, 1 \leq i, j \leq n.$$

On trouve une nouvelle matrice $D = (d_{ij}, 1 \leq i, j \geq n)$ *symétrique. Il est clair qu'après une redéfinition des coefficients, la valeur de la forme quadratique* $F(x)$ *reste inchangée :*

$$F(x) = x'Ax = x'Dx, \quad \forall x \in \mathbb{R}^n$$

Pour cela, il est naturel de considérer que la matrice d'une forme quadratique est toujours symétrique.

Exemple 1.1. *Soit la forme quadratique suivante :*

$$F(x) = 3x_1^2 + x_2^2 - 4x_1x_2 + 4x_2x_3.$$

La matrice symétrique associée à $F(x)$ *est :*

$$D = \begin{pmatrix} 3 & -2 & 0 \\ -2 & 1 & 2 \\ 0 & 2 & 0 \end{pmatrix}$$

1.3.1 Gradient et Hessien d'une forme quadratique

Définition 1.7. *Soit une fonction de classe* C^1*,* $F : \mathbb{R}^n \to \mathbb{R}$*. Le gradient de la fonction F est défini par :*

$$g(x) = \nabla F(x) = \begin{bmatrix} \frac{\partial F}{\partial x_1} \\ \frac{\partial F}{\partial x_2} \\ . \\ . \\ . \\ \frac{\partial F}{\partial x_n} \end{bmatrix} = 2Dx.$$

Avec $\frac{\partial F}{\partial x_i}$ *est la dérivée partielle de F par rapport à* x_i*,* $i = 1, 2, ..., n$*.*

Définition 1.8. *Soit une fonction de classe* C^2*,* $F : \mathbb{R}^n \to \mathbb{R}$*. Le hessien de la fonction F est défini par :*

$$H(x) = \nabla^2 F(x) = \begin{bmatrix} \frac{\partial^2 F}{\partial^2 x_1} & \frac{\partial^2 F}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 F}{\partial x_1 \partial x_n} \\ \frac{\partial^2 F}{\partial x_2 \partial x_1} & \frac{\partial^2 F}{\partial^2 x_2} & \cdots & \frac{\partial^2 F}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 F}{\partial x_n \partial x_1} & \frac{\partial^2 F}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 F}{\partial^2 x_n} \end{bmatrix}$$

Définition 1.9. *[8] Soit $F : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction de classe C^1. La dérivée directionnelle de F dans la direction d au point x est :*

$$\begin{aligned}\frac{\partial F}{\partial d}(x) &= \lim_{t \longrightarrow 0} \frac{F(x+td)-F(x)}{t}, \\ &= \frac{\partial F}{\partial x_1}(x+td)\mid_{t=0} d_1 + \cdots + \frac{\partial F}{\partial x_n}(x+td)\mid_{t=0} d_n, \\ &= d'\nabla F(x).\end{aligned}$$

1.3.2 Formes quadratiques définies et semi-définies positives

Soit la forme quadratique $F(x) = x'Dx$ [28].

Définition 1.10.

- *On dit que $F(x)$ est définie positive si $x'Dx > 0$, $\forall x \in \mathbb{R}^n$ et $x \neq 0$. Elle est dite semi-définie positive ou définie non négative si $x'Dx \geq 0$, $\forall x \in \mathbb{R}^n$.*
- *On dit que $F(x)$ est définie négative si $x'Dx < 0$, $\forall x \in \mathbb{R}^n$ et $x \neq 0$. Elle est dite semi-définie négative ou définie non positive si $x'Dx \leq 0$, $\forall x \in \mathbb{R}^n$.*

Définition 1.11. *Soit D une matrice symétrique $n \times n$. On dit que D est semi-définie positive ou définie négative et on note $D \geq 0$, lorsque :*

$$xDx' \geq 0, \forall \in \mathbb{R}^n \quad et \quad x \neq 0.$$

On dit que D est définie positive et on note $D > 0$, lorsque :

$$xDx' > 0, \forall \in \mathbb{R}^n \quad et \quad x \neq 0.$$

1.3.3 Propriétés des formes quadratiques définies et semi-définies positives

Les matrices symétrique définies ont des propriétés très intéressantes. En voici quelques unes :

Propriétés 1.12. *Un élément de la diagonale d'une matrice symétrique D définie non négative ne peut s'annuler que si les autres éléments de la même ligne et colonne s'annulent aussi.*

Propriétés 1.13. *Soit D une matrice symétrique définie non négative. Si x est un point quelconque mais fixé de $\mathbb{R}^n$ tel que $x'Dx = 0$, alors nous avons $Dx = 0$.*

Propriétés 1.14. *Soit la matrice D par blocs de la manière suivante :*

$$D = \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix}$$

Si $D > 0$ $(D \geq 0)$, alors les sous matrices principales D_{11} et D_{22} sont aussi définies positives (non négatives).
On générale, toute sous matrice principale d'une matrice définie positive (non négative) est définie positive (non négative).

Propriétés 1.15. *Soit D matrices symétrique $n \times n$. On note $\{\lambda_i\}_{i=1,\ldots,n}$ ses valeurs propres réelles. On a les équivalences suivantes :*

- $D \geq 0 \Leftrightarrow \lambda_i \geq 0,\ i = 1, \ldots, n.$
- $D > 0 \Leftrightarrow \lambda_i > 0,\ i = 1, \ldots, n.$
- *D est indéfinie si et seulement si certains λ_i sont positifs et d'autres négatifs.*

Définition 1.16. *On dit que $F(x)$ est une forme quadratique définie positive (resp. semi-définie positive), si la matrice D est définie positive (resp. semi-définie positive).*

1.4 Éléments d'analyse convexe

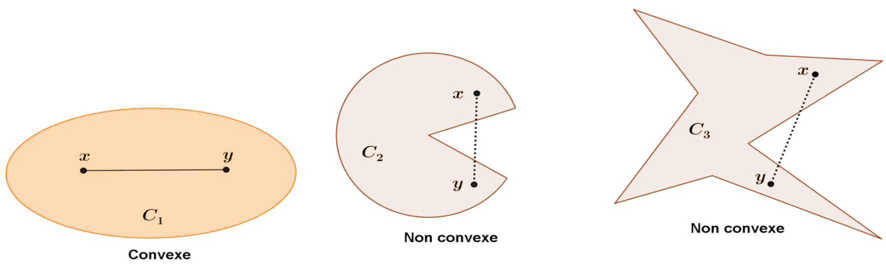
La convexité dans les problèmes de minimisation avec ou sans contraintes joue un rôle très important. Elle représente un outil indispensable pour la recherche des conditions à la fois nécessaires et suffisantes d'optimalité.

1.4.1 Ensembles convexes

On dit que un ensemble $C \subset \mathbb{R}^n$ est convexe si et seulement si pour deux points quelconques x et y on a :

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

Géométriquement, pour tout segment reliant deux points quelconques x et y de C, le segment $[x, y]$ doit être aussi dans C.



Exemple 1.2. *Soit $\mathbb{R}^n$ est convexe. On a les ensembles suivants :*
- $X = \{(x, y) \in \mathbb{R}^n, x^2 + y^2 \leq 1\}$ *est un ensemble convexe.*
- $Y = \{(x, y) \in \mathbb{R}^n, x^2 + y^2 \geq 1\}$ *est un ensemble non convexe.*

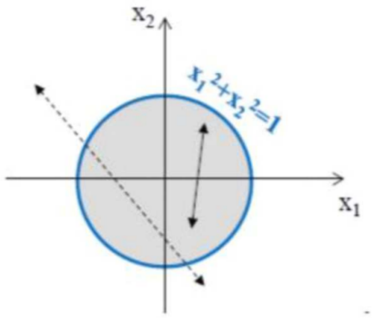


FIGURE 1.1 – Illustration d'ensemble convexe.

Définition 1.17. *Un ensemble de points C de $\mathbb{R}^n$ est convexe si et seulement si toute combinaison convexe de points de l'ensemble est aussi un point de l'ensemble.*

Théorème 1.18. *Soit C_1 et C_2 deux ensembles convexes de $\mathbb{R}^n$. Alors :*
- $C_1 \cap C_2$ *est convexe ;*
- $C_1 \pm C_2 = \{x_1 \pm x_2 | x_1 \in C_1, x_2 \in C_2\}$ *est convexe.*

Définition 1.19. *On dit que un ensemble C est un cône si $x \in C$ implique que $\lambda x \in C$ pour tout $\lambda > 0$. Si, de plus, C est convexe, alors C est appelé cône convexe.*

1.4.2 Fonctions convexes

Soit $C \subset \mathbb{R}^n$ un ensemble convexe et $F : C \longrightarrow \mathbb{R}$ une fonction.

Définition 1.20. *La fonction F est dite convexe si elle vérifie :*

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

Définition 1.21. *La fonction F est dite strictement convexe si l'inégalité stricte suivante est vérifiée :*

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

Définition 1.22. *La fonction F est dite concave si l'inégalité suivante est vérifiée :*

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

C'est à dire F est concave si seulement si $-F$ est convexe.

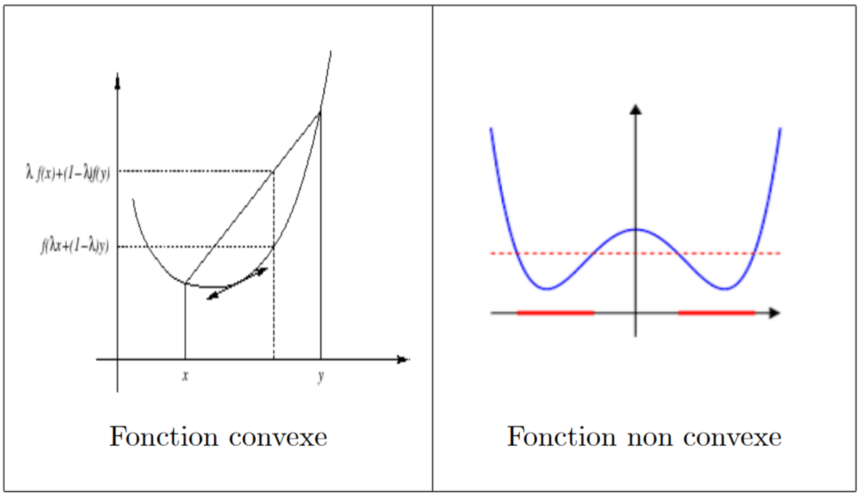


FIGURE 1.2 – Illustration de la convexité des fonctions.

Définition 1.23. *Soit $C \subset \mathbb{R}^n$ un ensemble non vide. L'ensemble*

$$\{(x, F(x)) : x \in C\} \subset \mathbb{R}^{n+1},$$

est appelée le graphe de la fonction F.

Propriétés 1.24. *Soit F une fonction réelle définie sur un ensemble convexe $C \subset \mathbb{R}^n$, F est convexe si et seulement si son épigraphe, noté $epi(F)$, est un convexe :*

$$epi(F) = \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R} : x \in C, F(x) \leq \alpha\}.$$

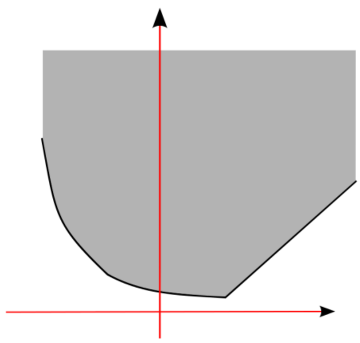


FIGURE 1.3 – L'épigraphe de la fonction est la zone grisée au-dessus du graphe de la fonction (en noir).

