



Université du Havre

**Résolution de problèmes non linéaires par les méthodes de points
intérieurs.
Théorie et algorithmes.**

Thèse présentée par
Mohammed OURIEMCHI

en vue de l'obtention du grade de
Docteur de l'Université du Havre
en Mathématiques Appliquées

Décembre 2005

© Mohammed OURIEMCHI, 2005.

Université du Havre

Thèse de Doctorat de l'Université du Havre

**Résolution de problèmes non linéaires par les méthodes de points
intérieurs.
Théorie et algorithmes.**

présentée par

Mohammed OURIEMCHI

devant le jury composé des

Professeur Jean-Pierre CROUZEIX,	Président du jury et rapporteur,
Professeur Jean-Rodolphe ROCHE,	rapporteur,
Professeur M. A. AZIZ-ALAOUI,	examineur,
Professeur Rabah LABBAS,	examineur,
Professeur Adnan YASSINE,	Directeur de thèse.

Thèse soutenue le 08 décembre 2005

À ma famille ...

REMERCIEMENTS

Je tiens tout d'abord à remercier Monsieur Adnan YASSINE, directeur du laboratoire de mathématiques appliquées du Havre et professeur à l'institut supérieur d'études logistiques, qui m'a fait l'honneur d'être mon directeur de thèse. Il a suggéré une grande part des sujets abordés dans cette thèse, et en a été le lecteur attentif, disponible et rigoureux. Son soutien moral et ses conseils m'ont beaucoup aidé dans la rédaction et la finalisation de ma thèse. Je lui en suis grandement reconnaissant.

Toute ma gratitude s'adresse à Monsieur Jean-Pierre CROUZEIX, Professeur à l'université Blaise Pascal de Clermont-Ferrand, je me réjouis qu'il ait accepté de juger mon travail en tant que rapporteur et en présider le jury.

J'aimerais remercier également Monsieur Jean-Rodolphe ROCHE, Professeur à l'université Henri-Poincaré (Nancy 1), d'avoir accepté d'être rapporteur de ma thèse, qu'il reçoit ici toute ma gratitude.

Mes plus vifs remerciements s'adressent à Messieurs M. A. AZIZ-ALAOUI et Rabah LABBAS, professeurs à l'université du Havre, qui ont accepté de participer à ce jury.

Je voudrais remercier tout particulièrement Monsieur Abdelkarim Kera-ghel, professeur à l'université de Setif en Algérie pour tous les discussions enrichissantes que nous avons eu autour de mon sujet de thèse, lors de ses visites au Havre.

Je remercie également Monsieur Martin Cadivel, maître de conférence à l'université du Havre, pour ses conseils et aides à la correction de cette thèse.

Je salue amicalement les membres du laboratoire LMAH notamment Kadi-ri, Tarik, Said, Hassan, Sara, Diallo et Sérigne avec qui j'ai passé des moments

de toute sympathie tout au long de ma thèse.

Enfin à tous ceux qui m'ont soutenu de près ou de loin et à tous ceux qui m'ont incité même involontairement à faire mieux, veuillez trouver ici le témoignage de ma profonde gratitude.

Symboles et Notations

- Étant donné $x \in \mathbb{R}^n$, on désigne par $\|x\|_p$ ($1 \leq p \leq \infty$) la norme p de x , en particulier $\|x\|_2$ et $\|x\|_\infty$ désignent la norme euclidienne et du max de x .
 $B(x, r)$ et $\bar{B}(x, r)$ sont les boules ouverte et fermée de centre x et de rayon r .
- Étant donnée une matrice A carrée d'ordre n , la norme matricielle p ($p \geq 1$) de A est la quantité $\|A\|_p = \sup\{\|Ax\|_p : \|x\|_p = 1\}$.
Si $x \in \mathbb{R}^n$, $X = \text{diag}(x)$ désigne la matrice diagonale dont les éléments diagonaux sont les composantes x_i de x .
- Étant donné $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $f \in \mathcal{C}^2$, on désigne par $\nabla f(x)$ et $\nabla^2 f(x)$ le gradient et le Hessien de f en x .
- Étant donné $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$, on notera par $g'(x)$ ou $J_g(x)$ le Jacobien de g en x , c'est une matrice $m \times n$.
- Étant donné le problème non linéaire suivant

$$\inf\{f(x) : x \in C\}$$

avec $C = \{x \in \mathbb{R}^n : g(x) \leq 0, h(x) = 0\}$

où $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ et $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$,

on notera $\tilde{C} = \{x \in \mathbb{R}^n : g(x) < 0, h(x) = 0\}$ et

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^m \lambda_{i=1} g_i(x) + \sum_{j=1}^p \mu_j h_j(x), \quad \lambda \geq 0$$

C et \tilde{C} sont appelés ensembles des solutions réalisables et strictement réalisables, L est le Lagrangien associé au problème.

INTRODUCTION GÉNÉRALE

Après la deuxième guerre mondiale, la méthode du *simplexe* s'est imposée comme la seule méthode efficace pour la résolution des programmes linéaires (PL). L'apparition de l'algorithme de *Karmarkar* en 1984 a permis le développement des méthodes de points intérieurs qui se sont révélées comme une véritable alternative à la méthode du simplexe.

La méthode du simplexe est basée sur le principe fondamental suivant : la valeur optimale (supposée finie) d'un problème linéaire est atteinte en au moins un sommet (ou point extrémal) du domaine des solutions réalisables, qui est un polyèdre convexe. La méthode du simplexe consiste, alors à parcourir les sommets du polyèdre jusqu'à trouver la solution optimale.

Les méthodes de points intérieurs partent d'un point intérieur au domaine des solutions réalisables, puis au moyen d'une stratégie fixée (trajectoire centrale, fonctions barrières,...) déterminent une valeur approchée de la solution optimale. Les avantages de ces méthodes par rapport à la méthode du simplexe sont nombreux : robustesse, complexité polynomiale et convergence rapide pour les problèmes réels de grandes tailles.

Depuis quelques années plusieurs chercheurs essaient de généraliser le principe de ces méthodes, d'abord pour la programmation mathématique quadratique [7, 47, 85], puis pour la programmation non linéaire [6, 8, 18, 21, 37, 44, 64].

Cette thèse est consacrée à l'étude du problème d'optimisation non linéaire et à sa résolution par des techniques numériques nouvelles. Notre travail de thèse s'articule principalement autour de deux axes.

Le premier est l'étude des méthodes numériques dédiées à l'optimisation

non linéaire, notamment les méthodes qui ont fourni des codes informatique pour résoudre des problèmes d'optimisation. Nous exposerons leurs avantages et leurs inconvénients.

Le deuxième axe est la présentation et l'analyse exhaustive d'une nouvelle méthode de points intérieurs. Plus précisément, nous présentons dans ce travail une étude théorique complète et une nouvelle stratégie qui garantit la stabilité numérique et assure la convergence. Nous exposerons les avantages de cette nouvelle méthode par rapport à celles existantes. Enfin une implémentation de la méthode et des tests numériques sont réalisés afin de montrer la robustesse de notre méthode.

Ce travail commence par une introduction générale à l'optimisation afin de poser le cadre général de notre étude. Cette introduction contient un survol rapide (historique, conditions d'optimalité, convergence) de la programmation linéaire, de la programmation quadratique et de la programmation non linéaire. Elle se termine par une démonstration du lien entre la programmation non linéaire et les méthodes de type barrière.

Cette introduction à l'optimisation et aux méthodes barrières est suivie par une introduction aux algorithmes D.C. (différence de deux fonctions convexes). Le chapitre 1 est consacré à l'optimisation D.C. Nous présentons quelques notions de l'analyse convexe ainsi que l'historique et la base théorique des algorithmes D.C. Le principe des méthodes D.C. consiste à décomposer une fonction non convexe en différence de deux fonctions convexes que nous pourrions traiter plus aisément grâce aux outils de l'optimisation convexe. La suite des itérés générés par une méthode D.C. et sa vitesse de convergence dépendent de cette décomposition, qui n'est pas unique. Nous avons donc intérêt à choisir une décomposition optimale permettant de bien définir la suite des itérés et d'obtenir des algorithmes robustes, performants et rapides. À la fin du chapitre 1

nous proposons une décomposition stable et performante avec des arguments montrant ses avantages et son optimalité. Nous utiliserons la programmation D.C. pour résoudre un problème de minimisation d'une forme quadratique sur une boule centrée à l'origine. Nous montrons que dans le cas sans contraintes, la méthode D.C. est finie et nous présentons des test numériques dans les deux cas convexe et non convexe.

Tous les sous-problèmes étudiés dans cette thèse sont de type quadratique. La méthode que nous utiliserons dans la suite (voir chapitre 5) pour résoudre le problème non linéaire sollicite une programmation quadratique séquentielle (connue sous le terme de SQP), ce qui demande une résolution de deux problèmes quadratiques à chaque itération, d'où la nécessité d'avoir des outils efficaces. Dans le chapitre 2 nous présentons la méthode du gradient conjugué qui est la principale méthode utilisée dans la littérature pour résoudre un problème quadratique sous région de confiance (contrainte quadratique sous forme de boule centrée à l'origine). Nous proposons une méthode D.C. comme alternative. Des comparaisons numériques sont données à la fin de la première section de ce chapitre pour montrer les avantages de notre méthode. La méthode D.C. sera généralisée dans la seconde section afin de résoudre le programme elliptique, c'est-à-dire, un programme dont l'objectif et la contrainte sont quadratiques. La troisième section de ce chapitre fournit plusieurs façons pour calculer la base tangente notamment la méthode QR que nous utiliserons pour réaliser nos simulations.

Le chapitre 3 représente le coeur théorique de notre étude. Nous présentons les conditions d'optimalité en présence de la qualification des contraintes. Ces conditions continuent à être l'objet de discussions au jour d'aujourd'hui, mais la qualification des contraintes n'est pas le seul sujet de discord. L'inconsistance et la redondance des contraintes interpellent beaucoup de chercheurs qui

pensent que ces inconsistances sont à l'origine de la divergence des codes et des méthodes existantes. Les deux dernières sections de ce chapitre traitent respectivement les méthodes primales et les méthodes primales-duales.

Dans le chapitre 4 nous présentons quatre codes qui sont largement utilisés dans le domaine de la programmation non linéaire. Ces méthodes numériques entrent dans le cadre des méthodes newtoniennes. La première méthode utilise la technique de recherche linéaire pour générer le pas de déplacement, alors que les deux suivantes utilisent la technique de région de confiance, ce qui leur assure des propriétés de convergence fortes. Elles traitent la convexité des problèmes mieux que la technique de la recherche linéaire, sur ce point, les méthodes utilisant une région de confiance sont plus robustes et permettent de trouver des solutions dans des cas très difficiles, tels ceux que l'on rencontre en commande optimale ou après discrétisation d'équations aux dérivées partielles. Le dernier code a été conçu pour résoudre des problèmes de grandes tailles avec un nombre limité de contraintes actives à l'optimum, l'intérêt de cette présentation est de vérifier le gain en efficacité pour des codes conçus sur mesure.

Dans le chapitre 5 nous décrivons les conditions d'optimalité du problème non linéaire. Ces conditions donnent lieu à un système qui sera perturbé par la pénalisation avec la technique des points intérieurs. Le système est ensuite préconditionné en utilisant un changement de variable qui est à l'origine de la stabilité et l'efficacité de notre méthode par rapport à celles existantes. De nombreux tests numériques viennent confirmer nos affirmations. L'efficacité de notre code utilisant la technique de la région de confiance et renforcé par l'application des algorithmes D.C. est présentée dans ce chapitre. Différentes simulations numériques nous aideront à mieux ajuster tous les paramètres dont le code a besoin et à choisir la meilleure fonction de mérite adéquate à notre

méthode. la section 5.9 et 5.10 montrent respectivement que la méthode ne souffre pas du mauvais conditionnement et qu'elle est globalement convergente.

Nous annexons à cette thèse, une présentation des méthodes de simplexe et de Karmarkar pour résoudre des problèmes linéaires et une section sur la modélisation dans l'environnement CUTEr et AMPL.

Nous avons fixé deux axes de recherche tout au long de cette thèse, l'étude des méthodes dédiées à l'optimisation non linéaire et la proposition de nouvelles méthodes. Notre expérience a montré que plusieurs méthodes souffrent énormément du mal-conditionnement, et l'étude approfondie de la base théorique de certaines d'entre elles a conduit à proposer une correction des conditions d'optimalité. Cette correction a fourni un gain concernant la vitesse de convergence de la méthode SDC proposée. L'efficacité observée a été renforcée par la résolution des sous-problèmes quadratiques par une méthode D.C. ce qui nous a permis de gagner au niveau du temps de calcul et surtout de réduire significativement les dimensions des problèmes quadratiques non convexes qu'il a fallu résoudre.

Nous pensons que la méthode peut encore évoluer dans deux sens, d'abord en profitant de la nature des matrices à savoir qu'elles soient creuses, puis en gérant mieux la fonction de mérite pondérée qui nécessite plus d'ajustement dynamique.

TABLE DES MATIÈRES

DÉDICACE	vi
REMERCIEMENTS	viii
INTRODUCTION GÉNÉRALE	xiv
Les méthodes barrières en optimisation	1
0.1 Introduction	1
0.2 La programmation mathématique	2
0.2.1 La programmation linéaire	2
0.2.2 La programmation quadratique	3
0.2.3 La programmation non linéaire	6
0.3 Les méthodes de type barrière	8
CHAPITRE 1 :INTRODUCTION AUX ALGORITHMES D.C.	13
1.1 Rappels de quelques outils d'analyse convexe	13
1.1.1 Définitions et notations	13
1.1.2 La fonction conjuguée	15
1.1.3 Le sous-différentiel	15
1.2 Optimisation D.C.	17
1.2.1 Historique	17
1.2.2 La base théorique	18
1.2.3 Algorithme D.C.	20
1.3 Exemples d'algorithmes D.C.	21
1.3.1 Décomposition classique	22
1.3.2 Décomposition optimale	23

1.3.3	Le cas quadratique sans contrainte	24
1.4	Discussion	26
1.4.1	Le cas quadratique convexe	27
1.4.2	Le cas quadratique non convexe	33

CHAPITRE 2 : LES OUTILS POUR LA PROGRAMMATION

	QUADRATIQUE	37
2.1	Le problème quadratique (RC)	39
2.1.1	Le point de Cauchy	40
2.1.2	Le gradient conjugué selon Steihaug	42
2.1.3	La méthode D.C.	44
2.1.4	Comparaison numérique	46
2.1.5	Autres alternatives pour la résolution du problème quadratique	49
2.2	Problème Quadratique sous Contrainte Quadratique (PQCQ)	51
2.2.1	Qualification des contraintes et conditions d'optimalité	53
2.2.2	Résolution du problème PQCQ	54
2.3	Calcul de bases tangentes et d'inverses à droite	59
2.3.1	Méthode 1 : factorisation QR	59
2.3.2	Méthode 2 : partitionnement de A_k^t	64
2.3.3	Méthode 3 : factorisation LU de A_k	65
2.3.4	Calcul d'un inverse à droite par les équations normales	65
2.3.5	Calcul d'un inverse à droite par système augmenté	66

CHAPITRE 3 : LA PROGRAMMATION NON LINÉAIRE 67

3.1	Définitions et notations	68
3.2	Conditions d'optimalité	72
3.3	Inconsistance et redondance des contraintes	74

3.4	Théorèmes de convergence	78
3.5	Méthodes primales	83
3.5.1	Calcul du pas de déplacement	84
3.5.2	Newton classique	84
3.5.3	Système augmenté	86
3.5.4	Extrapolation	86
3.5.5	Les fonctions barrières modifiées	87
3.6	Méthodes primales-duales	87
3.6.1	Les méthodes directes	89
3.6.2	La programmation quadratique séquentielle	90
3.6.3	Fonctions de mérite	93
3.6.4	Le cas non convexe	96
CHAPITRE 4 : MÉTHODES NUMÉRIQUES		97
4.1	Introduction	97
4.2	La méthode LOQO	98
4.3	La méthode KNITRO	102
4.4	La méthode filterSQP	107
4.5	La méthode SNOPT	111
4.6	Évaluation numérique	114
4.7	Étude antérieure	116
CHAPITRE 5 : LA MÉTHODE SDC		119
5.1	Les conditions d'optimalité	121
5.2	Hypothèses	125
5.3	Résolution des sous problèmes	126
5.4	Les multiplicateurs de Lagrange	134
5.5	Les contraintes emboîtées	136

5.6	La fonction de mérite	138
5.7	Correction du second ordre	139
5.8	Tests numériques	142
5.8.1	Impact du changement de variable	142
5.8.2	Choix d'une fonction de mérite	161
5.9	Le problème du conditionnement	187
5.10	Étude de la convergence	189
5.10.1	Cadre général de la convergence globale de la méthode des points intérieurs	190
5.10.2	Convergence de la méthode SDC	195
5.11	Conclusion	197
	CONCLUSION	199
	ANNEXE A : LA MÉTHODE DU SIMPLEXE	203
A.1	Notations	203
A.2	Algorithme du Simplexe	205
	ANNEXE B : LA MÉTHODE DE KARMARKAR	207
B.1	Résolution d'une fonction linéaire sur un sphère	207
B.2	L'algorithme	211
	ANNEXE C : MODÉLISATION	213
C.1	L'environnement CUTEr	215
C.2	Le langage de modélisation AMPL	243
	BIBLIOGRAPHIE	247

TABLE DES FIGURES

1	Exemple de problème non linéaire.	8
1.1	Comparaison en fonction de la valeur optimale.	28
1.2	Comparaison en fonction du nombre d'itérations.	29
1.3	Comparaison en fonction du temps CPU.	30
1.4	Comparaison de la valeur optimal avec et sans région de confiance.	31
1.5	Comparaisons des temps CPU pour deux valeurs de test d'arrêt.	32
1.6	Temps de calcul CPU pour le test d'arrêt le plus faible.	33
1.7	Comparaison en fonction de la valeur optimale.	34
1.8	Comparaison en fonction du temps CPU.	35
1.9	Comparaison en nombres d'itérations.	36
2.1	Comparaison par rapport à la valeur de la fonction objectif produite par chaque méthode.	46
2.2	Comparaison par rapport au nombre d'itérations que nécessite chaque méthode.	47
2.3	Comparaison par rapport au temps CPU que nécessite chaque méthode.	48
5.1	Exemple de sous-problème quadratique inconsistant.	127
5.2	Exemple de sous-problème quadratique réalisable, dans ce cas v est nul.	128
5.3	Exemple de sous-problème quadratique inconsistant perturbé et redevenu réalisable grâce à la programmation verticale.	128
B.1	Les chemins pris par la méthode du Simplexe et la méthode de Karmarkar	208

B.2 Illustration de la solution d'un problème linéaire sur un cercle . 209

LISTE DES ALGORITHMES

1	Algorithme DC-rajout	22
2	Algorithme DC-optimal	23
3	Calcul de point de Cauchy	41
4	L'algorithme de Steihaug	43
5	Algorithme DC	45
6	L'algorithme de Hebden	51
7	Programmation PQCQ, Procédure 1	58
8	Programmation PQCQ, Procédure 2	58
9	Programmation PQCQ, Procédure 3	59
10	L'orthogonalisation de Gram-Schmidt	63
11	Algorithme filtreSQP	110
12	L'algorithme de base de la résolution DC	131
13	Procédure DC pour résoudre le problème horizontal	132
14	Procédure DC pour résoudre le problème vertical	134
15	Correction du second ordre	140
16	L'algorithme général de la méthode SDC	141
17	Règle pour choisir le paramètre de pénalisation (règle 1)	164
18	Mise à jour du paramètre de pénalisation (règle 2)	187
19	L'algorithme du Simplexe	205
20	L'algorithme de base de Karmarkar	212
21	Les commandes de base pour le langage AMPL	244
22	Utiliser le code SDC en langage AMPL	244

LISTE DES TABLEAUX

5.1	Effet du changement de la variable sur un ensemble de problèmes de la bibliothèque CUTEr	147
5.2	Quand une composante de S s'approche de zéro le modèle explose.	150
5.3	Résolution du problème HS10 par la méthode 1.	154
5.4	Résolution du problème HS15 par la méthode 1.	155
5.5	Résolution du problème HS15 par la méthode 2.	155
5.6	Résolution du problème SIMPLLPA par la méthode 1.	157
5.7	Résolution du problème SIMPLLPA par la méthode 2.	157
5.8	Résolution du problème ZECEVC4 par la méthode 1.	159
5.9	Résolution du problème ZECEVC4 par la méthode 2.	161
5.10	Résultats en fonction de l'objectif pour les différentes fonctions de mérite.	170
5.11	Résultats en fonction du défaut de réalisabilité pour les différentes fonctions de mérite	175
5.12	Résultats en fonction du nombre d'itérations pour les différentes fonctions de mérite.	179
5.13	Résultats pour la fonction de mérite pondérée	182
5.14	la moyenne du nombre d'itérations normalisée par rapport au nombre de problèmes résolus	184
5.15	Résultats de la comparaison de l'objectif après révision	184
5.16	Résultats de la comparaison du défaut de réalisabilité après révision	185
C.1	Principales familles de problèmes continus.	214
C.2	Index des problèmes de la base CUTEr utilisées dans les expériences numériques.	243

CHAPITRE INTRODUCTIF

LES MÉTHODES BARRIÈRES EN OPTIMISATION

0.1 Introduction

Dans ce chapitre introductif, nous allons présenter les problèmes classiques d'optimisation, leurs conditions d'optimalité et définir quelques notions usuelles.

Un problème d'optimisation consiste à minimiser une fonction f définie sur $E \subset \mathbb{R}^n$ à valeurs réelles, $f : E \rightarrow \mathbb{R}$. Cette fonction est appelée *fonction objectif* et l'ensemble E est dit *domaine réalisable* ou ensemble des solutions réalisables. La formulation mathématique de ce problème est

$$(\mathcal{P}_{\text{inf}}) \begin{cases} \text{Trouver } x^* \in E \text{ tel que} \\ f(x^*) = f^* = \inf_{x \in E} f(x) \end{cases}$$

En toute généralité, la fonction objectif f peut ne pas avoir de borne inférieure et si elle en admet une, cette borne inférieure peut ne pas être atteinte. Dans le cas contraire, i.e. si f admet un minimum, le problème s'écrit alors

$$(\mathcal{P}_{\text{min}}) \begin{cases} \text{Trouver } x^* \in E \text{ tel que} \\ f(x^*) = \min_{x \in E} f(x) \end{cases}$$

Nous nous intéressons par la suite, aux problèmes où f admet un minimum et nous noterons x^* une solution de $(\mathcal{P}_{\text{min}})$. Une solution de $(\mathcal{P}_{\text{min}})$ est aussi appelée optimum, ou plus précisément optimum global.

On dit que x^* est un minimum local (ou optimum local) de f s'il existe un voisinage V de x^* dans E tel que x^* est un optimum de $f|_V$. $f|_V$ représente la restriction de f à V .

On dit que x^* est un minimum strict de f s'il existe un voisinage V de x^* dans E tel que

$$\forall x \in V, x \neq x^*, f(x) > f(x^*).$$

Un optimum local x^* est dit isolé s'il existe un voisinage V de x^* qui ne contient aucun optimum local autre que x^* .

0.2 La programmation mathématique

0.2.1 La programmation linéaire

La programmation linéaire traite la résolution des problèmes d'optimisation pour lesquels la fonction objectif et les contraintes sont affines. Sakarovitch [75] la considère comme étant la technique la plus célèbre de la recherche opérationnelle, celle qui a donné à cette discipline sa notoriété.

Un problème de programmation linéaire mis sous la forme standard peut être défini comme suit

$$m_p = \min_x \{c^t x : x \geq 0, Ax = b\} \quad (P)$$

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

Tout problème de programmation linéaire peut être mis sous cette forme.

Au problème (P), on associe son problème dual

$$m_d = \max_y \{b^t y : A^t y \leq c\} \quad (D)$$

On a toujours $-\infty \leq m_d \leq m_p \leq +\infty$.

Dès lors qu'un de ces deux problèmes est réalisable on a $m_d = m_p$. Si les deux problèmes sont réalisables alors $-\infty < m_d = m_p < +\infty$ et les deux

problèmes admettent des solutions optimales. Si un des deux problèmes admet une solution optimale, les deux sont réalisables.

Supposons que les deux problèmes sont réalisables, si \bar{x} est solution optimale de (P), les solutions du problème dual (D) sont les $y \in \mathbb{R}^m$ tel que $y \geq 0$, $A^t y \geq c$ et $y^t(b - A\bar{x}) = 0$.

Si \bar{y} est solution optimale de (D), les solutions de (P) sont les $x \in \mathbb{R}^n$ tel que $x \geq 0$, $Ax = b$ et $\bar{y}^t(b - Ax) = 0$.

La condition $\bar{y}^t(b - A\bar{x}) = 0$ liant les solutions optimales de (P) et (D) est appelée condition de complémentarité.

0.2.2 La programmation quadratique

La programmation quadratique est la minimisation d'une fonction objectif quadratique sous des contraintes linéaires. L'importance de la programmation quadratique provient du fait que plusieurs problèmes réels et académiques sont quadratiques, c'est le cas de l'optimisation du portefeuille dans la finance par exemple. Dans notre travail, l'optimisation quadratique est avant tout un outil de résolution de sous-problèmes. En effet, des techniques telles que la programmation quadratique séquentielle (voir paragraphe 3.6.2) proposent de traduire un programme non linéaire en une suite de problèmes quadratiques, en général non convexes, d'où la nécessité de disposer de méthodes numériques efficaces et robustes qui puissent résoudre toute sorte de problèmes quadratiques, convexe ou non, surtout de grande taille. C'est en effet, dans cet état d'esprit que nous proposerons les méthodes D.C. au chapitre 1 de cette thèse.

Un problème quadratique s'écrit sous la forme suivante,

$$\left\{ \begin{array}{ll} \min \frac{1}{2}x^T Qx + c^T x & \\ \text{s.c.} & \\ a_i^t x = b_i & i \in \mathcal{E} \\ a_i^t x \leq b_i & i \in \mathcal{I} \\ x \in \mathbb{R}^n & \end{array} \right. \quad (1)$$

où Q est une matrice symétrique $n \times n$, c est un vecteur de \mathbb{R}^n , $a_i \in \mathbb{R}^n$ et $b_i \in \mathbb{R} \forall i \in \mathcal{E} \cup \mathcal{I}$ où $\mathcal{E} = \{1, \dots, m\}$ et $\mathcal{I} = \{m+1, \dots, m+p\}$.

Si Q est une matrice semi-définie positive, le problème (1) est convexe, et on montre, dans ce cas que la solution, si elle existe, est unique.

Les résultats d'optimalité de la programmation linéaire du paragraphe précédent peuvent se généraliser au cas quadratique. Nous notons donc, $L(x, \lambda)$, le Lagrangien du problème (1) :

$$L(x, \lambda) = \frac{1}{2}x^T Qx + c^T x + \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i (a_i^t x - b_i) \quad (2)$$

Nous définissons par l'ensemble actif $\mathcal{A}(x^*)$ au point optimal x^* , l'ensemble des indices pour lesquels la contrainte en x^* est une égalité. Ces définitions nous permettent d'écrire les conditions nécessaires d'optimalité de premier ordre. En effet si x^* est une solution optimale du problème (1), alors il existe un vecteur λ^* dans \mathbb{R}^{m+p} tel que

$$\left\{ \begin{array}{ll} \nabla_x L(x^*, \lambda^*) = 0 & \\ a_i^t x^* - b_i = 0 & \forall i \in \mathcal{E} \\ a_i^t x^* - b_i \leq 0 & \forall i \in \mathcal{I} \\ \lambda_i^* \geq 0 & \forall i \in \mathcal{I} \\ \lambda_i^* (a_i^t x^* - b_i) = 0 & \forall i \in \mathcal{I} \end{array} \right. \quad (3)$$

Ces conditions sont aussi appelées conditions de Karush-Kuhn-Tucker (KKT). La dernière condition dans le système (3), appelée la condition de complémentarité, implique que $\lambda_i^* = 0$ pour $i \notin \mathcal{A}(x^*)$ ainsi

$$\nabla_x L(x^*, \lambda^*) = \nabla_x f(x^*) + \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* (a_i^t x^* - b_i) = 0 \quad (4)$$

Une condition suffisante pour que x^* soit un minimum local est que $Z^t Q Z$ soit définie positive, où Z est la base de l'espace généré par le noyau de la matrice jacobienne $[a_i^t]_{i \in \mathcal{A}(x^*)}$. Si Q est définie positive alors l'optimum est unique et tout minimum local est forcément global. En plus des difficultés d'unicité de l'optimum dans le cas non convexe, toute construction algorithmique est confrontée à la dégénérescence, qui peut être soit la dépendance linéaire des vecteurs de la matrice jacobienne, ce qui rendrait le calcul des éléments de Z plus complexe, soit la faiblesse d'activité d'une contrainte, dans ce cas l'algorithme aura du mal à décider si la contrainte est active ou non, par exemple on peut avoir $\lambda_i^* = 0$ pour un certain indice dans $\mathcal{A}(x^*)$.

Un problème particulièrement important et que nous traitons à plusieurs reprises dans ce travail est issu de la méthode de région de confiance. Il s'agit de minimiser une fonction \mathcal{C}^2 non nécessairement convexe sur l'espace \mathbb{R}^n tout entier (problème d'optimisation sans contraintes).

Partant de $x_0 \in \mathbb{R}^n$ et $r_0 > 0$ arbitraires, la méthode de région de confiance construit une suite (x_k, r_k) de la façon suivante :

- Calculer $g_k = \nabla f(x_k)$ et $H_k = \nabla^2 f(x_k)$.
- Déterminer d_k solution optimale de

$$\min_d \left\{ \frac{1}{2} \langle H_k d, d \rangle + \langle g_k, d \rangle : \|d\| \leq r_k \right\}$$

$$\tilde{x}_k = x_k + d_k.$$

- Accepter $x_{k+1} = \tilde{x}_k$ si \tilde{x}_k donne une descente suffisante de f et augmenter la valeur de r_{k+1} ; sinon diminuer r_k et rejeter la direction d_k .

Le problème à chaque itération est donc du type

$$\min_d \left\{ \frac{1}{2} \langle Hd, d \rangle + \langle g, d \rangle : \|d\| \leq \delta \right\} \quad (RC)$$

Cette approche de région de confiance a été introduite en 1944 par Levenberg [52] et Marquard en 1963 [56] pour résoudre des problèmes de moindres carrés et a été généralisée aux problèmes sans contraintes par Powell [69]. Cette technique a montré son efficacité d'une part pour la convergence globale (voir El Hallabi [37], Powell [69, 70], et Schultz, Schnabel et Byrd [74]) et d'autre part que la définie-positivité de W n'est pas nécessaire sous une région de confiance. Cette possibilité de traiter la non-convexité du problème non linéaire est un point fort de la technique. Si la région de confiance est inactive, le pas produit par la méthode sera exactement le pas de Newton, en revanche si la taille de la région de confiance est réduite, cela aura pour conséquence de rendre la direction du pas proche de la plus profonde descente du modèle, c'est-à-dire, l'approche produit au pire, la direction de plus profonde descente (steepest descent) (cf. El Hallabi et Tapia [36], El Hallabi [37] et Nocedal et Wright [62]).

0.2.3 La programmation non linéaire

La programmation non linéaire est la recherche de l'optimum d'une fonction non linéaire sur un sous-ensemble convexe ou non d'un espace donné.

Le problème non linéaire s'écrit sous la forme

$$\left\{ \begin{array}{l} \text{Minimiser } f(x) \\ \text{sous les contraintes :} \\ h(x) = 0 \\ g(x) \leq 0 \end{array} \right. \quad (PNL)$$

où $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$, $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ et $x \in \mathbb{R}^n$.

Ainsi la programmation non linéaire se présente comme étant une généralisation de la programmation quadratique et de la programmation linéaire.

Bien que cette généralisation soit évidente de nos jours, cet esprit d'unification n'est apparu qu'après 1984, date de l'introduction de l'algorithme de Karmarkar [48] pour la programmation linéaire. Les recherches qui ont suivi [23, 2, 6, 7, 8, 18, 47] ont pu bâtir des ponts entre deux mondes qui étaient jusqu'à alors tout à fait étrangers l'un à l'autre, nous parlerons depuis d'une révolution [62, 89].

Vu que l'ensemble des contraintes (le domaine réalisable) n'est pas généralement convexe, la solution n'est pas nécessairement unique, et par suite, nos méthodes essaient de trouver une solution optimale locale, d'où l'importance de l'étape de l'initialisation. Dans le cas où l'ensemble réalisable $\{x \in \mathbb{R}^n / h(x) = 0, g(x) \leq 0\}$ est strictement convexe, le point retrouvé est bien entendu l'unique solution optimale globale ou une approximation de celle-ci.

L'exemple suivant est exposé par Fiacco et McCormick dans [19].

$$\left\{ \begin{array}{l} \text{minimiser } f(x) = |x_1 - 2| + |x_2 - 2| \\ \text{s.c.} \\ h(x) = x_1^2 + x_2^2 - 1 = 0 \\ g(x) = x_2^2 - x_1 \leq 0 \end{array} \right. \quad (5)$$

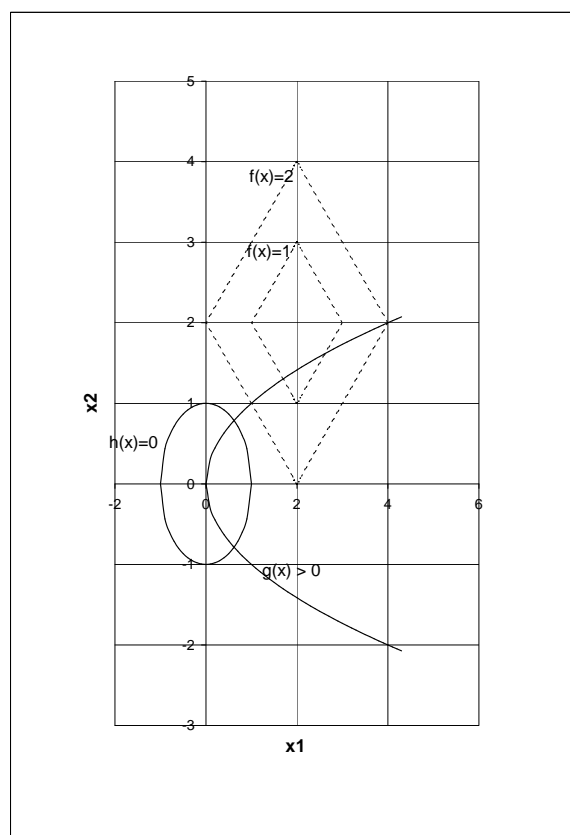


FIG. 1 – Exemple de problème non linéaire.

Le problème (5) est un problème convexe, qui admet le point $x^* = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$ comme optimum global pour lequel $f(x^*) = 4 - \sqrt{2}$. En prohibant les contraintes, la solution devient $x^* = (2, 2)$ avec une valeur optimale nulle de la fonction objective.

0.3 Les méthodes de type barrière

Dans leur livre, “*Non linéaire Programming*”, Anthony V. Fiacco et Garth P. McCormick [19], datent les débuts des méthodes barrières aux années quarante. En effet, en 1943 R. Courant [15] suggère l’étude des conditions de stationnarité de l’application $f(x) + t.g^2(x)$ quand $t \rightarrow +\infty$ afin d’avoir des informations

sur l'optimum du problème (6) d'égalité suivant,

$$\begin{cases} \min f(x) \\ \text{s.c.} \\ g(x) = 0 \end{cases} \quad (6)$$

Cette suggestion a été motivée par des considérations physiques et n'a jamais eu de suite théorique ou algorithmique pour résoudre le problème mathématique non linéaire.

En 1951, Kuhn et Tucker [51] publient leurs résultats concernant les conditions nécessaires et suffisantes caractérisant les solutions du problème non linéaire convexe et établissant l'équivalence entre ce problème et la définition du point-col (ou point-selle).

Cette année 1951 a vu également l'utilisation pour la première fois des techniques de gradients pour résoudre le problème non linéaire [3], s'en est suivi par une avalanche de variantes de ces méthodes de potentiel, dont les plus sérieuses étaient la méthode potentielle logarithmique de Frisch en 1954 [26] : $f(x) + \sum_{i=1}^m \alpha_i \ln g_i(x)$, celle de Ablow et Brigham en 1955 [1] (la contrainte d'inégalité étant $g(x) \geq 0$) : $f(x) - t \sum_{i=1}^m \min(g_i(x), 0)$ ou la fonction potentiel $f(x) + t \sum_{i=1}^m (\min(g_i(x), 0))^2$ pour les problèmes avec égalités ou aussi Carroll en 1959 [11] : $f(x) + \sum_{i=1}^m \frac{1}{g_i(x)}$.

Bien que les méthodes de type barrière ont vu le jour dans les années cinquante, ce n'est qu'aux années soixante que leur base théorique a été établie et ce grâce notamment aux travaux de Fiacco et McCormick. Le terme de *méthodes de points intérieurs* a été introduit pour la première fois dans leur livre : "*Nonlinear Programming, Sequential Unconstrained Minimization Techniques*", publié en 1968 [19]. Dans ce livre, les auteurs ont mis en valeur la relation entre les suites des solutions des fonctions de pénalité et les

solutions du problème initial. Or des études munies séparément à la fin des années soixante par Lootsma et Murray [53, 60] ont mis en évidence que le mal-conditionnement du Hessien de la fonction barrière s'aggravait au fur et à mesure qu'on s'approchait de la solution. Ce Hessien devient singulier à la limite. En parallèle à ces travaux, d'autres méthodes (SQP et Lagrangien augmenté) qui ne souffriraient pas de ce problème de mal-conditionnement se développaient, ce qui n'aménagea en rien les méthodes de points intérieurs, qui furent totalement abandonnées aux années 70. Pour relancer le débat au sujet de ces méthodes, il a fallu attendre la publication en 1984 de l'article de Karmarkar [48] qui proposa une méthode barrière projective afin de résoudre le problème linéaire (voir l'annexe B). L'intérêt de sa méthode est sa complexité polynomiale. En effet la méthode du simplexe, élaborée par Dantzig dans les années 40, n'a pas cessé d'évoluer mais sans assurer la convergence théorique qu'en un nombre exponentiel d'itérations.

Dans les années 90, plusieurs chercheurs ont essayé de généraliser les méthodes de points intérieurs du cas linéaire pour les appliquer au cas quadratique [47, 62] puis au cas non linéaire [8, 9, 58, 85]. En règle générale, une variable de rajout est introduite afin de transformer le problème non linéaire en une suite de minimisation d'une fonction potentiel avec barrière logarithmique sous des contraintes d'égalité.

Ainsi le problème non linéaire :

$$\left\{ \begin{array}{l} \min f(x) \\ \text{s.c.} \\ h(x) = 0 \\ g(x) \leq 0 \end{array} \right. \quad (PNL)$$

se transforme en une série de problèmes dits problèmes barrières ou pénalisés

$$\begin{cases} \min f(x) - \mu \sum_{i=1}^m \ln s_i \\ \text{s.c.} \\ h(x) = 0 \\ g(x) + s = 0 \end{cases} \quad (P_\mu)$$

où s est la variable auxiliaire, s_i est sa i^{me} composante : $s \in \mathbb{R}_+^m$ et $\mu > 0$ est dit paramètre de pénalisation.

En 1994, Austreicher et Vial ont développé une méthode de résolution du problème non linéaire convexe en résolvant une séquence de problèmes de barrière [4]. D'autres auteurs ont utilisé la méthode de Newton pour résoudre les conditions (*KKT*) du problème non linéaire non convexe ; El-Bakry, et cie montrent en 1999 [18] que sous certaines hypothèses, leur algorithme primal-dual converge globalement, en plus pour des valeurs μ_k bien choisies, la convergence est quadratique. Malheureusement cette méthode garantit seulement la condition du premier ordre, mais l'obtention d'une solution optimale n'a pas pu être démontrée.

Une quantité impressionnante de méthodes barrières ont été proposées à la fin du 20^e siècle mais juste une partie a été réellement implémentée parmi lesquelles : LOQO de Vanderbrei et Shanno [85], KNITRO de Nocedal et Waltz [8, 9], Minos de Murtagh et Saunders [58], SNOPT de Gill, Muray et Saunders [28], filterSQP de Fletcher et Leyffer [21] et MITR de Jonsson et Gilbert [46].

La plus grande partie de ces codes construit une suite de problèmes quadratiques, ce sont les techniques SQP que nous aurons le temps de présenter dans le paragraphe 3.6.2 du troisième chapitre. Deux familles d'approches se partagent la résolution de ces sous problèmes quadratiques : la recherche linéaire et la région de confiance, mais depuis la publication de l'article de Beigler et

Wächter [5] la balance se met plutôt au profit des méthodes de la région de confiance.

Au début de ce siècle les chercheurs essaient plutôt de rendre les méthodes existantes plus efficaces en résolvant une large variété de problèmes. Dans ce contexte s'impose notre premier apport personnel, à savoir le conditionnement de la matrice hessienne W . Nous revenons à cela lors de la présentation de notre code SDC.

Notre méthode SDC fait partie de la famille SQP-Région de confiance. Nous avons introduit la résolution de ces sous-problèmes quadratiques par les méthodes D.C., dont nous en exposerons les avantages au chapitre 1. En plus du conditionnement du Hessien W et de l'introduction de la méthode D.C., le chapitre 5 présente de nouvelles variantes de la fonction de mérite, suivies d'une étude complète de la convergence.

CHAPITRE 1

INTRODUCTION AUX ALGORITHMES D.C.

Ce chapitre est composé de quatre sections. La première section rappelle les notions de base d'analyse convexe qui nous serviront pour décrire l'optimisation D.C. La deuxième section présente un bref historique des méthodes D.C.. Nous énonçons les principaux résultats théoriques concernant ces méthodes, notamment le théorème de convergence globale. Ces résultats sont dus à Yassine [93]. Dans la troisième section nous fixons la décomposition que nous allons choisir pour construire notre algorithme, ce dernier sera utilisé dans toute la suite. La dernière section est une étude numérique qui nous aidera à répondre à la question suivante : quel seuil faut-il prendre pour le test d'arrêt ? Nous ferons la différence dans cette étude entre les problèmes convexes et non convexes.

1.1 Rappels de quelques outils d'analyse convexe

1.1.1 Définitions et notations

Définition. 1.1.1. (*La convexité*)

Un ensemble C est dit convexe si et seulement si, $\forall x \in C, \forall y \in C$ et $\forall \lambda \in [0, 1]$ alors

$$\lambda x + (1 - \lambda)y \in C \quad (1.1)$$

Une application f définie sur \mathbb{R}^n dans \mathbb{R} est dite convexe sur C si et seulement si $\forall x \in C, \forall y \in C$ et $\forall \lambda \in [0, 1]$ alors

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) \quad (1.2)$$

f est dite strictement convexe si l'inégalité ci-dessus est toujours stricte $\forall x \neq y$ et $\lambda \in]0, 1[$.

f est dite fortement convexe ou coercive de coefficient $\rho \geq 0$ si et seulement si

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) - \frac{\lambda(1 - \lambda)}{2} \rho \|x - y\|^2 \quad (1.3)$$

$\forall x \in C, \forall y \in C$ et $\forall \lambda \in [0, 1]$

□

Définition. 1.1.2. L'épigraphe d'une fonction f noté $\text{epi}(f)$ est l'ensemble défini par :

$$\text{epi}(f) = \{(\mu, x) \in \mathbb{R}^{n+1} \mid f(x) \leq \mu, x \in \mathbb{R}^n, \mu \in \mathbb{R}\} \quad (1.4)$$

□

Propriété. 1.1.1. Une fonction f est convexe si et seulement si $\text{epi}(f)$ est un ensemble convexe.

□

Définition. 1.1.3. On appelle domaine effectif de définition d'une fonction convexe f l'ensemble, noté $\text{dom}(f)$, défini par :

$$\text{dom}(f) = \{x \in \mathbb{R}^n \mid f(x) < +\infty\} \quad (1.5)$$

□

Remarque. 1.1.1. L'ensemble $\text{dom}(f)$ est la projection de $\text{epi}(f)$ sur \mathbb{R}^n .

□

Définition. 1.1.4. On dit qu'une fonction convexe f est propre si $\text{dom}(f)$ est non vide et si $f(x) > -\infty, \forall x \in \text{dom}(f)$.

□

1.1.2 La fonction conjuguée

Définition. 1.1.5. Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction convexe. On appelle fonction conjuguée de f , notée f^* , la fonction définie par :

$$\forall y \in \mathbb{R}^n, \quad f^*(y) = \sup_{x \in \mathbb{R}^n} \{y^t x - f(x)\}$$

La fonction conjuguée de f^* , notée f^{**} , est appelée fonction bi-conjuguée de f .

On dit que f est une fonction convexe fermée si et seulement si $f^{**} = f$

□

Définition. 1.1.6. Une fonction f est dite semi-continue inférieurement (sci) si

$$\liminf_{x \rightarrow x^*} f(x) = f(x^*)$$

On note $\Gamma_0(\mathbb{R}^n)$ l'ensemble des fonctions convexes, propres et s.c.i.

□

1.1.3 Le sous-différentiel

Pour une fonction convexe f différentiable, le gradient $\nabla f(x^*)$ en tout point x^* vérifie

$$f(x) \geq f(x^*) + \nabla f^t(x^*)(x - x^*) \quad (1.6)$$

Définition. 1.1.7. On appelle sous-gradient de f au point x^* tout vecteur $y \in \mathbb{R}^n$ vérifiant

$$f(x) \geq f(x^*) + y^t(x - x^*) \quad (1.7)$$

L'ensemble de tous les sous-gradients de f en x^* , noté $\partial f(x^*)$ est appelé sous différentiel de f au point x^* .

□

Rockafellar a montré en 1970 que toute fonction convexe propre définie sur \mathbb{R}^n admet un sous-gradient en tout point x^* appartenant à l'intérieur du domaine effectif de f , et que l'ensemble $\partial f(x^*)$ est convexe fermé.

Proposition. 1.1.1. y^* est un sous-gradient de f en x^* si et seulement si

$$f(x^*) + f^*(y^*) = (y^*)^t x^* \quad (1.8)$$

Preuve.

$$\begin{aligned} y^* \in \partial f(x^*) &\Leftrightarrow f(x) \geq f(x^*) + y^{*t}(x - x^*) \quad \forall x \in \mathbb{R}^n \\ &\Leftrightarrow (x^*)^t y^* - f(x^*) \geq y^{*t}x - f(x) \quad \forall x \in \mathbb{R}^n \\ &\Leftrightarrow (x^*)^t y^* - f(x^*) \geq \sup\{y^{*t}x - f(x) \mid x \in \mathbb{R}^n\}. \end{aligned}$$

Or x^* est un élément de \mathbb{R}^n donc on peut remplacer l'inégalité par une égalité.

CQFD □

Proposition. 1.1.2. Soit f une fonction convexe propre, x^* minimise f si et seulement si $0 \in \partial f(x^*)$.

Preuve.

$$\begin{aligned} 0 \in \partial f(x^*) &\Leftrightarrow f(x) \geq f(x^*) + 0 \cdot (x - x^*) \quad \forall x \in \mathbb{R}^n \\ &\Leftrightarrow x^* \text{ minimise } f \text{ sur } \mathbb{R}^n. \end{aligned}$$

1.2 Optimisation D.C.

1.2.1 Historique

L'optimisation D.C. (Différence de deux fonctions Convexes) consiste à résoudre le problème d'optimisation suivant :

$$(P) : \text{Min}\{f(x) = R(x) - T(x) \mid x \in \mathbb{R}^n\} \quad (1.9)$$

où R et T sont des fonctions convexes.

Le développement des algorithmes de sous gradients a commencé dans les années 70 (Rockafellar R.T. [73]) pour résoudre des problèmes d'optimisation non linéaire et non différentiable. C'est à la fin des années 70 que J.F. Toland [80, 81, 82] est arrivé à mettre en évidence les conditions d'optimalité locale et la dualité D.C.

Ces études ont été poursuivies dans les années 80 par plusieurs chercheurs surtout C. Lemarechal et J. B. Hiriart-Urruty [41, 42, 43]. Ce dernier a démontré l'optimalité globale des solutions dans le cas où T est polyédrale [42]. Il fallait attendre les années 90 pour que le développement des algorithmes de sous-gradients prend une grande valeur algorithmique et une immense importance pratique avec l'apparition d'une nouvelle génération d'algorithmes de sous-gradients dits DCA, développés par l'équipe d'optimisation de Grenoble, pour la résolution du problème (P). Ces algorithmes sont apparus dans les travaux de T. Pham Dinh [67] et par suite dans ceux de A. Yassine [91, 92, 93]. Malgré l'absence d'une preuve théorique de leur globalité, les algorithmes DCA ont été appliqués avec succès à de nombreux problèmes concrets de grande dimension [91, 92]. Ils ont fourni des solutions optimales globales pour certains problèmes. La globalité des solutions obtenues est vérifiée grâce à la connaissance à priori de la valeur optimale du problème traité, ou par comparaison

avec des méthodes globales. C'est le cas, par exemple, du problème MDS [91] et le calcul des valeurs propres extrêmes [92].

1.2.2 La base théorique

La problématique DC est donc la résolution du problème (1.2.2) définie par

$$m_p = \min\{f(x) = R(x) - T(x) \mid x \in \mathbb{R}^n\} \quad (P)$$

où R et T sont des fonctions de $\Gamma_0(\mathbb{R}^n)$.

Le problème dual associé au problème primal (P) est défini par :

$$m_d = \min\{T^*(y) - R^*(y) \mid y \in \mathbb{R}^n\} \quad (D)$$

Théorème. 1.2.1. *La dualité DC est stable c'est-à-dire le saut de dualité $m_d - m_p$ est nul.*

Preuve.

$$m_p = \inf\{R(x) - T(x) \mid x \in \mathbb{R}^n\}$$

or $T(x) = T^{**}(x) = \sup\{y^t x - T^*(y) \mid y \in \mathbb{R}^n\}$.

d'où $m_p = \inf\{R(x) - \sup\{y^t x - T^*(y) \mid y \in \mathbb{R}^n\} \mid x \in \mathbb{R}^n\}$

$$= \inf\{R(x) + \inf\{T^*(y) - y^t x \mid y \in \mathbb{R}^n\} \mid x \in \mathbb{R}^n\}$$

$$= \inf\{\inf\{R(x) + T^*(y) - y^t x \mid y \in \mathbb{R}^n\} \mid x \in \mathbb{R}^n\}$$

$$= \inf\{\inf\{R(x) + T^*(y) - y^t x \mid x \in \mathbb{R}^n\} \mid y \in \mathbb{R}^n\}$$

$$= \inf\{T^*(y) + \inf\{R(x) - y^t x \mid x \in \mathbb{R}^n\} \mid y \in \mathbb{R}^n\}$$

$$= \inf\{T^*(y) - \sup\{y^t x - R(x) \mid x \in \mathbb{R}^n\} \mid y \in \mathbb{R}^n\}$$

$$= \inf\{T^*(y) - R^*(x) \mid y \in \mathbb{R}^n\}$$

$$= m_d$$

CQFD \square

Théorème. 1.2.2. [93] Soit P l'ensemble des solutions optimales du problème primal (P), et soit Δ l'ensemble des solutions optimales du problème dual (D), alors :

1. $\partial T(x) \subset \partial R(x) \quad \forall x \in P.$
2. $\partial R^*(y) \subset \partial T^*(y) \quad \forall y \in \Delta.$
3. $\cup\{\partial T(x) \mid x \in P\} \subset \Delta$ (égalité si R^* est sous-différentiable dans Δ).
4. $\cup\{\partial R^*(y) \mid y \in \Delta\} \subset P$ (égalité si T est sous-différentiable dans P).

 \square

La première propriété du théorème (1.2.2) ci-dessus donne une condition nécessaire d'optimalité.

Théorème. 1.2.3. [93] Sous les mêmes notations ci-dessus, nous avons les deux assertions suivantes.

1. $x^* \in P \Leftrightarrow R(x^*) + R^*(y) \leq T(x^*) + T^*(y) \quad \forall y \in \mathbb{R}^n$
2. $y^* \in \Delta \Leftrightarrow T^*(y^*) + T(x) \leq R^*(y^*) + R(x) \quad \forall x \in \mathbb{R}^n$

 \square

Théorème. 1.2.4. [93] Si x^* admet un voisinage V tel que

$$\partial T(x) \cap \partial R(x^*) \neq \emptyset \quad \forall x \in V \quad (1.10)$$

alors x^* est un minimum local de $(R - T)$.

 \square

Ce résultat motive la définition suivante

Définition. 1.2.1. (*point critique*)

x^* est appelé point critique si

$$\partial T(x) \cap \partial R(x^*) \neq \emptyset$$

Tout minimum local est donc forcément un point critique.

1.2.3 Algorithme D.C.

Se basant sur les théorèmes et propositions ci-dessus, le principe d'une méthode DC est de construire une suite $(x^k)_k$ qui converge vers un point critique, on en demande en outre d'être telle que $(R - T)(x^k)$ est décroissante.