Démonstration. Supposons que F est convexe.
Soient $(x_1, \alpha), (x_2, \beta) \in epi(F)$.
D'après les définitions de l'épigraphe et de la fonction convexe, on a :

$$F(\lambda x_1 + (1-\lambda)x_2) \leq \lambda F(x_1) + (1-\lambda)F(x_2) \leq \lambda\alpha + (1-\lambda)\beta, \quad \forall \lambda \in [0, 1].$$

Comme C est un ensemble convexe, alors

$$\lambda x_1 + (1-\lambda)x_2 \in C.$$

D'où

$$(\lambda x_1 + (1-\lambda)x_2, \lambda\alpha + (1-\lambda)\beta) \in epi(F).$$

Donc $epi(F)$ est convexe.
Réciproquement, supposons que $epi(F)$ est convexe.
Soient $(x_1, F(x_1)), (x_2, F(x_2)) \in epi(F)$.
Alors, d'après la convexité de $epi(F)$, on a :

$$(\lambda x_1 + (1-\lambda)x_2, \lambda F(x_1) + (1-\lambda)F(x_2)) \in epi(F), \quad \forall \lambda \in [0,1].$$

Donc

$$F(\lambda x_1 + (1-\lambda)x_2) \leq \lambda F(x_1) + (1-\lambda)F(x_2), \quad \forall \lambda \in [0,1].$$

D'où F est convexe. □

Propriétés 1.25. *Soit F une fonction réelle définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors F est convexe si et seulement si :*

$$F\big(\sum_{i=1}^{k} \lambda_i x_i\big) \leq \sum_{i=1}^{k} \lambda_i F(x_i),$$

avec $x_i \in C, i = 1, \dots, k, \lambda \geq 0, \quad \sum_{i=1}^{k} \lambda_i = 1.$

Démonstration. On raison par récurrence, pour $k = 2$ on a :

$$F(\lambda_1 x_1 + \lambda_2 x_2) \leq \lambda_1 F(x_1) + \lambda_2 F(x_2),$$

avec $\lambda_1 + \lambda_2 = 1$ est vrai.
Supposons que la propriété est vrai pour k et on la démontre pour $(k+1)$ alors

$$F\big(\sum_{i=1}^{k+1} \lambda_i x_i\big) \leq \sum_{i=1}^{k+1} \lambda_i F(x_i)$$

avec $\sum_{i=1}^{k+1} \lambda_i = 1$. On a :

$$\begin{aligned} F(\sum_{i=1}^{k+1} \lambda_i x_i) &= F(\sum_{i=1}^{k} \lambda_i x_i + \lambda_{k+1} x_{k+1}) \\ &= F((1-\lambda_{k+1}) \sum_{i=1}^{k} \frac{\lambda_i}{1-\lambda_{k+1}} x_i + \lambda_{k+1} x_{k+1}) \end{aligned}$$

comme F est convexe, alors :

$$\begin{aligned}
F((1-\lambda_{k+1})\sum_{i=1}^{k}\frac{\lambda_i}{1-\lambda_{k+1}}x_i+\lambda_{k+1}x_{k+1}) &\leq (1-\lambda_{k+1})F(\sum_{i=1}^{k}\frac{\lambda_i}{1-\lambda_{k+1}}x_i)+\lambda_{k+1}F(x_{k+1})\\
&\leq (1-\lambda_{k+1})\sum_{i=1}^{k}\frac{\lambda_i}{1-\lambda_{k+1}}F(x_i)+\lambda_{k+1}F(x_{k+1}).\\
&= \sum_{i=1}^{k}\lambda_i F(x_i)+\lambda_{k+1}F(x_{k+1}).\\
&= \sum_{i=1}^{k+1}\lambda_i F(x_i)
\end{aligned}$$

D'où

$$F(\sum_{i=1}^{k+1}\lambda_i x_i)\leq \sum_{i=1}^{k+1}\lambda_i F(x_i).$$

□

Théorème 1.26.

- *Soit F une fonction convexe sur un ensemble convexe $C \subset \mathbb{R}^n$, et un nombre réel $\alpha \geq 0$. Alors, αF est aussi une fonction convexe sur C.*
- *Soient F_1, F_2 deux fonctions convexes sur un ensemble convexe C . Alors, $F_1 + F_2$ est une fonction convexe sur C.*
- *Soient $F_1, F_1, \ldots, F_m$ des fonctions convexes sur l'ensemble convexe C et $\alpha_1, \alpha_2, \ldots, \alpha_m \geq 0$ des nombres réels. Alors, $\sum_{i=1}^{m}\alpha_i F_i$ est encore une fonction convexe sur C.*

Théorème 1.27. *Les conditions (1) et (2) ci-dessous sont équivalentes si F est continûment différentiable. De plus, les condition (1), (2) et (3) ci-dessus sont équivalentes si F est deux fois continûment différentiable :*
(1) F est convexe.
(2) $\forall x \in C$, $\forall y \in C$, $F(y) - F(x) \geq \nabla F(x)'(y - x)$.
(3) $\forall x \in C$, le hessien $\nabla^2 F(x)$ est une matrice semi-définie positive.

Remarque 1.1. *D'après le théorème précédent, on déduit facilement qu'une fonction quadratique $F(x) = x'Dx$ est convexe si et seulement si sa matrice associée D est semi-définie positive.*

1.5 Conclusion

Dans ce chapitre, nous avons vu les notions de l'algèbre matricielle, les formes quadratiques et leurs propriétés, et nous avons présenté des notions sur la convexité, avec les ensembles et les fonction convexes et leurs propriétés.

Chapitre 2

La programmation quadratique convexe (PQC)

2.1 Introduction

Dans ce chapitre, nous allons rappeler brièvement les définitions des problèmes quadratiques sous contraintes d'égalités, les conditions nécessaires et suffisantes pour l'optimalité des problèmes quadratiques convexes, et la notion de la dualité.

2.2 Programmation quadratique sous contraintes

2.2.1 Formulation du problème

Un problème de programmation quadratique sous contraintes d'égalités est formulé de la manière suivante :

$$\begin{cases} min \quad F(x), \\ g_i(x) = A_i'x - b_i = 0, \forall i = 1, ..., m. \end{cases} \tag{2.1}$$

Avec $F : \mathbb{R}^n \longrightarrow \mathbb{R}$ est un fonction deux fois continûment différentiable, elle représente la fonction objectif du problème. b est un vecteur de $\mathbb{R}^m$, A est une matrice d'ordre $m \times n$, elle se compose des vecteurs colonnes et des vecteurs lignes suivants :

$$A' = (a_1, a_2, ..., a_n) = \begin{pmatrix} A_1' \\ A_2' \\ \vdots \\ A_m' \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}, \quad g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix},$$

est une fonction vectorielle définie de $\mathbb{R}^n \longrightarrow \mathbb{R}$.
Pour que l'ensemble des solutions réalisables, noté X, ne soit pas vide ou ne soit pas réduit à un point isolé, on considère que $rg(A) = m < n$.

$$X = \{x| \quad g_i(x) = 0, \; i = 1, ..., m\}.$$

Définition 2.1. ***(Minimum local)***

- *On dit que la fonction F admet un minimum local $x^* \in X$ s'il existe un réel $\epsilon > 0$ tel que :*
$$F(x^*) \leq F(x), \quad \forall x \in X \quad \textit{avec} \quad \|x - x^*\| < \epsilon.$$
- *On dit que la fonction F admet un minimum local strict $x^* \in X$ s'il existe un réel $\epsilon > 0$ tel que :*
$$F(x^*) < F(x), \quad \forall x \in X \quad \textit{avec} \quad \|x - x^*\| < \epsilon.$$

Définition 2.2. ***(Minimum global)***

- *On dit que la fonction F admet un minimum global $x^* \in X$ si :*
$$F(x^*) \leq F(x), \quad \forall x \in X.$$
- *On dit que la fonction F admet un minimum global strict $x^* \in X$ si :*
$$F(x^*) < F(x), \quad \forall x \in X.$$

Exemple 2.1. *Soit* $F(x) = 3e^{-x^2} + e^{-(x-3)^2}$.

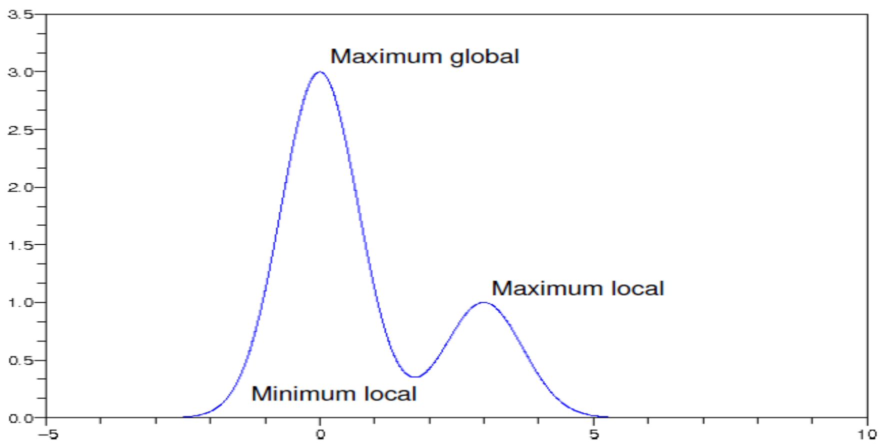


FIGURE 2.1 – Exemple de minima et maxima locaux et globaux pour la fonction F.

2.3 Conditions d'optimalité pour un problème quadratique sous contraintes d'égalités

Considérons le problème quadratique (2.1) et la fonction F de classe C^1.

Définition 2.3. *La fonction $\mathcal{L}(x, \lambda) = F(x) + \sum_{i=1}^{m} \lambda_i g_i(x)$ est dite fonction de Lagrange associée au problème* (2.1), *où le vecteur $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m) \in \mathbb{R}^m$, formé des multiplicateurs de Lagrange.*

Théorème 2.4. ***(Théorème de Lagrange)***
Soit x^* un minimum local (ou global) pour le problème (2.1). *Il existe alors un vecteur multiplicateur de Lagrange $\lambda^* \in \mathbb{R}^m$, tel que :*

$$\nabla \mathcal{L}(x^*, \lambda^*) = 0 \Leftrightarrow \begin{cases} \nabla_{\lambda^*} \mathcal{L}(x^*, \lambda^*) = 0, \\ \nabla_{x^*} \mathcal{L}(x^*, \lambda^*) = 0. \end{cases} \tag{2.2}$$

Le couple (x^*, λ^*) est appelé point stationnaire de la fonction de Lagrange.

2.3.1 Conditions d'optimalité du premier ordre

Théorème 2.5. *[21] Soit x^* un minimum du problème* (2.1), *alors il existe nécessairement un vecteur $\lambda \in \mathbb{R}^m$ vérifiant :*

$$\nabla F(x^*) + A'\lambda = 0. \tag{2.3}$$

Si, de plus, A est de rang complet en lignes, c-à-d, $rg(A) = m$, alors λ est unique, comme le montre le théorème suivant :

Théorème 2.6. *Le vecteur des multiplicateurs de Lagrange λ^* est unique si et seulement si $rg(A) = m$.*

En utilisant la fonction de Lagrange, la condition (2.3) peut être donnée comme suit :

$$\nabla_x \mathcal{L}(x, \lambda) = \nabla F(x) + A'\lambda = 0. \tag{2.4}$$

D'autre part, un minimum local est tout d'abord un point réalisable du problème (2.1), qui vérifie :

$$Ax - b = 0 \Rightarrow \nabla_\lambda \mathcal{L}(x, \lambda) = Ax - b = 0. \tag{2.5}$$

Donc, d'après les relations (2.4) et (2.5), on trouve la condition nécessaire d'optimalité du premier ordre pour le problème (2.1).

2.3.2 Conditions d'optimalité du second ordre

Considérons le problème (2.1) et, de plus, la fonction F de classe C^2.

Théorème 2.7. *[21]* ***(Condition nécessaire d'optimalité du second ordre)***
Soit x^* un minimum du problème (2.1) *et λ^* un vecteur multiplicateurs de Lagrange vérifiant* (2.5). *Alors, la matrice $\nabla^2 F(x^*)$ est semi-définie positive sur l'ensemble des points de la variété linéaire $Ay = 0$. En d'autres termes :*

$$y'\nabla^2 F(x^*)y \geq 0, \forall y \in N(A) = \{y \in \mathbb{R}^n : Ay = 0\}.$$

Théorème 2.8. ***(Condition suffisante d'optimalité du second ordre)***
Soit un couple de vecteurs (x^*, λ^*) qui satisfait la condition nécessaire d'optimalité du premier ordre du problème (2.1), *tel que :*

$$\nabla \mathcal{L}(x^*, \lambda^*) = 0.$$

Pour que x^* soit un point minimum local du problème (2.1), *il est suffisant que la matrice $\nabla^2 F(x^*)$ soit semi-définie positive sur le sous-espace vectoriel $N(A)$.*

2.4 Programmation convexe

L'hypothèse de la convexité apporte l'élégance et la simplicité à la théorie de l'optimisation. En particulier, les conditions nécessaires d'optimalité deviennent également suffisantes.
Un problème de programmation mathématique est convexe(respectivement strictement convexe), s'il consiste à minimiser une fonction convexe (respectivement 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'hypothèse de la convexité est cruciale en optimisation. Notons que :

- Les problèmes convexes sont synonymes de minimisation.
- Les problèmes convexes sont les meilleurs problèmes de la théorie : ceux pour lesquels il existe des algorithmes de résolution efficaces.
- L'hypothèse de convexité ne garantit cependant ni l'existence ni l'unicité d'une éventuelle solution.

Définition 2.9. *Le problème* (2.1) *est dit convexe si la fonction objectif F est convexe.*

Pour tous les problèmes de programmation convexe, on a les propriétés suivantes [4, 23] :

Propriétés 2.10. *Soit F une fonction convexe définie sur un ensemble convexe $C \subset \mathbb{R}^n$. Alors, l'ensemble des points où F atteint son minimum est convexe.*

Propriétés 2.11. *Soit $F : C \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction convexe. Alors tout minimum local est un minimum global.*

Propriétés 2.12. *Tout problème strictement convexe admet au plus une solution.*

Propriétés 2.13. *Si la fonction F est strictement convexe, alors son minimum global, lorsque qu'il existe, est atteint en un seul point x^*.*

Considérons maintenant le problème avec contraintes donnée sous la forme (2.1), où F est une fonction convexe de classe C^1.

Théorème 2.14. ***(Théorème de Karush-Kuhn-Tucker 1951)****[1] Si x^* est un minimum local de F sur X, alors il existe $\lambda^* = (\lambda_1^*, \lambda_2^*, ..., \lambda_m^*) \in \mathbb{R}^m$ tel que :*

- $\nabla F(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) = 0$ *(condition de stationnarité) ;*

Théorème 2.15. *[24] Soient F une fonction convexe et un couple (x^*, λ^*) de vecteurs vérifiant les conditions de Karush-Kuhn-Tucker suivante :*

- $\nabla \mathcal{L}_x(x^*, \lambda^*) = 0, \lambda^* \in \mathbb{R}^m$,

Alors, le vecteur x^* constitue un minimum global de F sur X.

Ainsi, les conditions de KKT sont donc à la fois nécessaires et suffisantes de minimalité (c'est le théorème de KKT-convexe).

2.5 Dualité en programmation quadratique convexe

Dans nombreux cas, il est plus facile de traiter la forme duale de notre problème que son primal, ainsi les variable duales ont des interprétation très utiles, par exemples : courant/tension dans les réseaux électriques, consommation/prix dans les modèles économiques, tensions/ déplacements dans la statique... etc.
On rappelle tout d'abord quelques définitions.

Définition 2.16. ***(Problème Primal)***
Le problème primal est un problème de minimisation qui consiste à trouver un vecteur x^*, s'il existe, tel que

$$\begin{cases} F(x^*) = min\ F(x), \\ x^* \in X = \{x \in \mathbb{R}^n | g_i(x) = 0, i = 1, ..., m\}, \end{cases} \tag{2.6}$$

avec la fonction F est convexe sur l'ensemble C.

Définition 2.17. ***(Problème dual)***
Le problème dual du (2.6) est défini comme un problème de maximisation, qui consiste à trouver deux vecteurs $\kappa^* \in \mathbb{R}^n$ et $\lambda^* \in \mathbb{R}^m$, s'ils existent, tel que

$$\begin{cases} \mathcal{L}(\kappa^*, \lambda^*) = max \quad \mathcal{L}(\kappa, \lambda), \\ (\kappa^*, \lambda^*) \in V = \{(\kappa, \lambda) \in \mathbb{R}^n \times \mathbb{R}^m / \nabla_\kappa \mathcal{L}(\kappa, \lambda) = 0, \quad \lambda \in \mathbb{R}^m\}. \end{cases} \tag{2.7}$$

Avec $\mathcal{L}(\kappa, \lambda) = F(x) + y'g(x)$ est la fonction de Lagrange associée au problème (2.6).

Les relations existantes entre le programme primal et son dual sont données par les deux théorèmes suivants [32] :

Théorème 2.18. ***(Théorème faible de la dualité, Wolfe 1961)***
Soient $F(x)$ et $\mathcal{L}(\kappa, \lambda)$ les fonctions objectifs du programme primal et de son dual respectivement. Nous avons alors

$$\mathcal{L}(\kappa, \lambda) \leq F(x), \quad \forall (\kappa, \lambda) \in V, \forall x \in X.$$

Théorème 2.19. ***(Théorème fort de la dualité, Wolfe 1961)***
Soit x^* une solution optimale du problème primal (2.6). Alors il existe $\lambda^* \in \mathbb{R}^m$, tel que (κ^*, λ^*) est une solution optimale du dual (2.7) et on a

$$F(x^*) = \mathcal{L}(\kappa^*, \lambda^*).$$

2.6 Domaines d'application de P.Q.C.

L'application de la programmation quadratique convexe est de plus en plus en expansion croissante et on trouve beaucoup d'applications dans plusieurs domaines, on peut

citer :
• La gestion des portefeuilles [22].
• L'analyse structurelle [14].
• Les machines à vecteurs de supports (SVM) [29, 27, 7].
• Le contrôl optimal [14].
• L'économie mathématique [2, 10].

2.7 Conclusion

Dans ce chapitre, nous avons présenté les conditions d'optimalité pour un problème quadratique convexe sous contraintes d'égalité, la notion de la dualité, et les applications de la programmation quadratique convexe.

Chapitre 3

Méthode adaptée pour la résolution d'un problème quadratique convexe à variables bornées

3.1 Introduction

Dans ce chapitre, on va présenter la méthode adaptée de Gabassov [19, 17, 18, 5] pour la minimisation d'un programme quadratique convexe sous la forme standard, où la matrice de la forme quadratique est semi-définie positive, et les variables du problème sont bornées.

3.2 Position du problème

Le problème de programmation quadratique convexe à variables bornées se présente sous la forme standard suivante :

$$min \quad F(x) = \frac{1}{2}x'Dx + c'x, \tag{3.1}$$

$$Ax = b, \tag{3.2}$$

$$l \leq x \leq u. \tag{3.3}$$

Où D est une matrice carrée symétrique d'ordre n et semi-définie positive, b est un m-vecteur, A est une matrice d'ordre $m \times n$, avec $A = A(I, J)$ et $rg(A) = m < n$; c, x, l et u sont des n-vecteurs.
$I = \{1, 2, ..., m\}$: l'ensemble des indice des linges de A.
$J = \{1, 2, ..., n\}$: l'ensemble des indice des colonnes de A.
Soit les partitions suivantes :

$$J = \{1, 2, ..., n\} = J_B \cup J_N \quad \text{avec} \quad J_B \cap J_N = \emptyset, |J_B| = m,$$
$$J_N = J_S \cup J_{NN}, J_S \cap J_{NN} = \emptyset.$$

Nous pouvons alors écrire et fractionner les vecteurs et la matrice A comme suit :
$x = x(J) = (x_j, j \in J)$,
$x = (\frac{x_B}{x_N}), x_B = x(J_B) = (x_j, j \in J_B), x_N = x(J_N) = (x_j, j \in J_N)$,
$c = (\frac{c_B}{c_N}), c_B = c(J_B) = (c_j, j \in J_B), c_N = c(J_N) = (c_j, j \in J_N)$,

$$A = A(I, J) = (a_{ij}, 1 \leq i \leq m, 1 \leq j \leq n); A = (a_j, j \in J), a_j = \begin{pmatrix} a_{1j} \\ \cdot \\ \cdot \\ \cdot \\ a_{mj} \end{pmatrix},$$

$A = (A_B|A_N), A_B = (I, J_B), A_N = (I, J_N)$.

Les conditions de Kuhn-Kuhn-Tucker (KKT) [1] sont des conditions nécessaires d'optimalité, valables dans le cadre général de l'optimisation non linéaire sous contraintes avec une fonction objectif différentiable. Elles s'appliquent donc bien sûr à la programmation quadratique et elles sont même suffisantes grâce à la propriété de convexité. Ici, on abordera le problème $(3.1) - (3.3)$ par une approche constructive.

3.3 Définitions

Définition 3.1. ***(Solution réalisable)***
Un n- vecteur x est dit solution réalisable du problème $(3.1)-(3.3)$ *s'il vérifie les contraintes* $(3.2) - (3.3)$.

Définition 3.2. ***(Solution optimale)***
Une solution x^* est dite optimale si :

$$F(x^*) = \frac{1}{2}(x^*)'Dx^* + c'x^* = min\ \frac{1}{2}x'Dx + c'x,$$

où x est pris parmi tous les vecteurs vérifiant les contraintes $(3.2) - (3.3)$.

Définition 3.3. ***(Solution suboptimale)***
Une solution x^ϵ est dite ϵ-optimale ou suboptimale si :

$$F(x^\epsilon) - F(x^*) = (\frac{1}{2}(x^\epsilon)'Dx^\epsilon + c'x^\epsilon) - (\frac{1}{2}(x^*)'Dx^* + c'x^*) \leq \epsilon,$$

avec x^* est une solution optimale du problème (3.1)-(3.3) et ϵ est un nombre arbitraire positif ou nul.

Définition 3.4. ***(Support)***

- *L'ensemble $J_B \subset J$, $|J_B| = m$ est appelé support des contraintes du problème (3.1)-(3.3) si :*

$$det(A_B) = det(A(I, J_B)) \neq 0.$$

- *L'ensemble $J_S \subset J_N$ est appelé support de la fonction objectif du problème (3.1)-(3.3) si la sous matrice $M_S = M(J_S, J_S)$ de M est non singulière, i.e,*

$$det(M(J_S, J_S)) \neq 0,$$

où

$$M = M(J_N, J_N) = Z'DZ, Z = Z(I, J_N) = \begin{pmatrix} -A_B^{-1}A_N \\ I_N \end{pmatrix},$$

I_N étant la matrice identité d'ordre $(n-m)$.