Nous proposons, ainsi, le schéma suivant :

1. $k = 0$ et $x^0 \in \mathbb{R}^n$ quelconque.
2. $y^k \in \partial T(x^k)$.
3. $x^{k+1} \in \partial R^*(y^k) \Leftrightarrow y^k \in \partial R(x^{k+1})$.
4. Arrêt si $x^{k+1} = x^k$ i.e. si $\partial T(x^*) \cap \partial R(x^*) \neq \emptyset$

Le schéma décrit ci-dessus est un schéma d'une méthode primale-duale.

Théorème. 1.2.5. [93] *Supposons que les deux suites $(x^k)_k$ et $(y^k)_k$ sont bien définies, alors on a les propriétés suivantes :*

1. $R(x^{k+1}) - T(x^{k+1}) \leq T^*(y^k) - R^*(y^k) \leq R(x^k) - T(x^k)$
si de plus $R(x^{k+1}) - T(x^{k+1}) = R(x^k) - T(x^k)$ alors
 $x^k \in \partial R^*(y^k)$ et $y^k \in \partial T(x^{k+1})$

2. *par dualité :*

- $T^*(y^{k+1}) - R^*(y^{k+1}) \leq R(x^k) - T(x^k) \leq T^*(y^k) - R^*(y^k)$ *si de plus*
 $T^*(y^{k+1}) - R^*(y^{k+1}) = T^*(x^k) - R^*(x^k)$ *alors*

$$y^k \in \partial T(x^{k+1}) \text{ et } x^k \in \partial R^*(y^k)$$

$$3. \text{ Si } m_p \text{ est fini alors : } \lim_{k \rightarrow +\infty} \{R(x^k) - T(x^k)\} = \lim_{k \rightarrow +\infty} \{T^*(y^k) - R^*(y^k)\} = \gamma \geq m_d = m_p$$

où m_p est l'argmin du problème (P) et m_d est l'argmin du problème (D).

4. Si m_p est fini et si $(x^k)_k$ et $(y^k)_k$ sont bornées alors $\forall x^* \in \Omega((x^k)_k)$, $(\forall y^* \in \Omega((y^k)_k) \text{ resp.})$ il existe $y^* \in \Omega((y^k)_k)$ ($x^* \in \Omega((x^k)_k) \text{ resp.}$) telle que :

$$(a) \ R(x^*) - T(x^*) = T^*(y^*) - R^*(y^*) = \gamma \geq m_d = m_p.$$

$$(b) \ \lim_{k \rightarrow +\infty} \{R(x^k) + R^*(y^k)\} = R(x^*) + R^*(y^*) = (y^*)^t x^*.$$

$$(c) \ \lim_{k \rightarrow +\infty} \{T(x^k) + T^*(y^k)\} = T(x^*) + T^*(y^*) = (y^*)^t x^*.$$

où $\Omega((z^k)_k)$ est l'adhérence de $(z^k)_k$

□

1.3 Exemples d'algorithmes D.C.

Considérons le problème d'optimisation suivant

$$\begin{cases} \min \frac{1}{2} x^t Q x + b^t x \\ \text{s.c.} \\ x \in C \end{cases} \quad (1.11)$$

où Q est une matrice symétrique non nécessairement définie positive et b est un vecteur de \mathbb{R}^n .

On notera les deux cas

$C = \mathbb{R}^n$ correspondant au problème sans contrainte.

$C = \bar{B}(0, r)$ où r est un réel strictement positif donné, ce qui correspond à des contraintes de région de confiance.

Le problème (1.11) est un problème quadratique qu'on peut l'écrire sous la forme sans contrainte suivante :

$$\min\left\{\frac{1}{2}x^tQx + b^tx + \chi_C(x) \mid x \in \mathbb{R}^n\right\} \quad (1.12)$$

où $\chi_C(x)$ est la fonction indicatrice de C .

Le grand problème consiste à choisir une décomposition qui soit la plus appropriée et dont le calcul des sous gradients soit simple à réaliser. Nous présentons ici deux décompositions qui nous semblent les plus naturelles.

1.3.1 Décomposition classique

L'idée de la méthode est d'ajouter et de soustraire la quantité $\frac{\rho}{2}\|x\|^2$ avec $\rho > \|Q\|_1$ et de choisir la décomposition suivante :

$$R(x) = \frac{1}{2}x^tQx + b^tx + \chi_C(x) + \frac{\rho}{2}\|x\|^2 \text{ et } T(x) = \frac{\rho}{2}\|x\|^2.$$

Ainsi l'algorithme issu de cette décomposition est le suivant :

Algorithme DC-rajout

1. $k = 0$, x^0 quelconque
2. $y^k = \nabla T(x^k) = \rho x^k$
 $x^{k+1} \in \partial R^*(y^k)$, i.e. $x^{k+1} \in \operatorname{argmin}\{\frac{1}{2}x^tQx + b^tx + \chi_C(x) + \frac{\rho}{2}\|x\|^2 - x^ty^k\}$
 (i.e. $x^{k+1} \in \operatorname{argmin}\{\frac{1}{2}x^t(Q + \rho I)x + (b - y^k)^tx : x \in C\}$)
3. Si $\|x^{k+1} - x^k\| \leq \varepsilon$, on s'arrête $x^* = x^{k+1}$ est la solution optimale, sinon $k = k + 1$ et on retourne à 2.

Algorithme 1: Algorithme DC-rajout

L'inconvénient de cette méthode est la nécessité de résoudre un problème quadratique convexe à chaque itération, ceci reste loin d'être l'idéal, cet inconvénient disparaît avec la version suivante.

1.3.2 Décomposition optimale

Cette méthode propose la décomposition suivante :

$$\begin{aligned} R(x) &= \frac{\rho}{2}\|x\|^2 + b^t x + \chi_C(x) & \text{et} \\ T(x) &= \frac{\rho}{2}\|x\|^2 - \frac{1}{2}x^t Q x \end{aligned}$$

Ainsi l'algorithme issu de cette décomposition est le suivant :

Algorithme DC-optimal

1. $k = 0$, x^0 quelconque
2. $y^k = \nabla T(x^k) = (\rho I - Q)x^k$
 $x^{k+1} \in \operatorname{argmin}\{\frac{\rho}{2}\|x\|^2 - x^t(y^k - b) \mid x \in C\}$
(i.e. $x^{k+1} \in \operatorname{argmin}\{\|x\|^2 - 2x^t(\frac{y^k - b}{\rho}) \mid x \in C\}$)
3. Si $\|x^{k+1} - x^k\| \leq \varepsilon$, on s'arrête $x^* = x^{k+1}$ est la solution optimale, sinon $k = k + 1$ et on retourne à 2.

Algorithme 2: Algorithme DC-optimal

En posant $z^k = \frac{y^k - b}{\rho}$, on remarque que x^{k+1} n'est autre que la projection de z^k sur la boule C . Cette projection est donnée par la formule suivante :

$$x^{k+1} = \begin{cases} z^k & \text{si } \|z^k\| \leq r \\ \frac{r z^k}{\|z^k\|} & \text{sinon} \end{cases} \quad (1.13)$$

On considère le test d'arrêt suivant $\|x^{k+1} - x^k\| \leq \varepsilon$ où ε est la précision recherchée. Ce test n'est pas unique on peut par la suite utiliser un autre test comparant les valeurs de la fonction objectif en x^{k+1} et x^k , ou toute autre forme permettant de mieux gérer la convergence. La convergence dépend aussi du paramètre ρ , en effet on peut choisir d'autres expressions de ce dernier. La plus grande valeur propre de la matrice Q permet d'obtenir de meilleurs résultats, malheureusement en présence de problème de grande taille, la recherche d'une valeur propre peut ralentir la méthode. Pour cette raison nous avons choisi de prendre la valeur $\rho = \|Q\|_1$ quelque soit la situation.

1.3.3 Le cas quadratique sans contrainte

La résolution d'un problème quadratique sans contrainte, i.e. $C = \mathbb{R}^n$, par l'algorithme DC-optimal n'est intéressant que dans le cas où la matrice Q est semi-définie positive. Dans le cas contraire, on peut avoir $m_p = -\infty$.

Supposons que la matrice Q est semi-définie positive. Alors \bar{x} est point critique si et seulement s'il est optimal. Nous avons dans ce cas $Q\bar{x} = -b$ (voir la remarque 1.3.1).

Dans le cas où la région de confiance est inactive, nous démontrons une proposition qui donne une estimation du nombre d'itérations nécessaire pour la convergence de la méthode. Ce cas est utile dans la pratique car la méthode de région de confiance converge vers une solution de second ordre et donc, le problème quadratique au voisinage de la solution devient convexe et sans contrainte.

Proposition. 1.3.1. *Supposons appliquer la méthode DC-optimale pour résoudre le problème (1.11) avec Q une matrice définie positive et $C = \mathbb{R}^n$. Alors la méthode est finie et le nombre d'itérations nécessaire à sa convergence est $k = \lceil \ln\left(\frac{\rho\varepsilon}{\|Qx_0+b\|} - \|I - \frac{Q}{\rho}\| \right) \rceil$.*

où $[a]$ désigne la partie entière de a .

Preuve.

Dans le cas d'un problème quadratique sans contrainte, notre algorithme se présente comme suit :

1. $k = 0$, x^0 quelconque.

2. $y^k = (\rho I - Q)x^k$.

3. $x^{k+1} = \frac{y^k - b}{\rho}$

Autrement dit $x^{k+1} = \frac{(\rho I - Q)x^k - b}{\rho} = Bx^k + c$

où $B = I - \frac{1}{\rho}Q$ et $c = -\frac{1}{\rho}b$

De proche en proche nous construisons la suite $(x^k)_k$:

$$x^{k+1} = Bx^k + c = \dots = B^k x^0 + B^{k-1}c + \dots + Bc + c \quad (1.14)$$

d'où

$$x^{k+1} - x^k = B^k((B - I)x^0 + c) \quad (1.15)$$

ainsi le test d'arrêt sera vérifié si

$$\|x^{k+1} - x^k\| = \|I - \frac{1}{\rho}Q\|^k \| -\frac{1}{\rho}Qx^0 - \frac{1}{\rho}b \| = \varepsilon \quad (1.16)$$

ou $\|I - \frac{1}{\rho}Q\|^k = \frac{\rho\varepsilon}{\| -\frac{1}{\rho}Qx^0 - \frac{1}{\rho}b \|}$

d'où

$$k = \operatorname{argmin}_k \left\{ \|I - \frac{1}{\rho}Q\|^k = \frac{\rho\varepsilon}{\| -\frac{1}{\rho}Qx^0 - \frac{1}{\rho}b \|} \right\} \quad (1.17)$$

ainsi

$$k = \lceil \ln\left(\frac{\rho\varepsilon}{\|Qx_0 + b\|} - \left\|I - \frac{Q}{\rho}\right\|\right) \rceil \quad (1.18)$$

CQFD \square

Remarque. 1.3.1. *Sous les notations de la proposition ci-dessus, la méthode DC-optimale converge si et seulement si les valeurs propres de la matrice B sont inférieures strictement à 1.*

En effet

$$\begin{aligned} B(x^k - \bar{x}) &= (I - \frac{1}{\rho}Q)(x^k - \bar{x}) \\ &= x^k - \bar{x} - \frac{1}{\rho}Qx^k + \frac{1}{\rho}Q\bar{x} \\ &= Bx^k - \bar{x} + c \end{aligned}$$

d'où

$$\begin{aligned} x^{k+1} - \bar{x} &= Bx^k + c - \bar{x} \\ &= B(x^k - \bar{x}) \\ &= B^{k+1}(x^0 - \bar{x}) \end{aligned}$$

\square

Ainsi, il y'a convergence si et seulement si $B^k \rightarrow 0$ quand $k \rightarrow +\infty$ c'est-à-dire si et seulement si les valeurs propres de B sont inférieures strictement à 1, ceci est sans doute le cas pour notre étude par définition du paramètre ρ et car Q n'a pas de valeurs propres négatives ou nulles, le problème quadratique étant convexe.

1.4 Discussion

Nous nous intéressons dans cette étude à la détermination d'une valeur optimale du paramètre ε dans le sens où la valeur de la fonction objectif dite valeur optimale soit assez proche de la valeur de l'objectif à l'optimum mais

avec le moindre coût possible en nombre d'itérations et en temps CPU nécessaires à la résolution des problèmes.

Soit le point x_ε l'optimum retrouvé par la méthode DC avec le test d'arrêt $\|x^{k+1} - x^k\| \leq \varepsilon$. Nous testons quatre valeurs du paramètre (on dira aussi valeurs de seuil) ε : $\varepsilon = 10^{-2}$, $\varepsilon = 10^{-3}$, $\varepsilon = 10^{-4}$ et $\varepsilon = 10^{-6}$. La solution optimale est l'itéré x^{k+1} . Notons que les solutions sont de la même grandeurs sans quoi il serait préférable d'utiliser la quantité $\frac{\|x^{k+1} - x^k\|}{\|x^k\|}$ pour effectuer de telles expériences, mais ce test d'arrêt est différent de celui utilisé dans la littérature de la méthode D.C. ce qui nous a amené à contrôler les solutions en contrôlant les problèmes que nous voudrions résoudre.

Dans un premier temps nous appliquons la méthode D.C. décrite dans le paragraphe (1.3.2) pour résoudre un ensemble de problèmes quadratiques convexes avec pour seule contrainte celle de la région de confiance. Dans un second lieu nous testons la même méthode avec les mêmes valeurs de ε , appliquée pour résoudre des problèmes quadratiques non convexes avec région de confiance.

1.4.1 Le cas quadratique convexe

Nous générons une suite de problèmes convexes aléatoirement, nous fixons le rayon de la région de confiance $r = 10$, et nous testons différentes valeurs de ε afin de choisir le meilleur test d'arrêt pour notre étude, les résultats numériques sont présentés dans les figures suivantes.

Les matrices et les vecteurs de la forme quadratique

$$q(x) = \frac{1}{2}x^t W x + b^t x$$

sont générés de la même façon que dans le problème vertical (pour l'instant un problème vertical est un problème quadratique convexe avec une seule

contrainte sous forme de boule centrée à l'origine de rayon fixe) en l'occurrence chaque matrice W est le résultat d'un produit d'une matrice A donnée, par son transposé $W = A^t A$ et $b = A^t c$ où c est un vecteur donné. Les données des problèmes (A et c) sont comprises entre -200 et 200 et nous n'acceptons que les problèmes dont les solutions sont de la même grandeur pour permettre à nos expériences sur le test d'arrêt de rester fiables. Cependant, comme pour toute étude numérique, les conclusions que nous en tirons sont liés au jeu de problèmes qui servent à les réaliser.

La première comparaison (fig. 1.1) se charge d'évaluer la valeur optimale de la fonction objectif $q(x_\varepsilon)$ où x_ε est le point optimal retrouvé par la méthode pour un problème numéro l de dimension $2l + 1$, on répète la résolution pour le problème numéro $l + 1$, $l = 1, \dots, 100$, en initialisant les données du problème à chaque fois.

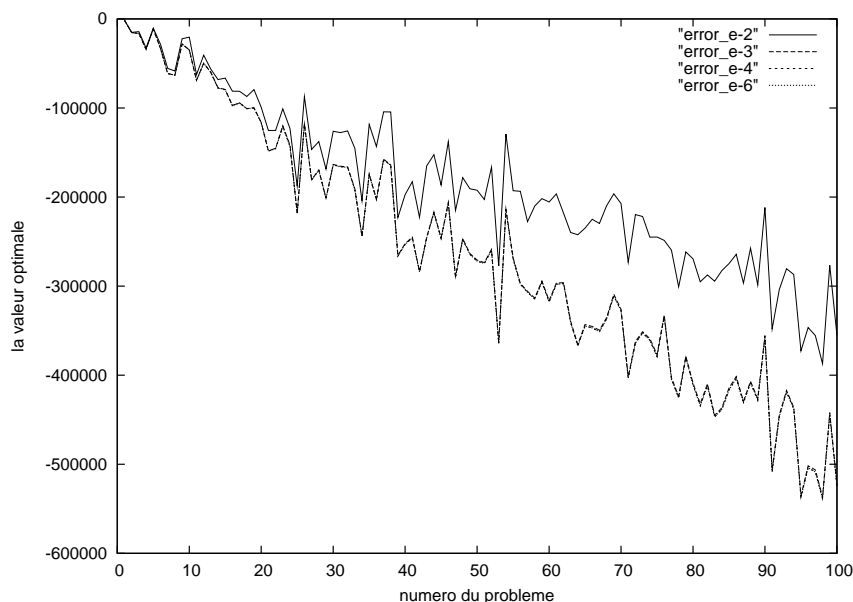


FIG. 1.1 – Comparaison en fonction de la valeur optimale.

Le graphe appelé *error_e-n* représente les valeurs optimales de la fonction

objectif lorsqu'on prend comme test d'arrêt l'inégalité : $\|x^{k+1} - x^k\| \leq 10^{-n}$ où $n = 2, 3, 4$ ou 6 .

Nous remarquons qu'à partir de la valeur $\varepsilon = 10^{-3}$ les graphes des valeurs optimales se superposent, nous concluons que le test d'arrêt $\|x^{k+1} - x^k\| \leq 10^{-3}$ constitue un bon seuil d'acceptation de l'optimalité d'un point trouvé par l'algorithme, en effet au delà de ce seuil, l'amélioration de la valeur de l'optimum i.e. $q(x_\varepsilon)$ n'est pas considérable, d'autant plus que les résultats concernant le nombre d'itérations et le temps CPU nécessaires pour avoir la convergence sont catastrophiquement élevés.

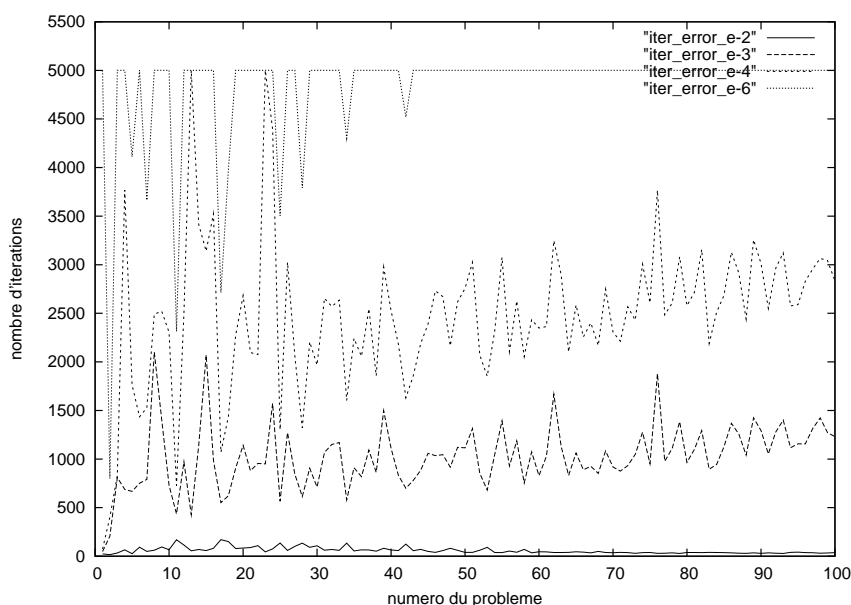


FIG. 1.2 – Comparaison en fonction du nombre d'itérations.

Le graphe *iter_error_e-n* dans la figure (1.2) représente le nombre d'itérations nécessaires pour avoir la convergence avec le seuil d'erreur 10^{-n} où $n = 2, 3, 4$ ou 6 .

La figure (1.2) permet la comparaison du nombre d'itérations pour chaque test d'arrêt. On remarque que pour le test d'arrêt avec $\varepsilon = 10^{-6}$ notre graphe

frôle à plusieurs reprises le nombre maximal d'itérations que nous imposons à savoir $k_{max} = 5000$ itérations, alors que le nombre d'itérations nécessaires reste élevé pour le test d'arrêt $\varepsilon = 10^{-4}$, raisonnable avec le seuil $\varepsilon = 10^{-3}$ et très attractif avec le seuil $\varepsilon = 10^{-2}$.

La deuxième remarque est que pour un seuil fixé, le nombre d'itérations ne change pas sévèrement et semble être insensible aux changements de la taille des problèmes. Nous remarquons même une légère diminution et une stabilisation pour le seuil $\varepsilon = 10^{-2}$ mais ceci n'est pas indicatif de l'efficacité de la méthode. En effet la figure (1.3) montre une autre réalité.

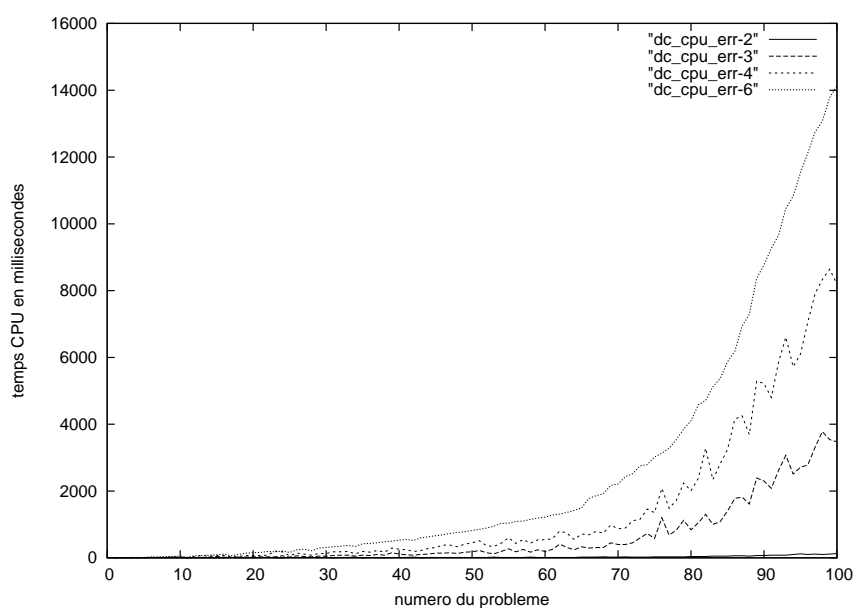


FIG. 1.3 – Comparaison en fonction du temps CPU.

Le graphe dc_cpu_err-n dans la figure (1.3) représente le temps CPU en millisecondes nécessaire pour avoir la convergence avec le seuil d'erreur 10^{-n} .

La figure (1.3) appuie notre dernière remarque, en effet le temps nécessaire pour la convergence augmente significativement en diminuant le seuil ε (i.e. en augmentant la précision), ainsi la valeur $\varepsilon = 10^{-3}$ semble être un seuil idéal

pour décréter l'optimalité atteinte.

Dans la figure (1.4), le graphe *dc_obj_unc* représente les valeurs optimales des problèmes sans contraintes de la région de confiance.

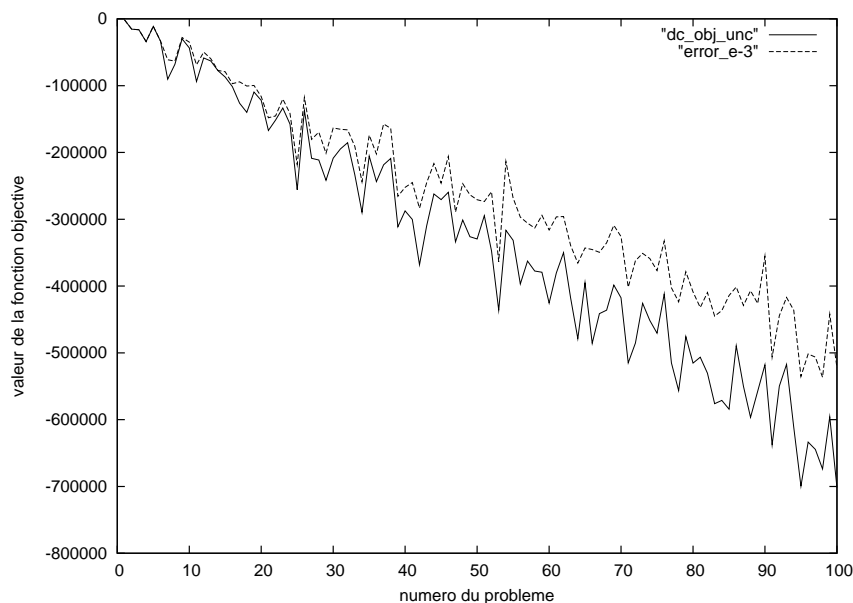


FIG. 1.4 – Comparaison de la valeur optimal avec et sans région de confiance.

La figure (1.4) représente une comparaison entre les valeurs de l'optimalité pour le seuil choisi i.e. $\varepsilon = 10^{-3}$ et les valeurs de l'optimalité pour les mêmes problèmes mais sans région de confiance. Cette comparaison nous permet de bien visualiser la qualité des résultats retrouvés avec le seuil $\varepsilon = 10^{-3}$. En effet, sans contrainte, on peut facilement donner la valeur optimale du problème quadratique d'autant plus que notre problème est modélisé à partir d'une équation linéaire. La solution du problème est donc évidente c'est le vecteur nul et la valeur optimale du problème quadratique n'est autre que l'opposé de la constante.

Nous remarquons aussi que pour la valeur $\varepsilon = 10^{-2}$, la méthode converge en très peu d'itérations et consomme peu de temps CPU, mais la valeur de la

fonction objectif produite nous laisse perplexes. En effet la résolution verticale (3.6.2) ne nécessite pas une grande sensibilité de la valeur optimale, généralement une approximation de celle-ci est suffisante, néanmoins dans cette étude nous avons choisi de garder la valeur $\varepsilon = 10^{-3}$ pour la résolution de tous les sous problèmes quitte à dépenser plus de temps et d'itérations, l'attitude change certainement pour des problèmes de grandes tailles.

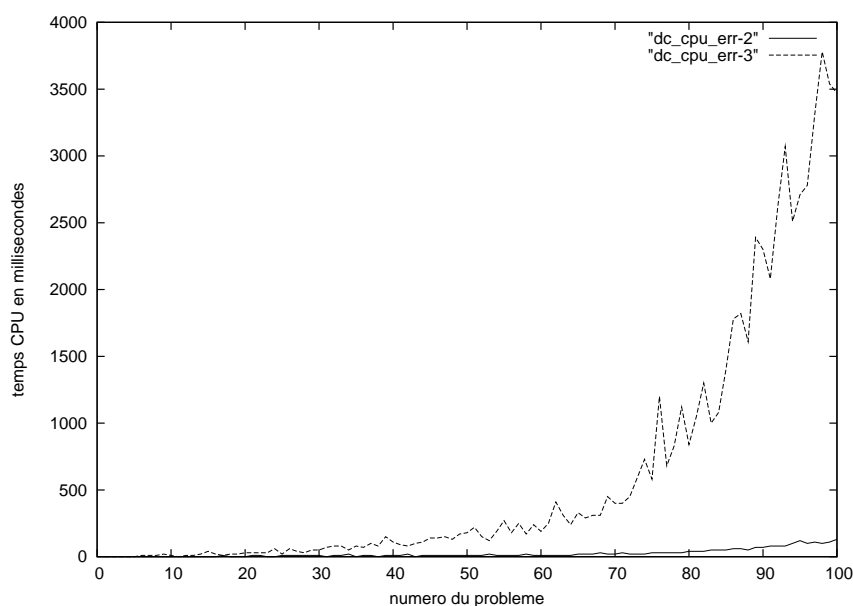


FIG. 1.5 – Comparaisons des temps CPU pour deux valeurs de test d'arrêt.

La figure (1.5) présente une comparaison entre le temps CPU nécessaire pour la convergence avec la valeur $\varepsilon = 10^{-2}$ et avec la valeur $\varepsilon = 10^{-3}$. On remarque clairement que la méthode est très sensible au test d'arrêt. On passe, pour les problème de taille supérieur à 90 (qui ne sont pas encore des problèmes de grande taille) d'une résolution en quelques dizaines de millisecondes à plus de 3,5 seconde. La méthode est parfois moins rapide quand on passe du seuil 10^{-2} au seuil 10^{-3} . Ceci n'est pas énorme si on cherche à résoudre quelques problèmes ou des problèmes de petites tailles mais pour la résolution

des problèmes non linéaires qui passe par la résolution de plusieurs problèmes quadratiques et qui nécessite une grande rapidité ceci risque d'être un point faible insurmontable de la méthode.

La figure (1.6) présente le temps CPU nécessaire pour la convergence avec la valeur $\varepsilon = 10^{-2}$. Nous allons voir par la suite (chapitre 1) que les dimensions ici correspondent au nombre de contraintes d'égalité et d'inégalité du problème non linéaire et que le nombre de variables n'intervient pas directement même s'il est très grand.

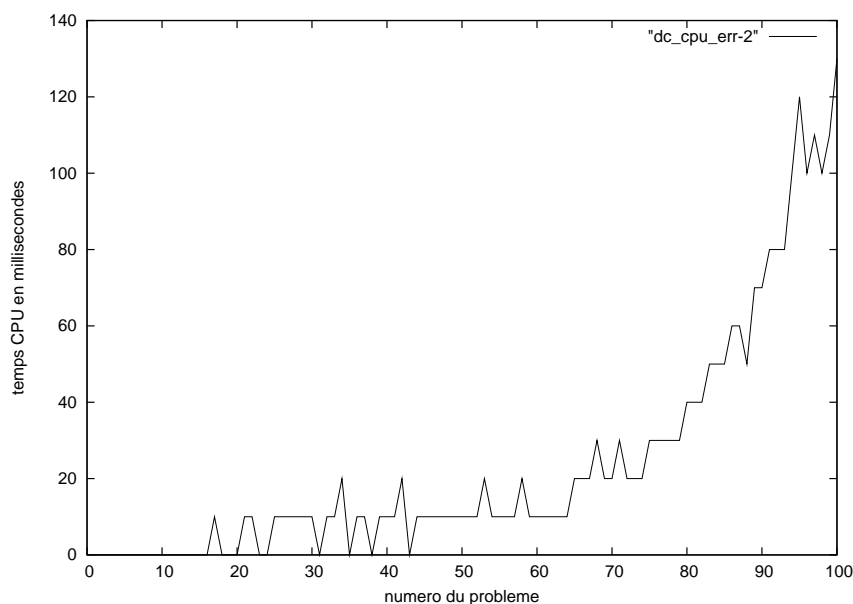


FIG. 1.6 – Temps de calcul CPU pour le test d'arrêt le plus faible.

1.4.2 Le cas quadratique non convexe

Une partie des problèmes traités dans cette dissertation est issue de la programmation quadratique séquentielle (3.6.2) et de la décomposition de Omojukum [63], qui nécessite la résolution d'un problème quadratique convexe dit

sous-problème vertical, et d'un problème quadratique non convexe dit sous-problème horizontal.

Nous cherchons dans ce paragraphe, de même manière que la précédente étude, de distinguer un seuil d'acceptation de l'optimalité qui nous servira par la suite de test d'arrêt de la procédure DC. Nous appliquons ainsi la méthode DC décrite dans le paragraphe (1.3.2) pour résoudre un ensemble de problèmes quadratiques non convexe avec seule contrainte la région de confiance, chaque problème est généré de façon aléatoire et numéroté de 1 à 100, le problème numéro l est de taille $l + 1$.

Nous avons comparé deux tests d'arrêt. Le premier avec le seuil $\varepsilon = 10^{-2}$ le deuxième avec le seuil $\varepsilon = 10^{-6}$, Grâce à trois représentations graphiques.

La première figure (1.7) représente les valeurs optimales retrouvées pour les deux tests.

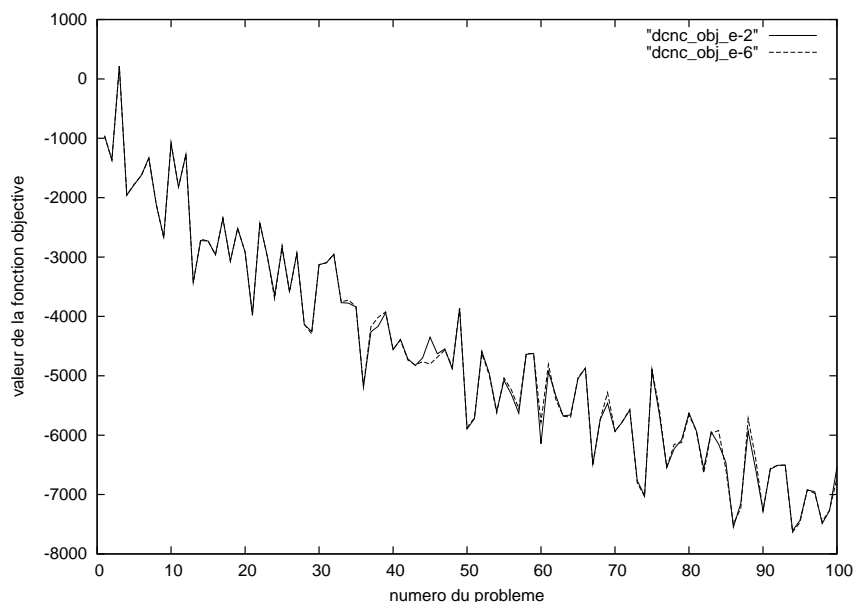


FIG. 1.7 – Comparaison en fonction de la valeur optimale.

Contrairement au cas convexe, on remarque ici que le test d'arrêt n'a pas

une grande incidence sur la valeur optimale même si un léger avantage est associé à la version de la méthode qui utilise une grande sensibilité. Pour certains problèmes la version faible (seuil $\varepsilon = 10^{-2}$) donne un résultat meilleur que la version forte, ceci est dû au fait que la méthode est initialisée à chaque fois et donc la méthode ne suit pas le même chemin après chaque initialisation, les erreurs d'arrondi en sont pour le reste.

La figure (1.8) illustre le temps de calcul pour chacun des test. On remarque très bien, grâce à cette représentation que les résultats obtenus sont d'une importance crucial par la suite pour la construction de notre algorithme.

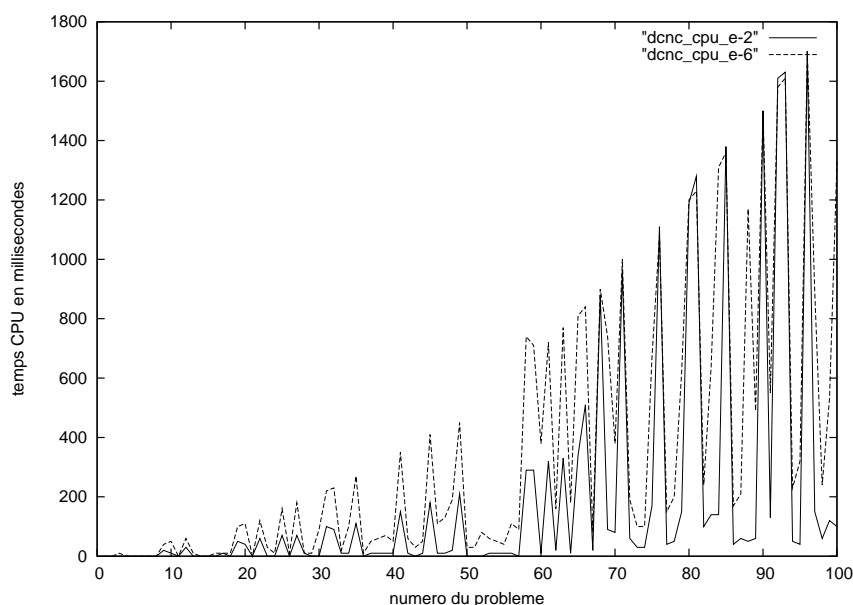


FIG. 1.8 – Comparaison en fonction du temps CPU.

Bien que les deux versions de la méthode trouve généralement des valeurs optimales identiques, la figure (1.8) montre qu'en terme de temps de calcul, l'avantage est de loin à la version que nous avons appelée faible, en effet tout au long de la minimisation la version qui utilise le seuil $\varepsilon = 10^{-2}$ donnait le résultat recherché bien avant la version qui utilise le seuil $\varepsilon = 10^{-6}$ sans que

cette dernière n'améliore grandement l'optimalité.

Finalement nous représentons les résultats concernant le nombre d'itérations nécessaires pour déclarer la convergence de chaque versions dans la figure (1.9). Nous remarquons sans ambiguïté l'intérêt du seuil $\varepsilon = 10^{-2}$ que nous concéderons largement suffisant pour décrire la convergence.

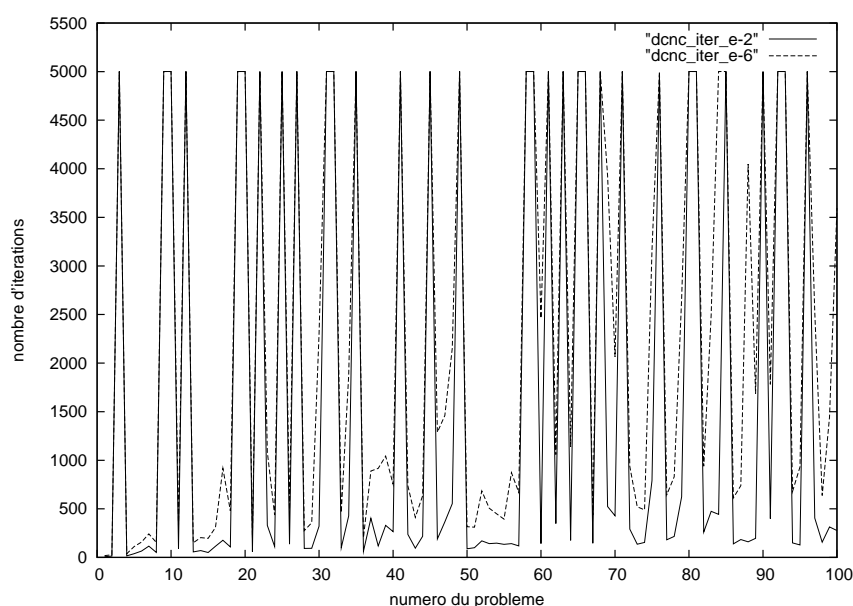


FIG. 1.9 – Comparaison en nombres d'itérations.

Cette remarque doit être prise dans son contexte, à savoir, nous nous intéressons ici à comparer des seuils entre eux, en revanche si l'utilisateur s'intéresse à plus de rigueur dans l'approximation des valeurs optimales, il serait souhaitable de prendre une plus grande puissance (un seuil plus petit) en particulier si on dispose suffisamment d'espace mémoire, et si le temps du calcul ne présente pas un inconvénient pour la détermination d'une solution optimale.

CHAPITRE 2

LES OUTILS POUR LA PROGRAMMATION QUADRATIQUE

Un problème quadratique s'écrit sous la forme suivante :

$$\begin{cases} \min_x \frac{1}{2}x^t Qx + c^t x \\ \text{s.c.} \quad A^t x + b = 0 \end{cases} \quad (2.1)$$

où Q est une matrice $n \times n$, c est un vecteur de \mathbb{R}^n , $b \in \mathbb{R}^m$ et A^t est une matrice $m \times n$.

Or notre étude de la programmation quadratique est imposée par le fait que la résolution du problème non linéaire peut conduire à deux types de problèmes quadratiques, le premier est un problème quadratique sous la contrainte de la région de confiance s'écrivant sous la forme :

$$\begin{cases} \min_x \frac{1}{2}x^t Qx + c^t x \\ \text{s.c.} \quad \|x\| \leq r \end{cases} \quad (2.2)$$

où $r \in \mathbb{R}_+^*$.

Le second problème quadratique est plus compliqué, car il est sous contrainte quadratique. Il peut s'écrire sous la forme (PQCQ) :

$$\begin{cases} \min_x \frac{1}{2}x^t Q_0 x + b_0^t x \\ \text{s.c.} \quad \frac{1}{2}x^t Q_1 x + b_1^t x + c_1 \leq 0 \end{cases} \quad (2.3)$$

où Q_0 est une matrice symétrique dans $\mathbb{R}^{n \times n}$ et Q_1 une matrice symétrique semi-définie positive dans $\mathbb{R}^{n \times n}$, $b_0 \in \mathbb{R}^n$, $b_1 \in \mathbb{R}^n$ et $c_1 \in \mathbb{R}$.

Pour simplifier, supposons que nous recherchons un minimum de la fonction

f définie de $\mathbb{R}^n \rightarrow \mathbb{R}$. Mathématiquement ceci se traduit par l'expression

$$\min_x f(x) \quad (2.4)$$

où $x \in \mathbb{R}^n$.

Supposons que la fonction f est de classe C^2 et notons $\nabla^2 f(x)$ son Hessien au point x . Soit p un vecteur de \mathbb{R}^n alors la formule de Taylor nous permet d'écrire

$$f(x+p) = f(x) + \nabla f(x)^t p + \frac{1}{2} p^t \nabla^2 f(x+tp) p \quad (2.5)$$

pour un scalaire $t \in [0, 1]$.

L'information recueillie de la fonction f nous permet donc, de construire un modèle quadratique $m(x_k)$ (pour simplifier on notera m_k) dont les caractéristiques au voisinage du point x_k (k représente l'instant actuel) sont similaires à celles de la fonction f à ce même point x_k . Et nous avons,

$$m(x_k) = f(x_k) + \nabla f(x_k)^t p + \frac{1}{2} p^t \nabla^2 f(x_k+tp) p \quad (2.6)$$

La première section de ce chapitre présente deux méthodes de résolution du problème (2.2), la première méthode est basée sur la technique des gradients conjugués alors que la seconde est la méthode DCA écrite pour résoudre des problèmes non convexe. Une comparaison numérique de ces deux méthodes nous permet de faire un choix de la méthode que nous utiliserons dans notre code (chapitre 5). D'autres alternatives sont possibles, nous en présenterons deux, la méthode de dogleg et la méthode de Hebden pour le cas convexe.

La dernière section de ce chapitre présente plusieurs outils algébriques pour calculer une base tangente et l'inverse à droite d'une matrice. Ces outils sont

nécessaires pour le développement de notre méthode dans le chapitre 5.

2.1 Le problème quadratique (RC)

Le problème que nous traitons dans ce chapitre est issu de la programmation quadratique séquentielle (voir paragraphe 3.6.2) s'écrivant sous la forme :

$$\begin{cases} \min_x \frac{1}{2}x^t Qx + c^t x \\ \text{s.c.} \quad \|x\| \leq r \end{cases}$$

En effet, comme le modèle quadratique m_k peut être une bonne ou une mauvaise approximation selon que x (un point de voisinage de x_k) est proche ou loin du point courant x_k alors nous imposons la contrainte $\|x\| \leq r$ dite région de confiance à notre problème quadratique, en espérant que cette zone représente un voisinage dans lequel le modèle quadratique est une bonne approximation de notre fonction f .

Comme $f(x_k)$ est une constante pour k fixé alors la minimisation de m_k revient à minimiser $\nabla f(x_k)^t p + \frac{1}{2}p^t \nabla^2 f(x_k) p$. Nous changeons de notations puisque la matrice hessienne Q n'est pas nécessairement le Hessien $\nabla^2 f(x_k)$ et que le vecteur c est éventuellement différent du vecteur $\nabla f(x_k)$. Nous expliciterons ces détails lors de la présentation de la programmation quadratique séquentielle au paragraphe (3.6.2), néanmoins, les résultats que nous présentons ici sont valables pour les deux expressions.

Le problème (2.2) peut être traité notamment de deux manières tout à fait différentes, nous pouvons d'abord le transformer en un problème quadratique sous contrainte quadratique :

$$\begin{cases} \min_x \frac{1}{2}x^t Qx + c^t x \\ \text{s.c.} \quad \frac{1}{2}x^t x - \frac{1}{2}r^2 \leq 0 \end{cases} \quad (2.7)$$

Nous pourrions, ainsi le résoudre par les méthodes proposées dans le paragraphe (2.2) suivant.

Un deuxième point de vue résout le problème (2.2) sans contrainte et considère la projection de la solution sur la boule $\{x \in \mathbb{R}^n : \|x\|_2 \leq r\}$.

Nous présentons ici deux méthodes de résolution et de projection. La première est une version de la méthode du gradient conjugué due à Steihaug [78]. Cette méthode a été utilisée par Hribar [9, 44] pour résoudre les sous-problèmes quadratiques issus de la méthode SQP, ses travaux ont été un élément essentiel dans l'élaboration du code KNITRO proposé par Gilbert, Waltz, Nocedal et Byrd [99].

Comme alternative à cette méthode nous proposerons la méthode DC (Différence de deux fonctions Convexes) que nous avons introduit au chapitre 1. Cette méthode présente des avantages numériques (voir notre comparaison en fin de ce chapitre) et théoriques importants pour résoudre des sous-problèmes verticaux ou horizontaux (Cf. paragraphe 2.1.4).

2.1.1 Le point de Cauchy

En pratique, il n'est pas nécessaire dans la méthode des région de confiance de calculer la solution optimale exacte du problème (RC). Il suffit de connaître une solution qui donne une réduction "suffisante" du modèle quadratique. Cette suffisance peut être quantifier par le point de Cauchy [62]. Notons ce point p_k^c .

Le point de Cauchy est déterminé en calculant d'abord la solution d'un problème linéaire facile à résoudre, puis en cherchant la solution d'un problème quadratique dans \mathbb{R} .

Supposons que $\|\nabla f(x_k)\| \neq 0$, soit

$$p_k^s = -\frac{r}{\|\nabla f(x_k)\|} \nabla f(x_k)$$

Calculer $\tau_k > 0$ la solution du problème

$$\begin{cases} \min_{\tau > 0} m(\tau p_k^s) \\ \text{s.c.} \quad \|\tau p_k^s\| \leq r \end{cases} \quad (2.8)$$

$$p_k^c = \tau p_k^s$$

Algorithme 3: Calcul de point de Cauchy

En fait p_k^s est la solution du problème lineaire suivant

$$\begin{cases} \min_{p \in \mathbb{R}^n} f(x_k) + \nabla f(x_k)^t p \\ \text{s.c.} \quad \|p\| \leq r \end{cases} \quad (2.9)$$

On vérifie facilement que τ_k vaut 1 si $\nabla f(x_k)^t \nabla^2 f(x_k) \nabla f(x_k) \leq 0$ et,

$$\tau_k = \min\left(\frac{1}{r} \frac{\|\nabla f(x_k)\|^3}{\nabla f(x_k)^t \nabla^2 f(x_k) \nabla f(x_k)}, 1\right)$$

sinon.

Le point de Cauchy est d'une importance cruciale. En effet, il est facile à calculer, son coût est très réduit et il permet de décider si la solution du modèle quadratique est acceptable. On montre [62] que la méthode de région de confiance est globalement convergente si ses itérations permettent une réduction suffisante de m_k , i.e. elles donnent une réduction du modèle m_k qui est au moins un certain multiple fixe de la diminution atteinte par le pas de Cauchy à chaque itération. Dans nos codes, nous avons utilisé systématiquement le

point de Cauchy comme point initial. Mais le point de Cauchy ne dépend de la matrice hessienne de son modèle que pour calculer sa longueur, et on ne peut pas avoir une convergence rapide (pour l'instant superlinéaire) si le Hessien n'intervient pas dans le calcul de la direction comme dans la longueur du pas de descente.

2.1.2 Le gradient conjugué selon Steihaug

La méthode du gradient conjugué produit un ensemble de vecteurs mutuellement conjugués, la principale propriété est que le calcul de la direction de descente d_k à l'itération k nécessite la connaissance seulement de la direction précédente d_{k-1} et aucune information sur les éléments d_0, d_1, \dots, d_{k-2} n'est demandée. Les vecteurs d_i ($i = 0, \dots, k$) sont mutuellement conjugués, c'est-à-dire, $d_i^t d_j = 0 \quad \forall i \neq j$.

La deuxième propriété importante est que la méthode est finie. En effet le nombre d'itérations nécessaire pour avoir la convergence de la méthode est "théoriquement" inférieur à la dimension du problème traité. Malheureusement cette propriété sera perdue avec l'utilisation de la projection, néanmoins Steihaug propose un test d'arrêt basé sur la dimension du problème, en fait il interdit à sa méthode de re-boucler plus que 2 fois la dimension du problème (voir Steihaug[78]).

Sans la projection, un vecteur d_k est donné par la formule $d_k = -(Qx_k + c) + \beta d_{k-1}$ où β est choisi tel que d_k et d_{k-1} soient orthogonaux. En général $\beta = \frac{(Qx_k + c)^t Q d_{k-1}}{d_{k-1}^t Q d_{k-1}}$ avec $d_0 = -(Qx_0 + c)$ pour un certain point initial x_0 . Le test d'arrêt serait $Qx_k + c = 0$.

Dans le cas de la présence d'une région de confiance, Steihaug propose deux tests d'arrêt supplémentaires : $\|d\| \leq r$ et la rencontre d'une direction de courbure négative en Q , *i.e.* un certain d_k vérifiant $d_k^t Q d_k \leq 0$. Ainsi l'algorithme

proposé par Steihaug est le suivant :

Soit $\varepsilon > 0$

Prendre $x_0 = 0$, $r_0 = c$ et $d_0 = -r_0$

Si ($\|r_0\| < \varepsilon$) **Alors**

 | x_0 est optimal

Fin Si

Pour k de 0 à \dots **faire**

Si ($d_k^t Q d_k \leq 0$) **Alors**

 | Chercher τ tel que $x = x_k + \tau d_k$ minimise $\frac{1}{2}x^t Q x + c^t x$ et satisfaisant $\|x\| = r$

 | x est optimal

Fin Si

$\alpha_k = r_k^t r_k / d_k^t Q d_k$

$x_{k+1} = x_k + \alpha_k d_k$

Si ($\|x_{k+1}\| \geq r$) **Alors**

 | Chercher τ tel que $x = x_k + \tau d_k$ vérifie $\|x\| = r$

 | x est optimal

Fin Si

$r_{k+1} = r_k + \alpha_k Q d_k$

Si ($\|r_{k+1}\| \leq \varepsilon \|r_0\|$) **Alors**

 | $x = x_{k+1}$ est optimal

Fin Si

$\beta_{k+1} = r_{k+1}^t r_{k+1} / r_k^t r_k$

$d_{k+1} = r_{k+1} + \beta_{k+1} d_k$

Fin Pour

[Remarque : La sortie est immédiate si la procédure déclare que x est optimal.]

Algorithme 4: L'algorithme de Steihaug

Steihaug a démontré que la suite générée par son algorithme vérifie les inégalités suivantes.

$$0 = \|x_0\| < \|x_1\| < \dots < \|x_k\| < \|x_{k+1}\| < \dots < \|x^*\| \leq r.$$

et que le vecteur post-initial $x_1 = \alpha_0 d_0 = -\frac{c^t c}{c^t Q c} c$ est le point de Cauchy, ceci est la raison pour laquelle le point initial est nul.

2.1.3 La méthode D.C.

Il s'agit d'appliquer la méthode DC présentée au chapitre 1. Nous écrivons la fonction objectif $\frac{1}{2}x^t Q x + c^t x$ sous forme de différence de deux fonctions convexes : $\frac{1}{2}x^t Q x + c^t x = R(x) - T(x)$.

$$\text{où } R(x) = \frac{\rho}{2}x^t x + b^t x, \text{ et } T(x) = \frac{\rho}{2}x^t x - \frac{1}{2}x^t Q x = \frac{1}{2}x^t(\rho I - Q)x.$$

ρ est un scalaire choisi de manière à avoir la matrice $\rho I - Q$ définie positive, en général, on prend $\rho = \|Q\|_1 + \varepsilon$ où $\varepsilon > 0$.

Le schéma d'une itération DC est le suivant :

Soit x_k l'itéré courant :

$$y_k = \nabla T(x_k) = (\rho I - Q)x_k.$$

$$x_{k+1} \in \partial R^*(y_k) \Leftrightarrow x_{k+1} \in \operatorname{argmin}\{R(x) - y_k^t x : \|x\| \leq r\}.$$

$$\text{ceci est équivalent à } x_{k+1} \in \operatorname{argmin}\{x^t x - \frac{2}{\rho}(y_k - b)^t x : \|x\| \leq r\}.$$

$$\text{ou bien } x_{k+1} \in \operatorname{argmin}\{\|x - \frac{y_k - b}{\rho}\|^2 \mid \|x\| \leq r\}.$$

ainsi

$$x_{k+1} = \begin{cases} z_k & \text{si } \|z_k\| \leq r \\ \frac{r z_k}{\|z_k\|} & \text{sinon} \end{cases} \quad (2.10)$$

$$\text{où } z_k = \frac{y_k - b}{\rho}.$$

Ainsi l'algorithme DC peut s'écrire sous la forme suivante :

Choisir $x_0 \in \mathbb{R}^n$ quelconque, $k = 0$, $\rho > \|Q\|_1$ et $\varepsilon > 0$

Tant que ($\|x_{k+1} - x_k\| > \varepsilon$) **faire**

 Calculer $z_k = \frac{(\rho I - Q)x_k - b}{\rho}$

Si ($\|z_k\| \leq r$) **Alors**

$x_{k+1} = z_k$

Sinon

$x_{k+1} = \frac{rz_k}{\|z_k\|}$

Fin Si

$k \leftarrow k + 1$

Fait

$x^* = x_{k+1}$

Algorithme 5: Algorithme DC

La projection est ainsi naturelle dans le sens où le point optimal minimisant $\|x - z_k\|$ sur la boule est soit z_k si $\|z_k\| \leq r$ soit sa projection sur la boule si la norme du vecteur dépasse le rayon de la boule.