- *L'ensemble $J_p = \{J_B, J_S\}$, où J_B est le support des contraintes et J_S support de la fonction objectif, est appelé support du problème (3.1)-(3.3).*

Définition 3.5. ***(Solution de support)***
Le couple $\{x, J_P\}$, formé d'une solution réalisable x et d'un support J_P, est appelé solution réalisable de support (SRS) du problème $(3.1)-(3.3)$.

Définition 3.6.

- *Le vecteur $g(x) = Dx + c$ représente le gradient de la fonction objectif F au point x, tel que :*

$$g' = (g'_B, g'_N), g'_B = g'(J_B) = (g_j, j \in J_B), g'_N = g'(J_N) = (g_j, j \in J_N).$$

- *Le vecteur π tel que $\pi' = g'_B A_B^{-1}$ est dit vecteur des potentiels.*
- *Le vecteur E tel que $E' = g' - \pi' A$, représente le vecteur des estimations ou vecteur des coûts réduits, tel que :*

$$E' = (E'_B, E'_S, E'_{NN}), E'_B = E'(J_B) = (E_j, j \in J_B) = 0, E'_S = E'(J_S) = (E_j, j \in J_S),$$
$$E'_{NN} = E'(J_{NN}) = (E_j, j \in J_{NN}).$$

Définition 3.7. ***(Solution de support non dégénérée)***
Une SRS est dite non dégénérée si :

$$l_j < x_j < u_j, j \in J_B.$$

Une SRS est dite accordée si $E_S = E(J_S) = 0$.

Définition 3.8. *Soit $d \in \mathbb{R}^n$. Le vecteur d est dit direction admissible du problème* $(3.1)-(3.3)$ *si $Ad = 0$. Une direction admissible d est dite direction d'amélioration au point x si $E'd < 0$.*

3.4 Formule d'accroissement de la fonction objectif

Soit $\{x, J_B\}$ une SRS et $\bar{x} = x + \Delta x$ une autre solution réalisable quelconque. Alors l'accroissement de la fonction objectif s'écrit comme suit :

$$\begin{aligned}
\Delta F = F(\bar{x}) - F(x) &= (\frac{1}{2}\bar{x}'D\bar{x} + c'\bar{x}) - (\frac{1}{2}x'Dx + c'x) \\
&= \frac{1}{2}(x+\Delta x)'D(x+\Delta x) + c'(x+\Delta x) - \frac{1}{2}x'Dx - c'x \\
&= \Delta x'(Dx + c) + \frac{1}{2}\Delta x'\Delta x \\
&= g'(x)\Delta x + \frac{1}{2}\Delta x'D\Delta x.
\end{aligned}$$

De plus, on a

$$\begin{cases} Ax = b \\ A\bar{x} = b \end{cases} \Leftrightarrow A\bar{x} = Ax = b \Rightarrow A\bar{x} - Ax = A\Delta x = 0 \Rightarrow A_B\Delta x_B + A_N\Delta x_N = 0,$$

où

$$\Delta x_B = \Delta x(J_B), \Delta x_N = \Delta x(J_N).$$

Donc

$$\Delta x_B = -A_B^{-1}A_N\Delta x_N.$$

En remplaçant dans la formule d'accroissement, on trouve :

$$\begin{aligned}
\Delta F &= g_B'\Delta x_B + g_N'\Delta x_N + \frac{1}{2}\Delta x'D\Delta x \\
&= g_B'(-A_B^{-1}A_N\Delta x_N) + g_N'\Delta x_N + \frac{1}{2}\Delta x'D\Delta x \\
&= (g_N' - g_B'A_B^{-1}A_N)\Delta x_N + \frac{1}{2}\Delta x'D\Delta x.
\end{aligned}$$

Alors

$$\Delta F = E_N'\Delta x_N + \frac{1}{2}\Delta x'D\Delta x. \tag{3.4}$$

D'autre part, nous avons :

$$\begin{aligned}
\Delta x'D\Delta x &= \begin{pmatrix} \Delta x_B \\ \Delta x_N \end{pmatrix}' D \begin{pmatrix} \Delta x_B \\ \Delta x_N \end{pmatrix} \\
&= \begin{pmatrix} -A_B^{-1}A_N\Delta x_N \\ \Delta x_N \end{pmatrix}' D \begin{pmatrix} -A_B^{-1}A_N\Delta x_N \\ \Delta x_N \end{pmatrix} \\
&= \Delta x_N{}' \begin{pmatrix} -A_B^{-1}A_N \\ I_N \end{pmatrix}' D \begin{pmatrix} -A_B^{-1}A_N \\ I_N \end{pmatrix} \Delta x_N \\
&= \Delta x_N{}'Z'DZ\Delta x_N.
\end{aligned}$$

D'ou

$$\Delta x' D \Delta x = \Delta x'_N M \Delta x_N. \tag{3.5}$$

En utilisant (3.5), la d'accroissement (3.4) prend la forme finale suivante :

$$\Delta F = F(\bar{x}) - F(x) = E'_N \Delta x_N + \frac{1}{2} \Delta x'_N M \Delta x_N, \tag{3.6}$$

avec M est une matrice semi-définie positive.

3.5 Critère d'optimalité

Théorème 3.9. *[18, 6] Soit $\{x, J_p\}$ une SRS accordée du problème $(3.1) - (3.3)$. Alors les relations*

$$\begin{cases} E_j \geq 0, & si \quad x_j = l_j; \\ E_j \leq 0, & si \quad x_j = u_j; \\ E_j = 0, & si \quad l_j < x_j < u_j, j \in J_{NN}, \end{cases}$$

sont suffisantes, et dans le cas de la non dégénérescence de $\{x, J_p\}$, elles sont aussi nécessaires pour l'optimalité de la solution réalisable x.

3.6 Critère suboptimalité

Définition 3.10. *Soit $\{x, J_p\}$ une SRS accordée du problème $(3.1) - (3.3)$. Le nombre*

$$\beta(x, J_p) = \sum_{j \in J_N, E_j > 0} E_j(x_j - l_j) + \sum_{j \in J_N, E_j < 0} E_j(x_j - u_j)$$

est dit estimation de suboptimalité.
De plus, on a toujours l'inégalité :

$$F(x) - F(x^*) \leq \beta(x, J_p).$$

Théorème 3.11. *Soit $\{x, J_p\}$ une SRS accordée du problème $(3.1)-(3.3)$ et ϵ un nombre arbitraire positif ou nul. Si $\beta(x, J_p) \leq \epsilon$, alors $\{x, J_p\}$ est une solution ϵ-optimale de ce problème.*

Démonstration. Soit x^* une solution optimale du problème $(3.1) - (3.3)$. En remplaçant $\bar{x}$ par x^* dans la formule d'accroissement (3.6) et en minorant l'expression, on aura :

$$\begin{aligned} F(x^*) - F(x) &= E'_N \Delta x_N + \frac{1}{2} \Delta x'_N M \Delta x_N + \sum_{j \in J_N} E_j(x_j^* - x_j) + \frac{1}{2} \Delta x'_N M \Delta x_N \\ &\geq \sum_{j \in J_N} E_j(x_j^* - x_j). \\ F(x) - F(x^*) &\leq \sum_{j \in J_N} E_j(x_j - x_j^*) = \sum_{j \in J_N, E_j > 0} E_j(x_j - x_j^*) + \sum_{j \in J_N, E_j < 0} E_j(x_j - x_j^*). \end{aligned}$$

Mais $l_j \leq x \leq u_j$, $j \in J_N$, ce qui implique :

$$\begin{aligned} E_j(x_j - x_j^*) &\leq E_j(x_j - l_j), \quad E_j > 0, \\ E_j(x_j - x_j^*) &\leq E_j(x_j - u_j), \quad E_j < 0. \end{aligned}$$

Alors

$$F(x) - F(x^*) \leq \sum_{j \in J_N, E_j > 0} E_j(x_j - l_j) + \sum_{j \in J_N, E_j < 0} E_j(x_j - u_j) \leq \epsilon.$$

Donc

- si $\beta(x) = 0$, alors x est une solution optimale.
- si $\beta(x) \leq \epsilon$, alors x est une solution suboptimale.
- si $\beta(x) > \epsilon$, il faut améliorer la solution x.

□

3.7 Une itération de la méthode adaptée

Soient $\{x, J_P\}$ une SRS accordée du problème et ϵ un nombre arbitraire positif ou nul choisi à l'avance. Une itération de l'algorithme de la méthode adaptée consiste à passer d'une SRS $\{x, J_P\}$ vers une autre SRS $\{\bar{x}, \bar{J_P}\}$ telle que $F(\bar{x}) \leq F(x)$ en suivant les directions d'améliorations primale et duale respectivement d et t, où $\bar{x} = x + \theta d, \quad \bar{E} = E + \theta t$.

3.7.1 Calcul des directions d et t

Si la condition de suboptimalité $\beta(x, J_P) \leq \epsilon$ n'est pas vérifiée, alors on construit une direction d'amélioration adaptée primale d tel que $d' = (d'_B, d'_S, d'_{NN})$, où

- les composantes du vecteur d_{NN} seront calculées comme suit :

$$d_j = \begin{cases} l_j - x_j, & \text{si} \quad E_j > 0, \\ u_j - x_j, & \text{si} \quad E_j < 0, \\ 0, & \text{si} \quad E_j = 0, \quad j \in J_{NN}; \end{cases} \tag{3.7}$$

- les composantes du vecteur d_S seront déduites de telle sorte que les composantes d'indices $j \in J_S$ du vecteur $\bar{E}(x + \theta d)$ soient nulles. En effet, on a

$$\bar{E_N}(\bar{x}) = E_N(x) + \theta M d_N.$$

On en déduit :

$$\begin{aligned} E_S(\bar{x}) = 0 &\Rightarrow E_S + \theta M(J_S, J_N) d_N = 0 \\ &\Rightarrow M(J_S, J_N) d_N = 0 \\ &\Rightarrow M(J_S, J_S) d_S + M(J_S, J_{NN}) d_{NN} = 0 \end{aligned}$$

Alors

$$d_S = -M(J_S, J_S)^{-1} M(J_S, J_{NN}) d_{NN}. \tag{3.8}$$

- les composantes du vecteur d_B seront déduites à partir de la formule $Ad = 0$:

$$d_B = -A_B^{-1}(A_S d_S + A_{NN} d_{NN}). \tag{3.9}$$

D'autre part, on calcule la direction duale t telle que $t = (t_B, t_S, t_{NN})$, avec

$$t_N = Md_N,$$

et

$$t_B = 0, \quad t_S = 0, \quad t_{NN} = D(J_{NN}, J)d - rA'_{NN}, \quad r = [A_B^{-1}]'D(J_B, J)d. \tag{3.10}$$

3.7.2 Changement de la solution et du vecteur des estimations

La nouvelle solution réalisable $\bar{x}$ et le nouveau vecteur des estimations $\bar{E}$ sont construits comme suit :

$$\bar{x} = x + \theta d, \quad \bar{E_N} = E_N + \theta t_N, \quad \bar{E_B} = 0, \tag{3.11}$$

où d est la direction d'amélioration primale définie par les formules (3.7)-(3.9), tandis que t est la direction d'amélioration duale définie par la formule (3.10).

3.7.3 Calcul du pas θ

Le nombre θ doit être pris comme suit :

$$\theta = \min(1, \theta_{j_b}, \theta_{j_s}, \sigma_f).$$

- Le nombre $\theta = 1$ représente le pas correspondant aux indices de J_{NN}.
- Le nombre θ_{j_b} est calculé de telle sorte que les contraintes de borne sur le vecteur $\bar{x}_B$ soient satisfaites, c-à-d :

$$l_j - x_j \leq \theta d_j \leq x_j - u_j, j \in J_B.$$

- Le nombre θ_{j_s} est calculé de telle sorte que les contraintes de borne sur le vecteur $\bar{x}_S$ soient satisfaites, c-à-d :

$$l_j - x_j \leq \theta d_j \leq x_j - u_j, j \in J_S.$$

Donc $\theta_{j_b} = \min \theta_j, j \in J_B$ et $\theta_{j_s} = \min \theta_j, j \in J_S$, avec

$$\theta_j = \begin{cases} \frac{(u_j - x_j)}{d_j}, & \text{si} \quad d_j > 0 \quad ; \\ \frac{(l_j - x_j)}{d_j}, & \text{si} \quad d_j < 0 \quad ; \\ \infty, & \text{si} \quad d_j = 0 \quad , j \in J_B \cup J_S. \end{cases} \tag{3.12}$$

- Le nombre σ_f se calcule de telle sorte le passage de x à $\bar{x}$ assure une diminution maximale de la fonction objectif tout en gardant le même signe pour E_j et $\bar{E}_j$, $j \in J_{NN}$. Par conséquent, en vertu de la formule (3.11), on aura :

$$\sigma_f = \sigma_{j_0} = \min \sigma_j, j \in J_{NN},$$

avec

$$\sigma_j = \begin{cases} \frac{-E_j}{t_j}, & \text{si} \quad E_j t_j < 0 \quad ; \\ \infty, & \text{sinon} \quad , j \in J_{NN} \quad . \end{cases} \tag{3.13}$$

Si $\theta = 1$, alors la nouvelle solution $\bar{x} = x + d$ est une solution optimale du problème (3.1)-(3.3). Sinon, on calcule la nouvelle estimation de suboptimalité de la nouvelle solution $\bar{x}$.

3.7.4 Calcul de la nouvelle estimation de suboptimalité

La nouvelle estimation de suboptimalité sera sous la forme suivante [16] :

$$\beta(\bar{x}, J_P) = (1-\theta)\beta(x, J_P) - \theta(1-\theta)t'_N d_N \leq \beta(x, J_P),$$

car

$$d'_N t_N = d'_N M d_N \geq 0.$$

Si $\beta(\bar{x}, J_P) \leq \epsilon$, alors $\bar{x}$ est une solution suboptimale. Sinon, on fait le changement de support.

3.7.5 Changement du support

- Si $\theta = \theta_{j_s}$, alors on pose

$$\bar{J}_B = J_B, \quad \bar{J}_S = J_S \backslash j_s, \quad \bar{J}_{NN} = J_{NN} \cup \{j_s\};$$

- Si $\theta = \sigma_f = \sigma_{j_0}$, alors on pose

$$\bar{J}_B = J_B, \quad \bar{J}_S = J_S \cup \{j_0\}, \quad \bar{J}_{NN} = J_{NN} \backslash \{j_0\};$$

- Si $\theta = \theta_{j_b}$, alors on a
  - Si $J_S = \emptyset$, alors on pose

$$\bar{J}_B = J_B \cup \{j_0\} \backslash \{j_b\}, \quad \bar{J}_S = \emptyset, \quad \bar{J}_{NN} = J_{NN} \cup \{j_b\} \backslash \{j_0\};$$

  - Si $J_S \neq \emptyset$, pour l'indice j_b, on a

$$d_{j_b} = -\sum_{j \in J_N} e'_{j_b} A_B^{-1} a_j d_j = -\sum_{j \in J_N} x_{j_b j} d_j \neq 0,$$

  où e_{j_b} est m-vecteur unitaire. Alors, $\exists j_* \in J_N$ tel que $x_{j_b j_*} \neq 0$.

- Si $\exists j_* \in J_S$ tel que $x_{j_b j_*} \neq 0$, alors on pose

$$\bar{J}_B = J_B \cup \{j_*\} \backslash \{j_b\}, \quad \bar{J}_S = J_S \backslash j_*, \quad \bar{J}_{NN} = J_{NN} \cup \{j_b\}.$$

- Si $x_{j_b j} = 0, \forall j \in J_S$, alors on pose

$$\bar{J}_B = J_B \cup \{j_0\} \backslash \{j_b\}, \quad \bar{J}_S = J_S, \quad \bar{J}_{NN} = J_{NN} \backslash j_0 \cup \{j_b\}.$$

Poser $x = \bar{x}, \quad J_B = \bar{J}_B, \quad J_S = \bar{J}_S, \quad J_{NN} = \bar{J}_{NN}, \quad J_P = \{J_B, J_S\}$ et faire une autre itération avec la nouvelle SRS $\{x, J_P\}$.

3.7.6 Schéma de la méthode adaptée

Soit $\{x, J_P\}$ une SRS accordée du problème (3.1)-(3.3). Le schéma de la méthode adaptée pour la résolution du programme quadratique convexe à variables bornées est décrit dans les étapes suivantes :

Algorithme 1. *(Algorithme de la méthode adaptée)*
(0) *Soit un nombre réel positif ou nul quelconque ϵ et un solution de support initiale $\{x, J_P\}$ tel que $J_P = \{J_B, J_S\}$, avec $J_S = \emptyset$ pour plus de facilité ;*
(1) *Calculer les matrices : $Z = Z(J, J_N) = \begin{pmatrix} -A_B^{-1} A_N \\ I_N \end{pmatrix}$ et $\quad M = M(J_N, J_N) = Z' D Z$;*
(2) *Calculer les vecteurs : $g(x) = Dx + c, \quad \pi' = g'_B A_B^{-1}, \quad E'_N = g'_N - \pi' A_N$;*
(3) *Calculer l'estimation de suboptimalité avec la formule :*

$$\beta(x, J_P) = \sum_{j \in J_{NN}, E_j > 0} E_j (x_j - l_j) + \sum_{j \in J_{NN}, E_j < 0} E_j (x_j - u_j)$$

(4) *Si $\beta(x, J_P) = 0$, alors l'algorithme s'arrête avec $\{x, J_P\}$ une SRS optimale.*
(5)*Si $\beta(x, J_P) \leq \epsilon$, alors l'algorithme s'arrête avec $\{x, J_P\}$ une SRS ϵ-optimale.*
(6) *Sinon, calculer le vecteur $d_{NN} = (d_j, j \in J_{NN})$:*

$$d_j = \begin{cases} l_j - x_j, & si \quad E_j > 0, \\ u_j - x_j, & si \quad E_j < 0, \\ 0, & si \quad E_j = 0 \quad , j \in J_{NN}, \end{cases}$$

(7) *Calculer les directions d et t en utilisant les relations :*

$$d_S = -M(J_S, J_S)^{-1} M(J_S, J_{NN}) d_{NN}, \quad d_B = -A_B^{-1}(A_S d_S + A_{NN} d_{NN});$$

$$t_B = 0, \quad t_N = M d_N;$$

(8) *Calculer $\theta_{j_b} = \theta_j, j \in J_B$ et $\theta_{j_s} = \min \theta_j, j \in J_S$, où θ_j est déterminé par la formule :*

$$\theta_j = \begin{cases} \frac{(u_j - x_j)}{d_j}, & si \quad d_j > 0 \quad ; \\ \frac{(l_j - x_j)}{d_j}, & si \quad d_j < 0 \quad ; \\ \infty, & si \quad d_j = 0 \quad , j \in J_B \cup J_S; \end{cases}$$

(9) *Calculer* $\sigma_f = \sigma_{j_0} = \min \sigma_j, j \in J_{NN}$, *où* σ_j *est déterminé par la formule :*

$$\sigma_j = \begin{cases} \frac{-E_j}{t_j}, & si \quad E_j t_j < 0 \quad ; \\ \infty, & sinon \quad , j \in J_{NN} \quad . \end{cases}$$

(10) *Calculer* $\theta = \min(1, \theta_{j_b}, \theta_{j_s}, \sigma_f)$ *;*
(11) *Calculer* $\bar{x} = x + \theta d$ *et* $\bar{E_N} = E_N + \theta t_N$;
(12) *Si* $\theta = 1$, *alors l'algorithme s'arrête avec* $\{\bar{x}, J_P\}$ *une SRS optimale.*
(13) *Sinon, calculer* $\beta(\bar{x}, J_P) = (1-\theta)\beta(x, J_P) - \theta(1-\theta)t'_N d_N$ *;*
(14) *Si* $\beta(\bar{x}, J_P) \leq \epsilon$, *alors l'algorithme s'arrête avec* $\{\bar{x}, J_P\}$ *une SRS* ϵ*-optimale.*
(15) *Appliquer le changement de support :*

(15.1) *Si* $\theta = \theta_{j_s}$, *alors poser*

$$\bar{J}_B = J_B, \quad \bar{J}_S = J_S \backslash j_s, \quad \bar{J}_{NN} = J_{NN} \cup \{j_s\};$$

(15.2) *Si* $\theta = \sigma_f = \sigma_{j_0}$, *alors poser*

$$\bar{J}_B = J_B, \quad \bar{J}_S = J_S \cup \{j_0\}, \quad \bar{J}_{NN} = J_{NN} \backslash \{j_0\};$$

(15.3) *Si* $\theta = \theta_{j_b}$, *alors*

(15.3.1) *Si* $J_S = \emptyset$, *alors poser*

$$\bar{J}_B = J_B \cup \{j_0\} \backslash \{j_b\}, \quad \bar{J}_S = \emptyset, \quad \bar{J}_{NN} = J_{NN} \cup \{j_b\} \backslash \{j_0\};$$

(15.3.2) *Si* $J_S \neq \emptyset$, *alors*

(15.3.2.1) *On a*

$$d_{j_b} = -\sum_{j \in J_N} e'_{j_b} A_B^{-1} a_j d_j = -\sum_{j \in J_N} x_{j_b j} d_j \neq 0,$$

Alors, $\exists j_* \in J_N$ *tel que* $x_{j_b j_*} \neq 0$.