L'unique test d'arrêt de l'algorithme représente la sensibilité recherchée (le nombre de chiffres significatifs après la virgule).

Dans le cas où la région de confiance n'existe pas (on dit que la région de confiance est inactive), la méthode est finie, et le nombre d'itérations permettant la convergence est donné par la formule :

$$k = \lceil \ln\left(\frac{\rho\varepsilon}{\|Qx_0 + b\|} - \left\|I - \frac{1}{\rho} \cdot Q\right\|\right) \rceil \quad (2.11)$$

où $[a]$ désigne la partie entière de a .

2.1.4 Comparaison numérique

Nous avons choisi de comparer les deux méthodes décrites dans ce chapitre en les appliquant pour résoudre les 100 problèmes quadratiques générés aléatoirement de l'étude du chapitre 1. Nous comparons pour cet ensemble le résultat de minimisation c'est-à-dire les valeurs optimales retrouvées par chaque méthode puis le nombre d'itérations et le temps de calcul nécessaire pour retrouver les points optimaux. La figure (2.1) montre que la méthode DC retrouve une meilleure optimalité que la procédure Steihaug pour l'ensemble des problèmes.

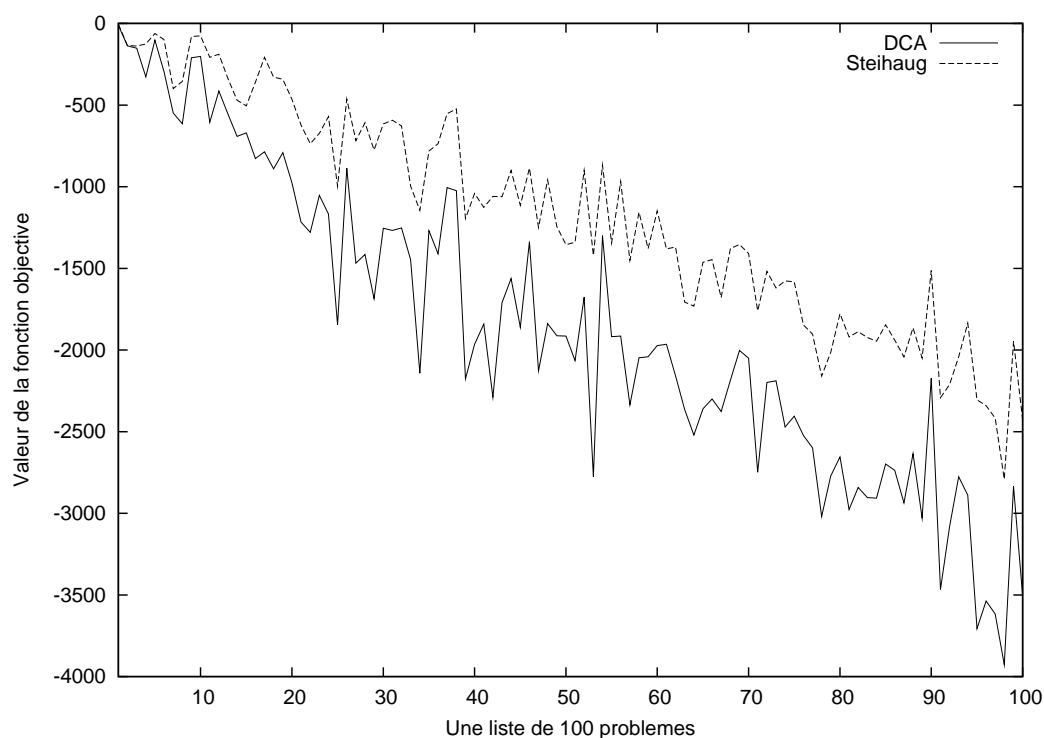


FIG. 2.1 – Comparaison par rapport à la valeur de la fonction objectif produite par chaque méthode.

La différence entre la valeur optimale retrouvée par la méthode DC et

celle retrouvée par la méthode Steihaug s'aggrave pour des problèmes de grande taille dans le sens où celle retrouvée par la méthode DC est toujours la meilleure. Dans la figure (2.2) nous rapportons le nombre d'itérations nécessaire pour la convergence de chaque méthode et nous traçons des courbes continues pour illustrer les changements.

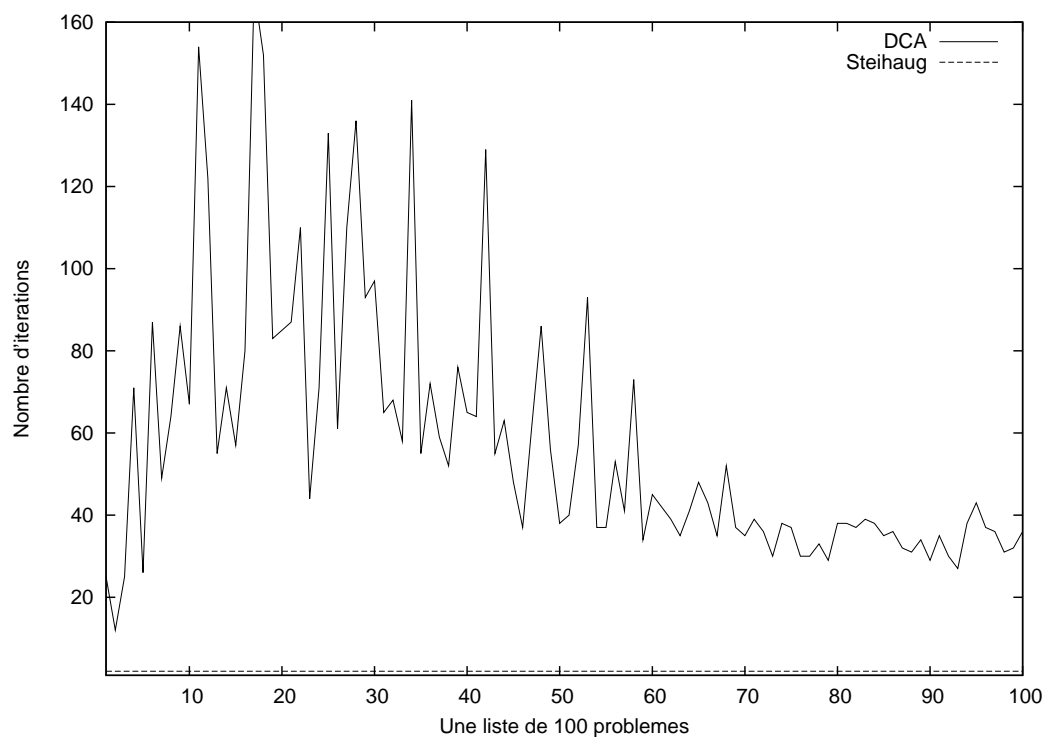


FIG. 2.2 – Comparaison par rapport au nombre d'itérations que nécessite chaque méthode.

Nous remarquons que la méthode de Steihaug converge en deux ou trois itérations, alors que la méthode DC réclame plus d'itérations pour satisfaire la convergence (entre 5 et 130 itérations pour la plupart de nos problèmes), ceci est très relatif, en effet en terme de temps de calcul, nous remarquons qu'en moyenne une itération DC est trois fois plus rapide en terme de temps de calcul qu'une itération Steihaug.

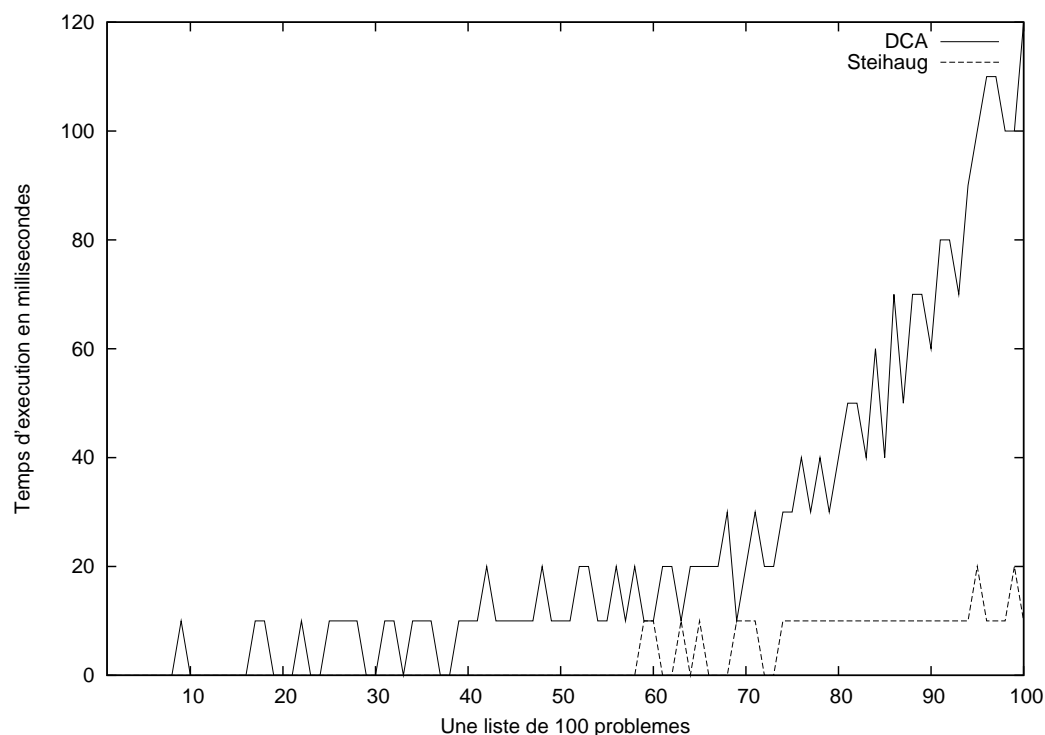


FIG. 2.3 – Comparaison par rapport au temps CPU que nécessite chaque méthode.

La figure (2.3) nous permet de remarquer que la différence en matière de temps de calcul se creuse pour les problèmes de grande taille. Vu ces expériences, nous pouvons affirmer que la méthode de Steihaug consomme moins de temps et d'itérations que la méthode DC mais retrouve des points stationnaires moins bons en terme d'optimalité que la méthode DC. Ce résultat n'affecte pas l'importance de la méthode DC en effet, dans le cas où nous n'avons pas besoin d'une grande précision du résultat de l'optimalité, c'est le cas de la programmation verticale (5.3) nous pourrions changer le test d'arrêt de la méthode afin de produire un minimum "acceptable" en temps raisonnable, soit en prenant un seuil plus grand que 10^{-3} pris pour notre étude ici, soit en comparant la valeur de la fonction objectif à chaque itération à une valeur que

nous jugeons acceptable ou en comparant les valeur de l'objectif entre deux itérations ce qui nous permet de définir une précision par rapport à l'objectif.

2.1.5 Autres alternatives pour la résolution du problème quadratique

Il existe d'autre méthodes, plus au moins connues, dédiées à la résolution du problème quadratique (2.2). Rappelons que pour le cas de la programmation verticale, il s'agit de résoudre le problème (2.12) suivant

$$\begin{cases} \min_{\bar{v}} \|Av + c\|^2 \\ \text{s.c.} \\ \|v\|_2 \leq \zeta r \end{cases} \quad (2.12)$$

où $\zeta \in]0, 1]$, c est un vecteur et A est une matrice de plein rang. Ce problème peut s'exprimer sous la forme quadratique :

$$\begin{cases} \min_v \frac{1}{2} v^t A^t A v + c^t A v \\ \text{s.c.} \\ \|v\|_2 \leq \zeta r \end{cases} \quad (2.13)$$

Le problème (2.13) est un problème quadratique avec une seule contrainte celle de la région de confiance. Plusieurs méthodes sont proposées dans la littérature pour résoudre ce genre de problème, nous avons présenté la méthode DC et la méthode de Steihaug mais d'autres alternatives sont possibles.

2.1.5.1 L'algorithme de dogleg

Quand la matrice Q du problème quadratique (2.2) est définie positive, c'est le cas de la programmation verticale avec la matrice $Q = A^t A$, la méthode dogleg de Powell [68] peut être intéressante, elle est relativement peu chère en terme de temps de calcul et d'espace mémoire. La trajectoire de dogleg est une combinaison linéaire reliant le point $v = 0$ au point de Cauchy $v_{cp} = -\alpha A c$, où

$$\alpha = \begin{cases} \frac{\|A^t c\|_2^2}{c^t (A A^t)^2 c} & \text{si } \frac{(c^t A A^t c)^{3/2}}{c^t (A A^t)^2 c} \leq \zeta r \\ \frac{\zeta r}{\|A^t c\|_2} & \text{si non} \end{cases} \quad (2.14)$$

Ensuite, à partir de ce point, la trajectoire de dogleg suit le pas complet de Newton. A est une matrice de plein rang, donc $A^t A$ est non singulière, et si la région de confiance devient inactive, alors la solution du problème (2.13) sera donnée par le pas de Newton : $v_N = -A^t (A^t A)^{-1} c$.

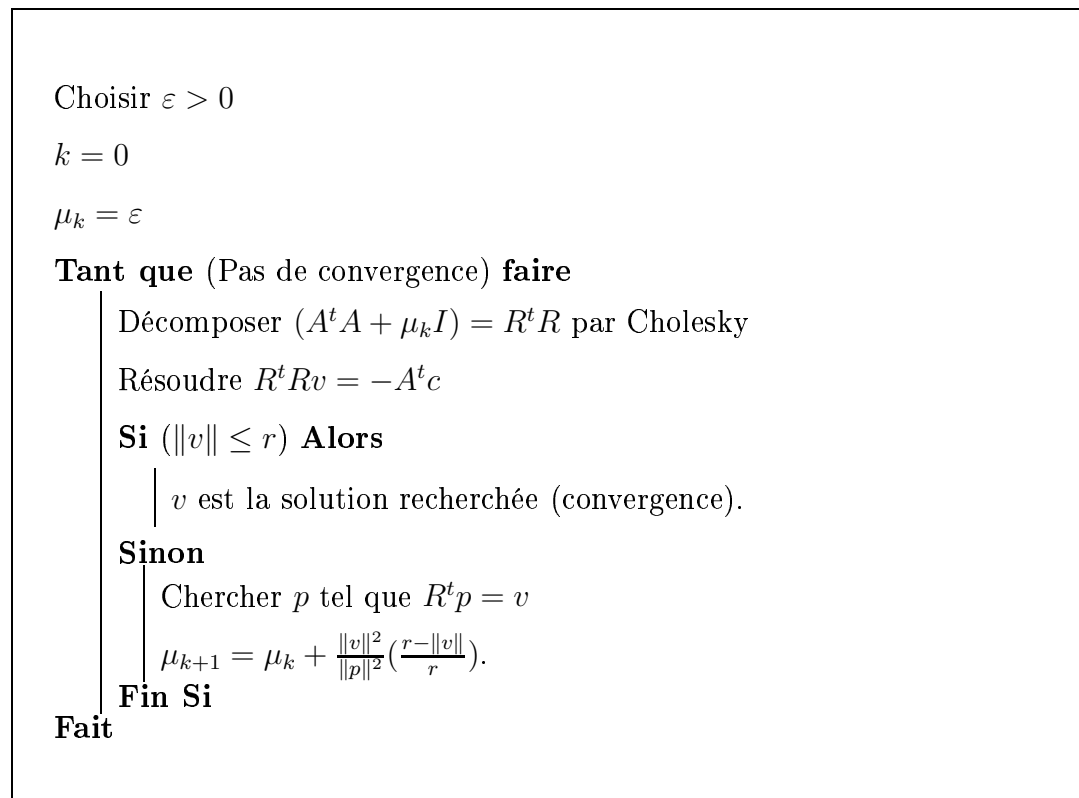
Ensuite Powell tronque le pas v par rapport à la région de confiance ζr , et il montre que son pas n'augmente pas la fonction objectif du problème vertical. Bien que cette façon de faire est très approximative, elle a été utilisée dans le code KNITRO [8] et dans le code MITR [46] pour son coût très raisonnable.

2.1.5.2 L'algorithme de Hebden

La matrice $A^t A$ est semi-définie positive, et la méthode de Hebden se présente comme suit.

Après une décomposition initiale de $(A^t A + \varepsilon I) = R^t R$ par Cholesky ($\varepsilon > 0$) et la résolution du problème linéaire $R^t R v = -A^t c$, Hebden vérifie si $\|v\| \leq r$, si c'est le cas alors la méthode est interrompue, et retourne la valeur de v comme valeur optimale, dans le cas contraire on construit une nouvelle matrice $A^t A + \mu_k I$ qui sera factorisée de nouveau par la méthode de Cholesky. L'algorithme

(6) suivant explicite la méthode de Hebden,



Algorithme 6: L'algorithme de Hebden

La méthode nécessite une factorisation de Cholesky à chaque itération, ce qui représente un inconvénient majeur, néanmoins sa convergence rapide retient toute notre attention.

2.2 Problème Quadratique sous Contrainte Quadratique (PQCQ)

Ce sujet de recherche est assez récent, les premières méthodes réservées exclusivement aux problèmes (PQCQ) remontent à la fin des années soixante-dix. En 1979 Panin publie un article [65] suivi par un deuxième en 1981 [66] faisant intervenir la programmation (PQCQ), il a prouvé que sa méthode est globalement convergente avec une convergence superlinéaire, en revanche les

hypothèses qu'il a supposé sont très restrictives et dures à satisfaire : il suppose que la fonction objectif et les contraintes sont fortement convexes (cf. le paragraphe 1.1), que le Hessien du Lagrangien est uniformément borné, et que les estimateurs de multiplicateurs de Lagrange sont uniformément bornés.

Vingt ans après, Kruk et Wolkowicz [49] ont proposé une méthode de type région de confiance-programmation quadratique convexe avec contrainte quadratique (PQCCQ) pour résoudre des problèmes d'optimisation (convexe ou non). Les sous-problèmes (PQCQ) sont résolus à chaque itération par une *relaxation SDP*. Dans un second article [50], ils étendirent leur discussion à l'effet Maratos et la convergence quadratique locale et globale, malheureusement l'utilisation de la relaxation SDP reste réservée aux problèmes de petites ou moyennes tailles.

En 2002, Anitescu [2] publie une étude du problème (PQCQ) afin de répondre à la dégénérescence du problème non linéaire. Sa méthode converge superlinéairement sous les hypothèses du qualification des contraintes de Mangasarian et Fromowitz (MFCQ) qui sont plus faibles que l'indépendance linéaire de colonnes de la matrice jacobienne des contraintes.

En 2003, Fukushima, Luo et Tseng [27] démontrent la convergence quadratique globale et locale sous la condition de Slater, et en supposant que le Hessien du lagrangien est uniformément défini positif, sans pour autant supposer la complémentarité stricte de la contrainte.

Dans cette dissertation, nous avons choisi de résoudre le problème non linéaire par une méthode SQP. La programmation (PQCQ) s'est imposée afin d'éviter le rejet de la direction de descente, ainsi le sous-problème quadratique que nous allons résoudre dans ce chapitre est non convexe.

2.2.1 Qualification des contraintes et conditions d'optimalité

La contrainte est convexe, et s'il existe un point réalisable x tel que $\frac{1}{2}x^t Q_1 x + b_1^t x + c_1 < 0$, autrement dit supposons que l'intérieur relatif est non vide, alors le problème (2.3) vérifie la condition de qualification des contraintes.

2.2.1.1 Conditions nécessaires d'optimalité

Soit $L(x, \lambda) = q_0(x) + \lambda q_1(x)$ $\lambda \in \mathbb{R}_+$, le Lagrangien du problème (2.3).
où $q_0(x) = \frac{1}{2}x^t Q_0 x + b_0^t x$ et $q_1(x) = \frac{1}{2}x^t Q_1 x + b_1^t x + c_1$.

Les conditions de Karush-Kuhn-Tucker s'écrivent sous la forme :

$$\begin{cases} (Q_0 x + b_0) + \lambda(Q_1 x + b_1) = 0 \\ q_1(x) \leq 0 \\ \lambda q_1(x) = 0 \\ \lambda \geq 0 \end{cases} \quad (2.15)$$

Et en rajoutant une variable d'écart t , nous retrouvons le système d'égalité suivant :

$$\begin{cases} (Q_0 x + b_0) + \lambda(Q_1 x + b_1) = 0 \\ \lambda q_1(x) = 0 \\ q_1(x) + t = 0 \\ \lambda \geq 0, t \geq 0 \end{cases} \quad (2.16)$$

2.2.1.2 Condition suffisante d'optimalité

Dans le cas où Q_0 est définie positive, la condition suffisante d'optimalité du second ordre est vérifiée pour tout triplet (x, λ, t) satisfaisant (2.16), c'est le cas notamment de la programmation verticale (voir paragraphe 5.3) .

En revanche si Q_0 n'est pas définie positive, la condition d'optimalité du

second ordre est la suivante :

Soit x^* une solution optimale alors il existe un scalaire $\lambda^* > 0$ tel que $p^t(Q_0 + \lambda^*Q_1)p \geq 0 \forall p \in \mathbb{R}^n$ vérifiant $(Q_0x^* + b^*)^t p = 0$.

2.2.2 Résolution du problème PQCQ

On dit qu'un triplet (x, λ, t) dans $\mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ est un point KKT s'il est solution du système suivant

$$\begin{cases} (Q_0x + b_0) + \lambda(Q_1x + b_1) = 0 \\ \lambda q_1(x) = 0 \\ q_1(x) + t = 0 \\ \lambda \geq 0, t \geq 0 \end{cases} \quad (2.17)$$

À partir d'un point initial nous construisons une suite convergente vers le point optimal du système (2.17) grâce à une méthode newtonienne. Pour la suite, le point (x, λ, t) désignera l'itéré courant, et le point (x^+, λ^+, t^+) désignera l'itéré suivant, c'est-à-dire $x^+ = x + d_x$, $\lambda^+ = \lambda + d_\lambda$ et $t^+ = t + d_t$, où le pas de déplacement (d_x, d_λ, d_t) est la solution du système linéaire suivant :

$$\begin{cases} (Q_0 + \lambda Q_1)d_x + (Q_1x + b_1)d_\lambda = -(Q_0x + b_0) - \lambda(Q_1x + b_1) & (a) \\ \lambda(Q_1x + b_1)^t d_x + q_1(x)d_\lambda = -\lambda q_1(x) & (b) \\ (Q_1x + b_1)^t d_x + d_t = -q_1(x) - t & (c) \end{cases} \quad (2.18)$$

2.2.2.1 Résolution du système (2.18) avec la méthode de Newton

Nous construisons une méthode primale de résolution du problème (PQCQ) en s'appuyant sur les relations suivantes :

$$t^+ = -(Q_1x + b_1)^t d_x - q_1(x) \quad (\text{d'après (2.18-c)}) \quad (2.19)$$

$$\lambda(Q_1x + b_1)^t d_x = -q_1(x)\lambda^+ \quad (\text{d'après (2.18-b)}) \quad (2.20)$$

et enfin l'équation (2.21) exprimant la relation entre t^+ et λ^+ . Cette relation est déduite en multipliant (2.18-c) par λ d'où : $\lambda(Q_1x + b_1)^t d_x + \lambda d_t = -\lambda q_1(x) - \lambda t$. puis (2.18-b) - (2.18-c) $\Rightarrow q_1(x)d_\lambda - \lambda d_t = t\lambda \iff$

$$q_1(x)d_\lambda = \lambda d_t + t\lambda = \lambda t^+ \quad (2.21)$$

2.2.2.2 Étude des cas

- Si $q_1(x) \neq 0$

$$(2.21) \iff d_\lambda = \frac{\lambda t^+}{q_1(x)} \quad (2.22)$$

donc,

$$(2.18-a) \iff (Q_0 + \lambda Q_1)d_x + (Q_1x + b_1)t^+ \frac{\lambda}{q_1(x)} = -(Q_0x + b_0) - \lambda(Q_1x + b_1)$$

$$\iff (Q_0 + \lambda Q_1)d_x - \frac{\lambda}{q_1(x)}(Q_1x + b_1)(Q_1x + b_1)^t d_x - \lambda(Q_1x + b_1) = -(Q_0x + b_0) - \lambda(Q_1x + b_1)$$

\iff

$$\left[Q_0 + \lambda Q_1 - \frac{\lambda}{q_1(x)}(Q_1x + b_1)(Q_1x + b_1)^t\right]d_x = -(Q_0x + b_0) \quad (2.23)$$

– Supposons que $q_1(x) = 0$, (2.18) devient

$$\begin{cases} (Q_0 + \lambda Q_1)d_x + (Q_1x + b_1)d_\lambda = -(Q_0x + b_0) - \lambda(Q_1x + b_1) \\ \lambda(Q_1x + b_1)^t d_x = 0 \\ (Q_1x + b_1)^t d_x + d_t = -t \end{cases} \quad (2.24)$$

d'où

$$\lambda t^+ = 0 \quad (2.25)$$

Ainsi sous l'hypothèse de la complémentarité stricte, $t^+ = 0$ c'est-à-dire si l'algorithme produit un point x_k appartenant à la frontière, alors tout itéré suivant x_k demeure sur la frontière, i.e. $q_1(x_l) = 0 \quad \forall l \geq k$.

Dans ce cas ($q_1(x) = 0$ et $\lambda > 0$),

$$(2.18\text{-a}) \Rightarrow d_x^t(Q_0 + \lambda Q_1)d_x - d_\lambda(Q_1x + b_1)^t d_x = -(Q_0x + b_0)^t d_x + \lambda(Q_1x + b_1)^t d_x$$

\Rightarrow

$$d_x^t(Q_0 + \lambda Q_1)d_x + (Q_0x + b_0)^t d_x = 0 \quad (2.26)$$

Cet équation n'admet de solution que si la matrice $Q_0 + \lambda Q_1$ est semi-définie positive, cette condition est à fortiori vérifiée si Q_0 est semi-définie positive.

– Cas de la dégénérescence

Ceci correspond au cas où $\lambda = 0$ et $q_1(x) = 0$. Le système du Newton devient,

$$\begin{cases} Q_0 d_x + (Q_1x + b_1)d_\lambda = -(Q_0x + b_0) \\ (Q_1x + b_1)^t d_x + d_t = -t \iff (Q_1x + b_1)^t d_x = t^+ \end{cases} \quad (2.27)$$

et x^+ peut être donné par

$$Q_0 x^+ = (Q_1 x + b_1) d_\lambda - b_0 \quad (2.28)$$

2.2.2.3 Condition supplémentaire sur les variables

Il est nécessaire, par ailleurs que les paramètres λ et t restent positifs à chaque itération, de manière à avoir (x, λ, t) un point KKT, i.e. $\lambda \geq 0$ et $t \geq 0$.

D'après (2.21) on a : $q_1(x) d_\lambda = \lambda(t + d_t)$ donc si $q_1(x) \leq 0$, $\lambda \geq 0$ et $t^+ \geq 0$ alors $-\lambda \leq d_\lambda \leq 0$. Nous avons ainsi besoin d'avoir

$$\lambda + d_\lambda \leq \lambda \quad (2.29)$$

Puisque $(\lambda_k)_{k \geq 0}$ décroît et elle est minorée par zéro, alors la suite converge vers une limite notée $\lambda^* \geq 0$.

La même hypothèse est supposée pour la variable d'écart, $t \geq 0$ et $t + d_t \geq 0$.

A partir de (2.18 - c) nous pouvons écrire

$$-(Q_1 x + b_1)^t d_x = (t + d_t) + q_1(x) \geq q_1(x) \quad (2.30)$$

Cette étude nous permet de proposer les procédures suivantes :

Procédure 1

Cas où $\lambda \neq 0$ et $q_1(x) \neq 0$

Résoudre l'équation linéaire (2.23) afin d'avoir d_x

$$t^+ = \max(-q_1(x) - (Q_1x + b_1)^t d_x, 0)$$

$$\lambda^+ = \max(\lambda(\frac{t^+}{q_1(x)} + 1), 0)$$

Algorithme 7: Programmation PQCQ, Procédure 1

Comme $t^+ \geq 0$ alors $1 + \frac{t^+}{q_1(x)} \leq 1$ d'où

$$\lambda^+ \leq \lambda \quad (2.31)$$

Procédure 2

Cas où $\lambda \neq 0$ et $q_1(x) = 0$

Résoudre l'égalité quadratique (2.26) pour avoir d_x

$$t^+ = 0$$

$$\bar{\lambda} \text{ est solution de } (Q_1x^+ + b_1)\lambda^+ + (Q_0x^+ + b_0) = 0$$

$$\lambda^+ = \max(\bar{\lambda}, 0)$$

Algorithme 8: Programmation PQCQ, Procédure 2

Cette valeur de $\bar{\lambda}$ est prise pour avoir (x^+, λ^+, t^+) comme point KKT et $\lambda^+ \leq \lambda$.

Procédure 3

Cas où $\lambda = 0$ et $q_1(x) = 0$

Résoudre l'équation linéaire $Q_0x^+ + b_0 = 0$ pour avoir x^+

$$t^+ = 0$$

$$\lambda^+ = 0$$

Algorithme 9: Programmation PQCQ, Procédure 3

Donc tout revient à résoudre, soit un système linéaire sous la forme $Ad_x + b = 0$ où A est une matrice symétrique et b un vecteur quelconque, soit à résoudre l'équation quadratique non convexe (2.26), ce qui revient à résoudre un problème de type (RC).

2.3 Calcul de bases tangentes et d'inverses à droite

La résolution du problème horizontal (problème quadratique non convexe sous la contrainte d'une région de confiance), nous confronte à la recherche d'une base tangente i.e. de l'espace noyau de A^t , mais en fait, on en a besoin dans plusieurs étapes du développement, le calcul des multiplicateurs de Lagrange et la correction du second ordre requièrent la résolution de systèmes linéaires, et un inverse à droite serait le bienvenu.

2.3.1 Méthode 1 : factorisation QR

Soit A une matrice de $\mathbb{R}^{(n+m) \times (m+p)}$ et Π est une permutation. La factorisation QR de la matrice A est une des méthodes les plus utilisées dans la pratique. Il s'agit de donner une matrice orthonormée Q et une matrice triangulaire supérieure R tel que

$$A\Pi = QR = [Q_1 Q_2] \begin{bmatrix} R \\ 0_{(n-m)(m+p)} \end{bmatrix}$$

où

$Q = [Q_1 \quad Q_2]$ est une matrice orthonormée de dimension $(n + m)$.

Q_1 est une matrice de dimension $(n + m) \times (m + p)$.

Q_2 est une matrice de dimension $(m + p) \times (n - p)$.

et R est une matrice triangulaire supérieure d'ordre $(m + p)$.

On montre aisément que les colonnes de Q_2 forment une base orthonormée de l'espace noyau $N(A^t)$. Soit $u \in \mathbb{R}^{n-p}$ et soit y un élément de \mathbb{R}^{n+m} dont les $m + p$ premières composantes sont nulles et les $n - p$ dernières composantes sont les composantes de $u : y = (0, 0, \dots, 0, u_1, \dots, u_{n-p})$.

Comme Q est une matrice orthonormée alors $Q^t A = Q^t Q R = R$ d'où $A^t Q = R^t$ donc $A^t Q y = R^t y$ et puisque les $n - p$ dernières colonnes de R^t ($n - p$ dernières lignes de R) sont toutes nulles, alors $A^t Q y = R^t y = 0$. Si on prend $Z = Q_2$ alors $A^t Z u = 0$ pour tout élément u de \mathbb{R}^{n-p} , ainsi la base recherchée est la matrice $Z = Q_2$.

La base $Z = Q_2$ fournit une matrice qui ne dégrade pas le conditionnement du système réduit issu de la programmation horizontale, mais malheureusement, la matrice Q_2 est en général pleine, ce qui fait perdre aux matrices de notre problème quadratique leur caractère creux.

Nous disposons actuellement de plusieurs méthodes pour calculer la factorisation QR d'une matrice (non carrée), nous en exposerons trois d'entre elles qui sont la transformation de Householder, l'orthogonalisation de Gram-Schmidt et la rotation de Givens.

2.3.1.1 Transformation de Householder

Soit I la matrice identité et v un vecteur. Une matrice de Householder est une matrice symétrique et orthogonale de la forme

$$P = I - \frac{2}{v^t v} v v^t \quad (2.32)$$

L'application de P sur un vecteur x donne $Px = x - (\frac{2v^t x}{v^t v})v$ et $P^2 = I$.

La question est : étant donnée deux vecteur x et y , existe-t-il une transformation de Householder telle que $Px = y$? Puisque P est orthogonale, nécessairement on doit avoir $\|x\| = \|y\|$. On a alors le lemme (2.3.1) suivant qui répond à notre question.

Lemme. 2.3.1. *Soit x et y deux vecteur, si $P = I - \frac{2}{v^t v} v v^t$ avec $v = x - y$ alors P est une matrice de Householder telle que $Px = y$*

Preuve.

Puisque P est orthogonale on doit avoir $\|x\|_2 = \|y\|_2$. Si on pose $v = x - y$ alors

$$\begin{aligned} v^t v &= (x - y)^t (x - y) \\ &= x^t x - 2x^t y + y^t y \\ &= (x^t x - x^t y) \\ &= x^t v \end{aligned}$$

Donc $2v^t x = v^t v$ d'où $Px = x - 2\frac{1}{2} \frac{v^t x}{v^t v} v = x - v = y$

CQFD \square

Dans la pratique y est choisi creux, $y = \sigma e_1$ où $\sigma = \pm \|x\|_2$. D'où $v = x - y = x - \sigma e_1$. En général nous choisissons $\text{sign}(\sigma) = -\text{sign}(x_1)$ où x_1 est la première composante de x .

La transformation de Householder ainsi constitue la première étape de la factorisation QR. En effet, appliquée sur la première colonne de la matrice A ,

Householder transforme cette dernière en une matrice intermédiaire dont les composantes de sa première colonnes sont nulles sauf la première. Le processus de la factorisation QR est une suite de transformation que nous pouvons illustrer à l'aide du schéma suivant

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{P_1} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{P_2} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{P_3} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \end{bmatrix} = \mathcal{R}$$

Donc à chaque étape, on s'attaque à une colonne, ainsi après, au plus $n+m$ étape, le processus est achevé. L'étape k est décrite comme suite. Soit $A = A_1$,

$$A_k = \begin{bmatrix} R_{k-1} & z_k & B_k \\ 0 & x_k & c_k \end{bmatrix}$$

où R_{k-1} est une matrice triangulaire supérieur d'ordre $(k-1)$.

Nous choisissons une matrice \tilde{P}_k telle que $\tilde{P}_k x_k = \sigma e_1$ et nous décrivons la matrice carrée P_k d'ordre $(m+p)$ comme suite :

$$P_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & \tilde{P}_k \end{bmatrix}$$

Pour calculer A_{k+1} nous avons besoin de calculer un produit de la forme $\tilde{P}_k c_k = (I - \frac{2vv^t}{v^t v})c_k = c_k - \beta v(v^t c_k)$ où $\beta = \frac{2}{v^t v}$. Donc on traduit le produit matrice-matrice en produit vecteur-matrice, ce qui coûte moins cher en terme d'opérations. Le coût totale d'une transformation de Householder est $2(m+p)^2(n + \frac{m}{3} - \frac{p}{3})$. En revanche pour expliciter la matrice Q nous avons besoin de

$4(n^2(m+p) + nm^2 + \frac{1}{3}(m^3 + p^3))$, ce qui est raisonnable pour des matrices de taille moyenne. Le deuxième avantage de l'utilisation d'une factorisation QR est sa stabilité numérique (voir à ce sujet Wilkinson [86, 87]).

2.3.1.2 L'orthogonalisation de Gram-Schmidt

La transformation de Householder n'est pas la seule façon pour calculer une factorisation QR. La méthode la plus ancienne est l'orthogonalisation de Gram-Schmidt qui coûte $2(m+p)^2(n+m)$ opérations sans expliciter la matrice Q . Soit a_j et q_j respectivement, la j^{eme} colonne de A et Q . L'algorithme suivant décrit cette méthode :

Pour j de 1 à m+p faire

Pour i de 1 à j-1 faire

$$r_{ij} = q_i^t a_j$$

Fin Pour

$$q'_j = a_j - \sum_{k=1}^{j-1} r_{kj} q_k$$

$$r_{jj} = \|q'_j\|_2$$

$$q_j = q'_j / r_{jj}$$

Fin Pour

Algorithme 10: L'orthogonalisation de Gram-Schmidt

2.3.1.3 La rotation de Givens

La dernière méthode de factorisation que nous présentons ici est la rotation de Givens. Soit G la matrice identité sauf les quatre composantes de coordonnées i et j :

$$G([i, j], [i, j]) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

où $c = \cos\theta$, $s = \sin\theta$ et θ est l'angle de la rotation.

Donc $y_j = 0$ si $s = \frac{x_j}{\sqrt{x_i^2 + x_j^2}}$ et $c = \frac{x_i}{\sqrt{x_i^2 + x_j^2}}$. Le coût total de cette transformation est $3(m+p)^2(n + \frac{2}{3}m - \frac{p}{3})$ donc 50% de plus qu'une transformation de Householder. Pour plus d'informations, le lecteur peut consulter Higham [40].

Pour les matrices de très grandes taille on peut utiliser d'autres méthodes que la factorisation QR, Nous présentons ici quelques idées exposés par Jonsson [46].

2.3.2 Méthode 2 : partitionnement de A_k^t

Supposons que la partition $A^t = [B \ N]$ où B est une matrice inversible, soit possible. alors on peut prendre :

$$Z = \begin{bmatrix} -B^{-1}N \\ I_{n-p} \end{bmatrix} \quad \text{et} \quad A^- = \begin{bmatrix} B^{-1} \\ 0_{n-p} \end{bmatrix}$$

En pratique, [24], c'est la factorisation LU appliquée à la matrice

$$\bar{A} = \begin{pmatrix} A^t \\ 0_{(n-p) \times (m+p)} \end{pmatrix}$$

qui trouve cette partition.

En suite à l'aide de méthode telle que MA28 de la librairie HSL, on factorise $B = LU$.

Cette méthode a l'avantage de conserver la forme creuse des matrices, mais nécessite deux factorisation LU , et ne préserve pas l'orthogonalité des déplacements verticaux.

Sautter lors des ses travaux concernant le pseudo-inverse à proposer une correction de v_k pour avoir l'orthogonalité de celle-ci , mais sa méthode souffre de la complexité de l'implémentation, et de son coût élevé en cas de problème de grande taille ; en plus la méthode est très instable si les matrices sont denses.

2.3.3 Méthode 3 : factorisation LU de A_k

Supposons que, après permutations on puisse factoriser A_k sous la forme

$$A_k = \begin{bmatrix} L_k \\ N_k \end{bmatrix} U_k$$

où $L_k \in \mathbb{R}^{(m+p) \times (m+p)}$, $N_k \in \mathbb{R}^{(n-p) \times (m+p)}$ et $U_k^{(m+p) \times (m+p)}$

alors le couple base tangente-inverse à droite à droite peut se donnée par :

$$Z = \begin{bmatrix} -L^{-t}N^t \\ I_{n-p} \end{bmatrix} \quad \text{et,} \quad A^- = \begin{bmatrix} L^{-t}U^{-t} \\ 0_{n-p} \end{bmatrix}$$

Cette méthode est plus avantageuse que la précédente de fait qu'elle ne nécessite qu'une factorisation LU . Si A_k est mal conditionnée, ce critère apparaîtra beaucoup plus dans la matrice U que dans la matrice L , ce qui représente un avantage énorme pour le calcul de la base tangente. Jonsson présente [46] une comparaison entre les deux dernières méthodes, et montre la suprématie de la dernière, bien que cette comparaison reste numérique.

2.3.4 Calcul d'un inverse à droite par les équations normales

C'est un calcul direct, supposons que nous cherchons à résoudre le système $A^t v = C$.

À l'aide d'une factorisation de Cholesky $AA^t = R^t R$ le système se résout par : $v = (R^t R)^{-1} AC$. Cette façon de faire présente beaucoup de failles :

1. Le mal conditionnement de A est aggravé par la multiplication.
2. Une grande perte de précision apparaît dans la solution.
3. La factorisation présente un éventuel échec, et elle est presque impossible si la taille du problème est grande.

2.3.5 Calcul d'un inverse à droite par système augmenté

On calcule le déplacement $v = -A^{-1}C$ de norme minimale en inversant dans $\mathbb{R}^{(n+m)}$ le système augmenté,

$$\begin{pmatrix} I_{n+m} & A^t \\ A & 0 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} = \begin{pmatrix} 0 \\ -C \end{pmatrix} \quad (2.33)$$

Ensuite on résout ce système, généralement indéfini, par l'algorithme de Bunch-Kaufmann (Pivot partiel), ou Bunch-Parlett (Pivot total). En fait une factorisation de la forme LDL^t est obtenue, D est une matrice diagonale par blocs de tailles 1×1 ou 2×2 , et L est une matrice triangulaire inférieure.

CHAPITRE 3

LA PROGRAMMATION NON LINÉAIRE

Vingt ans après la publication de l'article de Karmarkar pour la programmation linéaire, beaucoup de chercheurs continuent de s'inspirer de l'invention de la méthode de points intérieurs pour proposer des algorithmes d'optimisation. Le nombre d'articles couvrant la programmation non linéaire et les méthodes de points intérieurs est énorme, ainsi citer tous les travaux qui ont contribué à cette révolution est quasiment impossible (voir le site Web [101]), tout d'au moins, nous en citons tout au long de cette thèse un nombre d'articles qui a un lien direct avec les algorithmes que nous proposons et les articles fondateurs des méthodes que nous présenterons [8, 9, 21, 28, 58, 85]. Nous suivrons les notations données en introduction lors de la présentation de la programmation non linéaire.

Soit le problème d'optimisation non linéaire général suivant :

$$\left\{ \begin{array}{l} \min f(x) \\ \text{s.c.} \\ h_i(x) = 0 \quad i \in \mathcal{E} = \{1, \dots, p\} \\ g_i(x) \leq 0 \quad i \in \mathcal{I} = \{1, \dots, m\} \end{array} \right. \quad (PNL)$$

où f est une fonction numérique de classe C^2 , et les fonctions de contraintes g et h sont définies de \mathbb{R}^n dans \mathbb{R}^m et \mathbb{R}^p respectivement, g et h sont supposées de classe C^2 .

Le problème (PNL) est parfois appelé mixte, en fait dans plusieurs études les chercheurs font la différence entre un problème avec contraintes d'égalité, résolu le plus souvent avec des techniques SQP (voir paragraphe 3.6.2) et un

problème avec des contraintes d'inégalité, résolu avec des méthodes de points intérieurs (MPI). Nous traitons dans cette études les contraintes mixtes en combinant MPI et SQP.

3.1 Définitions et notations

On notera K l'ensemble des solutions réalisables de (PNL) :

$$K = \{x \in \mathbb{R}^n : g(x) \leq 0 \text{ et } h(x) = 0\}$$

et nous supposons que K est non vide.

Un vecteur $v \in \mathbb{R}^n$ est dit direction admissible au point $x_0 \in K$ si $v = 0$ ou s'il existe une suite $(x_n)_n$ dans K telle que $x_n \neq x_0$, $x_n \rightarrow x_0$ et $\frac{x_n - x_0}{\|x_n - x_0\|} \rightarrow \frac{v}{\|v\|}$, l'ensemble de ces directions est noté $C_K(x_0)$.

On pose $c(x) = (h^t(x), g^t(x))^t$ et $\mathcal{A}_{x_0} = \{i \in \{1, \dots, m\} : g_i(x_0) = 0\}$ l'ensemble des indices des contraintes saturées en x_0 appelé aussi l'ensemble actif associé au point x_0 . Soit G le cône défini par :

$$G(x_0) = \{v \in \mathbb{R}^n : \nabla g_i^t(x_0).v \leq 0 \quad \forall i \in \mathcal{A}_{x_0} \text{ et } \nabla h_i^t(x_0).v = 0 \quad \forall i \in \mathcal{E}\}$$

Si v est une direction admissible en x_0 alors v vérifie

$$\nabla g_i^t(x_0).v \leq 0 \quad \forall i \in \mathcal{A}_{x_0} \text{ et } \nabla h_i^t(x_0).v = 0 \quad \forall i \in \mathcal{E} \text{ i.e. } C_K(x_0) = G(x_0)$$

On note par $J_h(x) = [\nabla h_1(x), \dots, \nabla h_p(x)]^t$ et, $J_g(x) = [\nabla g_1(x), \dots, \nabla g_m(x)]^t$ les matrices jacobiennes des contraintes. La matrice $J(x) = (J_h^t(x), J_g^t(x))^t$ est le Jacobien de c au point x .

Nous notons $J_{\mathcal{A}}(x)$ le Jacobien composé uniquement des gradients actifs c'est-à-dire les gradients de h et de g_i pour $i \in \mathcal{A}_x$.

Contrairement à la programmation linéaire où les conditions d'optimalités sont nécessaires et suffisantes, la programmation non linéaire nécessite la qualification des contraintes, ce qui conduira des chercheurs (Kuhn et Tucker 1961, Abadie 1967) à introduire des conditions supplémentaires sur l'ensemble des solutions réalisables K afin de caractériser des solutions optimales locales.

Définition. 3.1.1. *On dit que K satisfait en $x_0 \in K$ l'hypothèse de la qualification des contraintes (QC) si et seulement si*

$$cl(C_K(x_0)) = G(x_0) \quad (3.1)$$

où $cl(A)$ est la clôture de A .

□

Évidemment la vérification de cette hypothèse n'est pas toujours simple, les lemmes suivants rassemblent les résultats les plus importants.

Lemme. 3.1.1. [57] *L'hypothèse (3.1) est vérifiée si une des deux assertions suivantes est satisfaite :*

1. *Les fonctions g_i sont convexes, les fonctions h_i sont linéaires et il existe $\bar{x} \in K$ tel que $g_i(\bar{x}) < 0$ et $h_i(\bar{x}) = 0 \quad \forall i \in \mathcal{I} \cup \mathcal{E}$.*
2. *En $x_0 \in K$ les colonnes de la matrice $J_A(x_0)$ sont linéairement indépendantes.*

□

La dernière assertion sera souvent notée "LICQ" : Qualification des Contraintes par Indépendance Linéaire. On trouve dans [54] une condition de qualification des contraintes plus intéressante, dans le sens où elle est toujours vérifiée si une assertion du lemme précédent est satisfaite. Cette condition est souvent

désignée par “MFCQ” portant les noms des mathématiciens qui l’ont proposée : Qualification des Contraintes de Mangasarian et Fromovitz.

Lemme. 3.1.2. [54] *Soit un problème non linéaire général, la condition de qualification des contraintes (3.1) est vérifiée au point x_0 , si x_0 est un point intérieur (strictement) ou, si les gradients des contraintes d’égalité au point x_0 sont linéairement indépendants et s’il existe un vecteur p tel que*

$$\nabla g_i^t(x_0).p < 0 \quad \forall i \in \mathcal{A}_{x_0} \text{ et } \nabla h_i^t(x_0).p = 0 \quad \forall i \in \mathcal{E}.$$

□

Dans le cas d’un problème non linéaire avec uniquement des contraintes d’inégalité, seule l’existence d’un vecteur non nul p vérifiant $\nabla g_i^t(x_0).p < 0 \quad \forall i \in \mathcal{A}_{x_0}$ est demandée, ce qui allège considérablement les conditions de qualification des contraintes.

Beaucoup de résultats théoriques utilisent la condition MFCQ mais la difficulté algorithmique a poussé les programmeurs à supposer la condition LICQ vérifiée.

La définition suivante est très utile pour toute la suite.

Définition. 3.1.2. *Soit la fonction*

$$L : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m \longrightarrow \mathbb{R}$$

$$(x, \lambda_h, \lambda_g) \longrightarrow L(x, \lambda_h, \lambda_g) = f(x) + \lambda_h^t h(x) + \lambda_g^t g(x)$$

L est appelé le Lagrangien associé au problème (PNL), λ_h et λ_g sont appelés multiplicateurs de Lagrange associés à x .

□

Définition. 3.1.3. *(Conditions KKT de premier ordre)*

On dit que les conditions KKT du premier ordre sont vérifiées au point x^ , ou que x^* est un point KKT de premier ordre, s’il existe deux vecteurs λ_h^* de \mathbb{R}^p*

et λ_g^* de \mathbb{R}_+^m appelés *multiplicateurs de Lagrange*, tels que :

$$\left\{ \begin{array}{ll} \nabla_x L(x^*, \lambda_h^*, \lambda_g^*) = 0 & \text{(stationnarité)} \\ h(x^*) = 0 & \\ g(x^*) \leq 0 & \text{(réalisabilité)} \\ (\lambda_g^*)_i g_i(x^*) = 0 \quad i = 1, \dots, m & \text{(complémentarité)} \end{array} \right.$$

□

A moins que le problème (PNL) n'est convexe, un point KKT de premier ordre n'est pas toujours un point optimal global ni même local, par conséquent on considère des conditions d'optimalité de second ordre.

Définition. 3.1.4. (*Ensemble des multiplicateurs acceptables.*)

Soit x_0 un point KKT (de premier ordre) pour le problème (PNL) alors on définit l'ensemble des multiplicateurs de Lagrange acceptable :

$$\mathcal{M}_\lambda(x) = \{\lambda \in \mathbb{R}^{m+p} \mid \nabla f(x_0) = J^t(x_0) \cdot \lambda, \lambda_g \geq 0 \text{ et } \lambda_g^t \cdot g(x_0) = 0\} \quad (3.2)$$

avec $\lambda^t = (\lambda_h^t, \lambda_g^t)$.

□

Cette définition montre entre autre que le multiplicateur λ_h n'est pas contraint en signe contrairement à λ_g et impose à chaque composante $(\lambda_g)_i$ d'être nulle si la contrainte associée (i.e. g_i) n'est pas active au point x_0 .

Un phénomène de dégénérescence peut se produire si $(\lambda_g)_i = 0$ pour une contrainte active, pour exclure ce cas on pose la définition suivante, dite de complémentarité stricte.

Définition. 3.1.5. (*La complémentarité stricte*)

Un point KKT x_0 vérifie la condition de la stricte complémentarité si $(\lambda_g)_i > 0$ pour toute contrainte g_i active au point x_0 .

□

Cette définition est très utile pour établir les théorèmes de convergence, malheureusement elle n'est pas toujours vérifiée, ce qui crée de sérieux problèmes pour tous les codes d'optimisation non linéaire, et expliquerait que ce point soit traité à part dans la littérature.

Remarque. 3.1.1. *Si la condition MFCQ est vérifiée pour un point KKT x_0 , alors l'ensemble $\mathcal{M}_\lambda(x_0)$ est bornée.*

□

Cette remarque est due à Gauvin (1977), on trouve sa démonstration rigoureuse dans [23].

3.2 Conditions d'optimalité

Avant de citer les différents résultats d'optimalité nous introduisons les définitions suivantes.

Définition. 3.2.1. (*Minimum local isolé*)

Un minimum local x_0 est dit isolé s'il existe un voisinage V de x_0 tel que V ne contient aucun autre minimum local que x_0 .

□

Cette définition est différente de la définition du minimum local strict i.e. un point dont sa valeur de la fonction objectif est inférieure strictement à la valeur de la fonction objectif de tout autre élément dans un intervalle donné.

Définition. 3.2.2. (*La matrice hessienne*)

On note par $W(x, \lambda) = \nabla^2 L(x, \lambda)$ la matrice hessienne du Lagrangien L au point (x, λ) .

□

Lemme. 3.2.1. [57] (Conditions nécessaires de second ordre)

Soit x^* un optimum local du problème (PNL) et que les contraintes sont qualifiées en x^* au sens de l'indépendance linéaire, alors il existe deux vecteurs λ_h^* de \mathbb{R}^p et λ_g^* de \mathbb{R}_+^m tels que : $(\lambda_g^*)^t \cdot g(x^*) = 0$, $\nabla f(x^*) = J^t(x^*) \cdot \lambda^*$ et

$$p^t W(x^*, \lambda^*) p \geq 0 \quad \forall p \in \mathbb{R}^{n+m} \text{ tel que } J_A(x^*) p = 0 \quad (3.3)$$

□

Remarque. 3.2.1. Si N^* est le noyau du Jacobien $J_A(x^*)$ alors la condition (3.3) est équivalente à $(N^*)^t W(x^*, \lambda^*) N^*$ est semi-définie positive.

□

En présence de la qualification des contraintes un minimum local isolé peut être caractérisé à l'aide des deux théorèmes suivants.