(15.3.2.2) *Si* $\exists j_* \in J_S$ *tel que* $x_{j_b j_*} \neq 0$, *alors poser*

$$\bar{J}_B = J_B \cup \{j_*\} \backslash \{j_b\}, \quad \bar{J}_S = J_S \backslash j_*, \quad \bar{J}_{NN} = J_{NN} \cup \{j_b\}.$$

(15.3.2.3) *Si* $x_{j_b j} = 0, \forall j \in J_S$, *alors poser*

$$\bar{J}_B = J_B \cup \{j_0\} \backslash \{j_b\}, \quad \bar{J}_S = J_S, \quad \bar{J}_{NN} = J_{NN} \backslash j_0 \cup \{j_b\}.$$

(16) *Poser* $x = \bar{x}, J_B = \bar{J}_B, J_S = \bar{J}_S, J_{NN} = \bar{J}_{NN}, J_P = \{J_B, J_S\}$ *et aller à l'étape (1).*

3.8 Application numérique

On illustre la méthode adaptée sur l'exemple suivant :

$$\begin{cases} min \quad F(x) = 4x_1^2 + x_2^2 - 4x_1 x_2 - x_3 + 6x_4, \\ x_1 - x_2 + x_3 = 4, \\ -x_1 + 2x_2 + x_4 = -2, \\ 0 \leq x_1 \leq 6, 0 \leq x_2 \leq 2, -1 \leq x_3 \leq 3, -9 \leq x_4 \leq 1. \end{cases} \tag{3.14}$$

On a
$$D = \begin{pmatrix} 8 & -4 & 0 & 0 \\ -4 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \; c = \begin{pmatrix} 0 \\ 0 \\ -1 \\ 6 \end{pmatrix}, \; b = \begin{pmatrix} 4 \\ -2 \end{pmatrix}, \; l = \begin{pmatrix} 0 \\ 0 \\ -1 \\ -9 \end{pmatrix}, \; u = \begin{pmatrix} 6 \\ 2 \\ 3 \\ 1 \end{pmatrix}.$$
On pose $\epsilon = 10^{-4}$. On commence par la $SRS\{x, J_P\}$, tel que

$$x = (6, 2, 0, 0)', J_B = \{3, 4\}, J_S = \emptyset, J_N = \{1, 2\}, J_{NN} = \{1, 2\}.$$

On obtient :

$$F(x) = 100, A_B = (a_3, a_4) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, A_N = (a_1, a_2) = \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}$$

- Calculer les matrices Z et M :
$$Z = Z(J, J_N) = \begin{pmatrix} -A_B^{-1} A_N \\ I_N \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 1 \\ 1 & -2 \end{pmatrix}$$

$$M = Z'DZ = \begin{pmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} 8 & -4 & 0 & 0 \\ -4 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 1 \\ 1 & -2 \end{pmatrix} = \begin{pmatrix} 8 & -4 \\ -4 & 2 \end{pmatrix}$$

- Calculer le vecteur gradient :

$$g(x) = Dx + c = \begin{pmatrix} 8 & -4 & 0 & 0 \\ -4 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 6 \\ 2 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -1 \\ 6 \end{pmatrix} = \begin{pmatrix} 40 \\ -20 \\ -1 \\ 6 \end{pmatrix},$$

avec
$$g_B = \begin{pmatrix} -1 \\ 6 \end{pmatrix} \quad et \quad g_N = \begin{pmatrix} 40 \\ -20 \end{pmatrix}$$

- Calculer le vecteurs des coûts réduits :

$$\begin{aligned} E_N = (E_1, E_2)' &= g_N - (g_B' A_B^{-1} A_N)' \\ &= \begin{pmatrix} 40 \\ -20 \end{pmatrix} - \left(\begin{pmatrix} -1 & 6 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix} \right)' = \begin{pmatrix} 47 \\ -33 \end{pmatrix} \end{aligned}$$

- Calculer l'estimation de suboptimalité :

$$\begin{aligned} \beta(x, J_p) &= \sum_{j \in J_{NN}, E_j > 0} E_j (x_j - l_j) + \sum_{j \in J_{NN}, E_j < 0} E_j (x_j - u_j). \\ &= E_1(x_1 - l_1) + E_2(x_2 - u_2) = 47(6 - 0) - 33(2 - 2) \quad = 282 > \epsilon. \end{aligned}$$

Itération 1 :

- Calculer les directions d'amélioration d et t :

$$E_j > 0 \Rightarrow d_1 = l_1 - x_1 = -6,$$
$$E_j < 0 \Rightarrow d_2 = u_2 - x_2 = 0.$$

donc $d_{NN} = \begin{pmatrix} -6 \\ 0 \end{pmatrix}$.

$$d_B = (d_j, j \in J_B) - A_B^{-1} A_{NN} d_{NN} = \begin{pmatrix} -1 & 1 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} -6 \\ 0 \end{pmatrix} = \begin{pmatrix} 6 \\ -6 \end{pmatrix}.$$

Et

$$t_N = \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = M d_N = \begin{pmatrix} 8 & -4 \\ -4 & 2 \end{pmatrix} \begin{pmatrix} -6 \\ 0 \end{pmatrix} = \begin{pmatrix} -48 \\ 24 \end{pmatrix}.$$

- Calcul du θ le long des directions :
  - Pour $j \in J_{NN}$, on a : $\theta = 1$.
  - Pour $j \in J_B$ on a : $\theta_{j_b} = min_{j \in J_B} \theta_j = min\{\theta_3, \theta_4\}$ tel que :

$$\begin{cases} \theta_3 = \frac{1}{2}. \\ \theta_4 = \frac{3}{2}. \end{cases}$$

$$\theta_{j_b} = min\{\frac{1}{2}, \frac{3}{2}\} = \theta_3 = \frac{1}{2} \Rightarrow j_b = 3.$$

  - Pour $j \in J_S$, on a : $J_S = \emptyset \Rightarrow \theta_{j_s} = \infty$.
  - Pour σ_f, on a $\sigma_{j_0} = min\{\sigma_1, \sigma_2\}$ tel que :

$$\begin{cases} \sigma_1 = \frac{-E_1}{t_1} = \frac{47}{48}. \\ \sigma_2 = \frac{-E_2}{t_2} = \frac{33}{24}. \end{cases}$$

$$\sigma_{j_0} = min\{\frac{47}{48}, \frac{33}{24}\} = \sigma_1 = \frac{47}{48} \Rightarrow j_0 = 1.$$

  Donc, on trouve :

$$\theta = min\{1, \frac{1}{2}, \infty, \frac{47}{48}\} = \theta_{j_b} = \frac{1}{2}.$$

- Calculer de la nouvelle solution réalisable et du nouveau vecteur des estimations :

$$\bar{x} = x + \theta d = \begin{pmatrix} 3 \\ 2 \\ 3 \\ -3 \end{pmatrix}, \bar{E}_N = E_N + \theta t_N = \begin{pmatrix} 23 \\ -21 \end{pmatrix}.$$

- Calcul de la nouvelle valeur l'estimation de suboptimalité :

$$\beta(\bar{x}, J_p) = (1-\theta)\beta(x, J_p) - \theta(1-\theta)t'_N d_N$$
$$= 69 > \epsilon.$$

Comme $\beta(\bar{x}, J_p) > \epsilon$, donc on applique le changement de support :

$$\bar{J}_B = \{1, 4\}, \bar{J}_S = J_S = \emptyset, \bar{J}_N = \{2, 3\}, \bar{J}_{NN} = \{2, 3\}, \bar{J}_P = \{\bar{J}_B, \bar{J}_S\}.$$

Itération 2 : On commence la deuxième itération par $SRS\{\bar{x}, \bar{J}_P\}$
on a : $A_B = (a_1, a_4) = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}$, $A_N = (a_2, a_3) = \begin{pmatrix} -1 & 1 \\ 2 & 0 \end{pmatrix}$.

- Calculer le vecteur gradient :

$$g(x) = Dx + c = \begin{pmatrix} 16 \\ -8 \\ -1 \\ 6 \end{pmatrix},$$

avec

$$g_B = \begin{pmatrix} 16 \\ 6 \end{pmatrix} \quad et \quad g_N = \begin{pmatrix} -8 \\ -1 \end{pmatrix}.$$

- Calculer le vecteurs des coûts réduits :

$$E_N = (E_2, E_3)' = g_N - (g'_B A_B^{-1} A_N)' = \begin{pmatrix} 2 \\ -23 \end{pmatrix}.$$

- Calculer la nouvelle valeur l'estimation de suboptimalité :

$$\beta(\bar{x}, \bar{J}_P) = E_2(\bar{x}_2 - l_2) + E_3(\bar{x}_3 - u_3) = 4 > \epsilon.$$

- Calculer les matrices Z et M :

$$Z = Z(J, J_N) = \begin{pmatrix} -A_B^{-1} A_N \\ I_N \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix}.$$

$$M = Z'DZ = \begin{pmatrix} 1 & 1 & 0 & -1 \\ -1 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} 8 & -4 & 0 & 0 \\ -4 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix} = \begin{pmatrix} 2 & -4 \\ -4 & 8 \end{pmatrix}.$$

- Calculer les directions d et t :

$$d(\bar{J}_{NN}) = \begin{pmatrix} d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}.$$

$$d(\bar{J}_B) = (d_j, j \in \bar{J}_B) - A_B^{-1} A_{NN} d_{NN} = \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \end{pmatrix} = \begin{pmatrix} -2 \\ 2 \end{pmatrix}.$$

Et

$$t_N = \begin{pmatrix} t_2 \\ t_3 \end{pmatrix} = M d_N = \begin{pmatrix} 2 & -4 \\ -4 & 8 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \end{pmatrix} = \begin{pmatrix} -4 \\ 8 \end{pmatrix}.$$

- Calculer θ le long des directions :
  - Pour $j \in \bar{J}_{NN}$, on a : $\theta = 1$.

- Pour $j \in \bar{J}_B$ on a : $\theta_{j_b} = min_{j\in\bar{J}_B}\theta_j = min\{\theta_1, \theta_4\}$ tel que :
$$\begin{cases} \theta_1 = \frac{3}{2}. \\ \theta_4 = 2. \end{cases}$$
$$\theta_{j_b} = min\{\frac{3}{2}, 2\} = \theta_1 = \frac{3}{2} \Rightarrow j_b = 1.$$
- Pour $j \in \bar{J}_S$, on a : $\bar{J}_S = \emptyset \Rightarrow \theta_{j_s} = \infty$.
- Pour σ_f, on a $\sigma_{j_0} = min\{\sigma_2, \sigma_3\}$ tel que :
$$\begin{cases} \sigma_2 = \frac{-E_2}{t_2} = \frac{2}{4}. \\ \sigma_3 = \frac{-E_3}{t_3} = \frac{23}{8}. \end{cases}$$
$$\sigma_{j_0} = min\{\frac{2}{4}, \frac{23}{8}\} = \sigma_2 = \frac{2}{4} \Rightarrow j_0 = 2.$$

Donc, on trouve :
$$\theta = min\{1, \frac{3}{2}, \infty, \frac{2}{4}\} = \sigma_2 = \frac{2}{4}.$$

- Calculer la nouvelle solution réalisable et du nouveau vecteur des estimations :
$$\bar{\bar{x}} = \bar{x} + \theta d = \begin{pmatrix} 2 \\ 1 \\ 3 \\ -2 \end{pmatrix}, \bar{E}_N = E_N + \theta t_N = \begin{pmatrix} 0 \\ -19 \end{pmatrix}.$$

- Calculer la nouvelle valeur l'estimation de suboptimalité :
$$\beta(\bar{\bar{x}}, \bar{J}_p) = (1-\theta)\beta(x, J_p) - \theta(1-\theta)t'_N d_N = 0.$$

Comme $\beta(\bar{\bar{x}}, \bar{J}_p)$, étant nulle, le critère d'optimalité est vérifié.
D'où $x^* = \begin{pmatrix} 2 \\ 1 \\ 3 \\ -2 \end{pmatrix}$ est une solution optimale, avec $F(x^*) = -6$.

3.9 Conclusion

Dans ce chapitre, nous avons vu certaines définitions dont nous avons besoin dans les étapes de la méthode adaptée, l'algorithme de résolution par la méthode, et enfin nous avons fourni un exemple numérique.

Chapitre 4

Mise en œuvre

4.1 Introduction

Dans ce chapitre, on va présenter les résultats numériques de la comparaison entre la méthode adaptée présentée dans le chapitre précédent et la méthode d'activation des contraintes du solveur **quadprog**, défini sous Matlab. La comparaison est faite sur des problèmes quadratiques convexes générés aléatoirement. La caractéristique de chaque problème est que la solution optimale x^*, ainsi que la valeur de la fonction objectif F^* à l'optimum sont connues à l'avance. Enfin, on va faire une simple application de la méthode adaptée sur le problème de classification binaire par SVM [31, 11].