Théorème. 3.2.1. [57] (Conditions suffisantes d'optimalité 1)

Le point x^* est un minimum local isolé du problème (PNL) si :

1. x^* est un point KKT et $\mathcal{M}_\lambda(x^*) \neq \emptyset$.
2. La condition MFCQ est vérifiée au point x^* .
3. $\forall \lambda \in \mathcal{A}_\lambda(x^*)$ et pour tout vecteur p non nul tel que $\nabla f^t(x^*) p = 0$, $J_h^t(x^*) p = 0$ et $\nabla g_i^t(x^*) p_i \leq 0$ pour tout indice $i \in \mathcal{A}_{x^*}$, alors il existe un scalaire ω positif tel que $p^t W(x^*, \lambda) p \geq \omega \|p\|^2$.

□

On trouve dans la littérature un théorème plus fort que le théorème 3.2.1 mais plus utilisé à cause de sa simplicité algorithmique.

Théorème. 3.2.2. [57] (Conditions suffisantes d'optimalité 2)

Le point x^* est un minimum local isolé du problème (PNL) si :

1. x^* est un point KKT et la condition de la stricte complémentarité est vérifiée au point x^* (dans ce cas $\mathcal{M}_\lambda(x^*) = \{\lambda^*\}$).
2. La condition LICQ est vérifiée au point x^* .
3. Pour tout vecteur non nul p tel que $J_{\mathcal{A}}(x^*)p = 0$ il existe un scalaire $\omega > 0$ tel que $p^t W(x^*, \lambda^*) p \geq \omega \|p\|^2$

□

La complémentarité stricte signifie que la i ème composante de l'unique multiplicateur de Lagrange λ_g^* est strictement positive pour tout $i \in \mathcal{A}_{x^*}$ et que $(\lambda_h^*)_i \neq 0$ pour tout $i \in \mathcal{E}$. La deuxième condition peut être vérifiée en calculant la factorisation QR ou LU de la matrice $J_{\mathcal{A}}(x^*)$. La troisième condition est équivalente à ce que le Hessien réduit soit défini positif au point x^* .

Le fait que g et h sont de classe C^2 (supposé au début de ce chapitre), est utilisé pour la démonstration des théorèmes 3.2.1 et 3.2.2 (Cf. [23] pour les démonstrations complètes). En effet, cette hypothèse nous permet de dire que pour une itération k (k assez grand), les contraintes actives au point x_k le sont au point limite x^* i.e. $\mathcal{A}(x_k) \subseteq \mathcal{A}(x^*)$.

3.3 Inconsistance et redondance des contraintes

Dans ce paragraphe nous allons montrer que le conditionnement des matrices et l'absence de la complémentarité stricte ne sont pas les seules causes de la dégénérescence. En effet, Forsgren, Gill et Wright [23] pensent que l'inconsistance topologique et la redondance causent plus de dégâts qu'un éventuel mal-conditionnement et proposent des théorèmes de convergence en vu de leurs résultats. Nous présentons ces théorème de convergence au paragraphe suivant.

Définition. 3.3.1. (*Points intérieurs*)

Soit K un sous-ensemble de \mathbb{R}^n . Un point x est dit point intérieur de K si

$x \in K$ et s'il existe un voisinage de x contenu entièrement dans K . L'intérieur de K noté $\text{int}(K)$ est la réunion de tous les points intérieurs de K .

□

Définition. 3.3.2. (*Points frontières*)

Soit K un sous-ensemble de \mathbb{R}^n . Un point x est dit point frontière de K si tout voisinage de x contient au moins un élément de K et un élément n'appartenant pas à K . La frontière de K , noté ∂K , est la réunion de tous les points frontières de l'ensemble K .

□

Ces définitions sont purement topologiques et différentes de la définition d'ensemble de solutions strictements réalisables donnée comme suit,

Définition. 3.3.3. (*Point strictement réalisable*)

Un point est dit strictement réalisable si $g_i(x) < 0 \quad \forall i \in \mathcal{I}$ et $h_i(x) = 0$ pour tout $i \in \mathcal{E}$. Si K représente l'ensemble des solutions réalisables, on notera $\text{ir}(K) = \{x \in \mathbb{R}^n : g_i(x) < 0 \quad \forall i \in \mathcal{I} \text{ et } h_i(x) = 0 \quad \forall i \in \mathcal{E}\}$ l'ensemble des solutions strictements réalisables. Cet ensemble est noté parfois $\text{strict}(K)$.

□

Les ensembles $\text{int}(K)$ et $\text{ir}(K)$ ne sont pas toujours identiques ce qui provoque une inconsistance topologique à l'origine de beaucoup de difficultés. L'exemple suivant [23] illustre ce cas de figure : Supposons vouloir minimiser une fonction sous les contraintes suivantes : $g_1(x) = x^2 \geq 0$ et $g_2(x) = x + \gamma \geq 0$ avec $\gamma > 0$.

L'ensemble réalisable est donc $K = \{x \in \mathbb{R} : x^2 \geq 0 \text{ et } x + \gamma \geq 0\}$, or pour tout $x \in \mathbb{R}$ on a $x^2 \geq 0$ donc cette contrainte n'a aucun rôle dans la définition de K , et le point $x = 0$ est un point intérieur de K , $0 \in \text{int}(K)$ or ce point est exclu de la définition de l'intérieur relatif de K en effet

$ir(K) = \{x : 0 > x > -\gamma\} \cup \{x : x > 0\} = int(K) \setminus \{0\}$. Si l'ensemble K est convexe cette difficulté n'a pas lieu d'être, mais si K est non convexe (cas de notre étude) cette difficulté peut surgir.

Afin d'éclaircir ce point, donnons les résultats suivants (pour plus de détails et notamment pour les démonstrations des théorèmes, le lecteur peut consulter [23]).

Définition. 3.3.4. (*Inconsistance topologique*)

Une contrainte $g_i(x) \leq 0$ est dite inconsistante topologiquement au point \hat{x} si $\hat{x} \in int(K)$ et $g_i(\hat{x}) = 0$. La contrainte $g_i(x) \leq 0$ est dite consistante topologiquement si pour tout $x \in K$, $g_i(x) = 0$ est vérifiée seulement si $x \notin int(K)$.

□

Cette inconsistance topologique provoque une redondance au niveau des contraintes. Le lemme suivant regroupe les résultats les plus intéressants.

Lemme. 3.3.1. [23] (*Redondance topologique*)

1. Supposons que la contrainte $g_i(x) \leq 0$ est inconsistante topologiquement au point \hat{x} et que $\nabla^2 g_i(\hat{x})$ soit définie négative alors il existe un voisinage de \hat{x} dans lequel tout point est strictement réalisable pour la contrainte $g_i(x)$.
2. Si g_i est concave dans \mathbb{R}^n et inconsistante topologiquement au point \hat{x} alors $\forall x \in \mathbb{R}^n$ on a $g_i(x) \leq 0$, dans ce cas la contrainte g_i est dite redondante.

Preuve.

g_i est inconsistante topologiquement pour \hat{x} donc par définition $g_i(\hat{x}) = 0$ et $\hat{x} \notin int(K)$. Or par définition de l'intérieur de K , il existe un voisinage

de \hat{x} contenu complètement dans K , c'est-à-dire $\exists \varepsilon > 0$ tel que pour tout x vérifiant $\|x - \hat{x}\| \leq \varepsilon$ on ait $g_i(x) \leq 0$ d'où $g_i(x) \leq g_i(\hat{x}) = 0$ ainsi \hat{x} est un maximum local de la fonction g_i or d'après l'hypothèse de la première assertion $\nabla^2 g_i(\hat{x})$ est définie négative ce qui signifie que \hat{x} est un maximum local isolé, donc il existe un voisinage V de \hat{x} tel que $\forall x \in V$ $g_i(x) < g_i(\hat{x}) = 0$ d'où le résultat de l'assertion. Si g_i est concave alors le maximum isolé \hat{x} est global d'où $\forall x \in \text{int}(K)$, on a $g_i(x) \leq g_i(\hat{x}) = 0$.

CQFD \square

En pratique il est très utile de détecter cette redondance, ce qui implique que tout point au voisinage d'un tel point (un point pour lequel la contrainte est inconsistante) est réalisable, et si la contrainte est concave alors elle est entièrement redondante, i.e. elle est réalisable pour tout élément de l'espace, c'était le cas de l'exemple précédent où la première contrainte $C_1(x) = -x^2 \leq 0$ est une fonction concave et inconsistante topologiquement au point $x = 0$.

Remarque. 3.3.1. *Toutes les contraintes sont consistantes topologiquement au point \hat{x} si $\forall i \in \mathcal{A}_{\hat{x}}$ on a $\nabla g_i(\hat{x}) \neq 0$ ou $\nabla^2 g_i(\hat{x})$ n'est pas définie négative.*

\square

En plus des difficultés causées par l'inconsistance topologique et le mauvais conditionnement des matrices, l'existence des contraintes d'égalités complique la donne c'est la raison pour laquelle beaucoup d'études sont consacrées aux problèmes non linéaires avec seulement des contraintes d'inégalités ou seulement des contraintes d'égalités [8, 62]. Le paragraphe 3.6.2 revient sur ces cas de figure.

3.4 Théorèmes de convergence

Soit le problème non linéaire sous contraintes d'inégalités :

$$\begin{cases} \min f(x) \\ \text{s.c.} \\ g(x) \leq 0 \end{cases} \quad (3.4)$$

où f est une fonction numérique de classe C^2 , et la fonction g est définie de \mathbb{R}^n dans \mathbb{R}^m de classe C^2 .

Les résultats que nous avons présenté dans les paragraphes précédents de ce chapitre servent à démontrer les théorèmes de convergence. Ces résultats donnent des conditions pour l'existence d'une suite de minimums locaux de la fonction barrière qui converge vers le minimum local de la fonction initiale, sachant que chaque minimum local de la fonction barrière est la solution d'un problème non linéaire sans contrainte.

La fonction barrière la plus utilisée est la fonction barrière logarithmique $\mu \sum_{i=1}^m \ln(-g_i(x))$ où μ est un scalaire positif.

Soient $K = \{x \in \mathbb{R}^n \mid g(x) \leq 0\}$ l'ensemble des solutions réalisables et $ir(K) = \{x \in \mathbb{R}^n \mid g(x) < 0\}$ l'ensemble des solutions strictement réalisables du problème (3.4).

Lemme. 3.4.1. [23] *Soit X un ensemble compact tel que $ir(K) \cap X$ est non vide. Soit $(y_k)_k$ une suite d'éléments de $ir(K) \cap X$ convergeant vers $\bar{y} \in X \cap \partial(K)$.*

Soit ϕ une fonction continue définie dans $ir(K) \cap X$ avec la propriété : $\phi(y_k)$ n'est pas bornée supérieurement quand $k \rightarrow \infty$ pour toute suite $(y_k)_k$ décrite ci-dessus. Alors ϕ atteint son minimum fini $\phi^ = \phi(x^*)$ sur $ir(K) \cap X$.*

□

Ce lemme montre que la fonction barrière logarithmique atteint sa limite à l'intérieur du domaine réalisable et jamais sur la frontière, ce qui explique que les minimums retrouvés par les méthodes de points intérieurs sont approximatifs.

Numériquement on s'aperçoit vite que si le minimum se trouve sur la frontière les valeurs $1/g_i(x)$ et $1/g_i^2(x)$ pour $i \in \mathcal{A}$ explosent, ce qui conduit à l'inefficacité des méthodes barrières, néanmoins nous introduisons une technique de correction au paragraphe 5.1 pour remédier à ce problème.

Le théorème de convergence suppose l'existence d'un ensemble d'isolation dont nous donnons la définition.

Définition. 3.4.1. (*Sous-ensemble isolé*)

Soit \mathcal{N} et \mathcal{N}^* deux ensembles de \mathbb{R}^n tels que $\mathcal{N}^* \subseteq \mathcal{N}$. L'ensemble \mathcal{N}^* est dit sous-ensemble isolé de \mathcal{N} s'il existe un ensemble fermé F tel que $\mathcal{N}^* \subset \text{int}(F)$ et $F \cap \mathcal{N} = \mathcal{N}^*$.

□

En fait c'est une sorte de séparation des éléments de \mathcal{N}^* du reste des éléments de \mathcal{N} .

Le lecteur peut consulter [19] et [23] à cet égard et notamment pour la démonstration du théorème suivant, or la démonstration dans [19] n'est pas complète du fait qu'elle omet une éventuelle négativité de la fonction barrière.

Théorème. 3.4.1. [23] (*Convergence locale des méthodes de points intérieurs*)

Soit le problème d'optimisation 3.4 où f et g sont continues. Soit K l'ensemble des solutions réalisables et \mathcal{N} l'ensemble, non vide, des minimums pour lesquels la fonction objectif prend la valeur optimale f^* . Soit $(\mu_k)_k$ une suite strictement décroissante du paramètre barrière telle que $\lim_{k \rightarrow \infty} \mu_k = 0$. Supposons que :

1. Il existe un ensemble compact non vide \mathcal{N}^* contenant des minimums locaux qui soit un ensemble isolé dans \mathcal{N} .

2. Au moins un point de \mathcal{N}^* est dans la fermeture de $ir(K)$.

Alors nous avons les résultats suivants :

1. Il existe un ensemble compact X tel que $\mathcal{N}^* \subset \text{int}(X)$ et que pour tout $\bar{x} \in X$ sans être dans \mathcal{N}^* , alors $f(\bar{x}) > f^*$.
2. Pour tout μ_k assez petit, il existe un minimum de la fonction barrière $B(x, \mu_k)$ dans $ir(K) \cap X$ qu'on notera y_k , $y_k \in ir(K) \cap \text{int}(X)$.
3. Chaque suite des y_k admet au moins une sous-suite $(x_k)_k$ convergente.
4. La sous-suite $(x_k)_k$ atteint sa limite x_∞ dans \mathcal{N}^* .
5. $\lim_{k \rightarrow \infty} f(x_k) = f^* = \lim_{k \rightarrow \infty} B(x_k, \mu_k)$.

□

Le théorème suppose à peine la continuité de f et de g et ne demande en aucun cas l'existence des dérivées premières et secondes, donc il peut s'appliquer même si les conditions suffisantes ne sont pas vérifiées. Le théorème prouve l'existence d'une sous-suite convergente dans \mathcal{N}^* et implicitement d'autres sous-suites peuvent converger vers des limites qui ne soient pas des minimums locaux du problème initial.

Le théorème suivant améliore un peu ces résultats théoriques mais nécessite des conditions beaucoup plus fortes.

Théorème. 3.4.2. [23] (*Convergence de la trajectoire barrière*)

Soit le problème d'optimisation (3.4) où f et g sont continues. Supposons que l'intérieur relatif du domaine réalisable est non vide. Soit x^* un minimum local du problème (3.4). Supposons que les conditions suffisantes d'optimalité soient satisfaites au point x^* :

1. x^* est un point KKT.
2. Les contraintes sont qualifiées au sens de la MFCQ au point x^* .

3. Il existe $\omega > 0$ tel que $d^t W(x^*, \lambda)d \geq \omega \|d\|^2$ pour tout multiplicateur acceptable λ et tout vecteur non nul d vérifiant :

$$\nabla_x f(x^*)d = 0 \text{ et } J_{\mathcal{A}}d \geq 0.$$

Supposons appliquer une méthode barrière logarithmique avec μ_k décroissant strictement vers 0, alors

1. Il existe au moins une sous-suite de minimum locaux de la fonction barrière $B(x, \mu_k)$ (minimisation sans contraintes) convergeant vers x^* . Notons par $(x_k)_k$ cette sous-suite.

2. La suite de multiplicateur $(\lambda_k)_k$, dont les composantes sont $-\mu_k/g_i(x_k)$, est bornée.

3. $\lim_{k \rightarrow \infty} \lambda_k = \bar{\lambda} \in \mathcal{M}_{\lambda}(x^*)$.

Si de plus La complémentarité stricte est vérifiée au point x^* , alors

4. $\bar{\lambda}_{\mathcal{A}} > 0$

5. Pour k assez grand, la matrice hessienne $\nabla^2 B(x_k, \mu_k)$ est définie positive.

6. Il existe une unique trajectoire $x(\mu)$ continue et différentiable pour μ proche de 0 positivement, où chaque valeur $x(\mu)$ est un minimum local de la fonction barrière $B(x, \mu)$.

7. $\lim_{\mu \rightarrow 0^+} x(\mu) = x^*$.

□

Le théorème suppose que x^* est un minimum local isolé dans la fermeture de $ir(K)$ donc le théorème (3.4.1) est automatiquement applicable à x^* . La complémentarité stricte est une hypothèse assez forte qui implique entre autre que le Hessien réduit est défini positif et qu'il existe un scalaire positif M tel que $\frac{\mu_k}{\|x_k - x^*\|} \geq M$, i.e. $\|x_k - x^*\| = \mathcal{O}(\mu_k)$ (voir déf. 3.4.2 ci-dessous). Ce résultat est très utile pour prouver que les multiplicateurs de la fonction barrière convergent vers un multiplicateurs $\bar{\lambda}$ qui vérifie la complémentarité

stricte, i.e. $\bar{\lambda}_{\mathcal{A}} > 0$. En effet

$$\bar{\lambda}_i = \frac{1}{(\nabla g_i(x^*))^t t} \quad \text{pour } i \in \mathcal{A}_{x^*} \quad (3.5)$$

où $t = \lim_{k \rightarrow \infty} \frac{x_k - x^*}{\mu_k}$.

Autrement dit, si on prend $x_k = x^* + \mu_k t_k$ alors $t_k \rightarrow t$ quand $k \rightarrow \infty$. L'égalité (3.5) est à l'origine de l'idée de l'approche barrière non tangentielle (voir [23] page 558).

Le théorème (3.4.2) montre aussi que l'élément crucial de la défini-positivité du Hessien, et donc de la vérification des conditions suffisante d'optimalité, est la positivité des multiplicateurs correspondants aux contraintes actives. L'unicité de la trajectoire $x(\mu)$ se démontre grâce au théorème des fonction implicites appliqué pour résoudre

$$\Phi(x, \mu) = 0 \quad (3.6)$$

où $\Phi(x, \mu) = \nabla f(x) - \mu J_{\mathcal{A}}^t(x) G^{-1}(x) e = 0$.

e représente le vecteur unité de \mathbb{R}^n , $G(x)$ est la matrice diagonale dont les éléments diagonaux sont les composantes $g_i(x)$ pour $i \in \mathcal{A}_x$ et $G^{-1}(x)$ est sa matrice inverse. On notera par $G^{-2}(x)$ la matrice diagonale dont les éléments diagonaux sont les $g_i^{-2}(x)$ pour $i \in \mathcal{A}_x$.

Ce qui montre encore une fois l'importance de la complémentarité stricte, en effet l'existence d'un multiplicateur λ_i nul pour les indices $i \in \mathcal{A}(x^*)$ remet en cause l'unicité de la trajectoire, alors qu'on ne peut pas assurer la convergence de toutes les trajectoires $x(\mu)$.

Rappelons la définition suivante :

Définition. 3.4.2. *Étant donnée une fonction réelle, vectorielle ou matricielle f d'une variable positive h , soit k_u et k_l des constantes et p fixe (le plus souvent*

un entier naturel).

On dit que $f = \mathcal{O}(h^p)$ s'il existe $k_u > 0$ tel que $\|f\| \leq k_u h^p$ pour h suffisamment petit.

On dit que $f = o(h^p)$ si pour tout $\varepsilon > 0$, $\|f\|/h^p < \varepsilon$ pour tout h suffisamment petit.

On dit que $f = \Omega(h^p)$ s'il existe $k_l > 0$ tel que $\|f\| \geq k_l h^p$ pour h suffisamment petit.

On dit que $f = \theta(h^p)$ s'il existe $k_l > 0$ et $k_u > 0$ tel que $k_l h^p \leq \|f\| \leq k_u h^p$ pour h suffisamment petit.

□

3.5 Méthodes primales

La recherche d'une trajectoire passe donc par la résolution de l'équation non linéaire (3.6), avec

$$\Phi(x, \mu) = \nabla f(x) - \mu J_A^t(x) G^{-1}(x) e = 0 \quad (3.7)$$

où x est appelé variable primale par analogie à la programmation linéaire. l'égalité (3.7) peut s'écrire sous la forme $F(x) = 0$ où F est une forme non linéaire continue et différentiable de \mathbb{R}^n dans \mathbb{R}^n (μ étant fixé).

La méthode de Newton permet la résolution itérative de cette équation, en effet lorsque $\nabla_x F$ est non singulier, alors on peut construire une suite $(d_k)_k$ vérifiant $\nabla F^t(x_k) d_k = -F(x_k)$ de façon à ce que la suite $(x_k)_k$ définie par

$$\begin{cases} x_0 \text{ donné} \\ x_{k+1} = x_k + d_k \end{cases} \quad (3.8)$$

converge vers un point x^* qui vérifie $F(x^*) = 0$.

3.5.1 Calcul du pas de déplacement

Deux approches sont utilisées pour calculer le pas de déplacement et améliorer la convergence. Les méthodes de la région de confiance calculent le pas de déplacement en résolvant le problème (RC) (voir notre introduction). L'approche de la recherche linéaire calcule une direction de descente puis l'améliore en résolvant le problème d'optimisation suivant :

$$\min_{\alpha} \{f(x_k + \alpha d_k) : \alpha \geq 0\}$$

3.5.2 Newton classique

La méthode classique de Newton est la résolution de l'équation linéaire

$$\nabla_{xx}^2 B d = -\nabla_x B \quad (3.9)$$

où B est le Hessien du problème non linéaire, c'est-à-dire

$$(\nabla_{xx}^2 f + \sum_{i=1}^m \frac{\mu}{g_i} \nabla^2 g_i + \mu J^t G^{-2} J) d = -\nabla_x f - \mu J^t G^{-1} e \quad (3.10)$$

La méthode est dite classique puisque elle fut la première utilisée dans les années soixante, c'est aussi une méthode primale car les itérations se font uniquement par rapport à la variable primale x .

Lorsque μ s'approche de zéro, la matrice $\nabla_{xx}^2 B$ sera dominée par la matrice $\mu J_{\mathcal{A}}^t g^{-2} J_{\mathcal{A}}$, donc si le nombre des contraintes actives est $m_{\mathcal{A}}$ est tel que $0 < m_{\mathcal{A}} < m$ alors la matrice souffrira d'une insuffisance au niveau du rang, la dimension de la matrice n'est pas impliquée directement dans cette insuffisance. Le théorème suivant résume les résultats de cette insuffisance, étudiée par Wright [88].

Théorème. 3.5.1. [88] (*Valeurs propres du Hessien*)

Soit x_δ un point strictement réalisable i.e. $x_\delta \in ir(K)$ vérifiant

$$\|x_\delta - x^*\| \leq \delta \text{ où } \delta \ll 1, \mu = \mathcal{O}(\delta) \text{ et } \min g_i(x_\delta) = \Theta(\mu) \forall i \in \mathcal{A}_{x^*} \quad (3.11)$$

où δ et μ sont suffisamment petits.

Soit $m_{\mathcal{A}}$ le nombre des contraintes actives au point x^* , et supposons que $0 < m_{\mathcal{A}} < m$. Soit $N_{\mathcal{A}}$ la matrice dont les colonnes forment une base orthonormée de l'espace noyau de $J_{\mathcal{A}}(x)$. Alors

1. La matrice $\nabla^2 B(x_\delta, \mu)$ est définie positive.
2. La matrice $\nabla^2 B(x_\delta, \mu)$ a $m_{\mathcal{A}}$ valeurs propres qui soient de l'ordre $\Theta(1/\mu)$ et $n - m_{\mathcal{A}}$ valeurs propres qui soient de l'ordre $\Theta(1)$ d'où $\text{cond}(\nabla^2 B(x_\delta, \mu)) = \Theta(1/\mu)$.
3. Soit $\{\nu_1, \dots, \nu_{n-m_{\mathcal{A}}}\}$ les $n - m_{\mathcal{A}}$ valeurs propres de $N_{\mathcal{A}}^t(x_\lambda)W(x_\lambda, \pi)N_{\mathcal{A}}(x_\lambda)$, où π sont des estimations des multiplicateurs. Les $n - m_{\mathcal{A}}$ valeurs propres de $\nabla^2 B(x_\delta, \mu)$ noté $\xi_1, \dots, \xi_{n-m_{\mathcal{A}}}$ vérifient

$$|\xi_k - \nu_k| = \mathcal{O}(\mu), \quad k = 1, \dots, n - m_{\mathcal{A}} \quad (3.12)$$

□

Dans une grande partie d'exemples numériques, la méthode barrière résout les problèmes beaucoup mieux qu'on s'attendait. Une étude au niveau des erreurs de calcul par Forsgren et al. [23] pourrait expliquer les raisons de cet enchantement, mais la méthode classique reste inefficace puisqu'elle produit des points non réalisables au cours des itérations.

3.5.3 Système augmenté

Afin de corriger les inconvénients de la méthode de Newton, des chercheurs ont proposé de résoudre un système augmenté qui ne souffre pas automatiquement du mal conditionnement des matrices lorsque μ tend vers zéro [33].

Soit $W(x, \pi) = \nabla^2 f(x) - \sum_{i=1}^m \pi_i \nabla^2 g_i(x)$ où $\pi_i = \pi_i(x, \mu) = -\mu/g_i(x)$. Le système proposé est le suivant :

$$\begin{pmatrix} W(x, \pi) & -J^t(x) \\ \Pi J(x) & G(x) \end{pmatrix} \begin{pmatrix} d \\ t \end{pmatrix} = - \begin{pmatrix} \nabla f(x) - J^t(x)\pi \\ 0 \end{pmatrix} \quad (3.13)$$

où d est le pas de déplacement de Newton recherché et Π est la matrice $\text{diag}(\pi)$.

Malheureusement la méthode produit des pas de mauvaise qualité dans le sens de la réduction de la fonction barrière (voir [23]).

3.5.4 Extrapolation

L'extrapolation voulait résoudre les difficultés rencontrées par la méthode du système augmenté, en effet en remarquant que la cause principale à ces difficultés est que même si les vecteurs $\pi_i(x, \mu) = -\mu/g_i(x)$ étaient de bonnes estimations aux multiplicateurs de Lagrange à une certaine itération (itération déjà réalisée), ce n'est pas toujours le cas des vecteurs $\pi_i(x, \hat{\mu}) = -\hat{\mu}/g_i(x)$ pour l'itération suivante (en cours), l'extrapolation propose donc de remplacer le paramètre barrière courant $\hat{\mu}$ par le paramètre barrière de l'itération précédente μ dans l'expression de π avant de résoudre le système

$$\begin{pmatrix} W(x, \pi) & -J^t(x) \\ \Pi J(x) & G(x) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix} = - \begin{pmatrix} \nabla f(x) - J^t(x)\pi \\ 0 \end{pmatrix} \quad (3.14)$$

ce qui produit de bons résultats, un procédé similaire à celui ci est utilisé pour corriger les estimations de multiplicateurs de Lagrange dans les codes KNITRO et MITR (voir le chapitre suivant).

3.5.5 Les fonctions barrières modifiées

A fin de remédier au problème de la singularité du Hessien lorsque μ tend vers zéro, la méthode de la barrière modifiée (voir [14, 23, 61]) se base sur la remarque que pour un paramètre de barrière μ fixé, les ensembles réalisables des contraintes $g_i(x) \leq 0$ et $-\mu \ln(1 - g_i(x)/\mu) \leq 0$ sont identiques. Malheureusement, la méthode n'est pas pratique, en effet, si pour une itération donnée le Hessien n'est pas suffisamment défini positif, alors le paramètre barrière sera réduit et l'identité qui relie les deux ensembles réalisables pourrait être perdu. Une correction est possible avec l'utilisation de la fonction barrière modifiée $f(x) - \sum_{i=1}^m \lambda_i \ln(1 - g_i(x)/s_i)$ où s_i peut être interprétée comme une variable auxiliaire.

Toutes ces difficultés ont permis aux méthodes primales-duales d'être plus attractives, les résultats numériques consolident cette popularité des méthodes.

3.6 Méthodes primales-duales

L'utilisation des méthodes primales-duales est en croissance monotone [8, 14, 23, 18, 64]. Ces méthodes suggèrent, à l'instar de leurs semblables proposées pour la programmation linéaire, d'étudier indépendamment l'évolution des variables primales x et les variables duales λ représentées par les multiplicateurs de Lagrange.

L'idée de base de toute méthode primale-duale est la recherche d'un point

réalisable (x_μ, λ_μ) qui résout l'équation non linéaire

$$F^\mu(x, \lambda) = 0 \quad (3.15)$$

où

$$F^\mu(x, \lambda) = \begin{pmatrix} \nabla f(x) - J^t(x)\lambda \\ G(x)\lambda - \mu e \end{pmatrix} \quad (3.16)$$

Ainsi à partir d'un point courant (x, λ) , la méthode de Newton donne la direction de descente (d_x, d_λ) solution du système linéaire suivant :

$$\begin{pmatrix} W(x_\mu, \lambda_\mu) & -J^t(x) \\ \Lambda J(x) & G(x) \end{pmatrix} \begin{pmatrix} d_x \\ d_\lambda \end{pmatrix} = \begin{pmatrix} \nabla f(x) - J^t(x)\lambda \\ G(x)(\lambda - \pi(x, \mu)) \end{pmatrix} \quad (3.17)$$

où π est le vecteur dont les composantes sont $-\mu/c_i(x)$, $i = 1, \dots, p + m$ et $\Lambda = \text{diag}(\lambda)$. Dans le cadre des méthodes de points intérieurs W représentera le Hessien du Lagrangien pénalisé et le système linéaire (3.17) est rendu symétrique afin de récupérer un bon modèle quadratique [8, 9, 64].

Le succès des méthodes primales-duales s'explique par leur efficacité à suivre la trajectoire barrière. En effet, soit $(x, \lambda) = (x(\mu), \lambda(\mu))$ le point courant, où μ est le paramètre de pénalisation. Supposons que $(x^+, \lambda^+) = (x(\mu^+), \lambda(\mu^+))$ est le point suivant ($0 < \mu^+ < \mu$). La direction de déplacement est donnée par

$$\begin{pmatrix} W(x_\mu, \lambda_\mu) & -J^t(x) \\ \Lambda J(x) & G(x) \end{pmatrix} \begin{pmatrix} d_x \\ d_\lambda \end{pmatrix} = \begin{pmatrix} 0 \\ (\mu - \mu^+)e \end{pmatrix} \quad (3.18)$$

ainsi on peut écrire

$$\begin{aligned}x(\mu^+) &= x(\mu) + (\mu^+ - \mu)x'(\mu) \text{ et} \\ \lambda(\mu^+) &= \lambda(\mu) + (\mu^+ - \mu)\lambda'(\mu)\end{aligned}$$

où $(x', \lambda') = (x'(\mu), \lambda'(\mu))$ dit point tangent est solution du problème linéaire (3.19) suivant

$$\begin{pmatrix} W(x_\mu, \lambda_\mu) & -J^t(x) \\ \Lambda J(x) & G(x) \end{pmatrix} \begin{pmatrix} x'_\mu \\ \lambda'_\mu \end{pmatrix} = \begin{pmatrix} 0 \\ e \end{pmatrix} \quad (3.19)$$

Cette propriété n'a pas lieu pour les méthodes primales.

En plein essor, les méthodes primales-duales essaient de répondre à quatre questions :

1. Formulation du système linéaire de Newton (pénalisation, symétrisation,...).
2. Choix de la stratégie de résolution (SLP, SQP, méthodes directes,...).
3. Mesurer l'amélioration vers l'optimalité (fonctions de mérite, potentiel, filtre).
4. Traiter la non convexité et les contraintes d'égalité.

3.6.1 Les méthodes directes

Supposons μ fixé, les itérations qui servent à produire une direction de déplacement sont dites internes. Lorsqu'un point est proche de la trajectoire des minimums pénalisés (minimums des problèmes pénalisés), les méthodes primales-duales sont très performantes et sous l'hypothèse de la complémentarité stricte, la vitesse de convergence externe est Q-quadratique [18]. Le coût de chaque itération externe est dominé par la résolution du système linéaire (3.17)

d'où l'importance de chercher des méthodes de plus en plus performantes pour résoudre le système linéaire et qui puissent garantir que la direction produite est une véritable direction de descente pour le problème non linéaire.

En plus de la recherche de nouvelles méthodes, l'approche la plus commune est la condensation des données, les matrices hessiennes et jacobiennes contiennent de larges blocs nulles, ce qui permettrait, dans le cas de dimensions raisonnables d'utiliser des méthodes de factorisation QR ou LU, dans d'autres cas on en préfère des méthodes itératives comme le gradient conjugué préconditionné ou les méthodes D.C. Dans cette étude on veut étaler et proposer des méthodes et techniques résolvant le problème non linéaire général de grande taille, ce qui remet en cause l'éventualité d'avoir recours aux méthodes directes, néanmoins le champ des méthodes hybrides reste ouvert [37].

3.6.2 La programmation quadratique séquentielle

La stratégie SQP traite essentiellement la programmation non linéaire avec contraintes d'égalité. Considérons le problème

$$\left\{ \begin{array}{l} \min f(x) \\ \text{s.c.} \\ h(x) = 0 \\ x \in \mathbb{R}^n \end{array} \right. \quad (3.20)$$

où $f : \mathbb{R}^n \rightarrow \mathbb{R}$ et $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$ sont deux fonctions de classe C^2 .

Soit $L(x, \lambda) = f(x) - \lambda^t h(x)$ le Lagrangien du problème (3.20). $\lambda \in \mathbb{R}^p$ est le multiplicateur de Lagrange.

Les conditions d'optimalité de premier ordre sont données par le système

suivant

$$\begin{cases} \nabla_x L(x, \lambda) = \nabla f(x) - J_h^t \lambda = 0 \\ h(x) = 0 \end{cases} \quad (3.21)$$

où J_h est la matrice jacobienne de la fonction $h(x)$. Soit

$$F(x, \lambda) = \begin{bmatrix} \nabla f(x) - J_h^t \lambda \\ h(x) \end{bmatrix} \quad (3.22)$$

Le système (3.21) est équivalent à un système non linéaire de $(n+p)$ inconnus (x, λ) et $(n+p)$ égalités $F(x, \lambda) = 0$. Résoudre ce système par la méthode de Newton revient à chercher une direction de déplacement $d^t = (d_x^t, d_\lambda^t)$ telle que

$$\nabla F^t(x, \lambda)d = -F(x, \lambda) \quad (3.23)$$

C'est-à-dire, construire une suite de vecteur (x_k, λ_k) à partir d'un point initial :

$$\begin{pmatrix} x_{k+1} \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ \lambda_k \end{pmatrix} + \begin{pmatrix} d_x \\ d_\lambda \end{pmatrix} \quad (3.24)$$

où

$$\begin{bmatrix} W_k & -J(x_k)^t \\ J(x_k) & 0 \end{bmatrix} \begin{bmatrix} d_x \\ d_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla f(x_k) + J(x_k)^t \lambda \\ -h(x_k) \end{bmatrix} \quad (3.25)$$

avec $W_k = W(x_k, \lambda_k)$ est le Hessien du Lagrangien évalué au point (x_k, λ_k) . La méthode de Newton est bien définie et sa vitesse de convergence devient

quadratique si le point initial est proche de l'optimum et si les hypothèses de la non singularité de la matrice jacobienne des contraintes et la condition KKT sont vérifiées à savoir :

1. $J_k = J_h(x_k)$ est une matrice de plein rang.
2. La matrice W_k est définie positive dans l'espace noyau de la jacobienne des contraintes i.e. $d^t W_k d > 0 \quad \forall d \neq 0$ et $J_k d = 0$.

La stratégie SQP constitue une alternative à la résolution directe du système linéaire (3.25), en effet considérons le problème quadratique suivant :

$$\begin{cases} \min \frac{1}{2} d^t W_k d + \nabla f_k^t d \\ \text{s.c. } J_k d + h_k = 0 \end{cases} \quad (3.26)$$

où W_k est une matrice carrée d'ordre $n + p$, $d \in \mathbb{R}^{n+p}$, ∇f_k est un vecteur de \mathbb{R}^{n+p} , J_k est une matrice $(p, n + p)$ et h_k est un vecteur de \mathbb{R}^p .

Dans le cas où le problème quadratique (3.26) est convexe alors son unique solution (p_k, μ_k) vérifie le système suivant

$$\begin{cases} W_k d_k + \nabla f_k - J_k^t \mu_k = 0 \\ J_k p_k + h_k = 0 \end{cases} \quad (3.27)$$

ou sous forme matricielle, en rajoutant le terme $J_k^t \lambda_k$ à gauche et à droite de la première équation

$$\begin{bmatrix} W_k & -J_k^t \\ J_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \mu_k - \lambda_k \end{bmatrix} = \begin{bmatrix} -\nabla f_k + J_k^t \lambda_k \\ -h_k \end{bmatrix} \quad (3.28)$$

En comparant (3.25) et (3.28) on en déduit le principe de la méthode SQP. Il s'agit de résoudre le problème quadratique (3.26) avec $W_k = W(x_k, \lambda_k)$, $\nabla f_k = \nabla f(x_k)$, $J_k = J_h(x_k)$, $h_k = h(x_k)$ et de prendre $d_x = p_k$ et $d_\lambda = \mu_k - \lambda_k$

ou tout simplement $\lambda_{k+1} = \mu_k$, ainsi la résolution de l'équation $F(x, \lambda) = 0$ se traite itérativement en résolvant une suite de sous-problèmes quadratiques, d'où l'appellation : Programmation Quadratique Séquentielle (SQP).

Il est certain que sous l'hypothèse de la non singularité de W_k , si f et h admettent des dérivées secondes lipschitziennes et si le point initial (x_0, λ_0) est proche de l'optimum alors la méthode converge Q-quadratiquement, mais l'origine même de la méthode (programmation non linéaire) remet en cause ces hypothèses, d'où la nécessité d'utiliser d'autres techniques. En général, une contrainte sous forme de région de confiance est imposée au sous-problème quadratique, c'est le cas pour la méthode SDC [64] que nous présenterons plus loin dans cette dissertation ou pour la méthode KNITRO [99], d'autres méthodes utilisent l'approximation linéaire à la place de l'approximation quadratique c'est le cas, notamment de la méthode LOQO de Vendrebrei [100].

3.6.3 Fonctions de mérite

Lorsque le sous-problème quadratique n'est pas convexe, l'unicité du vecteur (p_k, μ_k) comme solution du système (3.28) est remise en cause et l'existence d'une suite de direction de déplacement $(d_k)_k$ qui pourra garantir une amélioration c'est-à-dire $f(x_k + d_k) < f(x_k)$ ou éventuellement plus de réalisabilité n'est pas assurée non plus, d'où la nécessité d'introduire une fonction qui puisse évaluer cette amélioration.

L'idée de chaque fonction de mérite est de proposer à la fois une façon de mesurer la progression des itérations vers l'optimum et également une manière de peser l'importance des contraintes par rapport à la fonction objectif. En général une fonction de mérite s'écrit sous la forme suivante :

$$M(x) = \phi_1(f(x)) + \phi_2(h(x)) + \phi_3(g(x)) \quad (3.29)$$

où

$$\begin{aligned}\phi_1 &: \mathbb{R} \rightarrow \mathbb{R}, \\ \phi_2 &: \mathbb{R}^p \rightarrow \mathbb{R} \text{ et} \\ \phi_3 &: \mathbb{R}^m \rightarrow \mathbb{R}.\end{aligned}$$

Définition. 3.6.1. *On dira qu'un point x^+ est meilleur qu'un point x au sens de la fonction de mérite si $M(x^+) < M(x)$.*

On dira qu'un vecteur d_x améliore l'optimalité au point x si $M(x+d_x) < M(x)$.

□

Les fonctions ϕ_1 , ϕ_2 et ϕ_3 doivent être prises de façon à ce que pour tout point x^+ meilleur qu'un point x au sens de la fonction de mérite on ait $f(x^+) < f(x)$ ou $|h(x^+)| < |h(x)|$ ou $g^+(x^+) < g^+(x)$ où $g^+(x) = \max(g(x), 0)$. Ainsi une fonction de mérite évidente est

$$M(x, \mu) = f(x) + \frac{1}{\mu} \sum_{i=1}^p |h_i(x)| + \frac{1}{\mu} \sum_{i=1}^m g_i^+(x) \quad (3.30)$$

où μ est un scalaire strictement positive appelé le paramètre de pénalité ou charge de mérite, qui sert à déterminer le poids et donc l'importance à donner aux contraintes au cours des itérations. La fonction (3.30) est dite fonction de mérite l_1 elle peut être améliorée en affectant des charges paramétrées selon les composantes des contraintes,

$$M(x, \sigma_f, \sigma^h, \sigma^g) = \sigma_f f(x) + \sum_{i=1}^p \sigma_i^h |h_i(x)| + \sum_{i=1}^m \sigma_i^g g_i^+(x) \quad (3.31)$$

où $\sigma_f \in \mathbb{R}_+$ joue le rôle de μ dans la fonction (3.30) c'est-à-dire d'équilibrer le poids de l'objectif par rapport aux contraintes. Le paramètre σ^h équilibre intrinsèquement les rapports aux contraintes d'égalité, et σ^g équi-

libre intrinsèquement les contraintes d'inégalité suivant le schéma suivant. Soit $I \subset \{1, 2, \dots, m\}$ l'ensemble des indices pour lesquels $g(x) > 0$.

$$\sigma_i^g = \begin{cases} \frac{g_i(x)}{\Sigma_{i \in I} g_i(x) + \Sigma_{i=1}^p |h_i(x)|} & \text{si } i \in I \\ \sigma_c = \frac{\alpha}{\Sigma_{i \in I} g_i(x) + \Sigma_{i=1}^p |h_i(x)|} & \text{si } i \notin I \end{cases} \quad (3.32)$$

$$\sigma_i^h = \begin{cases} \frac{|h_i(x)|}{\Sigma_{i \in I} (g_i(x) + s_i) + \Sigma_{i=1}^p |h_i(x)|} & \text{si } h_i(x) \neq 0 \\ \sigma_c & \text{si } h_i(x) = 0 \end{cases} \quad (3.33)$$

où

$$\alpha = \min\{\min_{i \in I} g_i(x), \min_{i=1,2,\dots,p} |h_i(x)|\} \geq 0 \quad (3.34)$$

Le point faible de ces deux fonctions de type l_1 est la non différentiabilité par rapport à la variable x . Les fonctions de mérite de type l_2 résout ce problème, en particulier le Lagrangien augmenté de Fletcher :

$$M(x, \mu) = f(x) - \lambda(x)^t h(x) + \frac{1}{2\mu} \sum_{i=1}^p h_i^2(x) \quad (3.35)$$

où $\lambda(x) = [J(x)J^t(x)]^{-1} J(x) \nabla f(x)$ est l'estimation du multiplicateur des moindres carrés.

Cette fonction est généralisable aux problèmes non linéaires en utilisant des variables d'écart (variables auxiliaires).

Une autre fonction de mérite de type l_2 peut être utilisée,

$$M(x, \sigma) = f(x) + \sigma \|c(x)\|_2 \quad (3.36)$$

où σ est un réel strictement positif.

Une alternative aux fonctions de mérite est l'utilisation de filtres, cette technique a été proposée par Fletcher en 1997 [21].

Le filtre permet l'acceptation d'une direction de déplacement si cette dernière présente une réduction concernant l'optimalité ou la violation des contraintes. Nous détaillons le concept du filtre au paragraphe (4.4).

La méthode de filtre nécessite la sauvegarde de couples dominant ce qui la rend moins populaire que les fonctions de mérite qui reste très attrayantes pour les problèmes de grandes taille.

3.6.4 Le cas non convexe

Le principal atout de toute méthode proposée pour résoudre le problème non linéaire est sa capacité à traiter le cas non convexe. En effet nous avons très peu de résultats théoriques pour ce cas, et le peu qu'il y en a est asymptotique [23]. L'utilisation de la méthode des régions de confiance, et des fonctions de mérite permettraient de contourner ce problème, nous reviendrons à ce cas dans notre étude numérique.

CHAPITRE 4

MÉTHODES NUMÉRIQUES

4.1 Introduction

La résolution du problème d'optimisation général (contraintes d'égalités et d'inégalités) est un sujet moderne. Classiquement les chercheurs s'intéressent soit aux problèmes non linéaires avec des contraintes d'égalités soit aux problèmes non linéaires avec des contraintes d'inégalités. Cette classification influe largement les méthodes numériques existantes. Notre intérêt pour les méthodes de points intérieurs nous a pourtant poussé à choisir de présenter ici des méthodes de résolution de problèmes non linéaires avec contraintes d'inégalités, sachant que ses méthodes ont des extensions pour résoudre le problème général.

En général, on commence par identifier les équations à satisfaire (conditions d'optimalité) pour les solutions locales du problème à résoudre, ensuite, en perturbant les conditions d'optimalités retenues, on en déduit un algorithme de résolution. Les algorithmes diffèrent aussi en termes d'outils algébriques ou numériques mis en place pour analyser et résoudre les différents équations, inéquations et problèmes rencontrés.

Dans ce chapitre nous présentons quatre méthodes numériques basées sur quatre idées différentes de développements et de résolutions. Bien que LOQO et KNITRO se basent toutes les deux sur les techniques SQP et points intérieurs, la première utilise la recherche linéaire pour retrouver la direction de descente alors que la seconde opte pour la région de confiance. La troisième méthode, filterSQP, utilise également la technique SQP avec région de confiance ou recherche linéaire mais remplace la fonction de mérite (qui contrôle l'évolution des candidats) par un filtre considéré comme fonction de mérite de dimension

deux. La dernière méthode présentée a été conçue pour résoudre un problème industriel spécifique, et a été généralisée pour répondre à plus de demande, nous essayons de comprendre par cet exemple de logiciel l'intérêt de choisir, quand il le faut, une méthode conçue spécialement.

Nous terminons ce chapitre par l'analyse d'une comparaison numérique due à Morales et cie [59] de ces quatre méthodes leader sur le marché de l'optimisation numérique.

4.2 La méthode LOQO

Le code *LOQO* a été développé par R. J. Vanderbei [84, 100] pour résoudre des problèmes quadratiques par les méthodes de points intérieurs, il a été rapidement généralisé à la programmation non linéaire [85].

Considérons le problème d'optimisation non linéaire suivant :

$$\begin{cases} \min_x f(x) \\ \text{s.c. } h_i(x) \geq 0 \quad i = 1, \dots, m \end{cases} \quad (4.1)$$

où $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \rightarrow \mathbb{R}$ et $h_i : \mathbb{R}^n \rightarrow \mathbb{R} \quad i = 1, \dots, m$ sont des fonctions de classe C^2 .

La méthode de Vanderbei utilise une pénalisation de type points intérieurs afin de retrouver un problème d'optimisation non linéaire d'égalité :

$$\begin{cases} \min_x f(x) - \mu \sum_1^m \ln s_i \\ \text{s.c. } h(x) - s = 0 \end{cases} \quad (4.2)$$

$$\text{où } s = \begin{pmatrix} s_1 \\ \vdots \\ s_m \end{pmatrix} > 0 \text{ et } h(x) = (h_1(x), h_2(x), \dots, h_m(x)).$$

Le Lagrangien du problème (4.2) est donné par la fonction :

$$L(x, s, \lambda, \mu) = f(x) - \mu \sum_1^m \ln s_i - \lambda^t (h(x) - s) \quad (4.3)$$

Ce qui nous permet d'utiliser les techniques (SQP) pour déterminer la direction de déplacement $(\Delta x, \Delta s, \Delta \lambda)$, solution du système linéaire suivant

$$\begin{bmatrix} H & 0 & -A^t \\ 0 & \Lambda & S \\ A & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A^t \lambda \\ \mu e - S \Lambda e \\ -h + s \end{bmatrix} \quad (4.4)$$

où $H := H(x, \lambda) = \nabla^2 f(x) - \sum_{i=1}^m \lambda_i \nabla^2 h_i(x)$

$A := A(x) = \nabla h(x)$

Λ est la matrice diagonale dont les éléments diagonaux sont les λ_i , $i = 1, \dots, m$

et S est la matrice diagonale dont les éléments diagonaux sont les s_i , $i = 1, \dots, m$.

Le système (4.4) peut s'écrire sous une forme symétrique

$$\begin{bmatrix} H & 0 & -A^t \\ 0 & -S^{-1}\Lambda & -I \\ -A & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \sigma \\ -\gamma \\ \rho \end{bmatrix} \quad (4.5)$$

où $\sigma = \nabla f(x) - A^t \lambda$ mesure la réalisabilité duale, $\rho = s - h(x)$ mesure la réalisabilité primale et $\gamma = \mu S^{-1} e - \lambda$.

Grâce à l'élimination de Δs en utilisant la formule $\Delta s = S \Lambda^{-1} (\gamma - \Delta \lambda)$,

Vanderbei réduit (4.5) et résout ainsi le système KKT contracté suivant

$$\begin{bmatrix} H & -A^t \\ -A & -S\Lambda^{-1} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \sigma \\ \rho - S\Lambda^{-1}\gamma \end{bmatrix} \quad (4.6)$$

À chaque itération, le code LOQO résout le système (4.5) par une méthode de recherche linéaire après factorisation LDL^t . À partir d'un point initial (x^0, s^0, λ^0) , on construit une suite (x^k, S^k, λ^k) telle que

$$\begin{aligned} x^{k+1} &= x^k + \alpha^k \Delta x^k \\ s^{k+1} &= s^k + \alpha^k \Delta s^k \\ \lambda^{k+1} &= \lambda^k + \alpha^k \Delta \lambda^k \end{aligned} \quad (4.7)$$

Le pas α^k est calculé de manière à ce que la fonction de mérite utilisée décroît à chaque itération.

La méthode ainsi construite souffre de deux problèmes essentiels :

1. La matrice dans le système (4.5) peut être singulière.
2. La méthode peut converger vers un point stationnaire qui ne soit pas un minimum local.

LOQO utilise la fonction de mérite de type l_2 suivante :

$$M_{\beta, \mu}(x, s) = f(x) - \mu \sum_{i=1}^m \ln s_i + \frac{\beta}{2} \|\rho(x, s)\|_2^2 \quad (4.8)$$

où $\rho(x, S) = S - h(x)$.

Dans le cas où la matrice $N = H + A^t S^{-1} \Lambda A$ est définie positive (cas de la programmation convexe par exemple), on montre qu'il existe $\beta_{min} > 0$ tel que $\forall \beta > \beta_{min}$ la direction de déplacement fournie par une recherche linéaire est une direction de descente pour la fonction de mérite $M_{\beta, \mu}$.

Le paramètre β utilisé dans le code LOQO est initialisé à zéro ($\beta_0 = 0$)

et reste immuable tant que $(\Delta x, \Delta S)$ sera une direction de descente, sinon β prend la valeur $10\beta_{min}$.

Par contre, si la matrice N est indéfinie, une perturbation s'impose, les auteurs [85] suggèrent d'employer la matrice

$$\hat{H} = H + \nu I \quad (4.9)$$

où $\nu \geq 0$ est choisi de telle façon à ce que la matrice N soit définie positive.

Le théorème suivant donne une idée de l'impact d'une telle perturbation sur la progression de la fonction de mérite, la fonction barrière ou sur la réalisabilité.

Théorème. 4.2.1. [85] Soit

$$\bar{x} = x + t\Delta x, \quad \bar{\lambda} = \lambda + t\Delta\lambda, \quad \bar{S} = S + t\Delta S$$

$$\bar{\rho} = \rho(\bar{S}, \bar{x}), \quad \bar{\sigma} = \sigma(\bar{S}, \bar{x})$$

Alors

$$\begin{aligned} \bar{\rho} &= (1 - t)\rho + o(t) \\ \bar{\sigma} &= (1 - t)\sigma - t\lambda\Delta x + o(t) \\ \bar{S}^t \bar{\lambda} &= (1 - t(1 - \delta))S^t \lambda + o(t). \end{aligned} \quad (4.10)$$

avec $\delta = m\mu/S^t\lambda$.

□

Avec des valeurs du paramètre de pénalisation $0 < \mu < \frac{S^t\lambda}{m}$ et $\nu > 0$, on montre que l'algorithme, s'il converge, alors il convergera vers un minimum local.

En pratique, LOQO prend par défaut

$$\mu = \gamma \min\left(\left(1 - r\right)\frac{1 - \zeta}{\zeta}, 2\right)^3 \frac{S^t\lambda}{m} \quad (4.11)$$

où $r = 0.95$, $\gamma = 0.1$ et $\zeta = \frac{\min_i S_i^t \lambda_i}{S^t \lambda / m}$ ces valeurs sont purement empiriques.

Le paramètre ν n'est pas connu d'avance, donc plusieurs factorisations sont nécessaires à la recherche de \hat{H} ce qui réduit la robustesse du logiciel. La méthode est handicapée par le manque d'une théorie pour la convergence globale. L'utilisation de règle dynamique pour la détermination du paramètre μ demeure un obstacle de taille.

4.3 La méthode KNITRO

KNITRO (Non linéaire Interior-point Trust Région Optimizer) est un algorithme de résolution du problème non linéaire général par une méthode de pénalisation-SQP et une résolution des sous-problèmes quadratiques par des méthodes de région de confiance : dogleg, gradient conjugué version Steihaug. La méthode KNITRO a été proposée par Byrd, Gilbert et Nocedal en 1996 [8]. Dans cet article, les auteurs posent les bases théoriques de la méthode pour résoudre un problème d'optimisation non linéaire avec des contraintes d'inégalités, même si les hypothèses du théorème de convergence sont fortes [8], les résultats numériques [9, 10] confortent les avis de Byrd et cie. que la méthode est robuste. Le papier [9] présente les détails de l'implémentation de l'algorithme KNITRO.

Soit le problème suivant :

$$\begin{cases} \min_x f(x) \\ \text{s.c. } h(x) = 0 \\ g(x) \leq 0 \end{cases} \quad (4.12)$$

où $x \in \mathbb{R}^n$, les fonctions $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$ et $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ sont de classe C^2 .

La méthode pénalise les contraintes d'inégalités en définissant une variable

auxiliaire $s \in \mathbb{R}_+^m$ et un paramètre de pénalisation $\mu \geq 0$. Ainsi, on construit une suite de problèmes pénalisés comme suit :

$$\begin{cases} \min_x f(x) - \mu \sum_{i=1}^m s_i \\ \text{s.c. } c(x, s) = 0 \end{cases} \quad (4.13)$$

où

$$c(x, s) = \begin{pmatrix} h(x) \\ g(x) + s \end{pmatrix} \quad (4.14)$$

La méthode de Newton appliquée pour résoudre les conditions d'optimalité de premier ordre du problème (4.13) permet d'écrire le système linéaire symétrique et perturbé suivant :

$$\begin{pmatrix} \nabla_{xx}^2 L_\mu & 0 & A_h & A_g \\ 0 & \Sigma & 0 & I \\ A_h^t & 0 & 0 & 0 \\ A_g^t & I & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x \\ d_s \\ \lambda_h^+ \\ \lambda_g^+ \end{pmatrix} = - \begin{pmatrix} \nabla_x f \\ -\mu S^{-1} e \\ h(x) \\ g(x) + s \end{pmatrix} \quad (4.15)$$

où le Lagrangien du problème (4.13) au point courant $(x, s, \lambda_h, \lambda_g)$ est donné par l'expression suivante :

$$L_\mu(x, s, \lambda_h, \lambda_g) = f(x) - \mu \sum_{i=1}^m \ln s_i + \lambda_h^t h(x) + \lambda_g^t (g(x) + s) \quad (4.16)$$

avec $\lambda_h \in \mathbb{R}^p, \lambda_g \in \mathbb{R}^m$ sont respectivement les multiplicateurs de Lagrange d'égalités et d'inégalités à l'instant courant. Les valeurs des multiplicateurs à l'instant suivant sont notés λ_h^+ et λ_g^+ . A_h et A_g sont respectivement les matrices jacobiennes des contraintes d'égalités et d'inégalités, $S = \text{diag}(s_i) \quad i = 1, \dots, m$, $\Sigma = \mu S^{-2}$ pour la version primale de la méthode ou $\Sigma = S^{-1} \Lambda_g$ pour

la version primale-duale de la méthode, où $\Lambda_g = \text{diag}(\lambda_g^i) \quad i = 1, \dots, m$. La méthode SQP montre que la direction de déplacement est le vecteur $d = \begin{pmatrix} d_x \\ d_s \end{pmatrix}$ qui n'est autre que la solution du sous-problème quadratique (4.17). Cette solution est unique si le problème quadratique est convexe. Les auteurs ont choisi de résoudre les sous-problèmes quadratiques par une méthode de région de confiance, ainsi à chaque itération KNITRO résout le problème quadratique suivant :

$$\left\{ \begin{array}{ll} \min_{\bar{d}} \frac{1}{2} d^t W d + \nabla \phi^t d & \\ \text{s.c :} & \\ Ad + c = 0 & (a) \\ \|d\| \leq r & (b) \\ \text{et } d_s > \tau s & (c) \end{array} \right. \quad (4.17)$$

où

$$W = \begin{pmatrix} \nabla_{xx}^2 L & 0 \\ 0 & \mu S^{-2} \end{pmatrix} \quad (4.18)$$

$$A = \begin{pmatrix} A_h & A_g \\ 0 & S \end{pmatrix} \quad (4.19)$$

et

$$\nabla\phi = \begin{pmatrix} \nabla_x f \\ -\mu S^{-1}e \end{pmatrix} \quad (4.20)$$

La dernière contrainte $d_s \geq -\tau s$, $\tau > 0$ assure la positivité de la variable auxiliaire s au cours des itérations.

L'estimation du multiplicateur de Lagrange

$$\lambda = \begin{pmatrix} \lambda_h \\ \lambda_g \end{pmatrix} \quad (4.21)$$

est le résultat du programme suivant :

$$\min_{\lambda \in \mathbb{R}^{p+m}} \|A^t \lambda + \nabla\phi\| \quad (4.22)$$

Cette estimation n'assure pas la positivité des multiplicateurs λ_g ce qui provoque la non convexité du modèle quadratique si on considère le modèle primal-dual $\Sigma = S^{-1}\Lambda_g$. Byrd, Hribar et Nocedal [9] proposent d'introduire une heuristique dans le code KNITRO afin de conserver la convexité du modèle au cours des itérations :

$$\Sigma_{ii} = \begin{cases} \lambda_g^i / s_i & \text{si } \lambda_g^i > 0 \\ \mu / s_i^2 & \text{sinon} \end{cases} \quad (4.23)$$

donc on combine les deux approches primale et primale-duale. Les auteurs affirment que si aucune preuve n'est donnée pour la convergence de la méthode introduisant cette combinaison, les résultats numériques confirment la justesse de l'idée.

Chaque pas de déplacement d est la somme d'un pas vertical v et d'un pas

horizontal w . Le pas vertical est la solution d'un problème quadratique convexe avec comme seule contrainte la région de confiance. Le rôle de cette résolution est d'améliorer la réalisabilité et d'empêcher un blocage de la méthode si jamais les contraintes ne sont pas compatibles. Le pas horizontal est la solution d'un problème quadratique (généralement non convexe) avec région de confiance. Le problème horizontal est généralement réduit grâce à une factorisation QR de la matrice jacobienne des contraintes, ainsi la résolution horizontale se réduit à la résolution d'un problème quadratique dans \mathbb{R}^{n-p} et d'une factorisation QR . La direction résultante candidate $d = v + Z^t u$ où Z est la base de l'espace noyau $\ker(A^t)$ et u est la solution du problème horizontal réduit. Cette direction est acceptée si elle permet une réduction suffisante de la fonction de mérite

$$M(x, s, \mu, \nu) = f(x) - \mu \sum_{i=1}^m s_i + \nu \|g(x) + s\|_2, \quad \nu > 0 \quad (4.24)$$

Malheureusement aucun résultat théorique ne permet de prévoir si la direction candidate d permet la réduction souhaitée, et des rétrécissements de la région de confiance et plusieurs résolutions des problèmes verticaux et horizontaux sont nécessaires. Ce désagrément se désintègre pour de petite région de confiance, en effet le lemme (4.6) du [8] montre qu'il existe un rayon r_0 assez petit tel que pour tout $r \leq r_0$ la solution d permet la réduction souhaitée de la fonction de mérite. Les résultats de convergence sont démontrés pour le cas de problèmes sans contraintes d'égalité et pour des contraintes d'inégalité non saturées i.e. s_i suffisamment grand pour tout $i = 1, \dots, m$. La diagonale de la matrice jacobienne des contraintes A est perturbée par rajout des éléments s_i pour $i = n + 1, \dots, n + m$, cette perturbation est basée sur l'observation numérique que l'utilisation de cette matrice diminue l'impact qu'aurait la perturbation des conditions d'optimalités de premier ordre. La méthode subit également les conséquences de l'effet de Maratos [55], les auteurs proposent

l'utilisation d'une procédure de correction de second ordre, cette heuristique est très efficace.

4.4 La méthode filterSQP

La méthode de filtre définit une nouvelle relation d'ordre entre les itérés produits par un algorithme SQP avec région de confiance. Le nouveau concept se présente comme une fonction 2-d. Un filtre permet l'acceptation du pas de déplacement si ce dernier présente une réduction concernant l'optimalité ou la violation des contraintes. Fletcher et Leyffer ont été encouragés par les résultats numériques [21] favorisant ainsi l'apparition d'une nouvelle branche de recherche [6, 20, 83].

La technique a été introduite [21] initialement pour résoudre le problème d'optimisation non linéaire

$$\begin{cases} \min_x f(x) \\ \text{s.c. } c(x) \leq 0 \end{cases} \quad (4.25)$$

puis a été généralisée par Fletcher, Leyffer et Toint [22] pour résoudre le problème non linéaire général

$$\begin{cases} \min_x f(x) \\ \text{s.c. } c_i(x) = 0 \quad i \in \mathcal{E} \\ \quad c_i(x) \leq 0 \quad i \in \mathcal{I} \end{cases} \quad (4.26)$$

où $\mathcal{E} \cup \mathcal{I} = \{1, \dots, m\}$, $x \in \mathbb{R}^n$ et f et $c_i \quad i = 1, \dots, m$ sont des fonctions de classe C^2 .

La réduction actuelle mesure l'optimalité, est définie par $\Delta f = f(x) - f(x + d)$. La violation des contraintes est mesurée par la fonction h , définie par

$$h(c) = \|c_{\mathcal{I}}^+\|_1 + \|c_{\mathcal{I}}\|_1 \quad (4.27)$$

où $c_i^+ = \max(0, c_i)$ $i \in \mathcal{I}$ et c est le vecteur composé par l'évolution des contraintes au point courant x . S'il s'agit d'une itération k , on notera par x^k le point courant ainsi $h^{(k)} = h(c(x^{(k)}))$ et $f^{(k)} = f(x^{(k)})$.

Au lieu de combiner $h^{(k)}$ et $f^{(k)}$ pour mesurer l'amélioration apportée par une direction de déplacement, les méthodes de filtre proposent d'utiliser une mesure bi-dimensionnelle, comme l'explique les deux définitions suivantes :

Définition. 4.4.1. *On dira qu'un couple $(f^{(k)}, h^{(k)})$ domine un autre couple $(f^{(l)}, h^{(l)})$ si et seulement si $f^{(k)} \leq f^{(l)}$ et $h^{(k)} \leq h^{(l)}$.*

□

Définition. 4.4.2. *Un filtre est un ensemble de couples $(f^{(l)}, h^{(l)})$ tel que aucun couple ne domine l'autre.*

Un couple est dit acceptable pour introduction dans le filtre s'il n'est pas dominé par aucun élément du filtre.

□

A chaque itération la méthode vérifie si le minimum du sous problème quadratique (le candidat) est dominé par un élément du filtre, si c'est le cas le candidat est rejeté, si non il est introduit dans le filtre, ensuite on vérifie si le nouvel entrant domine un élément du filtre, ce dernier fait, dans ce cas, une sortie du filtre. Comme dans toute méthode SQP, la direction du déplacement

$d^{(k)}$ est la solution du problème quadratique.

$$\left\{ \begin{array}{l} \min_x \frac{1}{2} d^t W^{(k)} d + d^t g^{(k)} \\ \text{s.c.} \\ A^{(k)} d + c^{(k)} \leq 0 \\ \|d\|_\infty \leq r \end{array} \right. \quad (4.28)$$

où $W^{(k)}$ et $g^{(k)}$ sont respectivement le Hessien et le gradient du Lagrangien et $A^{(k)}$ est la matrice jacobienne des contraintes, r est le rayon de la région de confiance.

Le problème (4.28) peut être non réalisable c'est-à-dire aucun vecteur d ne vérifie à la fois la contrainte $A^{(k)t}d + c^{(k)} \leq 0$ et la contrainte de la région de confiance $\|d\|_\infty \leq r$, dans ce cas Fletcher et Leyffer [21] propose de rajouter une procédure appelée phase de restauration permettant ainsi la perturbation des contraintes $A^{(k)t}d + c^{(k)} \leq 0$ afin d'avoir un point de contact c'est-à-dire chercher ε tel que l'ensemble $\{d : A^{(k)t}d + c^{(k)} \leq \varepsilon, \|d\|_\infty \leq r\}$ soit non vide. Ainsi l'algorithme de la méthode est donné par le schéma de base suivant :

Algorithme filtreSQP

Soit x^0 et $r > 0$

$k = 0$

Tant que (pas de convergence) **faire**

Si (le problème (QP) n'est pas réalisable) **Alors**

 | Trouver un point $x^{(k+1)}$ par la phase de restauration

Sinon

 | Résoudre (QP) pour avoir $d^{(k)}$

 | Prendre provisoirement $x^{(k+1)} = x^{(k)} + d^{(k)}$

Fin Si

Si (Si $(f^{(k+1)}, h^{(k+1)})$ est acceptable pour le filtre) **Alors**

 | Accepter $x^{(k+1)}$ et ajouter $(f^{(k+1)}, h^{(k+1)})$ au filtre

 | Éventuellement élargir r

Sinon

 | Rejeter la direction $d^{(k)}$

 | Réduire r

Fin Si

$k = k + 1$

Fait

Algorithme 11: Algorithme filtreSQP

Si une courbure négative est détectée, le solveur QP (solveur de programme quadratique) propose de la suivre jusqu'à ce qu'une frontière d'une contrainte ou de la région de confiance soit rencontrée. Des résultats de la convergence ont été établis pour la méthode, néanmoins ça reste un vaste champ de recherche. Les résultats numériques sont l'atout principal de la méthode de filtre mais l'utilisation de techniques sophistiquées notamment de la mémoire pour le passage d'une itération à une autre reste un handicap majeur, en effet tant

qu'un couple n'est pas dominé il reste occupant de l'espace mémoire.

4.5 La méthode SNOPT

SNOPT (*Sparse Nonlinear Optimizer*) est un solveur de problèmes non linéaires. Le code de Gill, Murray et Saunders [28, 102] a été conçu pour résoudre des problèmes de grandes tailles avec des degrés de liberté, c'est-à-dire quand le nombre de contraintes actives à l'optimum est limité. C'est un cas typique du calcul de la trajectoire optimale dans l'industrie aérospatiale [39]. La méthode SNOPT exploite la forme creuse de la matrice jacobienne des contraintes et garde l'approximation quasi-newtonienne BFGS de la matrice H_k .

Soit Le problème non linéaire suivant :

$$\left\{ \begin{array}{l} \min_{x \in \mathbb{R}^n} f(x) \\ \text{s.c.} \quad l \leq \begin{pmatrix} x \\ c(x) \\ Ax \end{pmatrix} \leq u \end{array} \right. \quad (4.29)$$

La méthode commence par évaluer les contraintes linéaires en résolvant le problème linéaire suivant dit problème élastique :

$$\left\{ \begin{array}{l} \min_{x,v,w} e^t(v+w) \\ \text{s.c.} \quad l \leq \begin{pmatrix} x \\ Ax - v + w \end{pmatrix} \leq u, v \geq 0, w \geq 0 \end{array} \right. \quad (4.30)$$

Si le problème (4.30) fournit des solutions $v \neq 0$ ou $w \neq 0$, c'est-à-dire que les contraintes linéaires du problème (4.29) ne sont pas réalisables, dans ce cas SNOPT s'arrête sans évaluer les fonctions non linéaires. Dans le cas où les contraintes non linéaires ne sont pas réalisables, la méthode résout le problème

non linéaire suivant, dit problème non linéaire élastique, par des itérations SQP

$$\begin{cases} \min_{x,v,w} f(x) + \gamma e^t(v+w) \\ \text{s.c. } l \leq \begin{pmatrix} x \\ c(x) - v + w \\ Ax \end{pmatrix} \leq u, \quad v \geq 0, \quad w \geq 0 \end{cases} \quad (4.31)$$

où $\gamma \geq 0$ est un paramètre de pénalité.

À chaque itération, la méthode résout un sous problème quadratique où l'objectif est l'approximation quadratique du Lagrangien modifié.

Soit le problème non linéaire suivant

$$\begin{cases} \min_x f(x) \\ \text{s.c. } c(x) \geq 0 \end{cases} \quad (4.32)$$

où $x \in \mathbb{R}^n$, $c \in \mathbb{R}^m$ et f et c_i sont des fonctions de classe C^2 .

Le Lagrangien modifié du problème (4.32) est donné par la formule :

$$\mathcal{L}(x, x_k, \pi_k) = f(x) - \pi_k^t d_L(x, x_k) \quad (4.33)$$

où x_k est la variable primale à l'itération k (itération en cours).

π_k est la variable duale à l'itération k .

$$d_L(x, x_k) = c(x) - c_L(x, x_k)$$

$$c_L(x, x_k) = c(x_k) + J_k(x - x_k)$$

et J_k est la matrice jacobienne des gradients évaluée au point x_k .

Soit H_k une approximation quasi-newtonienne du Hessien $\nabla^2 \mathcal{L}$ au point (x_k, π_k) . La résolution du sous-problème quadratique suivant :

$$\begin{cases} \min_x f_k + \nabla f_k^t(x - x_k) + \frac{1}{2}(x - x_k)^t H_k(x - x_k) \\ \text{s.c. } c_k + J_k(x - x_k) \geq 0 \end{cases} \quad (4.34)$$

permet de donner une solution optimale $(\hat{x}_k, \hat{\pi}_k)$, ainsi l'itération suivante $(x_{k+1}, \pi_{k+1}, s_{k+1})$ est donnée par une recherche linéaire :

$$\begin{pmatrix} x_{k+1} \\ \pi_{k+1} \\ s_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ \pi_k \\ s_k \end{pmatrix} + \alpha_k \begin{pmatrix} \hat{x}_k - x_k \\ \hat{\pi}_k - \pi_k \\ \hat{s}_k - s_k \end{pmatrix} \quad (4.35)$$

$0 < \alpha_k \leq 1$ permet une réduction suffisante de la fonction de mérite :

$$M(x, \pi, s) = f(x) - \pi^t(c(x) - s) + \frac{1}{2}(c(x) - s)^t D(c(x) - s) \quad (4.36)$$

où D est la matrice diagonale dont les éléments diagonaux sont les paramètres de pénalité.

L'algorithme maintient la définie positivité du Hessien. Soit $\delta_k = x_{k+1} - x_k$ et $y_k = \nabla \mathcal{L}(x_{k+1}, x_k, \pi) - \nabla \mathcal{L}(x_k, x_k, \pi)$. L'estimation de la matrice H_{k+1} est donnée par :

$$H_{k+1} = H_k + \theta y_k y_k^t - \Phi_k q_k q_k^t \quad (4.37)$$

où $q_k = H_k \delta_k$, $\theta = 1/y_k^t \delta_k$ et $\Phi_k = 1/q_k^t \delta_k$.

Ainsi, si H_k est définie positive, alors la matrice H_{k+1} est définie positive si et seulement si l'approximation de la courbure $y_k^t \delta_k$ est suffisamment positive, i.e. il existe un scalaire positif σ telle que $y_k^t \delta_k > \sigma$. En revanche si ce n'est pas le cas alors SNOPT effectue des modifications en redéfinissant y_k et δ_k à partir de la solution \bar{x} du problème (4.34). $\delta_k = x_{k+1} - \bar{x}$ et

$$y_k = \nabla \mathcal{L}(x_{k+1}, x_k, \pi_{k+1}) - \nabla \mathcal{L}(\bar{x}, x_k, \pi_{k+1}).$$

$y_k^t \delta_k$ approche ainsi la courbure du Hessien réduit qui doit être semi-défini positif au voisinage du minimum local du problème (4.32). Si le point (x_k, π_k) n'est pas suffisamment proche du minimum local alors une deuxième modification s'impose : $y_k := y_k + \Delta y_k$, où $\Delta y_k = (J_{k+1} - J_k)^t \Omega d_L(x_{k+1}, x_k)$ et $\Omega = \text{diag}(\omega_i)$, $\omega_i \geq 0$, $i = 1, \dots, m$. Cette modification a été proposée par Powell [72], ω est solution du problème de moindre carrée ordinaire suivant :

$$\begin{cases} \min_{\omega} \|\omega\|^2 \\ \text{s.c. } a^t \omega = \beta, \quad \omega \geq 0 \end{cases} \quad (4.38)$$

où $\beta = \sigma_k - y_k^t \delta_k$ et $a_i = v_i w_i$ $i = 1, \dots, m$ avec $v = (J_{k+1} - J_k) \delta_k$ et $w = d_L(x_{k+1}, x_k)$.

Si le problème (4.38) n'admet pas de solution alors la méthode n'effectue aucune modification. Les sous-problèmes sont résolus par l'algorithme SQOPT [29] qui est un algorithme d'ensemble actif prenant en charge la résolution des problèmes élastiques si cela s'avère nécessaire.

La convergence de la méthode est atteinte si un itéré (x, π) satisfait les conditions d'optimalité de premier ordre. Le code n'a pas besoin des dérivées secondes des fonctions à chaque itération mais les nouvelles versions permettent leur utilisation si elles existent. La convergence globale est difficile à établir pour SNOPT, car la fonction de mérite (4.36) utilisée n'est pas nécessairement minimisée par une solution du problème non linéaire.

4.6 Évaluation numérique

Nous avons préféré de présenter au début de ce chapitre, la partie théorique de chaque méthode afin de donner une vision de l'utilité de chaque code puisqu'une étude numérique ne peut être que indicative, et ne donnera point un

classement des méthodes étudiées, en effet les algorithmes évoluent en permanence, à l'heure de la rédaction de cette dissertation il s'agit de la version 3.1 de KNITRO, de la version 6.06 de LOQO, de la version 6.0 de SNOPT et de la version 2.0 de filtreSQP. Une étude numérique se fait en testant les méthodes en question pour résoudre un ensemble de problèmes qu'on appelle problèmes tests. Plusieurs bibliothèques numériques proposent des problèmes tests modélisés selon des formats portables (voir pour ceci la modélisation AMPL [25, 95] et la bibliothèque CUTEr [96]), ces problèmes tests sont généralement divisés en trois groupes :

1. Académiques issus essentiellement de problèmes mathématiques ou physiques ;
2. Réels issus de l'industrie ;
3. De modélisation qui sont des problèmes fabriqués sur mesure pour évaluer telle ou telle caractéristique d'un tel code.

Ainsi une comparaison numérique ne peut donner d'affirmation qu'au sujet de l'ensemble sur lequel l'étude est faite ou sur la classe de problèmes que concerne la comparaison. Ces classes sont l'ensemble des problèmes sans contraintes, l'ensemble des problèmes avec des contraintes d'égalités, l'ensemble des problèmes non linéaire convexes, les problèmes non linéaires généraux, auxquels nous pourrions ajouter ou retirer l'hypothèse de la complémentarité stricte. Le choix de programmeur affecte la comparaison, en effet chaque code a ses propres spécificités d'implémentation : stratégies pour rechercher le pas de déplacement, traitement de la forme creuse des matrices, tests d'arrêts et choix algorithmiques concernant les fonctions de mérite.

Nous nous intéressons à la résolution de problème non linéaire général suivant,

$$\begin{cases} \min_x f(x) \\ \text{s.c. } h(x) = 0 \\ g(x) \leq 0 \end{cases} \quad (4.39)$$

où $x \in \mathbb{R}^n$ et les fonctions $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$ et $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ sont de classe C^2 .

Le code LOQO est une méthode de pénalisation avec recherche linéaire, alors que KNITRO est un algorithme de région de confiance, les techniques d'algèbre linéaire utilisés pour le calcul de pas de déplacement sont significativement différentes entre les deux méthodes. SNOPT et filterSQP sont deux méthodes d'ensemble actif. La première est une méthode à recherche linéaire utilisant des modèles convexes, la deuxième est à région de confiance qui accepte les modèles non convexes et qui utilise la notion de filtre au lieu des fonctions de mérite.

4.7 Étude antérieure

Une étude numérique a été menée par Morales et cie [59] pour comparer ces quatre méthodes. Pour les raisons que nous avons cités ci-dessus aucune comparaison numérique ne peut permettre de donner un classement de ces méthodes mais on cherche plutôt à détecter les familles de problèmes pour lesquelles une méthode pourrait être plus adaptée qu'une autre. Morales et cie classent l'ensemble de problèmes tests en trois familles :

1. Les problèmes sans contraintes.
2. Les problèmes avec contraintes d'égalités.
3. Les problèmes non linéaires généraux.

Mais à cause de leur importance théorique, la convexité et la complémentarité stricte peuvent faire l'objet d'études spécifiées ainsi notre comparaison peut inclure la famille des problèmes convexes et la famille des problèmes pour lesquels l'hypothèse de la complémentarité stricte est toujours vérifiée ou n'est jamais vérifiée, cette dernière partition n'est pas évidente en pratique.

Le paramètre de performance proposé par Dolan et Moré [17] permet l'analyse des sorties (résultats des minimisations).

Soit $t_{p,s}$ le temps nécessaire pour résoudre un problème p par un optimiseur s . on définit le ratio

$$r_{p,s} = \frac{t_{p,s}}{t_p^*} \quad (4.40)$$

où t_p^* est le temps minimal nécessaire permettant la résolution du problème p . Si aucun optimiseur n'arrive à résoudre le problème p alors on affecte un grand nombre M au ratio $r_{p,s}$. On définit la performance logarithmique pour chaque code par le paramètre π_s donné par la relation suivante :

$$\pi_s(\tau) = \frac{\text{Nombre de problèmes t.q. } \log_2(r_{p,s}) \leq \tau}{\text{Nombre total des problèmes}}, \quad \tau \geq 0 \quad (4.41)$$

Ce paramètre nous permet de comparer le taux de problèmes résolus par chaque solveur en temps limité, tout en donnant de moins en moins d'importance à la comparaison avec de grandes valeurs de seuil temps d'où l'introduction de la fonction logarithmique.

Dans cette étude, on considère également le nombre d'évaluations des fonctions. Pour la classe des problèmes sans contraintes LOQO est plus performant que KNITRO, en effet les itérations G.C. affectent le temps de calcul. Ce phénomène est observé pour les problèmes mal-conditionnés. SNOPT est plus lent, en fait, il fait beaucoup d'évaluations de la fonction objectif. filtreSQP est assez

similaire à KNITRO en terme d'évaluations des fonctions.

En l'absence des contraintes d'inégalités, les quatre algorithmes appliquent une SQP pour résoudre le problème (4.42) suivant,

$$\begin{cases} \min_x f(x) \\ \text{s.c. } h(x) = 0 \end{cases} \quad (4.42)$$

Néanmoins il demeure des différences algorithmiques importantes entre les méthodes étudiées. LOQO utilise une recherche linéaire alors que KNITRO est un code de région de confiance, SNOPT recherche la direction de déplacement par recherche linéaire en résolvant un système linéaire et en remplaçant $\nabla^2 L$ par une matrice définie positive, et filterSQP résout un sous problème quadratique avec région de confiance sauf que le filtre remplace la fonction de mérite. L'étude munie par Morales et cie [59] montre la suprématie de KNITRO en terme du temps de calcul mais aussi en terme d'évaluations des fonctions.

Les différences algorithmiques sont beaucoup plus nettes pour la résolution du problème général (4.39). Pour des périodes temporelles limitées (par exemple $\tau = 6$) nous remarquons que SNOPT est le plus efficace, il est de loin le code le plus rapide et il ne consomme relativement pas beaucoup d'évaluations de fonctions. KNITRO devient plus performant si on dispose de plus temps nécessaire à la résolution de nos problèmes, en revanche si on se préoccupe d'avantage de l'évaluation des fonction alors filterSQP est mieux adapté. Ainsi les deux méthodes de points intérieurs LOQO et KNITRO semblent être mieux adaptés aux problèmes de petites et moyennes tailles sans contraintes d'inégalités, et aux problèmes généraux de grandes tailles.

CHAPITRE 5

LA MÉTHODE SDC

Les bons résultats de la programmation D.C. présentés au chapitre (1) concernant la résolution d'un problème quadratique, avec contraintes quadratiques ou sans contraintes, nous ont encouragé à utiliser ces techniques dans la programmation non linéaire. La méthode proposée par Byrd et son équipe [8, 9] est basée sur la résolution séquentielle de problèmes quadratiques avec seule contrainte la région de confiance imposée au pas de déplacement. Cette stratégie nous permet de comparer le potentiel de la méthode D.C. au coeur d'un programme d'optimisation non linéaire.

Des problèmes de convergence apparaissent lorsque le point courant s'approche de la frontière de l'une des contraintes d'inégalités. En effet le théorème (3.4.2) montre que la suite des multiplicateurs de Lagrange $(\lambda_k)_k$ dont les composantes sont $-\mu_k/g_i(x_k)$ est bornée, ainsi les composantes du Hessien $-\mu/s_i^2$ pour la version primale ou $(\lambda_g)_i/s_i$ dans la version primale-duale tendent vers l'infini et le sous-problème quadratique se retrouve dominé par les composantes correspondantes, dans ce cas, l'algorithme, dans son processus de minimisation s'attaquera naturellement à la minimisation de

$$\frac{1}{2}d_s^t \Sigma d_s - \mu S^{-1} d_s$$

où Σ est une matrice diagonale, dont les termes diagonaux sont

$$\Sigma_{ii} = \begin{cases} \mu/s_i^2 & \text{dans le cas primal} \\ (\lambda_g)_i/s_i & \text{dans le cas primal-dual} \end{cases}$$

Quand une composante s_i est proche de zéro et surtout dans le cas où la région de confiance est petite, le modèle quadratique $\frac{1}{2}d_s^t \Sigma d_s - \mu S^{-1}d_s$ se retrouve à son tour dominé par $-\mu S^{-1}d_s$, ainsi la minimisation d'un sous-problème quadratique sera équivalente à la maximisation de $S^{-1}d_s$ ce qui génère des composantes de S^{-1} très grandes et le processus que nous décrivons se répète empêchant toute convergence (voir section 5.8 ci-dessous).

Ceci nous a conduit à revoir l'écriture des conditions d'optimalité, en effet nous avons effectué un changement de variable sur le pas de déplacement de la variable auxiliaire s . Ainsi la fonction objectif du sous-problème quadratique subit un re-dimensionnement des m dernières composantes de sa matrice et de son vecteur. La fonction objectif est ainsi plus stable numériquement. En effet on remplace dans le modèle de Byrd, les composantes $(\lambda_g/s)_i$ ou μ/s_i^2 , qui exploseraient si une composante s_i tend vers 0, par μ dans la version primale et par $(\lambda_g \cdot s)_i$ dans le cas de la version primale-duale et de remplacer les composante $-\mu/s_i \quad i = 1, \dots, m$ dans le vecteur $\nabla\phi$ par $-\mu$. Dans notre stratégie, l'importance est donnée (surtout à la fin de convergence) à la partie (n, n) du Hessien $\nabla_{xx}^2 L$ c'est-à-dire à la partie qui caractérise la fonction objectif du problème non linéaire initial. La réalisabilité demeure contrôlée à chaque itération par la fonction de mérite. Finalement quand μ est suffisamment proche de zéro (10^{-12} à 10^{-18}) on remplace le modèle quadratique de dimension $n + m$ par un modèle quadratique de dimension n dépendant uniquement de la variable x . Les tests numériques que nous décrivons à la fin de ce chapitre confirment notre conclusion et nous avons remarqué que la méthode continue à converger vers la solution(i.e. la précision est plus grande).

Les sous problèmes sont résolus par la méthode D.C. En plus des bons résultats numériques, la méthode nous a permis d'avoir à chaque itération une orthogonalité parfaite puisque la solution de la programmation horizontale

appartient au noyau $\text{Ker}(J)$ alors qu'à chaque itération, la solution verticale appartient à l'ensemble $\text{Im}(J^t)$ dès que le point initial de la résolution verticale est un élément dans l'ensemble $\text{Im}(J^t)$.

5.1 Les conditions d'optimalité

Soit le problème non linéaire (5.1) suivant

$$\left\{ \begin{array}{l} \min f(x) \\ \text{s.c.} \\ h(x) = 0 \\ g(x) \leq 0 \\ x \in \mathbb{R}^n \end{array} \right. \quad (5.1)$$

où f est une fonction numérique de classe C^2 , et les fonctions de contraintes g et h sont définies de \mathbb{R}^n dans \mathbb{R}^m et \mathbb{R}^p respectivement, g et h sont aussi supposées de classe C^2 .

La résolution du problème (5.1) passe par une pénalisation intérieure. On définit ainsi, une suite de problèmes pénalisés (5.2) avec des contraintes d'égalités.

$$\left\{ \begin{array}{l} \min f(x) - \mu \sum_{i=1}^m \ln(s_i) \\ \text{s.c.} \\ h(x) = 0 \\ g(x) + s = 0 \\ x \in \mathbb{R}^n, s \in \mathbb{R}_+^m \end{array} \right. \quad (5.2)$$

La solution de chaque problème (5.2) est notée x_μ , ce qui forme une famille convergente vers une solution x^* quand μ tend vers zéro. L'étude de

la convergence (voir chapitre 3) montre que sous certaines conditions, la suite $(x_\mu)_\mu$ existe et converge vers un minimum local du problème non linéaire initial (c'est bel et bien une suite indexée par un entier i tel que μ_i est la solution optimale du i ème problème pénalisé, pour simplifier on note $x_{\mu_i} = x_\mu$, i ou μ_i étant fixé).

Le Lagrangien associé au problème (5.2) au point courant $(x, s, \lambda_h, \lambda_g)$ est donné par l'expression suivante :

$$L_\mu(x, s, \lambda_h, \lambda_g) = f(x) - \mu \sum_{i=1}^m \ln s_i + \lambda_h^t h(x) + \lambda_g^t (g(x) + s) \quad (5.3)$$

Les vecteurs $\lambda_h \in \mathbb{R}^p, \lambda_g \in \mathbb{R}^m$ sont respectivement les multiplicateurs de Lagrange d'égalité et d'inégalité à l'instant courant.

Les conditions d'optimalité de premier ordre sont données par le système suivant

$$\left\{ \begin{array}{l} \nabla_x L_\mu(x, s, \lambda_h, \lambda_g) = 0 \\ \nabla_s L_\mu(x, s, \lambda_h, \lambda_g) = 0 \\ h(x) = 0 \\ g(x) + s = 0 \\ s \in \mathbb{R}_+^m \end{array} \right. \quad (5.4)$$

d'où le système (5.5) après multiplication de la deuxième équation par S

$$\left\{ \begin{array}{l} \nabla_x f(x) + J_h^t \lambda_h + J_g^t \lambda_g = 0 \\ \lambda_g S e = \mu e \\ h(x) = 0 \\ g(x) + s = 0 \\ s \in \mathbb{R}_+^m \end{array} \right. \quad (5.5)$$

où $S = \text{diag}(s_i) \quad i = 1, \dots, m$ et e est le vecteur unité de \mathbb{R}^m .

La méthode de Newton appliquée à la résolution du système non linéaire (5.5) permet d'écrire le système linéaire suivant de dimension $(n + 2m + p)$.

$$\begin{pmatrix} \nabla_{xx}^2 L_\mu & 0 & J_h^t & J_g^t \\ 0 & \Lambda_g & 0 & S \\ J_h & 0 & 0 & 0 \\ J_g & I & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x \\ d_s \\ \lambda_h^+ \\ \lambda_g^+ \end{pmatrix} = - \begin{pmatrix} \nabla_x f \\ -\mu e \\ h(x) \\ g(x) + s \end{pmatrix} \quad (5.6)$$

où $\Lambda_g = \text{diag}(\lambda_g)$.

Les multiplicateurs à l'instant suivant sont notés λ_h^+ et λ_g^+ . J_h et J_g sont respectivement les matrices jacobiennes des contraintes d'égalité et d'inégalité. Les vecteurs d_x et d_s désignent respectivement le pas de déplacement de la variable x et de la variable s . On remarque que ce système ne traite pas la positivité de s , en fait nous ajouterons une procédure que nous exposerons au paragraphe (5.5) qui se chargera de garder $s + d_s > 0$ à chaque itération.

Afin de pouvoir utiliser les techniques SQP (consulter [19] et notre paragraphe (3.6.2)) nous avons besoin que la matrice du système (5.6) soit symétrique ainsi nous introduisons le changement de variable suivant

$$d_s = S\tilde{d}_s$$

Ceci affectera la deuxième et la quatrième équation. Finalement nous obtenons le système linéaire suivant

$$\begin{pmatrix} \nabla_{xx}^2 L_\mu & 0 & J_h^t & J_g^t \\ 0 & \Lambda_g S & 0 & S \\ J_h & 0 & 0 & 0 \\ J_g & S & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x \\ \tilde{d}_s \\ \lambda_h^+ \\ \lambda_g^+ \end{pmatrix} = - \begin{pmatrix} \nabla_x f \\ -\mu e \\ h(x) \\ g(x) + s \end{pmatrix} \quad (5.7)$$

Dans une écriture plus synthétique on trouve,

$$\begin{pmatrix} W & J^t \\ J & 0 \end{pmatrix} \begin{pmatrix} \tilde{d} \\ \lambda^+ \end{pmatrix} = - \begin{pmatrix} \nabla_x \phi(x, s) \\ c(x, s) \end{pmatrix} \quad (5.8)$$

avec

$$W = \begin{pmatrix} \nabla_{xx}^2 L_\mu & 0 \\ 0 & \Lambda_g S \end{pmatrix}, \quad J = \begin{pmatrix} J_h & 0 \\ J_g & S \end{pmatrix}$$

$$d = \begin{pmatrix} d_x \\ \tilde{d}_s \end{pmatrix}, \quad \lambda^+ = \begin{pmatrix} \lambda_h^+ \\ \lambda_g^+ \end{pmatrix}$$

$$\nabla \phi = \begin{pmatrix} \nabla_x f \\ -\mu e \end{pmatrix} \quad \text{et} \quad c(x, s) = \begin{pmatrix} h(x) \\ g(x) + s \end{pmatrix}$$

Or les condition d'optimalité du problème quadratique

$$\left\{ \begin{array}{l} \min \frac{1}{2} d^t W d + \nabla \phi^t d \\ \text{s.c.} \\ Jd + c = 0 \\ d \in \mathbb{R}^{n+m} \end{array} \right. \quad (5.9)$$

sont

$$\begin{pmatrix} W & J^t \\ J & 0 \end{pmatrix} \begin{pmatrix} \tilde{d} \\ \lambda^+ \end{pmatrix} = - \begin{pmatrix} \nabla_x \phi \\ c \end{pmatrix} \quad (5.10)$$

où λ^+ est le multiplicateur de Lagrange associé à ce problème, d'où la connexion avec les conditions d'optimalité de dimension $(n + 2m + p)$ de notre problème non linéaire. La détermination de la direction de descente passe donc par la résolution d'un problème quadratique de dimensions $(n + m, m + p)$ dit sous-problème quadratique, ce problème est non convexe.

La solution du problème quadratique (5.9) sera recherchée dans une zone de confiance, ainsi le modèle quadratique résultant est une approximation du modèle non linéaire au voisinage du point courant. La région de confiance sera d'autant plus grande que l'approximation quadratique est meilleure.

La région de confiance sera élargie si on remarque que le modèle quadratique est une bonne approximation i.e., fournit des pas de déplacement de "bonne qualité", et sera rétrécie si aucune amélioration n'est produite. La convergence globale est liée à ce processus de mise à jour de la région de confiance. Une étude récente menée par Sartnear [77] a permis de décrire numériquement un bon processus de mise à jour de la région de confiance mais ceci reste très dépendant du problème étudié.

5.2 Hypothèses

Nous commençons par donner quelques hypothèses et définitions. Ce sont des hypothèses suffisantes pour que la méthode proposée soit bien définie. Soit $(x_k)_{k \geq 0}$ une suite d'itérés générée par l'algorithme. Pour simplifier l'exposition, on notera par $c_k = c(x_k) \in \mathbb{R}^{m+p}$, $f_k = f(x_k)$, $J_k = J(x_k)$ et $W_k = W(x_k) = \nabla_{xx}^2 L(x_k, s_k, \lambda_k)$.

Nous supposons que sont vérifiées les hypothèses suivantes :

Hypothèses. 5.2.1. (*Hypothèses de base*)

1. La fonction f et les contraintes sont différentiables sur un ouvert convexe X contenant tous les itérés x_k .
2. Les applications ∇f , c et J sont lipschitziennes sur X .
3. La suite $(f(x_k))_k$ est bornée inférieurement.
4. Les suites $(\nabla f(x_k))_k$, $(c(x_k))_k$, $(J(x_k))_k$ et $(W_k)_k$ sont bornées.
5. La plus petite valeur singulière de la matrice $(J_k)_k$ est supérieure à une certaine constante strictement positive pour tout k , i.e. $\exists \rho > 0$ tel que la plus petite valeur propre de $J_k J_k^t \geq \rho \quad \forall k \geq 0$.

□

La dernière hypothèse est plus forte que l'hypothèse de l'indépendance linéaire des colonnes de la matrice $J(x_k)$. La convergence de l'algorithme vers un point stationnaire n'est pas assurée si ces matrices ne sont pas de plein rang. Dans ce cas la seule chose que l'on puisse prouver est la convergence des itérés vers un point stationnaire de la mesure d'admissibilité $x \rightarrow \|c(x)\|$.

5.3 Résolution des sous problèmes

Nous avons choisi de contrôler le pas de déplacement par la région de confiance ainsi nous rajoutons une contrainte à notre sous-problème quadratique. Le modèle quadratique est :

$$\left\{ \begin{array}{l} \min \frac{1}{2} d^t W d + \nabla \phi^t d \\ \text{s.c.} \\ Jd + c = 0 \\ \|d\|_2 \leq r \\ d \in \mathbb{R}^{n+m} \end{array} \right. \quad (5.11)$$

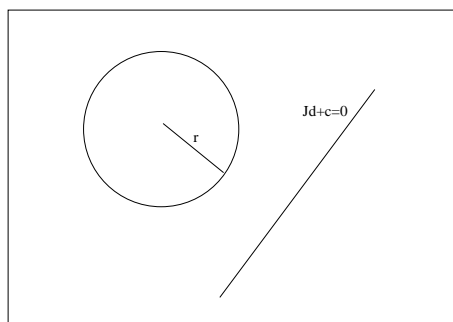


FIG. 5.1 – Exemple de sous-problème quadratique inconsistant.

où $r > 0$.

Afin de prévenir la non réalisabilité ou l'inconsistance du problème c'est-à-dire que l'ensemble

$$\{d \in \mathbb{R}^{n+m} / Jd + c = 0 \text{ et } \|d\|_2 \leq r\}$$

soit non vide (comme l'indiquent les dessins 5.1, 5.2 et 5.3), nous utilisons le schéma d'Omojukun [63]. En commençant par rechercher un vecteur v solution du problème (5.12) dit sous-problème quadratique vertical.

$$\begin{cases} \min \frac{1}{2} \|Jv + c\|_2^2 \\ \text{s.c.} \\ \|v\|_2 \leq \zeta r \end{cases} \quad (5.12)$$

où $\zeta \in]0, 1[$.

Ainsi au lieu de résoudre directement le sous-problème quadratique (5.11), on perturbe d'abord (si nécessaire) la contrainte linéaire $Jd + c = 0$ en la remplaçant par $Jd + c = Jv + c$ où v est la solution du sous-problème vertical

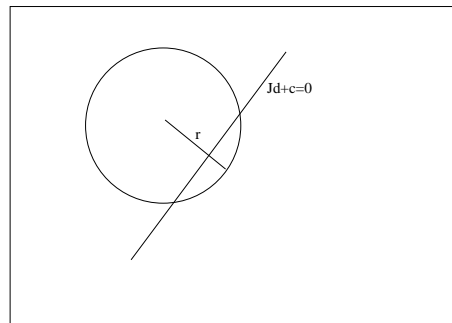


FIG. 5.2 – Exemple de sous-problème quadratique réalisable, dans ce cas v est nul.

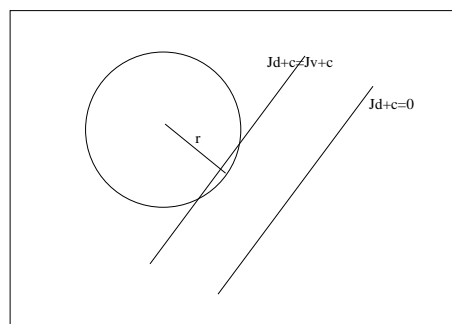


FIG. 5.3 – Exemple de sous-problème quadratique inconsistant perturbé et redevenu réalisable grâce à la programmation verticale.

(5.12), ensuite on résout le problème (5.13) issu de cette perturbation

$$\left\{ \begin{array}{l} \min \frac{1}{2} d^t W d + \nabla \phi^t d \\ \text{s.c.} \\ Jd + c = Jv + c \\ \|d\|_2 \leq r \\ d \in \mathbb{R}^{n+m} \end{array} \right. \quad (5.13)$$

ce qui est équivalent à

$$\left\{ \begin{array}{l} \min \frac{1}{2} d^t W d + \nabla \phi^t d \\ \text{s.c.} \\ J(d - v) = 0 \\ \|d\|_2 \leq r \\ d \in \mathbb{R}^{n+m} \end{array} \right. \quad (5.14)$$

on note $w = d - v$. Le problème suivant en w est appelé sous-problème quadratique horizontal.

$$\left\{ \begin{array}{l} \min \frac{1}{2} w^t W w + (\nabla \phi + W^t v)^t w \\ \text{s.c.} \\ Jw = 0 \\ \|w\|_2 \leq \bar{r} \\ d \in \mathbb{R}^{n+m} \end{array} \right. \quad (5.15)$$

où $\bar{r} = \sqrt{r^2 - \|v\|_2^2}$

La contrainte linéaire $Jw = 0$ traduit le fait que le vecteur w est un élément du noyau de J ainsi il existe une matrice Z (la base de l'espace généré par $\ker(J)$ de dimension $(n + m, n - p)$) tel que $JZ = 0$ et il existe un vecteur u dans \mathbb{R}^{n-p} tel que $w = Zu$. Ce changement de variable nous permet d'écrire le

problème (5.16) dit sous-problème quadratique réduit.

$$\left\{ \begin{array}{l} \min \frac{1}{2}u^t H u + (\nabla \phi_z)^t u \\ \text{s.c.} \\ \|u\|_2 \leq \bar{r} \\ u \in \mathbb{R}^{n-p} \end{array} \right. \quad (5.16)$$

où $H = Z^t W Z$ et $\nabla \phi_z = (\nabla \phi + W^t v) Z$.

Les deux sous-problèmes quadratiques (5.12 et 5.16) sont de la forme

$$\left\{ \begin{array}{l} \min Q(x) = \frac{1}{2}x^t G x + b^t x \\ \text{sc :} \\ \|x\|_2 \leq \delta \\ x \in \mathbb{R}^q \end{array} \right. \quad (5.17)$$

où $G = J^t J$, $b = c^t J$, et $\delta = \zeta r$ pour le sous-problème vertical, dans ce cas le problème (5.17) est convexe dans \mathbb{R}^{n+m} ($q = n + m$). Pour le sous-problème horizontal on pose $G = H$, $b = \nabla \phi_z$, et $\delta = \bar{r}$, ce dernier problème est non convexe par contre la résolution se fait dans \mathbb{R}^{n-p} ($q = n - p$) ce qui est très avantageux si p est grand.

On montre que si $\delta \neq 0$ alors le vecteur x est solution de (5.17) si et seulement s'il existe $\lambda \in \mathbb{R}$ tel que l'on ait

$$\left\{ \begin{array}{l} (G + \lambda I)x = -b \\ \|x\| \leq \delta \\ \lambda \geq 0 \\ \lambda(\|x\| - \delta) = 0 \\ (G + \lambda I) \text{ est semi-définie positive} \end{array} \right. \quad (5.18)$$

La résolution exacte du problème (5.18) n'est pas toujours possible surtout

dans le cas de problème de grandes tailles. La méthode de Newton tronquée a été introduite par Toint [79] et Steihaug [78]. Elle consiste à calculer par la méthode des gradients conjugués une solution approchée du système (5.18) en tenant compte de la région de confiance et de la courbure négative (pour la résolution du problème non convexe). Nous avons exposé cette approche au chapitre 2. En raison des expériences numériques du paragraphe (2.1.4), nous proposons de résoudre le sous-problème quadratique (5.17) en utilisant la programmation D.C.

La résolution du problème (5.17) passe par la programmation DC-optimale (1.3.2) que nous adaptons pour chaque problème. La règle générale de cette méthode est la production de deux suites x_k et z_k en suivant le schéma :

$$x_0 \in \mathbb{R}^q, k = 0, \varepsilon > 0, \rho > \|G\|_1$$

Tant que ($\|x_{k+1} - x_k\| > \varepsilon$) **faire**

$$k = k + 1$$

$$z_k = \frac{(\rho I - G)x_k - b}{\rho}$$

Si ($\|z_k\|_2 \leq \delta$) **Alors**

$$x_{k+1} = z_k$$

Sinon

$$x_{k+1} = \delta \frac{z_k}{\|z_k\|_2}$$

Fin Si

Fait

$$x^* = x_{k+1}$$

où ε est la précision recherchée.

Algorithme 12: L'algorithme de base de la résolution DC

L'étude numérique au chapitre 1, nous a permis de considérer $\varepsilon = 0.001$ comme étant une "bonne" précision. Le test d'arrêt peut être remplacé par un

test sur l'objectif i.e., tester si l'amélioration apportée à la fonction $\frac{1}{2}x^t Gx + b^t x$ est significative, c'est le cas de la programmation verticale où on cherche à perturber (5.11) de telle façon que l'ensemble réalisable soit non vide. On s'intéresse plus à la valeur de $Jv + c$ qu'au vecteur v .

La procédure DC pour résoudre le problème horizontal est la suivante :

$\rho > \|H\|_1$
 $u_0 \in \mathbb{R}^{n-p}, k = 0, \varepsilon > 0$

Tant que ($\|u_{k+1} - u_k\| > \varepsilon$) **faire**

$k \leftarrow k + 1, z_k = \frac{(\rho I - H)u_k - \nabla \Phi^t}{\rho}$

Si ($\|z_k\| \leq \bar{r}$) **Alors**

$u_{k+1} = z_k$

Sinon

$u_{k+1} = \bar{r} \frac{z_k}{\|z_k\|}$

Fin Si

Fait
 $u^* = u_{k+1}$
 $\tilde{w} = Zu^*$

Algorithme 13: Procédure DC pour résoudre le problème horizontal

Avant d'annoncer la procédure qui calcule la solution verticale, nous démontrons ce lemme (5.3.1) qui montre que la suite $(v_k)_k$ prend ses valeurs dans $Im(J^t)$.

Lemme. 5.3.1. *Soit le problème quadratique $\{\min Q(v) = \frac{1}{2}v^t J^t Jv + c^t Jv : \|v\| \leq \delta\}$*

où J est une matrice $(m \times n)$ et $c \in \mathbb{R}^m$ si le point initial v_0 est dans $Im(J^t)$ alors tous les itérés v_k donnés par la méthode DC-optimale que nous avons

décrit sont dans l'ensemble $Im(J^t)$ et la solution du problème quadratique est dans cet ensemble.

Preuve.

Considérons la décomposition $Q(v) = R(v) - T(v)$,

où $R(v) = \frac{1}{2}\rho\|v\|^2 + \langle J^t c, v \rangle$ et $T(v) = \frac{1}{2} \langle (\rho I - J^t J)v, v \rangle$

Comme JJ^t est une matrice semi-définie positive d'ordre n , il suffit de prendre $\rho > \max((JJ^t)_{ii} : i = 1, \dots, n)$.

Le principe de la méthode DC-optimale nous permet d'écrire

$$v_{k+1} = \frac{(\rho I - J^t J)v_k - J^t c}{\rho}$$

par récurrence, si à l'itération k nous avons $v_k \in Im(J^t)$ alors $\exists u_k \in \mathbb{R}^m$ tel que $v_k = J^t u_k$ d'où

$$\begin{aligned} v_{k+1} &= \frac{(\rho I_n - J^t J)v_k - J^t c}{\rho} \\ &= J^t \left(\frac{(\rho I_m - J J^t)u_k - c}{\rho} \right) = J^t u_{k+1} \end{aligned}$$

où

$$u_{k+1} = \frac{(\rho I_m - J J^t)u_k - c}{\rho} \quad (5.19)$$

avec I_n et I_m sont les matrices d'identité d'ordres n et m respectivement. Ainsi $\forall k$ nous avons $v_{k+1} \in Im(J^t)$. Ainsi pour avoir une solution du sous-problème vertical v dans l'ensemble $Im(J^t)$, il suffit de prendre un point initial v_0 dans cet ensemble, *i.e* $v_0 = J^t u_0$, où u_0 est arbitraire dans \mathbb{R}^m . Dans la pratique on prend le point de Cauchy comme point initial.

Finalement, remarquons qu'il suffit de prendre

$$\rho > \max_{i=1, \dots, n} (JJ^t)_{ii}$$

où JJ^t est une matrice d'ordre m .