4.2 Générateur d'exemples

Afin de bien tester et comparer l'efficacité de la méthode étudiée dans le chapitre précédent, nous avons développé un générateur de problèmes quadratiques convexes à variables bornées. Ensuite, résoudre ces problèmes générés avec la méthode adaptée ainsi que la méthode d'activation des contraintes (ASM) du Matlab. Puis, comparer les résultats obtenus.
Le programme du générateur aléatoire est une fonction appelée *generation exemples tests* :
$[xopt, fopt, D, c, A, b, l, u] = generateur\_exemple\_test(m, n)$;

Paramètres d'entrées :

n : nombre de variables.
m : nombre de contraintes.

Paramètres de sorties :

A : matrice d'ordre $m \times n$, notée la matrice des contraintes.
D : matrice carrée symétrique d'ordre n, elle est semi-définie positif.
c : n-vecteur.

b : m-vecteur.
generer : est une fonction qui génère des nombres aléatoires dans l'intervalle $[-1, 1]$, suivant une loi uniforme. En Langage Matlab *Urand*.

4.3 La méthode d'activation des contraintes pour la résolution de P.Q.C.

La méthode activation des contraintes (ASM : Active set method) ([15]) est une méthode classique développée au début des années soixante-dix pour la résolution des problèmes linéaires et quadratiques ; elle s'applique pour des problèmes d'optimisation avec contraintes linéaires de types inégalités ou mixtes.
Dans notre application, on a appelé la méthode d'activation des contraintes du solveur *quadprog* (disponible sous Matlab pour les versions), en l'appelant ainsi :
tic
$options = optimset('Algorithm','Active - set');$
$[X, f\_act, e\_act, output\_act] = quadprog(D, c, [], [], A, b, l, u)$
toc
Pour avoir en sortie le nombre d'itérations on aura à composer le code suivant : *output.iterations*.
Pour avoir le temps d'exécution de la méthode on utilise la commande *tic-toc*.
Pour plus d'informations sur ce solveur, vous composez ce code : *help quadprog*.

4.4 Implémentation de la méthode adaptée sous Matlab

La fonction implémentée sous MATLAB est un programme permet de résoudre des problèmes quadratiques convexes à variables bornées avec la méthode adaptée de support définie précédemment. Voici l'appel de ce programme :
tic
$[f, xopt, nbr\_iter, J_B] = support\_adaptee(D, c, A, b, l, u, J_B)$
toc

4.4.1 Exemple d'exécution

Nous avons déjà trouvé la solution du problème (3.14), et maintenant nous allons résoudre ce problème sous MATLAB par l'algorithme implémenté :

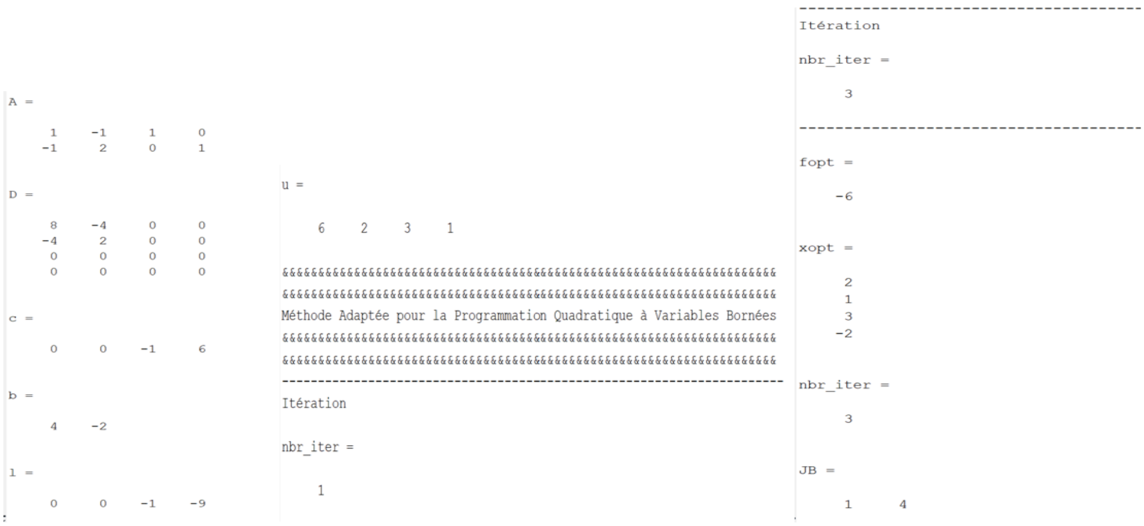


FIGURE 4.1 – Résolution de l'exemple en Matlab

4.4.2 Comparaison entre la méthode adaptée et ASM

Notation :

• CPU : Le temps d'exécution des problèmes tests en secondes. On a utilisé la fonction Matlab *tic-toc.*

• nit : Le nombre d'itérations requis pour la résolution des problèmes tests.

• ΔF : C'est la différence entre la valeur de la fonction objectif optimale et celles obtenues par les deux méthodes.

4.4.3 Discussion des résultats

Le tableau (4.4.2) présente les résultats comparatives des deux méthodes sur des problèmes générés sous Matlab. Les critères de comparaison sont le temps d'exécution, le nombre d'itérations ainsi que la différence ΔF entre la solution optimale du problème et celles données par les deux méthodes.
D'après le tableau, on remarque que les deux méthodes sont compétitives en termes de temps d'exécution et ΔF pour presque tous les problèmes générés.
En terme de nombre d'itérations, on remarque que la méthode ASM converge dans des cas avec 0 itération, cela le fait que la méthode utilise une technique de présolving qui permet de détecter une solution optimale avant d'entamer l'algorithme de résolution. D'autre part, la méthode adaptée trouve la solution avec quelques exemples de test après une seule itération, cela le fait de la métrique adaptée qui permet de changer tous les indices non

Taille		Méthode Adaptée			ASM		
m	n	ΔF	CPU	nit	ΔF	CPU	nit
1	2	9.4242×10^{-13}	0.3991	1	0	0.6149	0
2	3	6.6613×10^{-16}	0.5129	1	0	0.3291	0
2	4	2.2204×10^{-15}	0.4620	1	5.7070×10^{-10}	0.3854	5
3	5	9.3480×10^{-14}	0.4559	1	9.6655×10^{-9}	0.3832	4
3	6	4.4408×10^{-16}	0.4648	1	2.7093×10^{-8}	0.3573	4
5	10	1.7763×10^{-15}	0.6560	4	3.5527×10^{-15}	0.4309	4
6	10	4.7961×10^{-14}	0.6686	7	8.8817×10^{-16}	0.4123	4
6	12	8.5265×10^{-14}	0.6193	4	0	0.3349	0
10	15	4.6146×10^{-11}	0.7353	8	9.3223×10^{-7}	0.3459	4
10	20	2.7414×10^{-10}	1.1459	26	1.5276×10^{-12}	0.3586	5

TABLE 4.1 – Les résultats de comparaison entre la méthode adaptée et ASM.

optimaux à la fois, ce qui permet d'accélérer la convergence de l'algorithme.
Une autre particularité de l'algorithme est qu'il utilise l'estimation de suboptimalité qui permet d'arrêter l'algorithme avec une précision désirée et donc améliorer la convergence.

4.5 Application de la méthode adaptée sur les machines à vecteurs de support(SVM)

Les machines à vecteurs de support (SVM) ([31, 30, 13]) est une méthode de classification par l'apprentissage supervisé. Elle fut introduite par Vapnik en 1995 [29, 30]. Elle est devenue une technique d'apprentissage omniprésente dans les divers domaines de la communauté académique, ainsi que dans l'industrie.
La discrimination par SVM consiste à séparer les données en fonction de leurs étiquettes en deux (ou plusieurs) classes.
Dans le cas binaire, il n'y a que deux étiquettes possibles (par exemple, +1 ou -1). Ainsi, les SVM cherchent à déterminer une frontière (hyperplan séparateur) entre les deux catégories. Cette frontière est tracée en fonction d'exemples connus appelés vecteurs de support.
Un algorithme résolvant les SVM identifie parmi les exemples d'apprentissage quels sont les vecteurs de support, et il construit la frontière avec une combinaison linéaire de cette sélection.
Résoudre ce problème équivaut à résoudre un programme quadratique convexe avec une contrainte d'égalité et des variables bornées. Ce dernier peut être résolu par les techniques d'optimisation numérique.
Dans cette section, nous allons présenter la possibilité d'application de la méthode adaptée présentée dans le chapitre précédent pour la résolution du problème dual quadratique convexe d'un SVM. Cette idée d'application est le résultats des travaux [5, 11, 12, 13]

pour résoudre les problèmes de classification binaire SVM par la méthode adaptée.

4.5.1 Algorithme d'apprentissage

La recherche d'un hyperplan séparateur H (voir la figure), entre les deux classes de l'échantillon d'apprentissage par SVM revient à résoudre un programme quadratique convexe.
La formulation duale de ce problème d'optimisation est convexe, i.e., la fonction objectif

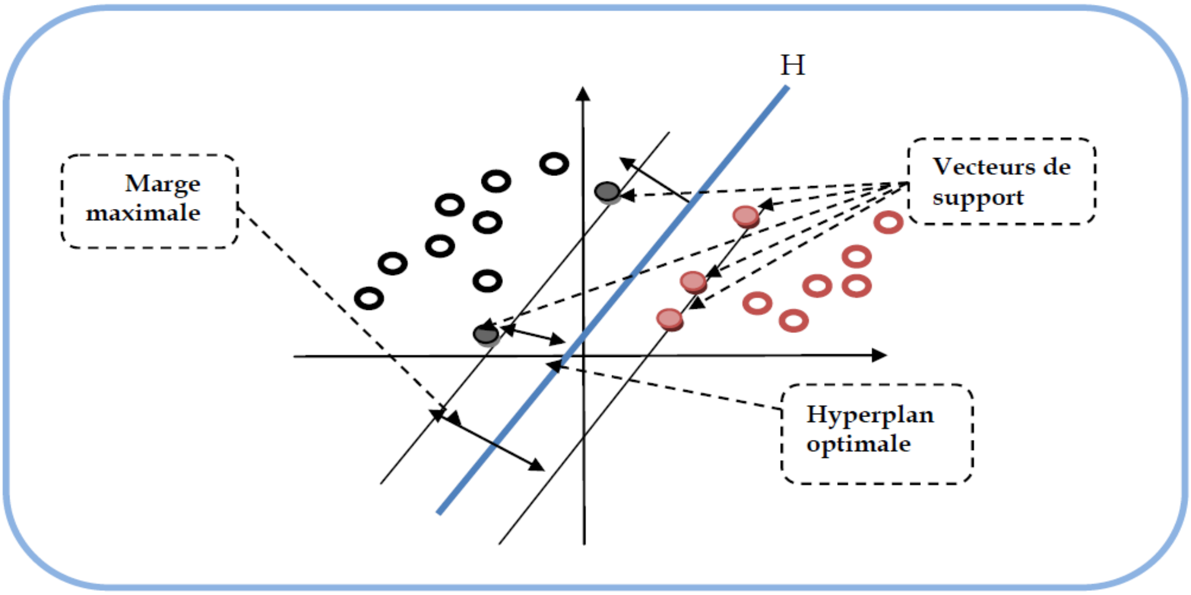


FIGURE 4.2 – Illustration de la séparation binaire par SVM

et l'ensemble de contraintes sont convexes. Par conséquent, le problème SVM ne possède pas plusieurs optima locaux, mais un minimum global assurée par une matrice hessienne semi-définie positive.
En effet, le problème de classification binaire par SVM est sous la forme suivante :

$$\min_{\alpha} \frac{1}{2}\alpha' D\alpha - e'\alpha, \tag{4.1a}$$

$$y'\alpha = 0, \tag{4.1b}$$

$$0 \leq \alpha \leq Ce, \tag{4.1c}$$

où le n-vecteur α est le vecteur des multiplicateurs de Lagrange associés aux contraintes du problème primal, e est un n-vecteur dont chaque composante est égale à 1, y est un n-vecteur et n la taille de l'ensemble d'apprentissage. La matrice D est carrée d'ordre n, symétrique et semi-définie positive, définie par ses éléments $d_{ij} = y_i y_j x_i x_j$, et $i, j \in I = \{1, \ldots, n\}$.