□

La procédure DC pour résoudre le problème vertical (5.12) est donnée par l'algorithme suivant :

$$\rho > \max_{i=1, \dots, m+p} (JJ^t)_{ii}$$

$$u_0 \in \mathbb{R}^{m+p}, k = 0, \varepsilon > 0$$

Tant que ($\|u_{k+1} - u_k\| > \varepsilon$) **faire**

$$k \leftarrow k + 1$$

$$z_k = \frac{(\rho I - JJ^t)u_k - c}{\rho}$$

Si ($\|J^t z^k\| \leq \zeta r$) **Alors**

$$u_{k+1} = z_k$$

Sinon

$$u_{k+1} = \zeta r \frac{z^k}{\|z^k\|}$$

Fin Si

Fait

$$u^* = u_{k+1}$$

la solution du problème vertical est $v = J^t u^*$

Algorithme 14: Procédure DC pour résoudre le problème vertical

Ceci permet d'une part une orthogonalisation parfaite, ce qui améliore considérablement la méthode et d'autre part de résoudre le problème vertical dans \mathbb{R}^{m+p} au lieu de le résoudre dans \mathbb{R}^{n+m} avec $p < n$.

5.4 Les multiplicateurs de Lagrange

À chaque itération, nous avons besoin de calculer une estimation des multiplicateurs de Lagrange $\lambda = (\lambda_h, \lambda_g)$ avec précision et ceci pour deux raisons :

1. D'abord les multiplicateurs interviennent dans le calcul du Hessien du

Lagrangien. Ils influent ainsi la convergence locale. Pour obtenir une bonne convergence locale (quadratique dans le cas newtonien), les multiplicateurs doivent fournir une bonne approximation du multiplicateur optimal λ^* .

2. Ensuite les conditions d'optimalités sont évaluées à chaque itération en (x_k, s_k, λ_k) ainsi pour détecter l'optimalité, nous avons besoin d'une bonne approximation des multiplicateurs.

Le système linéaire (5.5) nous permet d'écrire

$$\begin{cases} J_h^t \lambda_h + J_g^t \lambda_g &= -\nabla f(x) \\ \lambda_g S e &= \mu e \end{cases} \quad (5.20)$$

ou sous forme matricielle

$$\begin{pmatrix} J_h^t & J_g^t \\ 0 & S \end{pmatrix} \begin{pmatrix} \lambda_h \\ \lambda_g \end{pmatrix} = - \begin{pmatrix} \nabla f(x) \\ -\mu e \end{pmatrix} \quad (5.21)$$

ou bien sous la forme synthétique suivante :

$$J^t \lambda = -\nabla \phi \quad (5.22)$$

Donc à chaque itération, on peut soit utiliser une minimisation de moindres carrées pour résoudre

$$\frac{1}{2} \|J^t \lambda + \nabla \phi\|_2^2$$

soit procéder par étape, d'abord calculer λ_g par la formule $S\lambda_g = \mu e$, i.e. chaque composante de λ_g est prise comme rapport μ/s_i pour tout $i = 1, 2, \dots, m$. ceci à l'avantage d'assurer la positivité des multiplicateurs de Lagrange relatifs aux

contraintes d'inégalité. Puis on calcule λ_h par la formule

$$J_h^t \lambda_h = -(\nabla f(x) + J_g^t \lambda_g)$$

Dans la littérature, cette technique produit des estimations connues sous le nom de multiplicateurs barrière, du fait de l'utilisation des conditions (5.5) relatives à la résolution du problème pénalisé.

Une deuxième technique utilise plutôt, le système de Newton (5.6) pour donner λ^+ qui sera utilisé dans l'étape suivante. Cette technique nécessite un multiplicateur initial λ_0 .

5.5 Les contraintes emboîtées

Deux sortes de contraintes restent à traiter, d'abord les contraintes de la forme $a \leq x \leq b$, nous traitons ces contraintes sous deux formes :

1. Au point initial nous commençons par choisir un point de départ qui satisfait ces contraintes en suivant la règle suivante :

Si le point initial ne vérifie pas la contrainte alors nous avons une des trois situations suivantes :

- (a) Si $a_i = -\infty$ (ceci correspond au cas où cette contrainte est absente)
alors $x_i = b_i - \varepsilon$.
- (b) Si $b_i = +\infty$ (ceci correspond au cas où cette contrainte est absente)
alors $x_i = a_i + \varepsilon$.
- (c) si $a_i > -\infty$ et $b_i < +\infty$ alors $x_i = \frac{a_i + b_i}{2}$.

Si le point initial vérifie la contrainte alors ne rien faire.

Au cours des itérations, nous considérons les contraintes $a \leq x \leq b$ comme contraintes d'inégalité :

$$x_i - b_i \leq 0$$

$$\text{et } a_i - x_i \leq 0.$$

2. La deuxième sorte de contraintes est la positivité de la variable s . Une manière d'assurer la positivité stricte des itérés s_k est de forcer le déplacement d_s à satisfaire :

$$s_k + d_s \geq (1 - \tau)s_k \quad (5.23)$$

avec $\tau \in]0, 1[$.

Autrement dit, nous exigeons au pas d_s de vérifier :

$$d_s \geq -\tau s_k \quad (5.24)$$

or d'après notre changement de variable $d_s = S_k \tilde{d}_s$ où S_k est la matrice diagonale dont les éléments diagonaux sont les composantes de s à l'itération k . Ainsi la solution \tilde{d}_s de notre sous-problème quadratique doit vérifier :

$$\tilde{d}_s \geq -\tau e_m \quad (5.25)$$

où e_m est le vecteur unité de \mathbb{R}^m .

Cette dernière inégalité est garantie par la technique de rebroussement suivante :

Si $\tilde{d}_s < -\tau e_m$ alors, soit $\theta = \min\{\frac{-\tau}{(d_s)_i}, i = 1, 2, \dots, m\}$.

$$\tilde{d}_s \leftarrow \theta \tilde{d}_s.$$

Il est clair que le nouveau pas vérifie la contrainte de positivité de s et que sa norme est inférieure au rayon de la région de confiance. En effet, si $\tilde{d}_s < -\tau e_m$ alors il existe un certain indice $i_0 \in \{1, 2, \dots, m\}$ tel que

$(\tilde{d}_s)_{i_0} \leq (\tilde{d}_s)_i \quad \forall i \in \{1, 2, \dots, m\}$ et que $(\tilde{d}_s)_{i_0} \leq -\tau$. En prenant $\theta = \min\{\frac{-\tau}{(\tilde{d}_s)_i}, i = 1, 2, \dots, m\}$, $\theta \in]0, 1]$ ce qui prouve que $(\tilde{d}_s)_i \theta \geq (\tilde{d}_s)_{i_0} \theta = -\tau$ pour tout $i \in \{1, 2, \dots, m\}$ et que $\|\theta \tilde{d}_s\| = \theta \|\tilde{d}_s\| \leq \theta r \leq r$

5.6 La fonction de mérite

La fonction de mérite permet de contrôler le pas de déplacement et la progression des itérations. Comme nous l'avons indiqué au paragraphe (3.6.3) plusieurs formes de fonctions de mérite sont possibles. Une étude numérique s'impose pour classifier le potentiel de chaque fonction de mérite. Dans tous les cas, la réduction actuelle notée *ared*, est définie à l'instant donné par

$$ared(d) = M(x, s, \sigma) - M(x + d_x, s + d_s, \sigma)$$

où M est la fonction de mérite choisie. Le vecteur d améliore l'optimalité (voir définition 3.6.1) si *ared*(d) est strictement positive.

L'amélioration apportée par le pas de déplacement $d = v + w$ au problème quadratique sera quantifiée par la réduction prédite :

$$pred(d) = -q(w) + \sigma \sqrt{-m(v)}$$

où $q(w) = \frac{1}{2} w^t W w + (\nabla \phi + W^t v)^t w$

et $m(v) = \frac{1}{2} v^t J^t J v + c^t J v$

Enfin le pas de déplacement sera accepté s'il permet une réduction de la fonction de mérite proportionnelle à la réduction prédite :

$$ared(d) > \eta pred(d)$$

où $\eta \in]0, 1[$, ou bien

$$r(d) > \eta$$

en notant $r(d) = \text{ared}(d)/\text{pred}(d)$ la qualité de la réduction, η est dit seuil d'acceptation. La qualité de la réduction contrôle la réévaluation du rayon de confiance r par la règle suivante

$$r_{k+1} = \begin{cases} \alpha_1 \|d_k\| & \text{Si } r(d) \leq \eta_1 \\ r_k & \text{Si } \eta_1 < r(d) < \eta_2 \\ \max(r_k, \alpha_2 \|d_k\|) & \text{Si } r(d) \geq \eta_2 \end{cases} \quad (5.26)$$

où les paramètres α_1 , α_2 , η_1 et η_2 sont à déterminer.

Dans une étude de Sartnear [77], les auteurs proposent les valeurs suivantes : $\alpha_1 = 0.25$, $\alpha_2 \in [3.5, 5]$, $\eta_1 = 10^{-4}$ et $\eta_2 = 0.99$.

Notre étude numérique donne une proposition semblable de ces valeurs qui restent, comme nous l'avons précisé lors de l'introduction de ce chapitre, liées à l'ensemble de problèmes étudiés. Nous prenons comme adéquat à notre étude les paramètres suivants :

$$\alpha_1 = 0.25, \alpha_2 = 5, \eta_1 = 10^{-4} \text{ et } \eta_2 = 0.99$$

5.7 Correction du second ordre

Si la fonction de mérite n'est pas différentiable en un point satisfaisant les contraintes $c(x, s) = 0$, alors elle souffre de l'effet Maratos [55]. Dans ce cas même si le pas de déplacement d_k permet une convergence rapide et proche de l'optimum, la non dérivabilité de la fonction de mérite peut entraîner une croissance de l'objectif et de la norme des contraintes, d'où le rejet de d_k . Cet effet indésirable peut être éliminé en ajoutant à d_k un déplacement d_k^{soc} qui est

solution du problème (5.27) suivant

$$\min_d \|J_k d + c((x_k, s_k) + d_k)\| \quad (5.27)$$

On parle de correction du second ordre car $c((x_k, s_k) + d_k) = \mathcal{O}(\|d_k\|^2)$ d'où $d_k^{soc} = \mathcal{O}(\|d_k\|^2)$. L'effet Maratos est détecté si le pas vertical est petit devant le pas horizontal (voir [62, 46]). L'algorithme du second ordre est décrit par la procédure (15) suivante

Procédure Correction du second ordre(CSO)

Choisir η_1^{soc} , η_2^{soc} et η_3^{soc}

Si $((r(d) \leq \eta_1^{soc}) \text{ et } (\|v_k\| \leq \eta_2^{soc} \zeta r) \text{ et } (\|v_k\| \leq \eta_2^{soc} \|w_k\|))$ **Alors**

Calculer d_k^{soc} solution de (5.27)

Calculer $ared_k^{soc} = ared(d_k + d_k^{soc})$ et $r^{soc} = \frac{ared_k^{soc}}{pred}$

Si $(r^{soc} > r(d))$ **Alors**

$d_k \leftarrow d_k^{soc}$

$ared_k = ared_k^{soc}$ et $r(d) = r^{soc}$

$k \leftarrow k + 1$

Fin Si

Fin Si

Fin

Algorithme 15: Correction du second ordre

Généralement on garde le rayon de confiance invariant après une correction du second ordre réussi.

Finalement, la méthode SDC décrite ci-dessus produit l'algorithme 16 suivant :

Choisir ε_μ et ε_{tot}

Choisir $r_0 > 0$, $\mu_0 > 0$, $\sigma_0 \geq 0$, $s_0 > 0$ et $\theta \in]0, 1[$

$k = 0$

Tant que ($E(x_k, s_k, \mu_k) > \varepsilon_{tot}$) **faire**

Tant que ($E(x_k, s_k, \mu_k) > \varepsilon_\mu$) **faire**

 Calculer λ_k par (5.21)

 Évaluer le Hessien W et le Jacobien J

 Calculer $v = J^t u$ par la procédure (14)

 Calculer $w = Zu$ par la procédure (13)

$\tilde{d}_k = v + w$

 Reboisser d_k en respectant (5.24)

$d_k = S\tilde{d}_k$

 Évaluer σ_k et calculer la fonction de mérite $M(x_k, s_k, \sigma_k)$

Si ($ared > \eta pred$) **Alors**

$x_{k+1} = x_k + d_x$

$s_{k+1} = s_k + d_s$

$k \leftarrow k + 1$

 Évaluer le rayon de confiance r_k par la règle (5.26)

Sinon

 Utiliser la procédure *CSO* décrite par l'algorithme (15)

Fin Si

Fait

$\mu_k = \theta \mu_k$ et $\varepsilon_\mu = \theta \varepsilon_\mu$

Fait

Algorithme 16: L'algorithme général de la méthode SDC

5.8 Tests numériques

5.8.1 Impact du changement de variable

Nous nous intéressons dans un premier temps à l'impact du changement que nous avons introduit sur la résolution du problème non linéaire (5.1). Nous avons choisi de tester les performances de notre méthode sur un ensemble de problèmes issus de la bibliothèque CUTEr [96]. Nous avons demandé à notre code de résoudre à chaque étape le même problème d'abord sans la modification (méthode 1) apportée aux conditions d'optimalité puis avec la modification (méthode 2 du tableau (5.1)).

Trois groupes se distinguent à la fin de la minimisation :

1. L'ensemble des problèmes sans contraintes d'inégalité pour lesquels les résultats sont les mêmes pour les deux méthodes.
2. L'ensemble des problèmes avec contraintes d'inégalité non saturés pour lesquels les composantes de la variable s restent raisonnablement loin de zéro. Pour cet ensemble on remarque que les résultats sont globalement les mêmes pour les deux méthodes.
3. L'ensemble des problèmes avec contraintes d'inégalité pour lesquels on remarque que la variable s (plus précisément une de ses composantes) s'approche de zéro. C'est manifestement cet ensemble qui nous intéresse dans cette étude.

Ce dernier groupe se divise en deux sous-groupes. Nous remarquons que pour un certain nombre de problèmes, la variable s est certes petite mais garde cet ordre jusqu'à la fin de la minimisation. Pour le second sous-ensemble, la variable s tend vers zéro et ceci affecte, comme nous l'avons prédit, le processus de la minimisation de la première méthode, tandis que la deuxième méthode continue sa convergence.

Problème	méthode 1			méthode 2		
	iter	obj	cc	iter	obj	cc
AKIVA	46	6.17e+00	0.00e+00	46	6.17e+00	0.00e+00
ALLINITU	44	5.74e+00	0.00e+00	44	5.74e+00	0.00e+00
ALSOTAME	43	1.94e-01	1.11e-16	43	1.94e-01	1.11e-16
BARD	46	1.02e-02	0.00e+00	46	1.02e-02	0.00e+00
BEALE	46	4.12e-09	0.00e+00	46	4.12e-09	0.00e+00
BIGGS6	46	2.50e-01	0.00e+00	46	2.50e-01	0.00e+00
BOOTH	27	0.00e+00	9.68e-09	27	0.00e+00	9.68e-09
BOX3	46	4.15e-04	0.00e+00	46	4.15e-04	0.00e+00
BQP1VAR	1	6.32e-02	0.00e+00	1	6.32e-02	0.00e+00
BQPGASIM	1	-3.20e-06	0.00e+00	1	-3.20e-06	0.00e+00
BRKMCC	44	1.69e-01	0.00e+00	44	1.69e-01	0.00e+00
BT10	46	-1.00e+00	2.59e-05	46	-1.00e+00	2.59e-05
BT13	46	1.68e+02	2.41e-12	46	1.79e+02	6.20e-12
BT1	21	-3.17e-02	1.08e-18	21	-3.17e-02	1.08e-18
BT4	46	-1.86e+01	7.57e-09	46	-1.86e+01	7.57e-09
BT9	8	-3.71e-01	7.97e-01	8	-3.71e-01	7.97e-01
CANTILVR	46	1.38e+00	1.16e-01	46	1.38e+00	1.16e-01
CHACONN1	46	2.00e+00	0.00e+00	46	2.00e+00	0.00e+00
CHACONN2	46	2.04e+00	0.00e+00	46	2.11e+00	0.00e+00
CHANDHEQ	46	0.00e+00	4.00e-03	46	0.00e+00	4.00e-03
CHEBYQAD	1	1.72e-02	0.00e+00	1	1.72e-02	0.00e+00
CHNROSNB	46	5.84e-01	0.00e+00	46	5.84e-01	0.00e+00
CLIFF	46	2.01e-01	0.00e+00	46	2.01e-01	0.00e+00
... suite page suivante...						

... suite de la page précédente...						
CLUSTER	46	0.00e+00	4.25e-06	46	0.00e+00	4.25e-06
CUBENE	46	0.00e+00	2.56e-01	46	0.00e+00	2.56e-01
DECONVU	46	1.81e-03	0.00e+00	46	1.81e-03	0.00e+00
DENSCHNA	23	8.91e-17	0.00e+00	23	8.91e-17	0.00e+00
DENSCHNB	20	1.29e-17	0.00e+00	20	1.29e-17	0.00e+00
DENSCHNC	36	5.25e-19	0.00e+00	36	5.25e-19	0.00e+00
DENSCHND	46	8.76e-08	0.00e+00	46	8.76e-08	0.00e+00
DENSCHNE	46	9.99e-01	0.00e+00	46	9.99e-01	0.00e+00
DENSCHNF	24	9.32e-22	0.00e+00	24	9.32e-22	0.00e+00
EG1	1	1.41e-01	9.63e-01	1	1.41e-01	9.63e-01
EIGMAXB	10	-4.84e-01	1.83e-01	10	-4.84e-01	1.83e-01
EIGMINB	10	4.84e-01	1.83e-01	10	4.84e-01	1.83e-01
ERRINROS	46	4.02e+01	0.00e+00	46	4.02e+01	0.00e+00
EXTRASIM	1	4.64e-01	5.36e-01	1	4.64e-01	5.36e-01
GOTTFR	46	0.00e+00	1.80e-02	46	0.00e+00	1.80e-02
HATFLDA	1	2.85e+00	0.00e+00	4	8.96e-01	0.00e+00
HATFLDD	46	3.30e-03	0.00e+00	46	3.30e-03	0.00e+00
HATFLDE	46	1.46e-01	0.00e+00	46	1.46e-01	0.00e+00
HELIX	46	3.32e-03	0.00e+00	46	3.32e-03	0.00e+00
HILBERTA	29	8.00e-16	0.00e+00	29	8.00e-16	0.00e+00
HIMMELBA	36	0.00e+00	9.82e-09	36	0.00e+00	9.82e-09
HIMMELBB	7	8.42e-23	0.00e+00	7	8.42e-23	0.00e+00
HIMMELBC	15	0.00e+00	9.15e-09	15	0.00e+00	9.15e-09
HIMMELBE	45	0.00e+00	9.95e-09	45	0.00e+00	9.95e-09
... suite page suivante...						

... suite de la page précédente...						
HIMMELBG	18	1.58e-18	0.00e+00	18	1.58e-18	0.00e+00
HIMMELBH	23	-1.00e+00	0.00e+00	23	-1.00e+00	0.00e+00
HIMMELP2	5	8.60e+01	0.00e+00	5	8.60e+01	0.00e+00
HIMMELP4	46	-7.91e+00	0.00e+00	46	-7.91e+00	0.00e+00
HS10	46	-2.62e+03	1.84e+07	1	-2.00e+01	5.99e+02
HS110	2	-4.55e+01	0.00e+00	2	-4.55e+01	0.00e+00
HS11	2	-1.96e+01	5.35e+01	1	-2.50e+01	2.39e+01
HS15	9	1.03e+08	2.50e+02	1	3.00e+02	1.24e+00
HS16	1	6.52e+02	1.40e+00	1	6.52e+02	1.40e+00
HS18	2	6.54e+00	4.56e+01	1	1.22e+01	1.29e+01
HS22	1	2.23e+01	0.00e+00	1	1.76e+00	1.16e-04
HS25	1	3.28e+01	0.00e+00	1	3.28e+01	0.00e+00
HS29	2	-1.00e+00	0.00e+00	2	-1.00e+00	0.00e+00
HS34	46	-3.83e-04	0.00e+00	46	-3.83e-04	0.00e+00
HS39	8	-3.71e-01	7.97e-01	8	-3.71e-01	7.97e-01
HS41	3	1.85e+00	2.68e-01	3	1.85e+00	2.68e-01
HS44	11	-8.91e-02	1.43e-02	17	2.01e-05	3.21e-05
HS45	4	8.47e-01	5.43e-01	4	8.47e-01	5.43e-01
HS1	46	9.09e+02	0.00e+00	46	9.09e+02	0.00e+00
HS53	1	6.00e+00	6.90e+00	1	6.00e+00	6.90e+00
HS55	9	6.67e+00	1.46e-01	9	6.67e+00	1.46e-01
HS60	13	4.45e+00	1.02e-06	13	4.45e+00	1.02e-06
HS66	46	5.80e-01	0.00e+00	46	5.80e-01	0.00e+00
HS66	46	5.80e-01	0.00e+00	46	5.80e-01	0.00e+00
... suite page suivante...						

... suite de la page précédente...						
HS7	46	-1.73e+00	1.16e-11	46	-1.73e+00	1.16e-11
HS80	21	5.90e-02	1.77e+00	21	5.90e-02	1.77e+00
HS81	29	5.69e-02	2.00e-02	29	5.69e-02	2.00e-02
HS88	1	5.01e-01	1.42e-01	1	5.01e-01	1.42e-01
HS8	17	-1.00e+00	2.19e-09	17	-1.00e+00	2.19e-09
HYP CIR	20	0.00e+00	8.76e-09	20	0.00e+00	8.76e-09
KIWCRESC	2	2.42e+00	1.08e+00	6	2.06e+00	1.13e-02
KOWOSB	46	4.36e-04	0.00e+00	46	4.36e-04	0.00e+00
MANCINO	37	5.54e-22	0.00e+00	37	5.54e-22	0.00e+00
MARATOSB	46	9.96e-01	0.00e+00	46	9.96e-01	0.00e+00
MARATOS	46	-9.97e-01	9.79e-11	46	-9.97e-01	9.79e-11
METHANB8	46	0.00e+00	1.61e-01	46	0.00e+00	1.61e-01
MIFFLIN2	9	2.87e-01	5.70e-01	4	1.58e+00	1.70e-01
MINSURF	3	1.00e+00	8.15e-03	3	1.00e+00	8.15e-03
PALMER5C	36	2.13e+00	0.00e+00	36	2.13e+00	0.00e+00
PORTFL1	2	2.45e-02	4.08e-11	2	2.45e-02	4.08e-11
PORTFL2	2	3.43e-02	4.08e-11	2	3.43e-02	4.08e-11
PORTFL3	2	3.90e-02	4.08e-11	2	3.90e-02	4.08e-11
PORTFL4	2	3.05e-02	4.08e-11	2	3.05e-02	4.08e-11
PORTFL6	2	3.00e-02	4.08e-11	2	3.00e-02	4.08e-11
POWELLSQ	46	0.00e+00	6.82e-02	46	0.00e+00	6.82e-02
PRODPL0	0	0.00e+00	3.33e+00	0	0.00e+00	3.33e+00
PSPDOC	1	3.82e+01	0.00e+00	1	3.82e+01	0.00e+00
RECIPE	46	0.00e+00	2.41e-04	46	0.00e+00	2.41e-04
... suite page suivante...						

... suite de la page précédente...						
ROSENBR	46	1.44e-03	0.00e+00	46	1.44e-03	0.00e+00
S268	1	1.20e+04	5.32e-06	1	1.20e+04	5.32e-06
S277-280	1	2.44e+00	9.24e-01	1	2.44e+00	9.24e-01
S308	22	7.73e-01	0.00e+00	22	7.73e-01	0.00e+00
SENSORS	46	-2.02e+03	0.00e+00	46	-2.02e+03	0.00e+00
SIMPLLLPA	46	-7.43e+02	1.27e+03	1	-2.37e-02	8.17e-01
SIMPLLLPB	46	-4.00e+03	4.92e+03	1	1.03e+00	2.32e-01
SISSER	19	2.10e-12	0.00e+00	19	2.10e-12	0.00e+00
SUPERSIM	1	4.35e-01	2.79e-01	1	4.35e-01	2.79e-01
SYNTHESES1	3	3.31e+00	6.66e-01	2	1.00e+01	7.17e-05
SYNTHESES2	0	1.42e+02	1.00e+00	0	1.42e+02	1.00e+00
TOINTPSP	46	2.26e+02	0.00e+00	46	2.26e+02	0.00e+00
TRY-B	46	0.00e+00	8.20e-06	46	0.00e+00	8.20e-06
TWOBARS	33	2.07e+00	1.86e-04	46	2.89e+00	8.55e-02
VANDERM1	13	0.00e+00	4.94e-01	13	0.00e+00	4.94e-01
VANDERM2	13	0.00e+00	4.94e-01	13	0.00e+00	4.94e-01
WATSON	46	3.10e-03	0.00e+00	46	3.10e-03	0.00e+00
ZANGWIL2	11	-1.82e+01	0.00e+00	11	-1.82e+01	0.00e+00
ZANGWIL3	21	0.00e+00	9.53e-09	21	0.00e+00	9.53e-09
ZECEVIC2	12	-2.27e-01	2.16e-02	12	-2.27e-01	2.16e-02
ZECEVIC4	46	3.60e+01	2.00e+00	26	3.96e+01	2.51e-02

TAB. 5.1: Effet du changement de la variable sur un ensemble de problèmes de la bibliothèque CUTEr

Notons que $iter$ désigne le nombre d'itérations, obj est la valeur optimale de la fonction objectif et cc est le défaut de réalisabilité du problème pénalisé :

$$cc = \max\{\|h(x)\|_\infty, \|g(x) + s\|_\infty\}$$

Nous remarquons que les deux méthodes ne réagissent pas de la même façon pour certains problèmes, c'est le cas des problèmes : BT13, CHACONN2, HS10, HS11, HS15, HS18, HS22, HS44, KIWCRES, MIFFLIN2, SIMPLLP, SIMPLLPB, SYNTHES1 et ZECEVIC4, pour lesquels, on remarque que cc est loin d'être proche de zéro, et que le nombre d'itérations atteint son maximum 46. Ce plafond est pris implicitement à ce niveau à cause de la réduction de la variable de pénalisation, en effet à chaque itération μ est dévissée par 5 et comme nous avons imposé dans notre programme le test d'arrêt $\mu > \mu_\varepsilon = 10^{-32}$ alors $\frac{\mu_0}{5^{46}} \simeq 7\mu_0 10^{-33}$ d'où, si la valeur initial μ_0 est égal à 1 alors le nombre maximal d'itérations est 45 et si μ est initialement égale à 10 alors le nombre maximal d'itérations est 46. En fait dans cette première étude on s'intéresse plus à la réaction des deux méthodes face au changement de variable qu'à la précision de la valeur optimale, d'ailleurs les études suivantes montrent que ce plafond n'est pas nécessairement le mieux adapté, nous corrigerons ceci par la prise de μ_ε plus petit que 10^{-32} ou d'utiliser des fractions moins fortes que la divisions par 5 qui est prise dans notre étude.

Remarquons aussi que deux problèmes (BT13 et CHACONN2) réagissent mieux à la méthode 1 qu'à la résolution par la méthode 2, et que les problèmes HS10, HS11, HS15 et HS18 produisent des itérés non réalisables mais que pour ces quatre problèmes, les points "optimaux" produits par la méthode 2 sont meilleurs que ceux produits par la méthode 1.

Un affichage plus exhaustif nous permet de mieux comprendre ces résultats (voir tableau (5.2)).

Méthode	Problème	iter	obj	$ h $	$ g + s $	s_{\min}	r	μ
Méthode 1	BT13	46	1.68e+02	2.41e-12	7.02e+00	1.75e+02	1.49e+01	7.04e-32
Méthode 2	BT13	46	1.79e+02	6.20e-12	4.52e+00	1.75e+02	1.72e+01	7.04e-32
Méthode 1	CHACONN2	46	2.04e+00	0.00e+00	8.72e-10	3.13e-04	8.08e-08	7.04e-32
Méthode 2	CHACONN2	46	2.11e+00	0.00e+00	1.82e-09	2.37e-10	1.38e-08	7.04e-32
Méthode 1	HS10	46	-2.62e+03	0.00e+00	1.84e+07	1.12e-108	7.00e+01	7.04e-32
Méthode 2	HS10	1	-2.00e+01	0.00e+00	5.99e+02	3.93e-05	4.51e-19	2.00e+00
Méthode 1	HS11	2	-1.96e+01	0.00e+00	5.35e+01	1.96e-07	4.79e-19	4.00e-01
Méthode 2	HS11	1	-2.50e+01	0.00e+00	2.39e+01	3.93e-05	6.71e-19	2.00e+00
Méthode 1	HS15	9	1.03e+08	0.00e+00	2.51e+02	2.87e-01	1.73e-19	5.12e-06
Méthode 2	HS15	1	3.00e+02	0.00e+00	1.24e+00	2.61e-03	7.79e-19	2.00e+00
Méthode 1	HS18	2	6.54e+00	0.00e+00	5.38e+01	6.54e-06	2.71e-19	4.00e-01
Méthode 2	HS18	1	1.22e+01	0.00e+00	1.29e+01	6.55e-06	1.68e-19	2.00e+00
Méthode 1	HS22	1	2.23e+01	0.00e+00	5.46e-01	1.96e-05	1.97e-19	2.00e+00
Méthode 2	HS22	1	1.76e+00	0.00e+00	7.92e-01	1.96e-05	1.84e-19	2.00e+00
Méthode 1	HS44	11	-8.91e-02	0.00e+00	2.03e-02	3.93e-06	1.88e-19	2.05e-07
Méthode 2	HS44	17	2.01e-05	0.00e+00	8.18e-04	7.86e-04	9.03e-19	1.31e-11
Méthode 1	KIWCRESC	2	2.42e+00	0.00e+00	1.86e+00	2.28e-05	6.99e-19	4.00e-01
Méthode 2	KIWCRESC	6	2.06e+00	0.00e+00	1.17e-01	2.19e-05	9.54e-19	6.40e-04
Méthode 1	MIFFLIN2	9	2.87e-01	0.00e+00	6.39e-01	1.64e-03	1.85e-19	5.12e-06
Méthode 2	MIFFLIN2	4	1.58e+00	0.00e+00	8.19e-01	7.75e-05	1.94e-19	1.60e-02
Méthode 1	SIMPLLPA	46	-7.43e+02	0.00e+00	1.27e+03	2.79e-109	1.75e+01	7.04e-32
Méthode 2	SIMPLLPA	1	-2.37e-02	0.00e+00	9.02e-01	9.82e-06	7.67e-19	2.00e+00
Méthode 1	SIMPLLPB	46	-4.00e+03	0.00e+00	4.92e+03	2.23e-109	7.00e+01	7.04e-32
Méthode 2	SIMPLLPB	1	1.03e+00	0.00e+00	4.04e-01	7.86e-06	7.26e-19	2.00e+00
... suite page suivante...								

... suite de la page précédente...								
Méthode 1	SYNTHES1	3	3.31e+00	0.00e+00	1.20e+00	2.18e-06	7.89e-19	8.00e-02
Méthode 2	SYNTHES1	2	1.00e+01	0.00e+00	5.70e-04	4.36e-04	1.63e-19	4.00e-01
Méthode 1	ZECEVIC4	46	3.60e+01	0.00e+00	2.01e+00	7.36e-20	4.36e+00	7.04e-32
Méthode 2	ZECEVIC4	26	3.96e+01	0.00e+00	3.29e-02	1.05e-06	6.89e-19	6.71e-18

TAB. 5.2: Quand une composante de S s'approche de zéro le modèle explose.

Dans ce tableau $|h| = \|h(x)\|_\infty$, $|g + s| = \|g(x) + s\|_\infty$ représentent respectivement le défaut de réalisabilité par rapport aux contraintes d'égalités et d'inégalités et $s_{\min} = \min\{s_i, \quad i = 1, \dots, m\}$ la plus petite composante de la variable s à un instant donné (ici c'est l'instant final).

Les résultats du tableau (5.2) montrent que dans tous les cas, la contrainte d'égalité est bien traitée ceci rejoint les remarques de Morales, Nocedal, Waltz, Liu et Goux [59] concernant les problèmes d'égalité. La méthode semble fonctionner trop bien avec des contraintes d'égalité. La deuxième remarque est que dans le cas où la réalisabilité fait défaut correspond à chaque fois à une valeur s_{\min} tendant vers zéro. Les tableaux (5.3), (5.4), (5.5), (5.6), (5.7), (5.8) et (5.9) montrent que la situation s'aggrave à chaque itération, $s_{\min} \rightarrow 0$, du fait que le modèle pénalisé est dominé par la partie dépendant de s , ce qui pousse le pas de déplacement à vouloir minimiser s plutôt que $f(x)$ ou $c(x)$. Quant à la fonction de mérite, elle est unidimensionnelle, et elle n'a pas le potentiel de lutter contre ce phénomène. En effet la fonction de mérite contrôle la progression entre deux itérations successives. Cette progression est assurée puisque la minimisation de s minimiserait la fonction de mérite. Or les conditions d'optimalité ne sont pas satisfaites et s continue à tendre vers zéro sans l'atteindre,

ce qui implique que la méthode est dans ce cas infinie, dans les expériences reportées au tableau (5.2) la méthode s'arrête si μ devient trop petit de l'ordre de 10^{-32} ou si le rayon de la région de confiance est restreint à l'ordre de 10^{-19} .

Nous remarquons que le rayon de confiance est assez restreint pour la méthode 2 (excepté pour les problèmes BT13 et CHACONN2). Le rayon de confiance dépend dans sa progression de la qualité de réduction $r(d)$ et s'il arrive que cette quantité est inférieure à η_1 (voir la règle (5.26)) au voisinage d'un point stationnaire alors $\|d\|$ devient très petit et la valeur de r chute sans pouvoir se remettre réellement. Il est clair que la règle (5.26) améliore, au mieux, le rayon de confiance par

$$r_{k+1} = \max(r_k, \alpha_2 \|d_k\|)$$

avec $\alpha_2 = 5$.

Cette remarque nous a poussé à essayer des valeurs plus avantageuses pour r dans le cas où $r(d) \geq \eta_2$:

$$r_{k+1} = \max(10r_k, \alpha_2 \|d_k\|)$$

Dans l'étude de Byrd [9], on trouve la réévaluation suivante

1. Si $r(d) < \eta$ alors $r_{k+1} \in [0.1r_k, 0.5r_k]$.
2. Sinon

$$r_{k+1} = \begin{cases} r_k & \text{Si } \eta \leq r(d) < 0.3 \\ \max(2\|d_k\|, r_k) & \text{Si } 0.3 \leq r(d) < 0.9 \\ \max(7\|d_k\|, r_k) & \text{Si } r(d) \geq 0.9 \end{cases} \quad (5.28)$$

avec $\eta = 10^{-8}$.

Dans la thèse de Jonsson on trouve pour son algorithme PTR [46] la mise à

jour suivante :

1. Si $r(d) \leq \eta_1$ alors on prend $r_{k+1} \in]0, \omega r_k[$
2. Sinon

$$r_{k+1} = \begin{cases} \in]\omega r_k, r_k[& \text{Si } \eta_1 \leq r(d) \leq \eta_2 \\ \geq r_k & \text{Si } r(d) \geq \eta_2 \end{cases} \quad (5.29)$$

puis $r_{k+1} = \max(r_{\min}, r_{k+1})$ où r_0 est la valeur initiale du rayon de confiance, $r_{\min} \in]0, r_0]$ et $\omega \in]0, 1[$.

Jonsson cible par cette règle qu'un bon itéré ne puisse pas hériter un mauvais rayon de confiance (trop petit) provenant d'un mauvais itéré.

Reste à dire que le problème pénalisé, ne pénalise pas directement les contraintes d'inégalité mais la variable s et pousse $g(x) + s$ à tendre vers zéro, comme s'il s'agissait d'une contrainte d'égalité, avec la différence que les multiplicateurs de Lagrange sont pris positifs, ce qui rappelle qu'il s'agit bel et bien de contraintes d'inégalité. Ceci a pour conséquence que le phénomène que nous avons décrit ci-dessus peut se produire si s tend vers zéro sans pour autant que la contrainte d'inégalité ne soit saturée. Dans ce cas, le problème peut être contourné en prenant

$$s_i = \max(s_i, -g_i(x))$$

Nous retrouvons ce pas dans les algorithmes de Jonsson [46].

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	-2.00e+01	0.00e+00	5.99e+02	7.86e-03	5.99e+02	7.00e+01	1.00e+01
1	-2.00e+01	0.00e+00	5.99e+02	3.93e-05	5.99e+02	7.00e+01	2.00e+00
2	-2.57e+01	0.00e+00	1.40e+03	1.96e-07	1.40e+03	7.00e+01	4.00e-01
... suite page suivante...							

... suite de la page précédente...							
3	-3.39e+01	0.00e+00	3.20e+03	9.82e-10	3.20e+03	7.00e+01	8.00e-02
4	-4.72e+01	0.00e+00	7.21e+03	4.91e-12	7.21e+03	7.00e+01	1.60e-02
5	-6.93e+01	0.00e+00	1.62e+04	2.46e-14	1.62e+04	7.00e+01	3.20e-03
6	-1.05e+02	0.00e+00	3.65e+04	1.23e-16	3.65e+04	7.00e+01	6.40e-04
7	-1.64e+02	0.00e+00	8.22e+04	6.14e-19	8.22e+04	7.00e+01	1.28e-04
8	-2.27e+02	0.00e+00	1.51e+05	3.07e-21	1.51e+05	7.00e+01	2.56e-05
9	-2.90e+02	0.00e+00	2.41e+05	1.53e-23	2.41e+05	7.00e+01	5.12e-06
10	-3.53e+02	0.00e+00	3.53e+05	7.67e-26	3.53e+05	7.00e+01	1.02e-06
11	-4.16e+02	0.00e+00	4.85e+05	3.84e-28	4.85e+05	7.00e+01	2.05e-07
12	-4.79e+02	0.00e+00	6.38e+05	1.92e-30	6.38e+05	7.00e+01	4.10e-08
13	-5.42e+02	0.00e+00	8.13e+05	9.59e-33	8.13e+05	7.00e+01	8.19e-09
14	-6.05e+02	0.00e+00	1.01e+06	4.80e-35	1.01e+06	7.00e+01	1.64e-09
15	-6.68e+02	0.00e+00	1.22e+06	2.40e-37	1.22e+06	7.00e+01	3.28e-10
16	-7.31e+02	0.00e+00	1.46e+06	1.20e-39	1.46e+06	7.00e+01	6.55e-11
17	-7.94e+02	0.00e+00	1.72e+06	5.99e-42	1.72e+06	7.00e+01	1.31e-11
18	-8.57e+02	0.00e+00	2.00e+06	3.00e-44	2.00e+06	7.00e+01	2.62e-12
19	-9.19e+02	0.00e+00	2.30e+06	1.50e-46	2.30e+06	7.00e+01	5.24e-13
20	-9.82e+02	0.00e+00	2.62e+06	7.49e-49	2.62e+06	7.00e+01	1.05e-13
21	-1.05e+03	0.00e+00	2.97e+06	3.75e-51	2.97e+06	7.00e+01	2.10e-14
22	-1.11e+03	0.00e+00	3.33e+06	1.87e-53	3.33e+06	7.00e+01	4.19e-15
23	-1.17e+03	0.00e+00	3.72e+06	9.37e-56	3.72e+06	7.00e+01	8.39e-16
24	-1.23e+03	0.00e+00	4.13e+06	4.68e-58	4.13e+06	7.00e+01	1.68e-16
25	-1.30e+03	0.00e+00	4.55e+06	2.34e-60	4.55e+06	7.00e+01	3.36e-17
26	-1.36e+03	0.00e+00	5.00e+06	1.17e-62	5.00e+06	7.00e+01	6.71e-18
27	-1.42e+03	0.00e+00	5.47e+06	5.85e-65	5.47e+06	7.00e+01	1.34e-18
... suite page suivante...							

... suite de la page précédente...							
28	-1.49e+03	0.00e+00	5.96e+06	2.93e-67	5.96e+06	7.00e+01	2.68e-19
29	-1.55e+03	0.00e+00	6.48e+06	1.46e-69	6.48e+06	7.00e+01	5.37e-20
30	-1.61e+03	0.00e+00	7.01e+06	7.32e-72	7.01e+06	7.00e+01	1.07e-20
31	-1.67e+03	0.00e+00	7.56e+06	3.66e-74	7.56e+06	7.00e+01	2.15e-21
32	-1.74e+03	0.00e+00	8.14e+06	1.83e-76	8.14e+06	7.00e+01	4.29e-22
33	-1.80e+03	0.00e+00	8.74e+06	9.15e-79	8.74e+06	7.00e+01	8.59e-23
34	-1.86e+03	0.00e+00	9.35e+06	4.57e-81	9.35e+06	7.00e+01	1.72e-23
35	-1.93e+03	0.00e+00	9.99e+06	2.29e-83	9.99e+06	7.00e+01	3.44e-24
36	-1.99e+03	0.00e+00	1.07e+07	1.14e-85	1.07e+07	7.00e+01	6.87e-25
37	-2.05e+03	0.00e+00	1.13e+07	5.72e-88	1.13e+07	7.00e+01	1.37e-25
38	-2.12e+03	0.00e+00	1.20e+07	2.86e-90	1.20e+07	7.00e+01	2.75e-26
39	-2.18e+03	0.00e+00	1.28e+07	1.43e-92	1.28e+07	7.00e+01	5.50e-27
40	-2.24e+03	0.00e+00	1.35e+07	7.15e-95	1.35e+07	7.00e+01	1.10e-27
41	-2.30e+03	0.00e+00	1.43e+07	3.57e-97	1.43e+07	7.00e+01	2.20e-28
42	-2.37e+03	0.00e+00	1.51e+07	1.79e-99	1.51e+07	7.00e+01	4.40e-29
43	-2.43e+03	0.00e+00	1.59e+07	8.93e-102	1.59e+07	7.00e+01	8.80e-30
44	-2.49e+03	0.00e+00	1.67e+07	4.47e-104	1.67e+07	7.00e+01	1.76e-30
45	-2.56e+03	0.00e+00	1.75e+07	2.23e-106	1.75e+07	7.00e+01	3.52e-31
46	-2.62e+03	0.00e+00	1.84e+07	1.12e-108	1.84e+07	7.00e+01	7.04e-32

TAB. 5.3: Résolution du problème HS10 par la méthode 1.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	9.09e+02	0.00e+00	3.00e+00	2.62e-03	1.92e+03	7.00e+01	1.00e+01
1	3.00e+02	0.00e+00	1.24e+00	2.61e-03	1.24e+00	7.00e+02	2.00e+00
2	5.12e+02	0.00e+00	1.37e+00	1.31e-05	1.37e+00	7.00e+02	4.00e-01

3	3.45e+03	0.00e+00	2.35e+00	3.08e-07	2.35e+00	7.00e+02	8.00e-02
3	6.55e+04	0.00e+00	4.95e+00	3.51e-08	8.09e+02	7.00e+02	1.60e-02
5	9.73e+05	0.00e+00	1.11e+01	2.06e-05	1.22e+05	7.00e+02	3.20e-03
6	8.21e+06	0.00e+00	2.70e+01	1.41e-02	1.23e+06	7.00e+02	6.40e-04
7	1.64e+07	0.00e+00	1.00e+02	7.07e-05	3.12e+06	7.00e+02	1.28e-04
8	1.83e+07	0.00e+00	1.13e+02	4.09e-04	6.79e+05	7.00e+02	2.56e-05
9	1.03e+08	0.00e+00	2.50e+02	2.87e-01	4.02e+06	7.00e+02	5.12e-06

TAB. 5.4: Résolution du problème HS15 par la méthode 1.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	9.09e+02	0.00e+00	3.00e+00	2.62e-03	1.92e+03	7.00e+01	1.00e+01
1	3.00e+02	0.00e+00	1.24e+00	2.61e-03	1.24e+00	7.00e+01	2.00e+00

TAB. 5.5: Résolution du problème HS15 par la méthode 2.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	3.00e-01	0.00e+00	1.20e+00	1.96e-03	1.20e+00	7.00e+01	1.00e+01
1	-2.37e-02	0.00e+00	8.17e-01	9.82e-06	8.17e-01	1.75e+01	2.00e+00
2	-1.23e+00	0.00e+00	1.90e+00	4.91e-08	1.90e+00	1.75e+01	4.00e-01
3	-3.08e+00	0.00e+00	3.57e+00	2.46e-10	3.57e+00	1.75e+01	8.00e-02
4	-2.63e+00	0.00e+00	2.82e+00	1.23e-12	2.82e+00	1.75e+01	1.60e-02
5	4.84e+00	0.00e+00	8.17e+00	6.14e-15	8.17e+00	1.75e+01	3.20e-03
6	1.16e+01	0.00e+00	1.87e+01	3.07e-17	1.87e+01	1.75e+01	6.40e-04
7	1.54e+01	0.00e+00	3.11e+01	1.53e-19	3.11e+01	1.75e+01	1.28e-04
8	8.98e+00	0.00e+00	5.20e+01	7.67e-22	5.20e+01	1.75e+01	2.56e-05
9	-1.08e+01	0.00e+00	8.54e+01	3.84e-24	8.54e+01	1.75e+01	5.12e-06
... suite page suivante...							

... suite de la page précédente...							
10	-2.88e+01	0.00e+00	1.17e+02	1.92e-26	1.17e+02	1.75e+01	1.02e-06
11	-5.01e+01	0.00e+00	1.51e+02	9.59e-29	1.51e+02	1.75e+01	2.05e-07
12	-6.78e+01	0.00e+00	1.83e+02	4.80e-31	1.83e+02	1.75e+01	4.10e-08
13	-8.95e+01	0.00e+00	2.17e+02	2.40e-33	2.17e+02	1.75e+01	8.19e-09
14	-1.06e+02	0.00e+00	2.48e+02	1.20e-35	2.48e+02	1.75e+01	1.64e-09
15	-1.29e+02	0.00e+00	2.81e+02	5.99e-38	2.81e+02	1.75e+01	3.28e-10
16	-1.45e+02	0.00e+00	3.11e+02	3.00e-40	3.11e+02	1.75e+01	6.55e-11
17	-1.69e+02	0.00e+00	3.44e+02	1.50e-42	3.44e+02	1.75e+01	1.31e-11
18	-1.86e+02	0.00e+00	3.75e+02	7.49e-45	3.75e+02	1.75e+01	2.62e-12
19	-2.09e+02	0.00e+00	4.09e+02	3.75e-47	4.09e+02	1.75e+01	5.24e-13
20	-2.25e+02	0.00e+00	4.39e+02	1.87e-49	4.39e+02	1.75e+01	1.05e-13
21	-2.48e+02	0.00e+00	4.72e+02	9.37e-52	4.72e+02	1.75e+01	2.10e-14
22	-2.65e+02	0.00e+00	5.02e+02	4.68e-54	5.02e+02	1.75e+01	4.19e-15
23	-2.88e+02	0.00e+00	5.36e+02	2.34e-56	5.36e+02	1.75e+01	8.39e-16
24	-3.04e+02	0.00e+00	5.66e+02	1.17e-58	5.66e+02	1.75e+01	1.68e-16
25	-3.28e+02	0.00e+00	5.99e+02	5.85e-61	5.99e+02	1.75e+01	3.36e-17
26	-3.44e+02	0.00e+00	6.29e+02	2.93e-63	6.29e+02	1.75e+01	6.71e-18
27	-3.68e+02	0.00e+00	6.63e+02	1.46e-65	6.63e+02	1.75e+01	1.34e-18
28	-3.84e+02	0.00e+00	6.93e+02	7.32e-68	6.93e+02	1.75e+01	2.68e-19
29	-4.08e+02	0.00e+00	7.26e+02	3.66e-70	7.26e+02	1.75e+01	5.37e-20
30	-4.25e+02	0.00e+00	7.57e+02	1.83e-72	7.57e+02	1.75e+01	1.07e-20
31	-4.47e+02	0.00e+00	7.90e+02	9.15e-75	7.90e+02	1.75e+01	2.15e-21
32	-4.63e+02	0.00e+00	8.20e+02	4.57e-77	8.20e+02	1.75e+01	4.29e-22
33	-4.87e+02	0.00e+00	8.53e+02	2.29e-79	8.53e+02	1.75e+01	8.59e-23
34	-5.05e+02	0.00e+00	8.84e+02	1.14e-81	8.84e+02	1.75e+01	1.72e-23
... suite page suivante...							

... suite de la page précédente...							
35	-5.27e+02	0.00e+00	9.18e+02	5.72e-84	9.18e+02	1.75e+01	3.44e-24
36	-5.44e+02	0.00e+00	9.49e+02	2.86e-86	9.49e+02	1.75e+01	6.87e-25
37	-5.67e+02	0.00e+00	9.82e+02	1.43e-88	9.82e+02	1.75e+01	1.37e-25
38	-5.83e+02	0.00e+00	1.01e+03	7.15e-91	1.01e+03	1.75e+01	2.75e-26
39	-6.07e+02	0.00e+00	1.05e+03	3.57e-93	1.05e+03	1.75e+01	5.50e-27
40	-6.24e+02	0.00e+00	1.08e+03	1.79e-95	1.08e+03	1.75e+01	1.10e-27
41	-6.47e+02	0.00e+00	1.11e+03	8.93e-98	1.11e+03	1.75e+01	2.20e-28
42	-6.62e+02	0.00e+00	1.14e+03	4.47e-100	1.14e+03	1.75e+01	4.40e-29
43	-6.87e+02	0.00e+00	1.17e+03	2.23e-102	1.17e+03	1.75e+01	8.80e-30
44	-7.04e+02	0.00e+00	1.20e+03	1.12e-104	1.20e+03	1.75e+01	1.76e-30
45	-7.27e+02	0.00e+00	1.24e+03	5.58e-107	1.24e+03	1.75e+01	3.52e-31
46	-7.43e+02	0.00e+00	1.27e+03	2.79e-109	1.27e+03	1.75e+01	7.04e-32

TAB. 5.6: Résolution du problème SIMPLPA par la méthode 1.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	3.00e-01	0.00e+00	1.20e+00	1.96e-03	1.20e+00	7.00e+01	1.00e+01
1	-2.37e-02	0.00e+00	9.02e-01	9.82e-06	8.17e-01	1.75e+01	2.00e+00

TAB. 5.7: Résolution du problème SIMPLPA par la méthode 2.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	1.61e+02	0.00e+00	3.00e+00	1.31e-03	9.17e+00	7.00e+01	1.00e+01
... suite page suivante...							

... suite de la page précédente...							
1	1.26e+02	0.00e+00	2.36e+00	1.31e-03	1.43e+01	7.00e+01	2.00e+00
2	6.45e+01	0.00e+00	6.94e-01	6.54e-06	1.12e+01	4.36e+00	4.00e-01
3	3.79e+01	0.00e+00	2.89e-01	3.27e-08	6.32e+00	4.36e+00	8.00e-02
4	3.10e+01	0.00e+00	1.03e+00	1.64e-10	2.32e+00	4.36e+00	1.60e-02
5	3.26e+01	0.00e+00	1.64e+00	8.18e-13	4.43e+00	4.36e+00	3.20e-03
6	3.51e+01	0.00e+00	1.92e+00	4.09e-15	6.69e+00	4.36e+00	6.40e-04
7	3.58e+01	0.00e+00	1.99e+00	2.05e-17	7.19e+00	4.36e+00	1.28e-04
8	3.60e+01	0.00e+00	2.00e+00	1.02e-19	7.26e+00	4.36e+00	2.56e-05
9	3.60e+01	0.00e+00	2.00e+00	7.42e-20	7.27e+00	4.36e+00	5.12e-06
10	3.60e+01	0.00e+00	2.00e+00	7.40e-20	7.27e+00	4.36e+00	1.02e-06
11	3.60e+01	0.00e+00	2.00e+00	7.40e-20	7.27e+00	4.36e+00	2.05e-07
12	3.60e+01	0.00e+00	2.00e+00	7.39e-20	7.27e+00	4.36e+00	4.10e-08
13	3.60e+01	0.00e+00	2.00e+00	7.38e-20	7.27e+00	4.36e+00	8.19e-09
14	3.60e+01	0.00e+00	2.00e+00	7.38e-20	7.27e+00	4.36e+00	1.64e-09
15	3.60e+01	0.00e+00	2.00e+00	7.38e-20	7.27e+00	4.36e+00	3.28e-10
16	3.60e+01	0.00e+00	2.00e+00	7.38e-20	7.27e+00	4.36e+00	6.55e-11
17	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	1.31e-11
18	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	2.62e-12
19	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	5.24e-13
20	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	1.05e-13
21	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	2.10e-14
22	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	4.19e-15
23	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	8.39e-16
24	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	1.68e-16
25	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	3.36e-17
... suite page suivante...							

... suite de la page précédente...							
26	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	6.71e-18
27	3.60e+01	0.00e+00	2.00e+00	7.37e-20	7.27e+00	4.36e+00	1.34e-18
28	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	2.68e-19
29	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	5.37e-20
30	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	1.07e-20
31	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	2.15e-21
32	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	4.29e-22
33	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	8.59e-23
34	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	1.72e-23
35	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	3.44e-24
36	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	6.87e-25
37	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	1.37e-25
38	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	2.75e-26
39	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	5.50e-27
40	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	1.10e-27
41	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	2.20e-28
42	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	4.40e-29
43	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	8.80e-30
44	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	1.76e-30
45	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	3.52e-31
46	3.60e+01	0.00e+00	2.00e+00	7.36e-20	7.27e+00	4.36e+00	7.04e-32

TAB. 5.8: Résolution du problème ZECEVC4 par la méthode 1.

iter	obj	$ h $	$ g + s $	s_{\min}	KKT	r	μ
0	1.61e+02	0.00e+00	3.00e+00	1.31e-03	9.17e+00	7.00e+01	1.00e+01
1	1.26e+02	0.00e+00	2.36e+00	1.31e-03	1.43e+01	7.00e+02	2.00e+00
2	6.96e+01	0.00e+00	1.20e+00	6.54e-06	1.32e+01	2.71e+00	4.00e-01
3	5.70e+01	0.00e+00	1.26e+00	4.65e-06	1.14e+01	6.47e-01	8.00e-02
4	5.00e+01	0.00e+00	9.93e-01	3.30e-06	9.94e+00	6.47e-01	1.60e-02
5	4.55e+01	0.00e+00	3.95e-01	2.35e-06	9.00e+00	6.47e-01	3.20e-03
6	4.10e+01	0.00e+00	2.70e-01	1.22e-06	7.89e+00	6.47e-01	6.40e-04
7	4.02e+01	0.00e+00	7.95e-02	1.11e-06	7.92e+00	1.60e-01	1.28e-04
8	3.99e+01	0.00e+00	4.16e-02	1.08e-06	7.89e+00	3.97e-02	2.56e-05
9	3.97e+01	0.00e+00	3.05e-02	1.05e-06	7.83e+00	3.97e-02	5.12e-06
10	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.83e+00	9.28e-03	1.02e-06
11	3.96e+01	0.00e+00	3.31e-02	1.05e-06	7.82e+00	4.66e-04	2.05e-07
12	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.82e+00	4.66e-04	4.10e-08
13	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.82e+00	4.66e-04	8.19e-09
14	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.82e+00	6.99e-05	1.64e-09
15	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.82e+00	6.99e-05	3.28e-10
16	3.96e+01	0.00e+00	3.30e-02	1.05e-06	7.82e+00	6.99e-05	6.55e-11
17	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	1.31e-11
18	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	2.62e-12
19	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	5.24e-13
20	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	1.05e-13
21	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	2.10e-14
22	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	4.19e-15
23	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	8.39e-16
24	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	1.68e-16
... suite page suivante...							

... suite de la page précédente...							
25	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	3.36e-17
26	3.96e+01	0.00e+00	3.29e-02	1.05e-06	7.82e+00	6.99e-05	6.71e-18

TAB. 5.9: Résolution du problème ZECEVC4 par la méthode 2.

5.8.2 Choix d'une fonction de mérite

Dans l'étude suivante nous testons quatre formes de la fonction de mérite :

1. La fonction de mérite de type l_2 :

$$M_\sigma(x, s) = \phi(x, s) + \sigma(\|g(x) + s\|_2 + \|h(x)\|_2) \quad (5.30)$$

où σ est un réel et ϕ est l'objectif du modèle pénalisé. C'est cette fonction de mérite que nous avons utilisé pour réaliser les expériences ci-dessus.

2. La fonction de mérite de type l_1 :

$$M_\sigma(x, s) = \phi(x, s) + \sigma(\|g(x) + s\|_1 + \|h(x)\|_1) \quad (5.31)$$

3. La fonction de mérite de type l_∞ :

$$M_\sigma(x, s) = \phi(x, s) + \sigma(\|g(x) + s\|_\infty + \|h(x)\|_\infty) \quad (5.32)$$

4. La fonction de mérite pondérée que nous avons proposé au chapitre (3)

$$M(x, s, \sigma_f, \sigma^h, \sigma^g) = \sigma_f \phi(x, s) + \sum_{i=1}^p \sigma_i^h |h_i(x)| + \sum_{i=1}^m \sigma_i^g |g_i(x) + s_i| \quad (5.33)$$

où $\sigma_f \in \mathbb{R}_+$ joue le rôle d'équilibrage du poids de l'objectif par rapport aux contraintes, σ^h équilibre intrinsèquement les rapport aux contraintes d'égalité, et σ^g équilibre intrinsèquement les contraintes d'inégalité suivant le schéma suivant. Soit $I \subset \{1, 2, \dots, m\}$ l'ensemble des indices pour lesquels $|g(x) + s| > 0$.

$$\sigma_i^g = \begin{cases} \frac{|g_i(x) + s_i|}{\sum_{i \in I} |g_i(x) + s_i| + \sum_{i=1}^p |h_i(x)|} & \text{si } i \in I \\ \sigma_c = \frac{\alpha}{\sum_{i \in I} |g_i(x) + s_i| + \sum_{i=1}^p |h_i(x)|} & \text{si } i \notin I \end{cases} \quad (5.34)$$

$$\sigma_i^h = \begin{cases} \frac{|h_i(x)|}{\sum_{i \in I} |g_i(x) + s_i| + \sum_{i=1}^p |h_i(x)|} & \text{si } h_i(x) \neq 0 \\ \sigma_c & \text{si } h_i(x) = 0 \end{cases} \quad (5.35)$$

où

$$\alpha = \min\{\min_{i \in I} |g_i(x) + s_i|, \min_{i=1,2,\dots,p} |h_i(x)|\} > 0 \quad (5.36)$$

Dans le cadre de l'utilisation de la fonction de mérite de type l_2 (5.30), il est commun de poser

$$pred(d) = -q(d) + \sigma v pred(v) \quad (5.37)$$

où d est la solution du sous-problème quadratique dont l'objectif est la forme quadratique

$$q(d) = \frac{1}{2} d^t W d + \nabla \phi^t d$$

et $v pred(v) = \|c(x, s)\|_2 - \|Jv + c(x, s)\|_2$, où v est la solution optimale du problème vertical.

On a $\|c(x, s)\|_2^2 = c^t c$ et $\|Jv + c(x, s)\|_2^2 = v^t J^t J v + 2c^t J v + c^t c$.

Or v est une solution du problème vertical donc $v^t J^t J v + 2c^t J v \leq 0$ d'où

$\|Jv+c(x, s)\|_2^2 = v^t J^t Jv + 2c^t Jv + c^t c \leq c^t c$, ainsi $\|Jv+c(x, s)\|_2 \leq \|c(x, s)\|_2$
d'où $vpred(v) \geq 0$.

Si $vpred(v) = 0$ alors on peut considérer que $v = 0$ puisque la minimisation de l'objectif du problème vertical aurait produit un vecteur qui n'améliore pas l'objectif plus que le vecteur $v = 0$, ceci se produira s'il n'y a pas lieu d'inconsistance des contraintes du sous-problème quadratique.

D'autre part

$$\begin{aligned} q(d) &= \frac{1}{2}d^t Wd + \nabla\phi^t d \\ &= \frac{1}{2}(v+w)^t W(v+w) + \nabla\phi^t(v+w) \\ &= \frac{1}{2}w^t Ww + (\nabla\phi + W^t v)^t w + \frac{1}{2}v^t Wv + \nabla\phi^t v \\ &= \frac{1}{2}u^t Hu + \nabla\phi_z^t u + \frac{1}{2}v^t Wv + \nabla\phi^t v \end{aligned}$$

On pose

$$hpred(u) = -\left(\frac{1}{2}u^t Hu + \nabla\phi_z^t u\right)$$

où u est une solution du problème horizontal réduit, d'où $hpred(u) \geq 0$.

On note $\chi(v) = \frac{1}{2}v^t Wv + \nabla\phi^t v$ ainsi dans l'expression

$$pred(d) = hpred(d) + \sigma vpred(v) - \left(\frac{1}{2}v^t Wv + \nabla\phi^t v\right)$$

les valeurs $hpred(d)$ et $vpred(v)$ sont positives, or on ne peut rien affirmer au sujet de $\chi(v)$, qui est tantôt positif, tantôt négatif.

Il est également commun de renforcer la positivité de $pred(d)$ en cherchant un paramètre σ tel que

$$pred(d) \geq \zeta \sigma vpred(v) \tag{5.38}$$

où $\zeta \in]0, 1[$ ce qui implique que

$$hpred(d) + \sigma vpred(v) - \chi(v) \geq \zeta \sigma vpred(v) \quad (5.39)$$

c'est-à-dire

$$\sigma \geq \frac{\chi(v) - hpred(d)}{(1 - \zeta)vpred(v)} \quad (5.40)$$

Ainsi à chaque itération, nous choisissons le paramètre σ selon la règle donnée par l'algorithme 17.

Procédure Règle pour choisir σ (règle 1)

Si	($vpred(v) > 0$)	Alors
		$\sigma_k = \max(\sigma_{k-1}, \frac{\chi(v) - hpred(d)}{(1 - \zeta)vpred(v)})$
Sinon		
		$\sigma_k = \sigma_{k-1}$
Fin Si		
Fin		

Algorithme 17: Règle pour choisir le paramètre de pénalisation (règle 1)

Dans cette étude numérique, nous commençons par tester les réactions des deux méthodes lorsqu'on change de fonction de mérite. Les tableaux 5.10, 5.11 et 5.12 rassemblent les différents résultats. Nous avons appliqué les deux méthodes pour résoudre une série de problèmes de la collection CUTER, dont quelques uns proviennent de l'étude précédente (Tab. 5.2). L'introduction de la notion de réduction prédite a, en général, amélioré les résultats, bien que des exceptions sont remarquées. Par exemple, le problème MIFFLIN2 avait été résolu (dans l'étude précédente reportée sur la figure 5.2) en 9 itérations avec la méthode 1 et 4 itérations avec la méthode 2. Dans la présente étude, la résolution du même problème nécessite 2 itérations avec la méthode 1 et

3 itérations pour la méthode 2 mais avec un défaut de réalisabilité qui varie entre 4.83 et 51.86 (!!) alors que ces valeurs étaient de 0.639 et 0.819 lors de l'étude précédente. En effet, l'expression $ared(d) > \zeta_{pred}(d)$ contraint l'algorithme à rejeter les pas de déplacement de mauvaise qualité (70 rejets pour cet exemple). En revanche, nous remarquons que le problème BT13 réagit mieux cette fois, avec la méthode 2 qu'avec la méthode 1, bien qu'il reste toujours loin d'être résolu en 46 itérations pour la méthode 1 et 7 pour la méthode 2, avec une valeur exorbitante de cc : entre 24293.03 et 51813.97. Ce que nous avons changé pour avoir ces résultats c'est le rayon initial r_0 (nous l'avons divisé par 10). Ces résultats vont dans la même direction de plusieurs études récentes (voir par exemple Sartnear [76]) qui montrent que les algorithmes à région de confiance sont sensibles au rayon initial.

Problème	méthode 1			méthode 2		
	L 1	L 2	L inf	L 1	L 2	L inf
3PK	9938,84	9874,18	9938,84	9938,84	9900,4	9938,84
AIRCRFTA	0	0	0	0	0	0
AIRCRFTB	1059408,53	2872,69	2872,69	1059408,53	2872,69	2872,69
AKIVA	6,17	6,17	6,17	6,17	6,17	6,17
ALLINIT	10,04	7,04E+009	5,44E+009	13	12,55	10,43
ALLINITU	5,74	5,74	5,74	5,74	5,74	5,74
ALSOTAME	1,06	1,01	1,01	1,06	1,01	1,01
BARD	0,02	0,02	0,02	0,02	0,02	0,02
BEALE	0,02	0,02	0,02	0,02	0,02	0,02
BIGGS6	4978857,05	4978857,05	4978857,05	4978857,05	4978857,05	4978857,05
BOOTH	0	0	0	0	0	0
BOX2	1,88	1,88	1,88	1,88	1,88	1,88
... suite page suivante...						

... suite de la page précédente...						
BOX3	0	0	0	0	0	0
BQP1VAR	0,2	0,2	0,2	0,28	0,28	0,28
BQPGASIM	0	0,01	0	0	0	0
BRKMCC	0,17	0,17	0,17	0,17	0,17	0,17
BT10	-2	-2	-2	-2	-2	-2
BT11	13,65	8,8	1	13,65	1	1
BT12	5	5,93	5,89	5	5,93	5,89
BT13	227,93	230,71	227,93	221,48	238,78	238,77
BT1	59,61	3226,53	59,6	59,61	3226,53	59,6
BT2	106,04	81	106,04	106,04	81	106,04
BT4	-18,61	-18,62	-18,62	-18,61	-18,62	-18,62
BT5	975,96	965,5	965,76	975,96	965,43	965,76
BT9	-2,16	-2,08	-2	-2,16	-2,08	-2
CAMEL6	0,02	0,02	0,02	0,02	2,6	0,02
CANTILVR	0,31	-0,48	0,31	0,31	0,31	0,31
CLIFF	0,2	0,2	0,2	0,2	0,2	0,2
CLUSTER	0	0	0	0	0	0
COSHFUN	0	0,56	0,6	0,04	0,16	0
CUBENE	0	0	0	0	0	0
DECONVB	110,35	110,35	110,35	110,34	110,35	110,35
DECONVU	28,99	28,99	28,99	28,99	28,99	28,99
DENSCHNA	0	0	0	0	0	0
DENSCHNB	0	0	0	0	0	0
DENSCHNC	0	0	0	0	0	0
... suite page suivante...						

... suite de la page précédente...						
DENSCHNE	1	1	1	1	1	1
DENSCHNF	0	0	0	0	0	0
EG1	0	33516,69	31663,88	0	0,1	0,07
EXTRASIM	1,13	1	1,13	0,53	0,53	0,53
GOTTFR	0	0	0	0	0	0
HART6	-2,41	-2,41	-2,41	-2,41	-0,76	-2,41
HATFLDA	1,44	1,44	1,44	0,85	0,85	0,85
HELIX	1871,25	1871,25	1871,25	1871,25	1871,25	1871,25
HILBERTA	0	0	0	0	0	0
HIMMELBA	0	0	0	0	0	0
HIMMELBB	0	0	0	0	0	0
HIMMELBC	0	0	0	0	0	0
HIMMELBE	0	0	0	0	0	0
HIMMELBG	0	0	0	0	0	0
HIMMELBH	-1	-1	-1	-1	-1	-1
HIMMELP1	86	78,48	78,48	86	79,3	79,28
HIMMELP2	86	85,96	85,96	86	86	86
HS105	1253,77	1214,34	1222,04	1253,77	1288,39	1300,28
HS107	4853,33	-4,51E+006	-1,43E+009	4853,33	-4,51E+008	-1,43E+009
HS10	-20	-0,01	-20	-20	-20	-20
HS110	-43,14	-43,14	-43,14	-43,14	-43,14	-43,14
HS111LNP	-42,95	-249,76	-40,66	-42,95	-249,77	-40,66
HS111	-22,52	-21,01	-21,01	-22,52	-21,01	-21,01
HS1	23,16	4,08E+011	23,16	23,16	4,03E+011	23,16
... suite page suivante...						

... suite de la page précédente...						
HS22	1	1	1	2,34	2,34	2,34
HS25	32,84	32,84	32,84	32,84	32,84	32,84
HS27	1,82	6,2	0,02	1,82	6,17	0,02
HS29	-1	-1	-1	-1	-1	-1
HS30	3	4,14	4,14	3	3	3
HS31	19	9,95	9,95	19	19	19
HS33	-3	20,16	-315,99	-3	-3	-3
HS34	-0,04	-1,08	-1,08	0	0	0
HS37	-999,98	-999,67	-999,67	-999,98	-999,67	-999,67
HS39	-2,16	-2,08	-2	-2,16	-2,08	-2
HS41	-6	-1007,19	-1007,83	0,85	1,01	-30,02
HS42	76,86	76,86	76,86	76,86	76,86	76,86
HS45	1,73	-2,53	-2,53	1,73	1,73	1,73
HS46	3,34	3,34	3,34	3,34	3,34	3,34
HS47	20,74	20,85	20,85	20,74	20,85	20,85
HS4	3,79	3,68	3,79	3,31	3,31	3,31
HS5	-0,22	-0,22	-0,22	-0,22	-0,22	-0,22
HS60	1,49	1,45	2,55	1,49	1,45	2,55
HS62	-19119,85	-25698,3	-25698,3	-19119,85	-25698,3	-25698,3
HS66	-0,73	-0,73	-0,73	0,58	0,58	0,58
HS7	-1,89	-1,73	-1,89	-1,89	-1,73	-1,89
HS81	-0,5	-0,49	-4,82	-0,5	-0,49	-4,82
HS8	-1	-1	-1	-1	-1	-1
HYPICIR	0	0	0	0	0	0
... suite page suivante...						

... suite de la page précédente...						
KIWCRESC	0	0,01	0,04	0	0	0
KOWOSB	0	0	0	0	0	0
LSQFIT	1,02	24,48	24,48	1,02	1,02	1,02
MARATOSB	1	1	1	1	1	1
MARATOS	-1	-1	-1	-1	-1	-1
METHANB8	0	0	0	0	0	0
MIFFLIN2	0	5,45	5,45	0	0,12	0
PALMER5C	2,13	2,13	2,13	2,13	2,13	2,13
PORTFL1	0,02	0,03	0,03	0,03	0,03	0,03
PORTFL2	0,03	0,03	0,03	0,03	0,03	0,03
PORTFL3	0,04	0,04	0,04	0,04	0,04	0,04
PORTFL4	0,03	0,03	0,03	0,03	0,03	0,03
PORTFL6	0,03	0,03	0,03	0,03	0,03	0,03
POWELLSQ	0	0	0	0	0	0
PSPDOC	3,53	3,53	3,53	3,53	3,54	3,53
RECIPE	0	0	0	0	0	0
S308	0,77	0,77	0,77	0,77	0,77	0,77
SIMPLLPA	0,3	0,45	0,45	0,17	-0,22	-0,22
SIMPLLPB	0,25	4,83	4,83	0,25	0,25	0,25
SISSER	0	0	0	0	0	0
SUPERSIM	0	0	0	0,67	0	0
TRY-B	1,32	1,43	1,3	1,32	28,36	1,3
TWOBARS	1,41	0,9	0,89	1,11	1,22	1,41
ZECEVIC2	-1,3	-1,55	-3,51	0,12	0,12	-0,24
... suite page suivante...						

... suite de la page précédente...						
ZECEVIC3	294,1	161,76	161,76	294,1	287,03	244,77
ZECEVIC4	160,87	81,44	81,44	160,87	155,72	72,47
Somme	6039186,9	4,15E+011	4,20E+009	6039193,04	4,02E+011	1,23E+009

TAB. 5.10: Résultats en fonction de l'objectif pour les différentes fonctions de mérite.

Problème	méthode 1			méthode 2		
	L 1	L 2	L inf	L 1	L 2	L inf
3PK	0,72	0,71	0,72	0,72	0,72	0,72
AIRCRFTA	2,84	2,84	2,84	2,84	2,84	2,84
AIRCRFTB	0,01	1,37	1,37	0,01	1,37	1,37
AKIVA	0	0	0	0	0	0
ALLINIT	1	264,56	142,03	1,99	1	1
ALLINITU	0	0	0	0	0	0
ALSOTAME	0,84	0,84	0,84	0,84	0,84	0,84
BARD	0	0	0	0	0	0
BEALE	0	0	0	0	0	0
BIGGS6	0	0	0	0	0	0
BOOTH	1,07	1,07	1,07	1,07	1,07	1,07
BOX2	1,92E-005	1,92E-005	1,92E-005	1,92E-005	1,92E-005	1,92E-005
BOX3	0	0	0	0	0	0
BQP1VAR	0	0	0	0	0	0
BQPGASIM	1,50E-005	0,01	0	3,43E-007	5,34E-006	2,12E-005
... suite page suivante...						

... suite de la page précédente...						
BRKMCC	0	0	0	0	0	0
BT10	6	6	6	6	6	6
BT11	1,87	2,95	11,76	1,87	11,76	11,76
BT12	7,61	7,23	7,25	7,61	7,23	7,25
BT13	51842,07	51813,97	51842,07	18168,95	24293,03	24293,2
BT1	0,59	32,26	0,59	0,59	32,26	0,59
BT2	6514,69	11001,76	6514,69	6514,69	11001,76	6514,69
BT4	0	0	0	0	0	0
BT5	12,19	0,47	0,99	12,19	2,77	0,99
BT9	10,32	10,03	10	10,32	10,03	10
CAMEL6	0	0	0	0	0	0
CANTILVR	124	65,81	124	124	124	124
CLIFF	0	0	0	0	0	0
CLUSTER	1,59E-005	1,74E-005	1,97E-005	1,59E-005	1,74E-005	1,97E-005
COSHFUN	1	5,72	6,01	1,03	1,47	1
CUBENE	0,45	0,58	0,43	0,45	0,58	0,43
DECONVB	0,01	0,01	0,01	0	0	0
DECONVU	0	0	0	0	0	0
DENSCHNA	0	0	0	0	0	0
DENSCHNB	0	0	0	0	0	0
DENSCHNC	0	0	0	0	0	0
DENSCHNE	0	0	0	0	0	0
DENSCHNF	0	0	0	0	0	0
EG1	1	38,37	38,65	1	0,95	0,94
... suite page suivante...						

... suite de la page précédente...						
EXTRASIM	1,22	2	1,22	0,47	0,47	0,47
GOTTFR	1,79	1,79	1,79	1,79	1,79	1,79
HART6	0	0	0	0	0	0
HATFLDA	0	0	0	0	0	0
HELIX	0	0	0	0	0	0
HILBERTA	0	0	0	0	0	0
HIMMELBA	0,01	9,30E-009	9,30E-009	0,01	9,30E-009	9,30E-009
HIMMELBB	0	0	0	0	0	0
HIMMELBC	0	2,05E-007	2,05E-007	0	2,05E-007	2,05E-007
HIMMELBE	0,39	0,38	0,39	0,39	0,38	0,39
HIMMELBG	0	0	0	0	0	0
HIMMELBH	0	0	0	0	0	0
HIMMELP1	0	0	0	0	0	0
HIMMELP2	0	0,11	0,11	0	0	0
HS105	5	0	0	5	0	0
HS107	0,8	320,7	4842,74	0,8	1436,89	4842,74
HS10	599	629,18	599	599	598,73	599
HS110	0	0	0	0	0	0
HS111LNP	0,4	9,07	0,4	0,4	9,07	0,4
HS111	1,11	1,3	1,3	1,11	1,3	1,3
HS1	0	41,32	0	0	44,33	0
HS22	2	2	2	0,16	0,16	0,16
HS25	0	2,72E-006	1,33E-010	0	1,33E-010	1,33E-010
HS27	1	7	1,04	1	7	1,04
... suite page suivante...						

... suite de la page précédente...						
HS29	0	0	0	0	0	0
HS30	0	0,13	0,13	0	2,88E-008	1,15E-007
HS31	2,22E-016	1,01	1,01	2,22E-016	2,22E-016	2,22E-016
HS33	0	16,59	16,6	0	9,05E-006	2,06E-005
HS34	4,2	8,53	8,53	0	0	0
HS37	0	0	0	0	0	0
HS39	10,32	10,03	10	10,32	10,03	10
HS41	8	39,66	39,67	3,03	2,62	6,88
HS42	6,04E-007	9,67E-007	6,04E-007	6,04E-007	9,67E-007	6,04E-007
HS45	1	1,5	1,5	0,99	1	1
HS46	5,28E-010	8,99E-007	8,99E-007	5,28E-010	8,99E-007	8,99E-007
HS47	6,71E-009	0,01	0,01	6,71E-009	0,01	0,01
HS4	0	0	0	0	0	0
HS5	0	0	0	0	0	0
HS60	2,34	4,81	2,97	2,34	4,81	2,97
HS62	3,89E-016	2,78E-017	2,78E-017	3,89E-016	6,94E-017	2,78E-017
HS66	6,53	6,53	6,53	0	0	0
HS7	4,75	3,96E-005	4,75	4,75	3,96E-005	4,75
HS81	4	4	4,03	4	4	4,03
HS8	0,01	0	0	0,01	0	0
HYP CIR	0	8,82E-009	8,45E-009	0	8,82E-009	8,45E-009
KIWCRESC	4,25	4,25	4,25	4,25	4,25	4,25
KOWOSB	0	0	0	0	0	0
LSQFIT	0	4,52	4,52	0	9,31E-005	9,31E-005
... suite page suivante...						

... suite de la page précédente...						
MARATOSB	0	0	0	0	0	0
MARATOS	9,28E-006	2,53E-006	1,33E-005	9,28E-006	2,53E-006	1,33E-005
METHANB8	1,02	1,02	1,02	1,02	1,02	1,02
MIFFLIN2	4,75	51,86	51,91	4,75	4,83	4,75
PALMER5C	0	0	0	0	0	0
PORTFL1	7,96E-011	4,00E-010	4,00E-010	7,96E-011	4,00E-010	4,00E-010
PORTFL2	7,96E-011	4,00E-010	4,00E-010	7,96E-011	4,00E-010	4,00E-010
PORTFL3	7,96E-011	4,00E-010	4,00E-010	7,96E-011	4,00E-010	4,00E-010
PORTFL4	7,96E-011	4,00E-010	4,00E-010	7,96E-011	4,00E-010	4,00E-010
PORTFL6	7,96E-011	4,00E-010	4,00E-010	7,96E-011	4,00E-010	4,00E-010
POWELLSQ	0,12	0,12	0,12	0,12	0,12	0,12
PSPDOC	0	0	0	0	0	0
RECIPE	0,11	0,11	0,11	0,11	0,11	0,11
S308	0	0	0	0	0	0
SIMPLLPA	1,02	1,57	1,57	1,13	0,9	0,9
SIMPLLPB	0,9	10,24	10,24	0,9	0,9	0,9
SISSER	0	0	0	0	0	0
SUPERSIM	2	2	2	1,11E-016	2	2
TRY-B	4,61	4,74	4,57	4,61	27,36	4,57
TWOBARS	0,58	0,73	0,74	0,63	0,59	0,58
ZECEVIC2	0,08	0,03	0	0,1	0,1	0,04
ZECEVIC3	1,01	1,06	1,06	1,01	1,01	1,01
ZECEVIC4	3	5,02	5,02	3	2,94	2,95
Moyenne	528,84	584,55	583,96	228,01	346,6	335,16
... suite page suivante...						

... suite de la page précédente...

TAB. 5.11: Résultats en fonction du défaut de réalisabilité
pour les différentes fonctions de mérite

Problème	méthode 1			méthode 2		
	L 1	L 2	L inf	L 1	L 2	L inf
3PK	1	46	1	1	1	1
AIRCRFTA	0	0	0	0	0	0
AIRCRFTB	1	46	46	1	46	46
AKIVA	46	46	46	46	46	46
ALLINIT	1	46	46	1	3	3
ALLINITU	29	29	29	29	29	29
ALSOTAME	1	1	1	1	1	1
BARD	46	46	46	46	46	46
BEALE	46	46	46	46	46	46
BIGGS6	46	46	46	46	46	46
BOOTH	46	14	14	46	14	14
BOX2	46	46	46	46	46	46
BOX3	46	46	46	46	46	46
BQP1VAR	1	46	1	2	46	46
BQPGASIM	2	46	46	2	46	46
BRKMCC	46	46	46	46	46	46
BT10	0	46	46	0	46	46
BT11	19	10	46	19	46	46
... suite page suivante...						

... suite de la page précédente...						
BT12	46	46	46	46	46	46
BT13	4	46	7	19	7	7
BT1	2	46	46	2	46	46
BT2	46	0	46	46	46	46
BT4	0	46	46	0	46	46
BT5	46	43	46	46	46	46
BT9	2	46	46	2	46	46
CAMEL6	1	1	1	1	1	1
CANTILVR	0	46	46	0	46	46
CLIFF	46	46	46	46	46	46
CLUSTER	46	46	46	46	46	46
COSHFUN	0	46	46	46	18	18
CUBENE	46	46	46	46	46	46
DECONVB	1	1	1	3	1	1
DECONVU	46	46	46	46	46	46
DENSCHNA	46	46	46	46	46	46
DENSCHNB	17	17	17	17	17	17
DENSCHNC	46	46	46	46	46	46
DENSCHNE	46	46	46	46	46	46
DENSCHNF	46	46	46	46	46	46
EG1	0	46	46	0	1	1
EXTRASIM	1	0	1	19	18	18
GOTTFR	1	1	1	1	1	1
HART6	1	46	46	1	46	46
... suite page suivante...						

... suite de la page précédente...						
HATFLDA	1	46	1	1	1	1
HELIX	46	46	46	46	46	46
HILBERTA	21	21	21	21	21	21
HIMMELBA	46	41	41	46	41	41
HIMMELBB	46	46	46	46	46	46
HIMMELBC	46	46	46	46	46	46
HIMMELBE	46	14	17	46	17	17
HIMMELBG	16	16	16	16	16	16
HIMMELBH	21	21	21	21	21	21
HIMMELP1	0	4	4	0	4	4
HIMMELP2	0	2	2	0	1	1
HS105	0	46	46	0	46	46
HS107	0	23	46	0	46	46
HS10	0	46	0	0	0	0
HS110	1	1	1	1	1	1
HS111LNP	3	1	2	3	2	2
HS111	1	0	0	1	0	0
HS1	1	46	1	1	1	1
HS22	0	0	0	1	1	1
HS25	0	4	1	0	1	1
HS27	4	1	46	4	46	46
HS29	1	1	1	1	1	1
HS30	0	4	4	0	1	1
HS31	0	1	1	0	0	0
... suite page suivante...						

... suite de la page précédente...						
HS33	0	46	46	0	2	2
HS34	1	2	2	1	3	3
HS37	6	46	46	6	46	46
HS39	2	46	46	2	46	46
HS41	0	46	46	26	8	8
HS42	2	2	2	2	2	2
HS45	0	1	1	6	46	46
HS46	0	1	1	0	1	1
HS47	0	46	46	0	46	46
HS4	1	46	1	1	1	1
HS5	1	1	1	1	1	1
HS60	46	38	46	46	46	46
HS62	1	0	0	1	0	0
HS66	1	1	1	1	10	10
HS7	46	38	46	46	46	46
HS81	0	46	46	0	46	46
HS8	46	46	46	46	46	46
HYP CIR	46	37	40	46	40	40
KIWCRESC	0	19	18	0	17	17
KOWOSB	46	46	46	46	46	46
LSQFIT	0	2	2	0	46	46
MARATOSB	46	46	46	46	46	46
MARATOS	10	46	46	10	46	46
METHANB8	0	0	0	0	0	0
... suite page suivante...						

... suite de la page précédente...						
MIFFLIN2	0	3	3	0	19	19
PALMER5C	46	46	46	46	46	46
PORTFL1	1	0	0	1	0	0
PORTFL2	1	0	0	1	0	0
PORTFL3	1	0	0	1	0	0
PORTFL4	1	0	0	1	0	0
PORTFL6	1	0	0	1	0	0
POWELLSQ	46	46	46	46	46	46
PSPDOC	1	46	1	1	1	1
RECIPE	46	2	2	46	2	2
S308	24	24	24	24	24	24
SIMPLLPA	0	3	3	21	20	20
SIMPLLPB	0	46	46	0	1	1
SISSER	19	19	19	19	19	19
SUPERSIM	0	46	46	46	46	46
TRY-B	46	46	46	46	46	46
TWOBARS	0	46	46	4	46	46
ZECEVIC2	1	2	46	1	2	2
ZECEVIC3	0	46	46	0	46	46
ZECEVIC4	0	46	46	0	46	46
Moyenne	16,02	28,4	27,17	17,68	25,89	25,89

TAB. 5.12: Résultats en fonction du nombre d'itérations pour les différentes fonctions de mérite.

Pour la fonction de mérite pondérée nous avons obtenu les résultats suivants :

Problème	méthode 1			méthode 2		
	obj	cc	iter	obj	cc	iter
3PK	9869.154129	0.711381	46	9914.845457	0.715770	46
AIRCRFTB	2872.689084	1.372437	46	2872.689084	1.372437	46
AKIVA	6.169897	0.000000	46	6.169897	0.000000	46
ALLINITU	5.744385	0.000000	29	5.744385	0.000000	29
BARD	0.016289	0.000000	46	0.016289	0.000000	46
BEALE	0.020053	0.000000	46	0.020053	0.000000	46
BIGGS6	4.97E+06	0.000000	46	4.97E+06	0.000000	46
BOX2	1.884491	0.000019	46	1.884491	0.000019	46
BOX3	0.000543	0.000000	46	0.000543	0.000000	46
BQP1VAR	-0.250000	0.500000	46	-0.250000	0.500000	46
BQPGASIM	0.123187	0.046227	46	0.123187	0.046227	46
BRKMCC	0.169048	0.000000	46	0.169048	0.000000	46
BT11	4.010501	2.552912	46	4.010501	2.552912	46
BT4	-65.177006	1.809396	46	-65.177006	1.809396	46
BT5	976.775907	12.698661	25	976.775907	12.698661	25
CAMEL6	0.016548	0.000000	1	0.016548	0.000000	1
CLIFF	0.201062	0.000000	46	0.201062	0.000000	46
DECONVB	2322.859476	1.624325	7	110.319736	0.000875	46
DECONVU	28.992610	0.000000	46	28.992610	0.000000	46
DENSCHNA	0.000000	0.000000	46	0.000000	0.000000	46
DENSCHNB	0.000000	0.000000	17	0.000000	0.000000	17
DENSCHNC	0.000000	0.000000	46	0.000000	0.000000	46
... suite page suivante...						

... suite de la page précédente...						
DENSCHNE	0.998845	0.000000	46	0.998845	0.000000	46
DENSCHNF	0.000000	0.000000	46	0.000000	0.000000	46
EXPFITB	92.219733	0.003201	3	92.219733	0.003201	4
EXTRASIM	1.000001	1.999991	3	1.000000	1.999999	16
GOTTFR	0.000000	2.375000	46	0.000000	2.375000	46
HELIX	1871.250300	0.000000	46	1871.250300	0.000000	46
HILBERTA	0.000000	0.000000	21	0.000000	0.000000	21
HIMMELBB	0.000156	0.000000	46	0.000156	0.000000	46
HIMMELBC	0.000000	9.000000	1	0.000000	9.000000	1
HIMMELBE	0.000000	2.200000	46	0.000000	2.200000	46
HIMMELBG	0.000000	0.000000	16	0.000000	0.000000	16
HIMMELBH	-1.000000	0.000000	21	-1.000000	0.000000	21
HIMMELP1	75.763740	0.000000	4	79.312589	0.000000	1
HIMMELP2	79.312781	0.000000	2	79.312781	0.000000	27
HS105	1290.077170	78.030050	46	1234.646191	0.000000	46
HS110	-45.023446	0.000000	8	-45.023446	0.000000	8
HS1	23.164300	0.000000	3	23.164300	0.000000	3
HS25	32.835000	0.000006	4	32.835000	0.000000	1
HS27	0.035408	0.161887	46	0.035408	0.161887	46
HS29	-0.017472	0.000000	1	-0.017472	0.000000	1
HS31	681.944982	7.582547	46	18.858175	0.007743	46
HS42	14.000005	0.999997	5	14.000005	0.999997	5
HS46	3.337630	0.000004	46	3.337630	0.000004	46
HS4	3.784809	0.000000	46	3.314164	0.000000	46
... suite page suivante...						

... suite de la page précédente...						
HS5	-1.913219	0.000000	20	-1.913219	0.000000	20
HS60	1.906673	2.027768	46	1.906673	2.027768	46
HS7	-1.894876	4.746544	46	-1.894876	4.746544	46
HS81	-0.480121	4.000132	46	-0.480121	4.000132	46
KOWOSB	0.000509	0.000000	46	0.000509	0.000000	46
LSQFIT	1.078157	0.274783	2	1.122431	0.274807	7
MANCINO	3.58E+11	0.000000	46	3.58E+11	0.000000	46
MARATOSB	0.995892	0.000000	46	0.995892	0.000000	46
MARATOS	-1.000007	0.000074	46	-1.000007	0.000074	46
PALMER5C	2.128087	0.000000	46	2.128087	0.000000	46
PORTFL1	2.966959	9.110618	2	2.966959	9.110618	16
PORTFL2	3.997729	12.478277	4	3.997729	12.478277	15
PORTFL3	2.122685	9.494619	4	2.122685	9.494619	13
PORTFL4	2.919132	19.725226	46	2.919132	19.725226	46
PORTFL6	8.017043	45.462432	6	8.017043	45.462432	15
POWELLSQ	0.000000	11.677419	46	0.000000	11.677419	46
PSPDOC	7.523294	0.000000	4	7.523294	0.000000	19
RECIPE	0.000000	2.340570	1	0.000000	2.340570	1
S308	0.773199	0.000000	24	0.773199	0.000000	24
SISSER	0.000000	0.000000	19	0.000000	0.000000	19
SUPERSIM	0.000003	1.999992	2	0.000003	1.999991	17
TRY-B	1.323708	4.613363	46	1.323708	4.613363	46
comparaison	4990177,55	3,76	29,96	4987295,31	2,45	32,19

TAB. 5.13: Résultats pour la fonction de mérite pondérée

Revenons maintenant aux tableaux 5.10, 5.11 et 5.12. Sur la première colonne est affichée une série de problèmes choisis arbitrairement de la collection CUTEr. Les trois colonnes suivantes regroupent les résultats pour la méthode 1 : $L1$ avec la fonction de mérite de type l_1 (5.31), $L2$ avec la fonction de mérite de type l_2 (5.30) et $Linf$ avec la fonction de mérite de type l_∞ (5.32). Le tableau 5.10 affiche les valeurs de l'objectif alors que les tableaux 5.11 et 5.12 affichent respectivement les valeurs cc et le nombre d'itérations. Pour la fonction de mérite Pondérée (5.33), nous avons regroupé ses résultats dans le tableau 5.13. La dernière ligne des quatre tableaux essaye de donner un critère global de comparaison entre les quatre versions de chaque méthode. Il s'agit de la somme pour l'objectif et la moyenne pour le défaut de réalisabilité et pour le nombre d'itérations.

Le premier critère que nous vérifions est la moyenne en terme du défaut de réalisabilité. Nous rappelons que ce défaut doit être nul ou proche de zéro pour conclure que la méthode trouve une solution réalisable au problème en question. Les moyennes affichées par le tableaux 5.11 montrent que globalement, la méthode 2 est meilleure que la méthode 1 pour les trois formes de la fonction de mérite. Malheureusement, dans tous les cas, le chiffre affiché est supérieur à 228, ce qui est énorme s'il s'agissait d'un problème unique. En fait lorsqu'on regarde de très près notre tableau, on remarque que deux problèmes ont été mal résolus, il s'agit de BT13 et BT2 ($cc=18168.95$ et 6514.69 respectivement) ce qui influe fortement la moyenne cc .

Nous remarquons aussi (tableau 5.12) que pour les deux méthodes, la version L_1 affiche une moyenne du nombre d'itérations légèrement inférieure à la moyenne affichée par la version $L2$ ou $Linf$, et que cette moyenne pour la méthode 1 (16.02) est inférieure à la moyenne (17.68) pour la méthode 2. Or cette moyenne dépend du nombre de zéros dans la colonne. En effet, 0 indique que

la méthode n'a pas pu trouver un point optimal meilleur que le point initial. Pour la méthode 1, il s'agit de 33 cas, alors que ne sont que 26 cas pour la méthode 2 sur notre liste de 112 problèmes. En normalisant par rapport au nombre de problèmes réellement résolus, ces chiffres devienne 22.71 pour la méthode 1 et 23.02 pour la méthode 2. Les autres résultats sont donnés dans le tableau 5.14 suivant :

méthode	méthode 1			méthode 2		
Version	L 1	L 2	L inf	L 1	L 2	L inf
moyenne	22.71	32.03	30.13	23.02	28.71	28.71

TAB. 5.14: la moyenne du nombre d'itérations normalisée par rapport au nombre de problèmes résolus

Globalement toujours, la valeur objectif obtenue avec L_1 est d'ordre 10^6 est largement inférieure à celles obtenues avec L_2 qui est de l'ordre de 10^{11} et 10^9 pour L_{inf} . Ces grandeurs sont largement biaiser par les résultats de quelques problèmes. Par exemple, le problème HS1 donne un résultat de l'ordre de 10^{11} pour la version L_2 alors qu'il affiche une valeur de 23.16 pour la version L_1 . En éliminant cet exemple de notre tableau, on retrouve les tableaux de comparaison (Tab.5.15) pour l'objectif et (Tab.5.16) pour le défaut de réalisabilité suivants :

méthode	méthode 1			méthode 2		
Version	L 1	L 2	L inf	L 1	L 2	L inf
Somme obj	6.03E+06	7.04E+09	5.44E+09	6.03E+06	4.97E+06	4.96E+06

TAB. 5.15: Résultats de la comparaison de l'objectif après révision

méthode	méthode 1			méthode 2		
Version	L 1	L 2	L inf	L 1	L 2	L inf
Moyenne cc	8,17	12,19	10,97	7,99	8,59	7,98

TAB. 5.16: Résultats de la comparaison du défaut de réalisabilité après révision

La fonction de mérite pondérée telle que nous l'avons utilisé a dégradé les résultats, alors qu'on s'attendait à une accélération de la convergence, celle là est plus lente, et beaucoup plus de points stationnaires ont été détectés comme étant des points optimaux. Afin de mieux comprendre ce qui vient de se passer, rappelons que la fonction de mérite est introduite pour décider si un pas de déplacement d sera accepté ou non.

On dit qu'une application $(x, s) \mapsto M(x, s, \sigma)$ est une fonction de mérite (ou de pénalisation) exacte si tout minimum local de (5.2) est un minimum local de cet application. Han et Mangasarian [38] ont montré que si (x^*, s^*, λ^*) est un minimum local de (5.2) vérifiant les conditions suffisantes du second ordre de ce problème, alors x^* est un minimum local strict de $M(x, s, \sigma) = f(x) - \mu \sum_i s_i + \sigma \|c(x, s)\|$ dès que

$$\sigma > \|\lambda^*\|$$

La norme du multiplicateur est ainsi, une borne théorique pour le facteur σ .

Jusqu'au là, nous n'avons pas utilisé cette condition. Et ceci pour deux raisons :

1. Nous avons préféré d'imposer à la réduction prédite d'être suffisamment

positive avec la condition (5.38) :

$$pred(d) \geq \zeta \sigma vpred(v)$$

ce qui a impliqué pour σ la satisfaction de la condition (5.40) suivante

$$\sigma \geq \frac{\chi(v) - hpred(d)}{(1 - \zeta)vpred(v)}$$

si $vpred(v) > 0$.

2. Nous avons pris un paramètre σ initial relativement très grand, et avec la règle donnée par l'algorithme 17, nous sommes assurés (d'au moins pour l'ensemble de problèmes que nous avons traité jusqu'au là) que la condition $\sigma > \|\lambda^*\|$ reste toujours vérifiée.

Or il n'est jamais bien de prendre σ très grand, mieux encore (ou pire), il est parfois nécessaire de le faire décroître, mais cette stratégie est délicate, d'abord on ne connaît pas la valeur exacte de λ^* , donc on risque de perdre la notion de fonction de mérite exacte, mais aussi le paramètre σ peut (si on ne le contrôle pas) devenir assez petit au point de permettre à l'algorithme de minimiser l'objectif sans se préoccuper des contraintes, ce qui produirait des points non réalisables à la limite et mettrait en cause la convergence globale de l'algorithme. Afin de prendre en compte toutes ces remarques, nous proposons la mise à jour (procédure 18) du paramètre σ .

Procédure mise à jour du paramètre de pénalisation(règle 2)

<p>Si ($c_i \neq 0$) Alors</p> <p style="padding-left: 20px;">$\sigma_i = \max(\frac{c_i(x,s)}{\ c(x,s)\ _1}, \lambda_i)$</p> <p>Sinon</p> <p style="padding-left: 20px;">$\sigma_i = \max(\frac{\sigma_c}{\ c(x,s)\ _1}, \lambda_i)$</p> <p>Fin Si</p>	<p>Fin</p>
--	-------------------

Algorithme 18: Mise à jour du paramètre de pénalisation (règle 2)

Le rôle de σ_f (ou $\frac{1}{\sigma_f}$) n'est plus de renforcer la positivité de $pred$ puisque si on définit la réduction prédite par la formule :

$$pred(d) = -\sigma_f q(d) - \sigma^t Jv$$

alors, $pred$ est toujours positif (ceci est vrai du moment où le point initial de la minimisation verticale est pris dans l'orthant positif), mais σ_f traite la relation entre l'objectif et les contraintes, donc σ_f doit décroître à chaque itération et permettre que

$$\frac{\sigma_i}{\sigma_f} > |\lambda_i|$$

pour toute les composantes $i = 1, 2, \dots, m + p$. Ceci est évidemment vérifié si $0 < \sigma_f < 1$. Nous imposons donc à σ_f initial d'être compris entre 0 et 1 puis à chaque itération $0 < \sigma_f^k < \sigma_f^{k-1}$.

5.9 Le problème du conditionnement

L'attrait des méthodes de pénalisation, consistant à remplacer la solution d'un problème non linéaire par une suite de sous-problèmes non linéaires plus

simples à résoudre, a diminué vers 1970 après l'apparition des articles de Lootsma et Murray [53, 60]. Ces derniers avaient mis en évidence le mal conditionnement des sous-problèmes lorsque le paramètre de pénalisation (μ dans nos notations) se rapproche de zéro. Ce qui rend le problème difficile à résoudre et se reflète ainsi en une inefficacité de tout algorithme utilisé pour solutionner ces sous-problèmes.

Rappelons que le conditionnement d'un problème d'optimisation est lié au rapport des valeurs propres extrêmes de son Hessien.

Soit le problème non linéaire général (5.1) dont le sous-problème pénalisé associé est le problème non linéaire (5.2) est

$$\left\{ \begin{array}{l} \min f(x) - \mu \sum_{i=1}^m \ln(s_i) \\ \text{s.c.} \\ h(x) = 0 \\ g(x) + s = 0 \\ x \in \mathbb{R}^n, s \in \mathbb{R}_+^m \end{array} \right. \quad (5.41)$$

Le Lagrangien du problème pénalisé s'exprime au point $(x, s, \lambda_h, \lambda_g)$ par

$$L_\mu(x, s, \lambda_h, \lambda_g) = f(x) - \mu \sum_{i=1}^m \ln s_i + \lambda_h^t h(x) + \lambda_g^t (g(x) + s) \quad (5.42)$$

Son Hessien est donc, la matrice

$$W = \begin{pmatrix} \nabla_{xx}^2 L_\mu(x, s) & 0 \\ 0 & \Lambda_g S \end{pmatrix} \quad (5.43)$$

Le premier terme de ce Hessien se rapproche (en autant que x se rapproche de x^*) de $\nabla_{xx}^2 L_\mu(x^*, s^*)$, que l'on peut supposer bien conditionnée. Si

$\nabla_{xx}^2 L_\mu(x^*, s^*)$ est mal conditionnée, alors le problème est difficile, et le mauvais conditionnement ne provient pas de la méthode utilisée. Par conséquent, les valeurs propres extrêmes de $\nabla_{xx}^2 L_\mu(x^*, s^*)$ sont supposées finies et éloignées de zéro.

Le dernier terme du Hessien est la matrice $\Lambda_g S$. Cette écriture est propre à notre méthode SDC. La matrice $\Lambda_g S$ est diagonale dont les termes diagonaux sont $(\lambda_g)_i S_i$ $i = 1, \dots, m$. Or d'après le théorème de convergence (3.4.2) les termes $(\lambda_g)_i S_i$ sont bornés, ce qui réduit considérablement les chances d'avoir un mal conditionnement. Cependant à l'approche de la solution, la matrice $\Lambda_g S$ contient autant de valeurs propres nulles (ou proche de zéro) que de contraintes actives ce qui peut provoquer un mal conditionnement dans le cas où des valeurs propres de $\nabla_{xx}^2 L_\mu(x^*, s^*)$ sont très grandes. Néanmoins notre hypothèse du bon conditionnement du problème impose que les valeurs propres extrêmes de $\nabla_{xx}^2 L_\mu(x^*, s^*)$ sont finies et bien bornées ce qui réduit l'importance de l'éventualité d'un quelconque mal conditionnement.

5.10 Étude de la convergence

On considère ici le cas du problème d'optimisation non linéaire (5.44) avec seulement des contraintes d'inégalité (sans contraintes d'égalités), et soit μ fixé. Supposons que nous appliquons l'algorithme SDC pour résoudre ce problème d'optimisation. Les suites générées par l'algorithme seront indexées par l'entier naturel k .

$$\left\{ \begin{array}{l} \min f(x) \\ \text{s.c.} \\ g(x) \leq 0 \\ x \in \mathbb{R}^n \end{array} \right. \quad (5.44)$$

5.10.1 Cadre général de la convergence globale de la méthode des points intérieurs

Soit R l'ensemble des solutions réalisables, i.e. $R = \{x \mid g(x)_i \leq 0, i = 1, \dots, m\}$ et soit $ir(R) = \{x \mid g(x)_i < 0, i = 1, \dots, m\}$, l'ensemble des solutions strictement réalisables du problème (5.44).

Soit φ une fonction réelle de x définie sur \mathbb{R}^n . Supposons que φ est continue sur l'intérieur relatif de l'ensemble réalisable et que sa valeur explose quand les composantes s'approchent de zéro. Ceci se traduit par l'hypothèse suivante :

Hypothèses. 5.10.1. *Soit φ une fonction réelle, définie sur \mathbb{R}^n vérifiant les deux assertions suivantes*

1. *la fonction φ est continue sur $ir(R)$,*
2. *Si $(x_k)_k$ est une suite de points de $ir(R)$ convergeant vers un point x_B qui vérifie $g(x)_i(x_B) = 0$ pour au moins un indice i , alors $\lim_{k \rightarrow \infty} \varphi(x_k) = +\infty$*

□

Soit $s(\mu)$ une fonction réelle d'une variable positive μ qui vérifie les deux propriétés suivantes :

1. Si $\mu_1 > \mu_2 > 0$ alors $s(\mu_1) > s(\mu_2) > 0$,
2. si $(\mu_k)_k$ est une suite telle que $\lim_k \mu_k = 0$ alors $\lim_k s(\mu_k) = 0$

Une méthode de points intérieurs (voir notre introduction des méthodes barrière 0.3) est une technique qui construit, à partir d'un point initial (x_0, μ_0) , une fonction objectif pénalisée : $F(x, \mu) = f(x) + s(\mu) \cdot \varphi(x)$ telle que les deux fonctions φ et s vérifient les propriétés ci-dessus et $0 < \mu_1 < \mu_0$. Le minimum de la fonction F est noté (x_1, μ_1) et à partir de ce dernier point on reconstruit le modèle pénalisé pour avoir l'itéré (x_{k+1}, μ_{k+1}) suivant, ainsi de suite jusqu'à la convergence vers un minimum local du problème non linéaire (5.44).

Avant de donner le théorème de convergence, explicitons quelques définitions et lemmes nécessaires pour notre étude.

Lemme. 5.10.1. (*Fiacco-Mc Cormick [19]*)

Soit X un ensemble compact, non vide. Si la fonction objectif f est continue sur X alors il existe une valeur finie f^ et un point x^* de X tel que*

$$f(x^*) = f^* = \min_X f(x)$$

□

Le corollaire suivant nous aide à prouver l'existence d'un minimum local du problème pénalisé et du problème non linéaire.

Corrolaire. 5.10.1. *Supposons que l'ensemble R est fermé, que X est un ensemble compact et que $ir(R) \cap X \neq \emptyset$. Soit $F(x)$ une fonction continue sur $ir(R) \cap X$ et telle que pour toute suite $(x_k)_k$ de $ir(R) \cap X$ qui converge vers un point $y \in \partial(R) \cap X$ nous avons $\lim_{k \rightarrow \infty} F(x_k) = +\infty$, alors il existe une valeur finie \bar{F} et un point $\bar{x} \in ir(R) \cap X$ tel que $F(\bar{x}) = \bar{F} = \min_{ir(R) \cap X} F(x)$*

Preuve.

Soit x_0 un point de $ir(R) \cap X$ et soit $S = \{x \mid F(x) \leq F(x_0)\}$.

Si $(x_k)_k$ est une suite telle que pour tout k on ait $x_k \in S$, et comme $R \cap X$ est compact alors $x_k \rightarrow y$ implique que $y \in R \cap X$. Ainsi soit $y \in ir(R) \cap X$ soit $y \in \partial(R) \cap X$.

Si y est un élément de la frontière alors d'après la définition de la fonction F on a $\lim_{k \rightarrow \infty} F(x_k) = +\infty$ d'où pour k très grand x_k n'est pas un élément de S , d'où la contradiction. Ainsi $y \in ir(R) \cap X$.