En comparant la formulation du problème *pqsvm* par la formulation du problème (3.1) − (3.3) que nous avons vu dans le chapitre précédent, on trouve :

$$\begin{cases} L = F, \alpha = x, e = c, \\ y = A, b = 0, \\ Ce = u, l = (0, ..., 0). \end{cases}$$

Alors, la formulation du problème SVM est adéquate pour appliquer la méthode adaptée, avec la particularité $|J_B| = 1$. En effet, on a une seule contrainte d'égalité. Donc, l'application de la méthode adaptée sur cette formulation est plus simple que la formulation standard de la P.Q.C.

4.5.2 Exemple de résolution d'un SVM linéaire par la méthode adaptée

La base de données *linsep* contient des données de classification linéaire par SVM représenté par les vecteurs X et Y, tel que :
• X représente les caractéristiques des individus ;
• Y représente la classe des individus.
Pour charger la base de données dans l'espace de travail en Matlab on tape la commande *load linsep*.
On applique la méthode adaptée pour séparer les individus en deux classes. La figure (4.3) représente une capture des résultats obtenus.

```
Y =

    -1
     1
     1
    -1
     1
    -1
    -1


X =

    1.0000    1.0000
    3.0000    3.0000
    1.0000    3.0000
    3.0000    1.0000
    2.0000    2.5000
    3.0000    2.5000
    4.0000    3.0000


n =

     7
```

```
 ***** Méthode adaptée  *****
Méthode adaptée

D =

    3.5997    1.7529   -0.4693    1.3775   -0.1408   -0.9703   -2.8640
    1.7529    0.8751   -0.0138    0.8640   -0.0041   -0.4403   -1.3195
   -0.4693   -0.0138    2.2084    1.7529    0.6625    0.4486    1.1249
    1.3775    0.8640    1.7529    2.2664    0.5259   -0.0814   -0.4196
   -0.1408   -0.0041    0.6625    0.5259    0.1988    0.1346    0.3375
   -0.9703   -0.4403    0.4486   -0.0814    0.1346    0.3099    0.8847
   -2.8640   -1.3195    1.1249   -0.4196    0.3375    0.8847    2.5418

alpha*y=0.0000
```

```
Le temps de SVMAdapt=0.1493
Le nombre de vecteurs supports==3

alpha =

         0
    9.1429
         0
         0
         0
    6.5714
    2.5714
```

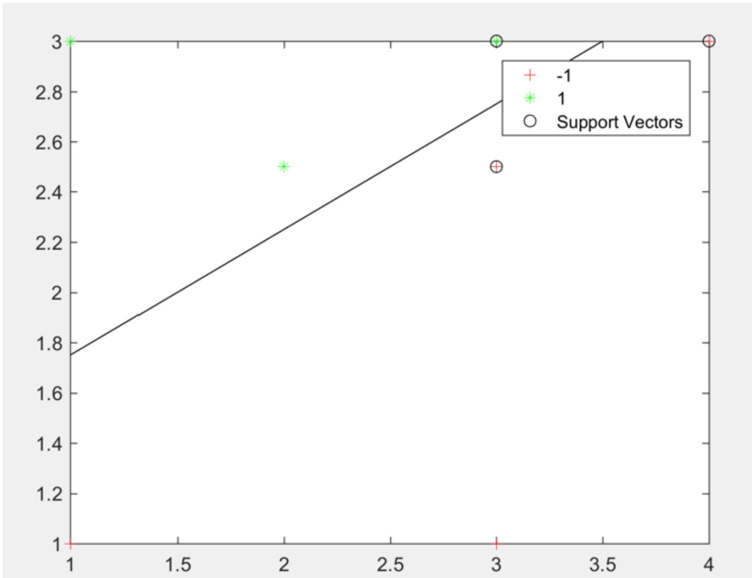


FIGURE 4.3 – Résolution d'un exemple simple de SVM par la méthode adaptée.

4.6 Conclusion

Dans ce chapitre, nous avons comparé numériquement la méthode adaptée et celle d'activation des contraintes implémentées sous l'environnement Matlab. Cette comparaison est faite sur des problèmes de minimisation quadratiques convexes générés aléatoirement. Ensuite, on a présenté un cas d'application de la méthode adaptée pour la résolution du problème de classification par SVM.

Conclusion Générale

L'objectif principal de ce mémoire est d'étudier la méthode adaptée de Gabassov pour la résolution des problèmes de minimisation quadratiques convexes. Elle appartient à la classe des méthodes primales-duales. Elle prend en compte les spécificités des contraintes et les traite comme elles se présentent, sans chercher à les transformer. Cela permet d'éviter d'étendre les dimensions du problème et préserve donc l'espace mémoire

Dans un premier temps, nous avons rappelé dans le premier chapitre les notions fondamentales d'algèbre linéaire et de la convexité. Ensuite, dans le deuxième chapitre, nous avons présenté l'optimisation quadratique convexe avec contraintes.

Puis, dans le troisième chapitre, nous avons étudié la méthode adaptée de résolution des problèmes de minimisation quadratiques convexes, avec une illustration numérique.

Dans le quatrième chapitre, nous avons présenté des expérimentations numériques de la méthode étudiée implémentée sous l'environnement MATLAB en comparaison avec la méthode ASM appelée à travers le solveur quadprog. Pour cela, nous avons implémenté un générateur des problèmes de minimisation quadratiques convexes. Puis, appliquer les deux algorithmes implémentés et faire des comparaisons numériques entre les résultats en termes de temps d'exécution, de nombre d'itérations, ainsi que ΔF. Enfin, nous avons vu un cas d'application de la méthode adaptée pour la résolution des problèmes de classification par SVM.

Perspectives

• Résoudre un problème de minimisation quadratique convexe par d'autres méthodes telles que : la méthode directe de support, la méthode du gradient projeté, la méthode du simplexe,...,etc.

• Appliquer la méthode adaptée pour résoudre d'autres cas pratiques comme : la gestion de portefeuille, la gestion de la production,...,etc.

Bibliographie

[1] J. Abadie, On the Kukn-tucker theorem, In J. Abadie, editor, Nonliear Programing. NorthHoland, Amesterdam, 1967.

[2] K. J. Arrow, G. Debreu. Existence of an equilibrium for a competitive economy. Econometrica, 22 :265–290, 1954.

[3] E.W. Barankin and R. Dorfman, On Quadratic Programming, vol.2, University of California, Publication in statistics, pp. 285-318, 1958.

[4] M. O. Bibi. Cours de programmation Mathématique 4eme année Recherche Opérationnelle, Université de Béjaia, 1998.

[5] B. Brahmi and M. O. Bibi, Dual support method for solving convex quadratic programs, Optimization, vol. 59, pp. $851 - 872, 2010$.

[6] A. Chernushevich, Algorithms for solving linear-quadratic extremal problems, PhD thesis, Institute of Mathematics, BSSR Academy of Sciences, Minsk, 1987(in Russian).

[7] N. Cristianini and J. Shawe-Taylor. Introduction to Support Vector Machines and other kernelbased learning methods. Cambridge University Press, United Kingdom, 2000.

[8] Y. H. Dai and Y. Yuan, Anonlinear conjugate gradient method with a strong global, SIAMJ. Optim. 1999.

[9] G.B. Dantzig. Linear Programming and Extensions, Princeton University Press, Princeton, New-Jersey, 1963.

[10] G. Debreu. Integration of correspondences. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2 : Contributions to Probability Theory, Part 1. The Regents of the University of California, 1967.

[11] S. Djemai, B. Brahmi, and M .O. Bibi, A primal-dual method for SVM training. Neurocomputing, 2016.

[12] S. Djemai, B. Brahmi, and M .O. Bibi, Méthode primale-duale pour l'apprentissage des svm, In : COSI'2014, pp. 189-197, 2014.

[13] S. DJEMAI. Résolution des problèmes de classification SVM par la méthode adaptée. Thèse de doctorat, Université de Béjaia, 2016.

[14] *H. J.* Ferreau. An online active set strategy for fast solution of parametric quadratic programs with applications to predictive engine control. *PhD* thesis, university of Heidelberg, 2006.

[15] R. Fletcher, P. E. Gill, al. Active-set methods, 1971, 1978.

[16] R. Gabasov et F. M. Kirillova. Méthodes de programmation linéaire, volumes 1, 2 et 3. Edition de l'Université, Minsk, 1977, 1978 et 1980 (en Russe).

[17] R. Gabasov, F. Kirillova, and O. Kostyukova, Solution of linear quadratic extremal problems, Soviet. Math. Dokl., 31, pp. $99-103, 1985$.

[18] R. Gabasov, F. Kirillova, V. Raketsky, and O. Kostyukova, Constructive methods of optimization, Volume 4 : Convex Problems, Minsk University Press, 1987(in Russian).

[19] R. Gabasov, F. Kirillova, and V. Raketsky, On methods for solving the general problem of convex quadratic programming, Soviet. Math. Dokl, 23, pp. $653-657$, 1981.

[20] R. Goldstein, L. Po-lyak. The gradient projection Method , 1960, 1964, 1966.

[21] E. A. Kostina and O. I Kostykova. Algorithme for programs equality and inequality constraints. Journal of applied Mathematics and physics (Russia), 42(7) : 1012-1026, 2001.

[22] H. M. Markowitz. Portfolio selection. Journal of Finance, 7(1) :77–91, 1952.

[23] N. Meddeouer. Contribution à l'étude des problèmes d'optimisation avec des données quadratiques. Thèse de Magister présentée au département de mathématiques, Université de Constantine (2003).

[24] M. Minoux, Programmation Mathématique, Théorie et Algorithmes, vol 1, Bordas et C.N.E.T-ENST, Paris, 1983.

[25] Y. E. Nesterov, A. S. Nemirovsky. Interior point methods, Interior-Point Polynomial Methods in Convex Programming, SIAM Publications, 1994.

[26] J. Nocedal and S. Wriht, Numerical optimization, Springer series in Operations Research, 1st edition, New York, 1999.

[27] B. Schlkopf and A. J. Smola. Learning With Kernels : Support Vector Machines, Regularization, Optimization and Beyond. MIT Press, 2002.

[28] W. Sun and Ya-Xiang Yuan. Optimization theory and methods, Nonlinear Programming. Springer, 2006.

[29] V. N. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, 1995.

[30] V. Vapnik. The Nature of Statistical learning Theory. Information Science and Statistics, Springer-Verlag, New York, 1995.

[31] V. Vapnik, S. Golowich, and A. Smola, Support vector method for function approximation, regression estimation, and signl processing, The Advances in Neural Information Processing Systems, MIT Press, pp. 281-287, 1997.

[32] P. Wolfe. A duality theorem for nonlinear programming. Quart. Appl. Math, pages 239-244, 1961.

[33] P. Wolfe, The Simplex Method for Quadratic Programming, Journal of the Econometrica Society, pp. 382-398, 1959.