Puisque pour tout k nous avons $F(x_k) \leq F(x_0)$ et comme F est continue sur $ir(R) \cap X$ alors $F(y) \leq F(x_0)$, d'où l'ensemble S est un ensemble compact, ainsi F atteint son minimum sur S d'où le résultat car $\inf_{ir(R) \cap X} F(x) = \min_S F(x)$.

CQFD \square

Nous rappelons la définition d'un ensemble isolé.

Définition. 5.10.1. Soit M^* un ensemble non vide tel que $M^* \subset M$. l'ensemble M^* est dit un ensemble isolé de l'ensemble M s'il existe un ensemble fermé E tel que $\overset{\circ}{E} \supset M^*$ et si $x \in E \setminus M^*$ alors $x \notin M$.

\square

Théorème. 5.10.1. (Fiacco-Mc Cormick [19])

Soit A^* un ensemble de minimums locaux correspondant à la valeur minimale f^* du problème non linéaire 5.44. Si A^* n'est pas vide et est isolé dans l'ensemble $A = \{x \in R \mid f(x) = f^*\}$ alors il existe un ensemble compact X tel que $A^* \subset \overset{\circ}{X}$ et pour tout point $y \in R \cap X$, si $y \notin A^*$ alors $f(y) > f^*$.

\square

Ce Théorème est fondamental dans la démonstration du théorème de convergence des méthodes de points intérieurs. Il montre qu'il est nécessaire d'avoir un ensemble compact X des minimums locaux, ceci est un point des hypothèses de base (5.2.1) que nous avons supposé pour construire notre code SDC.

Théorème. 5.10.2. (*Théorème de convergence*)

Si les assertions suivantes sont vérifiées

1. les fonctions f et g_1, g_2, \dots, g_m sont continues,
2. la fonction $F(x, \mu) = f(x) + s(\mu)\varphi(x)$ est la fonction objectif pénalisée tel que φ et s vérifient les propriétés des hypothèses 5.10.1.
3. l'ensemble A^* des minimums locaux associé à la valeur minimal f^* est non vide et est compact et isolé.
4. au moins un point de A^* est dans l'adhérence de $ir(R)$, et
5. μ_k est une suite strictement décroissante vers zéro, alors
 - (a) il existe un ensemble compact X (du théorème précédent) tel que $A^* \subset \overset{\circ}{X}$ et pour des paramètres μ_k assez petit, le minimum du problème pénalisé dans $ir(R) \cap \overset{\circ}{X}$ existe, et toute limite de chaque sous-suite $(x_k)_k$ des minimums est dans A^* .
 - (b) $\lim_{k \rightarrow \infty} s(\mu_k)\varphi(x(\mu_k)) = 0$,
 - (c) $\lim_{k \rightarrow \infty} f(x(\mu_k)) = f^*$,
 - (d) $\lim_{k \rightarrow \infty} F(x(\mu_k), \mu_k) = f^*$,
 - (e) la suite $(f(x(\mu_k)))_k$ est décroissante, et
 - (f) la suite $(\varphi(x(\mu_k)))_k$ est croissante.

□

Le lecteur peut consulter la preuve de (a)-(d) dans la démonstration du théorème 10 dans [19] traitant des méthodes de points extérieurs et la preuve de (e) et (f) peut être consultée à la démonstration du théorème 8 de la même référence.

La première condition du théorème est liée aux données du problème. Cette condition est donc vérifiée par hypothèse sur le problème initial. Il est facile,

maintenant de vérifier, que la fonction pénalisée de notre code SDC satisfait la deuxième condition du théorème. En effet, la fonction s n'est autre que l'identité, elle est continue et vérifie les propriétés de s , alors que la fonction φ est la fonction suivante

$$\begin{aligned} \varphi : \mathbb{R} &\longrightarrow \mathbb{R} \\ x &\longrightarrow \sum_{i=1}^m \ln(-g_i(x)) \end{aligned} \tag{5.45}$$

Cette fonction est continue sur $ir(R)$, et ses valeurs explosent quand une composante de x s'approche de la frontière, i.e. dès qu'un certain $g_i(x)$ est suffisamment proche de zéro.

Mais la méthode SDC n'est pas une méthode de points intérieurs classique dans le sens où elle traite les problèmes non linéaires avec seulement des contraintes d'inégalités. La méthode SDC a pour objectif de résoudre un problème non linéaire général d'où l'idée d'utiliser les technique SQP (voir paragraphe 3.6.2) ainsi le problème pénalisé n'est pas un problème sans contrainte et la fonction objectif pénalisée utilise une pénalisation en s_i (des variables auxiliaires), au lieu de la fonction φ de l'expression (5.45), nous utilisons la fonction suivante

$$\varphi(x) = \sum_{i=1}^m \ln(s_i)$$

où $s_i \in \mathbb{R}_+$ pour tout $i = 1, \dots, m$.

Pour les méthodes de points intérieurs classiques, c'est la fonction objectif pénalisée qui contrôle la progression des itérés. En effet, dans la démonstration du corrolaire (5.10.1), nous avons construit l'ensemble $S = \{x \mid F(x) \leq F(x_0)\}$, et tout point x_k doit appartenir à cet ensemble. Ceci n'est pas le cas pour les méthodes modernes. Pour les méthodes récentes, un itéré peut dégrader l'optimalité ou transgresser les contraintes. Ainsi on peut améliorer la construction

de l'ensemble S en y'injectant les contraintes, c'est le cas des méthodes de filtre (voir notre section 4.4 ou [6, 20, 21, 83]), ou bien utiliser une fonction de mérite au lieu de la fonction F . C'est le cas des trois autres méthodes que nous avons présenté au chapitre 3 et également de la méthode SDC.

5.10.2 Convergence de la méthode SDC

Dans des études plus récentes [8, 46], on trouve les résultats de convergence suivants. Ces résultats restent valables pour la méthode SDC.

$$\left\{ \begin{array}{l} \min f(x) - \mu \sum_{i=1}^m \ln(s_i) \\ \text{s.c.} \\ g(x) + s = 0 \\ x \in \mathbb{R}^n, s > 0 \end{array} \right. \quad (5.46)$$

Soit la mesure d'admissibilité (ou de réalisabilité) suivante

$$\Theta(x) = \frac{1}{2} \|c(x)^+\|^2$$

Cette fonction est différentiable et son gradient est $\nabla\Theta(x) = J(x)^t c(x)^+$. La suite $(x_k)_{k \geq 0}$ sera dite asymptotiquement réalisable si $c(x)^+ \rightarrow 0$. On dira aussi que la suite $(c(x_k), J(x_k))_k$ a un point limite (\bar{c}, \bar{J}) ne satisfaisant pas la condition de qualification des contraintes actives au sens de l'indépendance linéaire si la famille $\{\bar{J}^{(i)} : \bar{c}^{(i)} = 0\}$ de colonnes de \bar{J} n'est pas linéairement indépendante.

Sous les hypothèses de base (5.2.1) nous avons le théorème de convergence suivant dû à Byrd, Gilbert et Nocedal [8]. Le lecteur intéressé par une preuve de convergence de l'algorithme barrière consultera cette référence.

Théorème. 5.10.3. [8]

Supposons que l'algorithme SDC est appliqué au problème barrière défini en (5.44) et que les hypothèses (5.2.1) sont vérifiées. Alors,

1. La suite $(s_k)_k$ des variables d'écart est bornée,
2. $J(x)^t(c(x_k + s_k) \rightarrow 0$ et $S_k(c(x_k + s_k) \rightarrow 0$.

D'autre part, une des trois situations suivantes a lieu.

1. La suite des itérés $(x_k)_k$ n'est pas asymptotiquement réalisable. Dans ce cas

$$J(x)^t c(x_k)^+ \rightarrow 0$$

c'est-à-dire x_k tend vers un point critique de la mesure d'admissibilité $\Theta(x)$, et le facteur de pénalisation σ_k tend vers l'infini.

2. La suite des itérés $(x_k)_k$ est asymptotiquement réalisable mais $(c(x_k), J(x_k))_k$ a un point limite (\bar{c}, \bar{J}) ne satisfaisant pas la condition de qualification des contraintes actives au sens de l'indépendance linéaire et dans ce cas aussi, le facteur de pénalisation σ_k tend vers l'infini.
3. La suite des itérés $(x_k)_k$ est asymptotiquement réalisable et la suite $(c(x_k), J(x_k))_k$ a un point limite (\bar{c}, \bar{J}) qui satisfait la condition de qualification des contraintes actives au sens de l'indépendance linéaire. Dans ce cas chaque composante de $(s_k)_k$ est bornée inférieurement, le facteur de pénalisation σ_k est constant, et toutes les composantes de $c(x_k)$ sont négatives pour k assez grand.

D'autre part, les itérés convergent vers un point stationnaire du problème barrière, c'est-à-dire qu'on a

$$\nabla f(x_k) + J(x_k)^t \lambda_k \rightarrow 0$$

si le multiplicateur est défini par $\lambda_k = \mu S_k^{-1} e$.

□

Les deux premières situations sont importantes, et identifient deux situations pour les quelles les conditions d'optimalité KKT ne sont pas satisfaites. Dans les deux cas on remarque que σ_k tend vers l'infini. En pratique cette remarque est utilisée comme indice que les KKT ne sont peut être pas satisfaites.

5.11 Conclusion

Notre expérience a montré que plusieurs méthodes souffrent énormément du mauvais conditionnement. L'étude approfondie de la base théorique de certaines d'entre elles nous a conduit à proposer une correction des conditions d'optimalités qui a fourni un gain concernant la convergence de la méthode SDC proposée. L'efficacité observée a été renforcée par la résolution des sous-problèmes quadratiques par une méthode D.C. ce qui nous a permis de gagner au niveau du temps de calcul et surtout de réduire significativement les dimensions des sous-problèmes quadratiques horizontaux (non convexes) qu'il a fallu résoudre. En effet nous avons effectivement résolu les sous-problèmes dans \mathbb{R}^{m+p} au lieu de les résoudre dans \mathbb{R}^{n+m} et ceci est très avantageux car nous avons le plus souvent $p < n$.

Nous pensons que la méthode peut encore évoluer dans deux sens, d'abord en profitant de la nature des matrices à savoir qu'elles soient des matrices creuses, puis en gérant mieux la fonction de mérite pondérée dont les paramètres nécessitent plus d'ajustement dynamique.

CONCLUSION

Ce travail a été consacré à l'étude des problèmes d'optimisation non linéaire avec des contraintes mixtes d'égalités et d'inégalités et à leurs résolutions par des techniques numériques modernes. L'analyse des performances des méthodes existantes nous a permis de cibler nos principaux axes de recherche.

Nous avons exposé les résultats théoriques de l'optimisation non linéaire, notamment au sujet des conditions d'optimalité, de la redondance et de la qualification des contraintes et de la convergence. Nous avons souligné la relation entre les contraintes actives et la divergence des méthodes de points intérieurs. Le changement de variables que nous avons apporté au pas de déplacement s'est avéré être une réponse efficace à cette divergence. Les nombreux tests numériques que nous avons menés témoignent de la stabilité retrouvée.

Les fonctions de mérite contrôlent le pas de déplacement et ont, ainsi, un impact considérable sur la vitesse de convergence de la méthode. Nous avons proposé une fonction de mérite pondérée pour équilibrer, d'abord le poids de la fonction objectif par rapport à l'ensemble des contraintes, ensuite pour équilibrer intrinsèquement les contraintes. La première version de cette nouvelle fonction a légèrement dégradé les résultats au niveau du nombre d'itérations. En revanche, l'étude théorique nous a permis de proposer de nouvelles règles de mise à jour afin d'améliorer son efficacité.

Notre première étude numérique a été consacrée à la résolution des problèmes quadratiques convexes et non convexes par une méthode D.C. Cette dernière nous a amené à choisir numériquement un seuil de précision pour sa convergence. Nous avons complété cette étude par une comparaison entre la méthode G.C. de Steihaug et la méthode D.C., en terme du nombre d'itérations, de la valeur de la fonction objectif et du temps CPU nécessaire à la

convergence. Les résultats ont montré que la méthode D.C. donne toujours des solutions meilleures que celles retrouvées par la méthode de Steihaug, même si cette dernière consomme plus de temps de calcul. Cette remarque est modérée par le fait qu'une itération D.C. coûte, en moyenne, trois fois moins de temps CPU qu'une itération G.C-Steihaug. On peut, également arrêter la méthode D.C. à l'instant même où la méthode de Steihaug s'arrête, pour affirmer que les résultats de cette dernière sont moins intéressants que ceux retrouvés par la première.

En plus de ces résultats numériques performants, nous montrons qu'une bonne adaptation de la méthode D.C. permet de résoudre le problème vertical dans \mathbb{R}^{m+p} au lieu de \mathbb{R}^{n+m} ($p < n$), ce qui représente un gain considérable contribuant à l'efficacité de notre méthode. La procédure D.C-verticale fournit, grâce au bon choix du point initial, une orthogonalité théorique et numérique parfaite.

Nous avons utilisé des techniques de rebroussement pour garantir la positivité des variables auxiliaires. Une correction du second ordre est introduite pour répondre à un éventuel effet de Maratos, dans le cas où la fonction de mérite n'est pas différentiable.

Notre méthode est appelée SDC, en analogie avec les techniques SQP qu'elle utilise pour produire les sous problèmes quadratiques. Grâce aux hypothèses classiques, nous avons démontré le théorème de convergence de la méthode SDC. Nous avons, également, mis en évidence le fait que notre méthode ne souffre pas du mauvais conditionnement.

Les résultats numériques des versions suivantes de la fonction de mérite pondérée sont encourageants mais ne sont pas suffisamment considérables pour être concluants. Ainsi son amélioration fait partie de nos perspectives. La généralisation de la méthode D.C. pour résoudre les problèmes quadratiques avec

des contraintes quadratiques reste un problème ouvert. Cette généralisation peut améliorer l'efficacité de la méthode, en réduisant le nombre de directions rejetées par la méthode de région de confiance.

Au niveau numérique, la version CUTEr de la méthode SDC est pratiquement opérationnelle, alors que la version AMPL mérite plus de développement.

ANNEXE A

LA MÉTHODE DU SIMPLEXE

Le développement de la méthode du simplexe par Dantzig de l'université de Princeton à New Jersey, à la fin des années quarante a marqué le début de l'ère de l'optimisation moderne, ceci coïncida avec la mise en œuvre des premiers ordinateurs, ainsi le simplexe devint vite un des plus importants logiciels pour la programmation linéaire.

De nos jours le simplexe est largement utilisé par les praticiens de tout bord : management, économie, finance, problème de transport, ingénierie, et sciences de la matière.

La force de la méthode, est qu'à partir d'un minimum potentiel on arrive à trouver le point optimal exact, en un nombre d'itérations fini (si le point optimal n'existe pas ou la fonction objectif n'est pas borné inférieurement, alors la méthode arrive à déceler ces deux cas aussi).

A.1 Notations

Soit le problème d'optimisation linéaire standard suivant :

$$(PLS) \begin{cases} \min c^T x \\ \text{s.c.} \\ Ax = b \\ x \geq 0 \end{cases} \quad (A.1)$$

où c et x sont deux vecteurs de \mathbb{R}^n , et A une matrice $m \times n$ de \mathbb{R} , et $b \in \mathbb{R}^m$.

Supposons que $m < n$.

Soit x un sommet de K , où K est l'ensemble réalisable du problème (A.1)

Notons $I_x = \{i \in \{1, \dots, n\} : x_i > 0\}$

et $S_x = (A^i)_{i \in I_x}$

où A^i est la i^{me} colonne de la matrice A .

Si $\text{Card}(I_x) = m$ alors $B_x = S_x$ est une base de \mathbb{R}^m .

Si $\text{Card}(I_x) < m$ alors on complète le système S_x par des colonnes de A de sorte à obtenir une base B_x de \mathbb{R}^m .

on pose $\beta_x = \{i \in \{1, \dots, m\} : A^i \in B_x\}$ cet ensemble sera le plus souvent confondu avec la base B_x elle même.

Si $I_x = \beta_x$ alors on dit que x est un sommet non dégénéré.

Si $I_x \subsetneq \beta_x$ alors on dit que x est un sommet dégénéré.

ainsi toute colonne de A peut s'écrire dans la base B_x :

$$\forall j \in \{1, \dots, n\} : A^j = \sum_{i \in \beta_x} \alpha_i^j A^i$$

on pose $Z_j = c_j - \sum_{i \in \beta_x} \alpha_i^j c_i$, le coût marginal.

A.2 Algorithme du Simplexe

Soit x un sommet de K et β_x une base associée à x

on a une des éventualités suivantes :

Si ($Z_j \geq 0 \forall j \notin \beta_x$) **Alors**

| x est une solution optimale de (A.1)

Fin Si

Si ($\exists j \notin \beta_x$ tel que $Z_j < 0$) **Alors**

| on pose $J = \{j \notin \beta_x : Z_j < 0\} \neq \emptyset$.

| **Si** ($\exists j \in J$ tel que $\alpha_i^j \leq 0 \forall i \in \beta_x$) **Alors**

| | (A.1) n'admet pas de solution

| **Fin Si**

| **Si** ($\forall j \in J \exists i \in \beta_x$ tel que $\alpha_i^j > 0$) **Alors**

| | **Si** ($\exists j^+ \in J$ tel que $\{i \in \beta_x : \alpha_i^{j^+} > 0\} \subset I_x$) **Alors**

| | | on pose $\theta^{j^+} = \min\{\frac{x_i}{\alpha_i^{j^+}} : i \in \beta_x, \alpha_i^{j^+} > 0\}$, $\theta^{j^+} > 0$

| | | soit $j^- \in \beta_x$ tel que $\alpha_{j^-}^{j^+} > 0$ et $\theta^{j^+} = \frac{x_{j^-}}{\alpha_{j^-}^{j^+}}$

| | | soit

$$\bar{x} = \begin{cases} \bar{x}_i = x_i - \theta^{j^+} \alpha_i^{j^+} & \text{si } i \in \beta_x \\ \bar{x}_{j^+} = \theta^{j^+} \\ \bar{x}_i = 0 & \text{si } i \notin \beta_x \cup \{j^+\} \end{cases}$$

| | \bar{x} est un sommet de K différent de x tel que $c^t \bar{x} < c^t x$ et

| | $\beta_{\bar{x}} = (\beta_x \setminus \{j^-\}) \cup \{j^+\}$ est une base de \mathbb{R}^m associée à \bar{x}

| | **Sinon**

| | | i.e. $\forall j \in J \exists i \in \beta_x$ tel que $\alpha_i^j = 0$ et $x_i = 0$

| | | soit j^+ un indice quelconque dans J et soit $j^- \in \beta_x$ tel que

| | | $\alpha_{j^-}^{j^+} > 0$ et $x_{j^-} = 0$

| | | l'ensemble $\beta'_x = (\beta_x \setminus \{j^-\}) \cup \{j^+\}$ est une autre base de

| | | \mathbb{R}^m différente de β_x associée au même sommet x

| | **Fin Si**

| **Fin Si**

Fin Si

Pour déterminer un sommet initial et une base associée à ce sommet, on considère le problème auxiliaire :

$$(PLS_0) \begin{cases} \min \sum_{i=1}^m v_i \\ \text{s.c.} \\ (x, v) \in K_0 = \{(x, v) \in \mathbb{R}^n \times \mathbb{R}^m : x \geq 0, v \geq 0 \text{ et } Ax + v = b\} \end{cases} \quad (\text{A.2})$$

On montre que $K_0 \neq \emptyset$ et que $(0, b)$ est un sommet de K_0 associé à la base canonique. Le problème (PLS_0) admet une solution optimale (x^*, v^*) .

si $v^* \neq 0$ alors $K = \emptyset$.

si $v^* = 0$ alors x^* est un sommet de (A.1) associée à la base $B_{(x^*, v^*)}$.

Le grand défaut de la méthode de Simplexe n'est certainement pas son implémentation ni des failles lors de la résolution du problème de grand taille, mais les soucis dus à sa complexité qui est dans le pire des cas exponentielle et de fait que la méthode exploite tous les point extrémaux, sommet par sommet en commençant par le point initial.

Par souci de recherche d'une méthode polynomiale, Khachian a proposé en 1979 sa méthode d'ellipsoïdes qui a été proposé au départ pour la programmation non-linéaire, malheureusement la méthode de Khachian sera vite abandonnée à cause des difficultés d'implémentation qu'elle présente. Il fallait attendre 1984, pour voir naître une méthode polynomiale qui va révolutionner l'optimisation mathématique.

Rappelons enfin que pour un problème donnée (P), on désigne par taille du problème, le nombre de cases de mémoire qui permettent d'écrire les données du problème (P). On dira qu'un algorithme est de complexité algorithmique $g(N)$ si sa mise en œuvre pour résoudre un problème de taille N nécessite un nombre d'opérations borné par $g(N)$ et il n'existe pas de fonction h ayant la même propriété et telle que $\lim_{N \rightarrow \infty} \sup (h(N)/g(N)) = 0$.

ANNEXE B

LA MÉTHODE DE KARMARKAR

L'outil le plus utilisé pour la programmation linéaire et le plus célèbre de la recherche opérationnelle est la méthode du simplexe. Bien que cette méthode est réputé très efficace, en théorie, la méthode du simplexe ne permet la résolution d'un problème linéaire avec n variable et m contraintes qu'en un temps exponentiel, c'est-à-dire le nombre d'itérations est de l'ordre de 2^n .

En 1984 Karmarkar [48] propose un algorithme plus attractif notamment par sa complexité polynomiale, c'est-à-dire, le nombre d'itérations est un polynôme en n et m . L'algorithme minimise une fonction linéaire qui vaut zéro à l'optimum sur un simplexe régulier S qui est un ensemble de n -vecteurs positifs dont la somme des composantes est égale à une constante. À chaque itération, Karmarkar utilise une transformation projective de S dans lui même et qui envoie la solution réalisable courante au centre de S . Le dessin (B.1) montre la différence entre les chemins pris par les deux méthodes pour retrouver l'optimum. On remarque que, la méthode de Dantzig exploite tous les sommets à partir d'un sommet initial x_0 alors que la méthode de Karmarkar essaye de trouver un chemin plus court en passant à l'intérieur du simplexe.

B.1 Résolution d'une fonction linéaire sur un sphère

Soit le problème de minimisation suivant :

$$\begin{cases} \min c^t x \\ \text{s.c.} \\ x \in E \end{cases} \quad (\text{B.1})$$

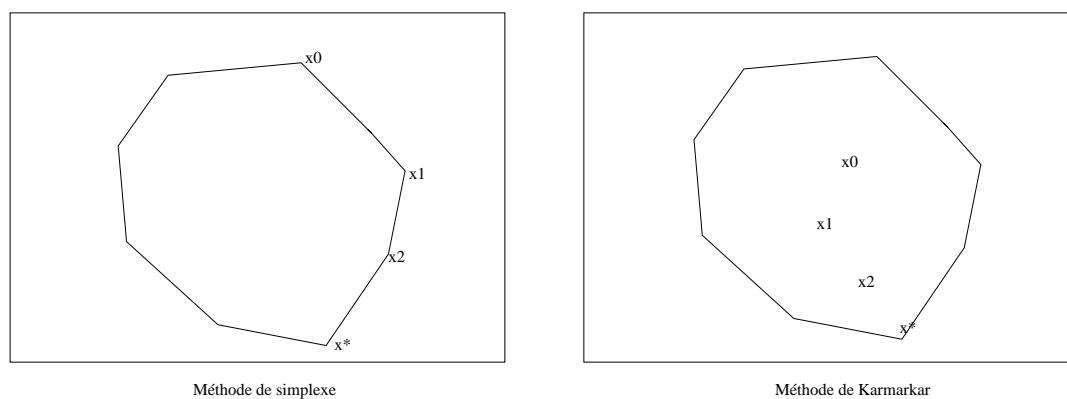


FIG. B.1 – Les chemins pris par la méthode du Simplexe et la méthode de Karmarkar

où E est une sphère de centre x_0 et de rayon r . Nous avons le lemme (B.1.1) suivant

Lemme. B.1.1. *La solution du problème linéaire (B.1) est donné par :*

$$x^* = x_0 - \frac{c}{\|c\|_2} r$$

Preuve.

soit $d = x - x_0$ alors résoudre le problème (B.1) est équivalent à la résolution de (B.2)

$$\begin{cases} \min c^t d \\ \text{s.c.} \\ \|d\|_2^2 \leq r^2 \end{cases} \quad (\text{B.2})$$

Il est facile de voir (voir dessin B.2) que la solution d^* de ce problème vérifie :

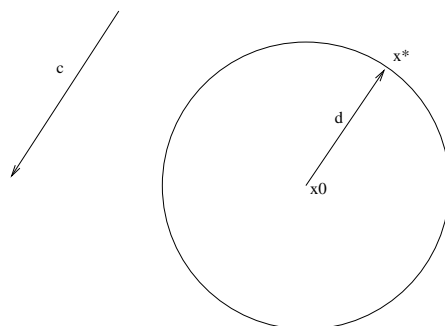


FIG. B.2 – Illustration de la solution d'un problème linéaire sur un cercle

$\|d^*\|_2 = r$ et $\frac{d^*}{\|d^*\|_2} = -\frac{c}{\|c\|_2}$ d'où $d = -\frac{c}{\|c\|_2}r$
 ainsi $x^* = x_0 + d^* = x_0 - \frac{c}{\|c\|_2}r$

On généralise facilement ce résultat au cas où l'ensemble E est un ellipsoïde. Algébriquement, on peut écrire

$$E = \{x \in \mathbb{R}^n : (x - x_0)^t A (x - x_0) \leq r^2\}$$

où A est une matrice symétrique définie positive.

De la même manière que dans le lemme précédent, on pose $d = x - x_0$ d'où E devient

$$\{d \in \mathbb{R}^n : d^t A d \leq r^2\}$$

et le problème à résoudre est (B.3)

$$\begin{cases} \min c^t d \\ \text{s.c.} \\ d^t A d \leq r^2 \end{cases} \quad (\text{B.3})$$

On retrouve l'expression recherchée en utilisant les conditions d'optimalité.

en effet, d^* est la solution de ce problème si et seulement s'il existe un réel $\lambda^* \geq 0$ tel que

$$\begin{cases} c + \lambda^* Ad^* & = 0 \\ \lambda^*(d^{*t}Ad - r^2) & = 0 \end{cases}$$

d'où $\lambda^* > 0$, $d^* = -\frac{1}{\lambda^*}A^{-1}c$ et $d^{*t}Ad = r^2$

ce qui nous permet d'écrire

$$d^* = \frac{-rA^{-1}c}{\sqrt{c^tA^{-1}c}} \text{ et } x^* = x_0 - \frac{rA^{-1}c}{\sqrt{c^tA^{-1}c}}.$$

Comme l'intersection d'un espace affine et une sphère est une sphère de dimension plus petite, alors la projection d'un problème linéaire sur une sphère est facile à résoudre vu le résultat du lemme (B.1.1).

Afin de construire les ellipsoïdes E et E' de rayon νr ($\nu > 1$) dont le rôle serait de contrôler la convergence, Karmarkar propose la transformation projective T que nous décrivons ci-dessous.

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

Soit $a \in \mathbb{R}^n$ un point intérieur à P et soit D la matrice diagonale : $\text{diag}(a)$.

Finalement notons S_{n+1} le simplexe de dimension $(n+1)$ suivant

$$S_{n+1} = \{x \in \mathbb{R}^{n+1} : x \geq 0, e_{n+1}^t x = 1\}$$

où e_{n+1} est le vecteur unité de \mathbb{R}^{n+1} c'est-à-dire toutes ces composantes sont égales à 1.

La transformation projective T est une fonction de \mathbb{R}_+^n dans \mathbb{R}_+^{n+1} définie par : $y = T(x)$ avec

$$\begin{cases} y_i & = \frac{x_i/a_i}{1 + \sum_{i=1}^n \frac{x_i}{a_i}} & i = 1, \dots, n \\ y_{n+1} & = 1 - \sum_{i=1}^n y_i \end{cases}$$

On montre facilement que T est une bijection, que $T(\mathbb{R}_+^n) = S_{n+1}$ et que

$T(a) = x_0 = \frac{1}{n+1}e_{n+1}$ (c'est le centre de S_{n+1}).

La plus grande sphère inscrite dans S_{n+1} est

$$B(x_0, r) = \{x \in \mathbb{R}^{n+1} : e_{n+1}x = 1, \|x - x_0\| \leq r\}$$

et la plus petite sphère contenant S_{n+1} est

$$B(x_0, R) = \{x \in \mathbb{R}^{n+1} : e_{n+1}x = 1, \|x - x_0\| \leq R\}$$

où $r = \frac{1}{\sqrt{n(n+1)}}$ et $R = \frac{\sqrt{n}}{\sqrt{n+1}}$

B.2 L'algorithme

Karmarkar résout le programme linéaire suivant

$$\begin{cases} \min c^t d \\ \text{s.c.} \\ Ax = 0 \\ x \in S_n \end{cases} \quad (\text{B.4})$$

sous les hypothèses dites hypothèses de Karmarkar :

Hypothèses. B.2.1. (*hypothèses de Karmarkar*)

1. La valeur optimale est nulle.
2. Le centre du simplexe x_0 est une solution réalisable du problème (B.4).
3. la matrice A est de plein rang.

L'algorithme de Karmarkar part d'un point x_0 et construit une suite de points x_n qui converge vers l'optimum noté x^* en un temps polynômial. A

chaque étape, on ramène x_k au centre de S_n par la transformation projective suivante :

$$T_k : x \in S_n \rightarrow T_k(x) = y \in S_n$$

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

$$\text{où } D_k = \text{diag}(x_k).$$

On teste à chaque fois la mesure d'optimalité $c^t x^k \leq \varepsilon$ où ε est la précision recherchée.

Initialisation $x_0 = \frac{1}{n}e_n$, $k = 0$

Tant que $(e^t x_k > \varepsilon)$ **faire**

$$D_k = \text{diag}(x_k)$$

$$A_k = AD_k$$

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

$$p_k = (I - B_k^t (B_k B_k^t)^{-1} B_k) D_k c$$

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

$$y_k = x_0 - \alpha r d_k \text{ où } r = \frac{1}{\sqrt{n(n-1)}}, \alpha \in]0, 1[$$

$$x_{k+1} = \frac{D_k y_k}{e_n^t D_k y_k}$$

$$k = k + 1$$

Fait

Algorithme 20: L'algorithme de base de Karmarkar

Karmarkar choisit $\alpha = .25$ et il montre que si $0 \leq \alpha \leq 0.25$ alors son algorithme converge après $\mathcal{O}(nq + n \log n)$ itérations, où $\varepsilon = 2^{-q}$

ANNEXE C

MODÉLISATION

À travers cette dissertation, nous admettons que l'ultime objectif est la résolution d'un problème d'optimisation dit non linéaire (C.1) pour lequel, l'objectif est une fonction non linéaire régulière et les contraintes sont de deux sortes, des contraintes d'égalités notés $h(x) = 0$ où h est une fonction non linéaire régulière, et des contraintes d'inégalité notés $g(x) \leq 0$ avec $g(x)$ une fonction non linéaire régulière.

$$\begin{cases} \min f(x) \\ h_i(x) = 0 \quad i = 1, 2, \dots, p \\ g_i(x) \leq 0 \quad i = 1, 2, \dots, m \\ x \in \mathbb{R}^n \end{cases} \quad (\text{C.1})$$

À croire GILL, Murray et Wright [30], la majeure partie des problèmes d'optimisation s'écrit sous cette forme, même si construire un bon algorithme nécessite de prendre en compte la forme et les caractéristiques de la famille de problèmes que nous souhaitons résoudre. Le tableau (Tab.C.1) suivant montre les principales familles de problème dans le cas continu, que nous pourrions rencontrer dans la pratique.

Propriété de l'objectif	Propriétés des contraintes
fonction linéaire	variables en boîtes
moindre carrée	fonctions linéaires
fonction quadratique	fonctions linéaires
reste non linéaire	fonctions non linéaires régulières
... la suite sur la page suivante...	

... suite de la page précédente...	
fonction non linéaire régulière	fonctions non linéaires régulières
fonction non linéaire	fonctions non linéaires non régulières
fonction non linéaire non régulières	avec ou sans contraintes

TAB. C.1: Principales familles de problèmes continus.

La deuxième caractéristique qui intervient dans le choix de la méthode est la taille du problème à résoudre, qui affecte en amont, toutes les techniques algébriques ou algorithmiques qu'on puisse utiliser et en aval, l'effort de calcul dont l'algorithme a besoin pour retrouver la solution optimale. Une technique qui va être très efficace pour une famille de problèmes peut s'avérer inutile par sa lourdeur pour exploiter une autre famille. La machine utilisée intervient également dans le choix des problèmes à résoudre et par conséquent de la méthode (le problème ainsi posé peut apparaître insensé mais pour tester l'efficacité d'une méthode nous avons besoin d'un jeu de tests, le choix de cet ensemble dépend aussi de la machine dont nous disposons). Pour notre étude nous avons exploité nos méthodes sur un processeur Intel(R) Pentium(R) 4 CPU 2.60GHz avec une fréquence de 2594.002 MHz avec une mémoire totale de 1033580 kB dont 185804 kB de mémoire cachée et avec une mémoire swap totale de 2104504 kB dont 91612 kB de swap caché géré par un système d'exploitation Linux Suze 9.2. Avec des ressources limitées, il faut exploiter au maximum les caractéristiques des problèmes et notamment le fait qu'une grande partie de nos matrices sont creuses.

Dans la pratique, il est nécessaire de prendre en considération toutes les particularités du problème et les moyens mis en place. Dans cette dissertation nous avons essayé de proposer des méthodes générales pour résoudre un en-

semble non homogène de problèmes ce qui donne à nos méthodes un potentiel global de résolution.

C.1 L'environnement CUTEr

La version initial de CUTEr est CUTE (Constrained and Unconstrained Testing Environment, r pour revisited) a été présenté en 1995 par Conn, Gould et Toint [13] mais dès 1993 CUTE été largement utilisé par la communauté scientifique. Grâce à un ensemble d'outils, CUTEr permet l'écriture de nouveau codes, l'unification des outils de programmation et de tests permettent des comparaisons plus objectives et rationnelles des codes existants. Les outils sont écrit en Fortran 77, mais le programmeur C peut utiliser CUTEr grâce à des fichier d'en-tête spécifique conçus pour cet effet.

Une panoplie d'outils a été introduite pour faciliter l'exploitation des fichiers contenant les problèmes, ils permettent le calcul de la valeur de l'objectif, des contraintes, des gradients de différentes fonctions, le Hessien ainsi que d'autres fonctionnalité comme le temps de calcul ou le nombre d'itérations.

Les fichiers contenant les données concernant les problèmes font partie d'un ensemble dit MASTSIF, chaque problème à une étiquette de classification qui le décrit

classification ABCx-DE-n-m

Le premier caractère A décrit la fonction objectif et porte une des possibilités suivantes

N Aucune fonction objectif n'est défini,

C l'objectif est une constante,

L l'objectif est linéaire,

Q l'objectif est quadratique,

S l'objectif est une somme de fonction quadratique, et finalement

O si l'objectif n'est d'aucune des possibilités ci-dessus.

Le second caractère B décrit le type des contraintes

U si le problème est sans contraintes,

X si le problème admet des variables fixés comme contraintes,

B si le problème admet des variables emboîtés comme contraintes,

N si les contraintes sont sous la forme d'une matrice issus d'un réseau linéaire,

L si les contraintes sont linéaires,

Q si les contraintes sont quadratiques, et finalement

O si aucune des situation décrite ci-dessus n'a lieu.

Le troisième caractère C décrit la régularité des problèmes, il admet deux valeurs

R si le problème est régulier c'est-à-dire, les fonction sont de classe C^2 , ou

I si le problème n'est pas régulier.

La lettre C est suivie par un nombre x qui peut prendre les valeurs 0,1 ou 2. x est le degré le plus haut de l'ordre de différentiabilité fourni analytiquement dans la description du problème.

La lettre D indique l'origine du problème :

A si le problème est académique,

M si le problème provient d'une modélisation, et

R si le problème ou sa résolution est une application réelle souvent de type industriel.

La lettre E est remplacé soit par

Y si la description du problème (le fichier) contient des variables internes explicites, soit par

N si ce n'est pas le cas.

Le nombre n est le nombre de variable du problème, si ce nombre est remplacée par V cela voudra dire que le nombre de variables est flottant (variable).

Et finalement le nombre m est le nombre de contraintes du problème, si ce nombre est remplacée par V cela voudra dire que le nombre de contraintes est variable. Un problème d'optimisation non linéaire sera codé à partir du schéma suivant :

$$\left\{ \begin{array}{l} \min F(x) = \sum_{i \in I_0} g_i(\sum_{j \in J_i} \omega_{i,j} f_j(\bar{x}_j) + a_i^t x - b_i) + \frac{1}{2} \sum_{j=1}^n \sum_{k=1}^n h_{jk} x_j x_k \\ \text{s.c.} \\ g_i(\sum_{j \in J_i} \omega_{i,j} f_j(\bar{x}_j) + a_i^t x - b_i) = 0 \quad i \in I_{\text{Egalité}} \\ 0 \begin{pmatrix} \leq \\ \geq \end{pmatrix} g_i(\sum_{j \in J_i} \omega_{i,j} f_j(\bar{x}_j) + a_i^t x - b_i) \begin{pmatrix} \leq \\ \geq \end{pmatrix} r_i \quad i \in I_{\text{Inégalité}} \\ l_i \leq x_i \leq u_i \end{array} \right. \quad (\text{C.2})$$

avec les définitions suivantes

g_i est dit groupe de fonctions, il permet de regrouper différentes partie des données en des ensembles homogènes, par exemples des fonctions *sinus* déclarées dans plusieurs endroits seront déclarées une seule fois grâce à un groupe SINUS qui sera utilisé par la suite.

$\sum_{j \in J_i} \omega_{i,j} f_j(\bar{x}_j)$ est dit le i^{eme} groupe.

f_j est l'élément non linéaire.

\bar{x}_j est un élément constitué de quelque composantes de x .

$\omega_{i,j}$ sont des poids

$a_i^t x - b_i$ est l'élément linéaire.

$\frac{1}{2} \sum_{j=1}^n \sum_{k=1}^n h_{jk} x_j x_k$ est le groupe quadratique.

Les outils SifDec permettent de décoder les problèmes qui sont écrits sous la forme SIF (Standard Input File), et sont nécessairement installés avant l'utilisation de CUTER. Le site [96] propose une documentation riche et complète, le choix d'une famille de problèmes précise est possible sur le site donnée dans la référence [97].

La construction de nouvelle interface SDC, par exemple, passe par les quatre étapes suivantes :

1. Écrire le driver `sdcm`, qui contient le code de la méthode.
2. Compiler le fichier source par la commande "`cc -c sdm.c`" et copier le fichier objet dans le répertoire "`$MYCUTER/double/bin`".
3. Écrire le prototype `sdsdc.pro` et `sd.c.pro`.
4. Créer un nouveau répertoire SDC qui contiendra les fichiers relatif à notre code.

Deux remarques sont nécessaires pour bien comprendre la relation entre le nombre m et les valeurs m et p de notre étude, premièrement, le nombre `cuter m`, ne fait pas la différence entre contraintes d'égalités et contraintes d'inégalités, deuxièmement les contraintes relatives aux variables ne sont pas comptabilisés. Par exemple un problème qui contient une contrainte relative à une variable fixée ou des variables emboîtés sera considéré comme un problème sans contraintes. Ainsi, lors de l'écriture du nouveau driver, il est important de vérifier la concordance entre le fichier des données et la forme du problème à résoudre. En effet, un problème non linéaire s'écrit sous la forme (C.1) alors que pour l'environnement CUTER, le même problème est modélisé avec l'expression (C.2). Ainsi les contraintes doivent être traduites quand on passe d'un environnement à un autre et les dimensions sont à revoir, et il n'est pas toujours vrai de dire que la dimension CUTER est l'addition de $m + p$. Retrouvons

ces différences dans le tableau Tab.C.2 qui explicite les noms et dimensions des problèmes de la base CUTEr que nous avons utilisés dans les expérimentations numériques de cette dissertation.

Problème	n CUTEr	m CUTEr	n	m	p
3PK	30	0	30	30	0
AGG	163	488	163	615	36
AIRCRFTA	8	5	8	0	8
AIRCRFTB	8	0	8	0	3
AIRPORT	84	42	84	210	0
AKIVA	2	0	2	0	0
ALJAZZAF	1000	1	1000	1000	1
ALLINITC	4	1	4	3	2
ALLINIT	4	0	4	3	1
ALLINITU	4	0	4	0	0
ALSOTAME	2	1	2	4	1
ARGAUSS	3	15	3	0	15
ARGLALE	200	400	200	0	400
ARGLBLE	200	400	200	0	400
ARGLCLE	200	399	200	0	399
ARGTRIG	200	200	200	0	200
AVGASA	8	10	8	26	0
AVGASB	8	10	8	26	0
AVION2	49	15	49	98	15
BARD	3	0	3	0	0
BATCH	48	73	48	155	12
... la suite sur la page suivante...					

... suite de la page précédente...					
BDEXP	5000	0	5000	5000	0
BDQRTIC	5000	0	5000	0	0
BEALE	2	0	2	0	0
BIGGS3	6	0	6	0	3
BIGGS5	6	0	6	0	1
BIGGS6	6	0	6	0	0
BIGGSB1	5000	0	5000	9998	0
BIGGSC4	4	7	4	21	0
BLOWEYA	4002	2002	4002	4002	2002
BLOWEYB	4002	2002	4002	4002	2002
BLOWEYC	4002	2002	4002	4002	2002
BOOTH	2	2	2	0	2
BOX2	3	0	3	0	1
BOX3	3	0	3	0	0
BQP1VAR	1	0	1	2	0
BQPGABIM	50	0	50	92	4
BQPGASIM	50	0	50	100	0
BRATU1D	5003	0	5003	0	2
BRITGAS	450	360	450	474	360
BRKMCC	2	0	2	0	0
BROWNALE	200	200	200	0	200
BROWNAL	200	0	200	0	0
BROWNBS	2	0	2	0	0
BROWNDEN	4	0	4	0	0
... la suite sur la page suivante...					

... suite de la page précédente...					
BROYDN7D	5000	0	5000	0	0
BRYBND	5000	0	5000	0	0
BT10	2	2	2	0	2
BT11	5	3	5	0	3
BT12	5	3	5	0	3
BT13	5	1	5	1	1
BT1	2	1	2	0	1
BT2	3	1	3	0	1
BT3	5	3	5	0	3
BT4	3	2	3	0	2
BT5	3	2	3	0	2
BT6	5	2	5	0	2
BT7	5	3	5	0	3
BT8	5	2	5	0	2
BT9	4	2	4	0	2
BYRDSPHR	3	2	3	0	2
CAMEL6	2	0	2	4	0
CAMSHAPE	800	1603	800	4004	0
CANTILVR	5	1	5	6	0
CATENARY	3003	1000	3003	0	1004
CATENA	3003	1000	3003	0	1004
CATMIX	2403	1600	2403	1602	1602
CBRATU2D	3200	2888	3200	0	3200
CBRATU3D	3456	2000	3456	0	3456
... la suite sur la page suivante...					

... suite de la page précédente...					
CHAIN	802	401	802	0	403
CHAINWOO	4000	0	4000	0	0
CHANDHEQ	100	100	100	100	100
CHEBYQAD	100	0	100	200	0
CHENHARK	5000	0	5000	5000	0
CHNROSNB	50	0	50	0	0
CLIFF	2	0	2	0	0
CLPLATEA	5041	0	5041	0	71
CLPLATEB	5041	0	5041	0	71
CLPLATEC	5041	0	5041	0	71
CLUSTER	2	2	2	0	2
CONCON	15	11	15	5	11
COOLHANS	9	9	9	0	9
CORE1	65	59	65	139	41
CORE2	157	134	157	299	108
COSHFUN	61	20	61	20	0
COSINE	10000	0	10000	0	0
CRAGGLVY	5000	0	5000	0	0
C-RELOAD	342	284	342	684	200
CRESC50	6	100	6	105	0
CUBENE	2	2	2	0	2
CUBE	2	0	2	0	0
DALLASM	196	151	196	392	151
DALLASS	46	31	46	92	31
... la suite sur la page suivante...					

... suite de la page précédente...					
DECONVB	61	0	61	72	0
DECONVC	61	1	61	51	11
DECONVNE	61	40	61	0	40
DECONVU	61	0	61	0	0
DEGENLPA	20	15	20	40	15
DEGENLPB	20	15	20	40	15
DEMBO7	16	20	16	53	0
DENSCHNA	2	0	2	0	0
DENSCHNB	2	0	2	0	0
DENSCHNC	2	0	2	0	0
DENSCHND	3	0	3	0	0
DENSCHNE	3	0	3	0	0
DENSCHNF	2	0	2	0	0
DISC2	29	23	29	20	17
DISCS	36	66	36	60	21
DITPERT	1133	1034	1133	1233	1034
DIXCHLNG	10	5	10	0	5
DIXMAANA	3000	0	3000	0	0
DIXMAANB	3000	0	3000	0	0
DIXMAANC	3000	0	3000	0	0
DIXMAAND	3000	0	3000	0	0
DIXMAANE	3000	0	3000	0	0
DIXMAANF	3000	0	3000	0	0
DIXMAANG	3000	0	3000	0	0
... la suite sur la page suivante...					

... suite de la page précédente...					
DIXMAANH	3000	0	3000	0	0
DIXMAANI	3000	0	3000	0	0
DIXMAANJ	3000	0	3000	0	0
DIXMAANK	15	0	15	0	0
DIXMAANL	3000	0	3000	0	0
DIXON3DQ	10000	0	10000	0	0
DJTL	2	0	2	0	0
DNIEPER	61	24	61	112	28
DQDRTIC	5000	0	5000	0	0
DQRTIC	5000	0	5000	0	0
DRCV1LQ	4489	0	4489	0	520
DRCV2LQ	4489	0	4489	0	520
DRCV3LQ	4489	0	4489	0	520
DRUGDISE	603	500	603	698	504
DTOC3	4499	2998	4499	0	3000
DTOC4	4499	2998	4499	0	3000
EDENSCH	2000	0	2000	0	0
EG1	3	0	3	4	0
EG2	1000	0	1000	0	0
EIGMAXA	101	101	101	202	101
EIGMAXB	101	101	101	202	101
EIGMINA	101	101	101	202	101
EIGMINB	101	101	101	202	101
ELATTAR	7	102	7	102	0
... la suite sur la page suivante...					

... suite de la page précédente...					
ENGVAL1	5000	0	5000	0	0
ENGVAL2	18	9	18	15	8
ERRINROS	50	0	50	0	0
EXPFITA	5	22	5	22	0
EXPFITB	5	102	5	102	0
EXPFIT	2	0	2	0	0
EXPLIN2	1200	0	1200	2400	0
EXPLIN	1200	0	1200	2400	0
EXPQUAD	1200	0	1200	200	0
EXTRASIM	2	1	2	1	1
EXTROSNB	1000	0	1000	0	0
FCCU	19	8	19	19	8
FEEDLOC	90	259	90	462	22
FMINSRF2	5625	0	5625	0	0
FMINSURF	5625	0	5625	0	0
FREUROTH	5000	0	5000	0	0
GENHS28	10	8	10	0	8
GENHUMPS	5000	0	5000	0	0
GENROSE	500	0	500	0	0
GILBERT	5000	1	5000	1	1
GOFFIN	51	50	51	50	0
GOTTFR	2	2	2	0	2
GPP	250	498	250	498	0
GRIDGENA	6218	0	6218	11120	658
... la suite sur la page suivante...					

... suite de la page précédente...					
GROWTHLS	3	0	3	0	0
GROWTH	3	12	3	0	12
GULF	3	0	3	0	0
HAGER1	5001	2500	5001	0	2501
HAGER2	5001	2500	5001	0	2501
HAGER3	5001	2500	5001	0	2501
HAGER4	5001	2500	5001	2500	2501
HAIRY	2	0	2	0	0
HALDMADS	6	42	6	42	0
HART6	6	0	6	12	0
HATFLDA	4	0	4	4	0
HATFLDB	4	0	4	5	0
HATFLDC	25	0	25	48	0
HATFLDD	3	0	3	0	0
HATFLDE	3	0	3	0	0
HATFLDF	3	3	3	0	3
HATFLDG	25	25	25	0	25
HATFLDH	4	7	4	21	0
HEART6LS	6	0	6	0	0
HEART6	6	6	6	0	6
HEART8LS	8	0	8	0	0
HEART8	8	8	8	0	8
HELIX	3	0	3	0	0
HILBERTA	2	0	2	0	0
... la suite sur la page suivante...					

... suite de la page précédente...					
HILBERTB	10	0	10	0	0
HIMMELBA	2	2	2	0	2
HIMMELBB	2	0	2	0	0
HIMMELBC	2	2	2	0	2
HIMMELBD	2	2	2	0	2
HIMMELBE	3	3	3	0	3
HIMMELBF	4	0	4	0	0
HIMMELBG	2	0	2	0	0
HIMMELBH	2	0	2	0	0
HIMMELBI	100	12	100	212	0
HIMMELBJ	45	14	45	43	16
HIMMELBK	24	14	24	24	14
HIMMELP1	2	0	2	4	0
HIMMELP2	2	1	2	5	0
HIMMELP6	2	5	2	9	0
HONG	4	1	4	8	1
HS100LNP	7	2	7	0	2
HS105	8	1	8	17	0
HS107	9	6	9	8	6
HS108	9	13	9	14	0
HS109	9	10	9	20	6
HS10	2	1	2	1	0
HS110	10	0	10	20	0
HS111LNP	10	3	10	0	3
... la suite sur la page suivante...					

... suite de la page précédente...					
HS111	10	3	10	20	3
HS112	10	3	10	10	3
HS114	10	11	10	28	3
HS116	13	14	13	41	0
HS118	15	17	15	59	0
HS119	16	8	16	32	8
HS11	2	1	2	1	0
HS12	2	1	2	1	0
HS13	2	1	2	3	0
HS1	2	0	2	1	0
HS21MOD	7	1	7	9	0
HS21	2	1	2	5	0
HS23	2	5	2	9	0
HS25	3	0	3	6	0
HS26	3	1	3	0	1
HS27	3	1	3	0	1
HS28	3	1	3	0	1
HS29	3	1	3	1	0
HS2	2	0	2	1	0
HS30	3	1	3	7	0
HS31	3	1	3	7	0
HS35I	3	1	3	7	0
HS35MOD	3	1	3	3	1
HS35	3	1	3	4	0
... la suite sur la page suivante...					

... suite de la page précédente...					
HS36	3	1	3	7	0
HS38	4	0	4	8	0
HS39	4	2	4	0	2
HS3MOD	2	0	2	1	0
HS3	2	0	2	1	0
HS40	4	3	4	0	3
HS41	4	1	4	8	1
HS42	4	2	4	0	2
HS45	5	0	5	10	0
HS46	5	2	5	0	2
HS47	5	3	5	0	3
HS48	5	2	5	0	2
HS49	5	2	5	0	2
HS4	2	0	2	2	0
HS50	5	3	5	0	3
HS51	5	3	5	0	3
HS52	5	3	5	0	3
HS53	5	3	5	10	3
HS54	6	1	6	12	1
HS55	6	6	6	8	6
HS56	7	4	7	0	4
HS57	2	1	2	3	0
HS5	2	0	2	4	0
HS60	3	1	3	6	1
... la suite sur la page suivante...					

... suite de la page précédente...					
HS61	3	2	3	0	2
HS62	3	1	3	6	1
HS63	3	2	3	3	2
HS64	3	1	3	4	0
HS65	3	1	3	7	0
HS67	3	14	3	20	0
HS68	4	2	4	8	2
HS69	4	2	4	8	2
HS6	2	1	2	0	1
HS70	4	1	4	9	0
HS77	5	2	5	0	2
HS78	5	3	5	0	3
HS79	5	3	5	0	3
HS7	2	1	2	0	1
HS80	5	3	5	10	3
HS81	5	3	5	10	3
HS85	5	21	5	48	0
HS86	5	10	5	15	0
HS87	6	4	6	12	4
HS88	2	1	2	1	0
HS89	3	1	3	1	0
HS8	2	2	2	0	2
HS90	4	1	4	1	0
HS91	5	1	5	1	0
... la suite sur la page suivante...					

... suite de la page précédente...					
HS92	6	1	6	1	0
HS99EXP	6	2	6	8	0
HS99	31	21	31	14	24
HS9	7	2	7	14	2
HUBFIT	2	1	2	0	1
HUES-MOD	2	1	2	2	0
HUESTIS	5000	2	5000	5000	2
HUMPS	5000	2	5000	5000	2
HVYCRASH	2	0	2	0	0
HYDC20LS	4004	3000	4004	4004	3002
HYDCAR20	99	0	99	0	0
HYDCAR6	99	99	99	0	99
HYDROELL	29	29	29	0	29
HYDROELM	1009	1008	1009	4030	2
HYDROELS	505	504	505	2014	2
HYP CIR	169	168	169	670	2
INDEF	2	2	2	0	2
JENSMP	2	0	2	0	0
JNLBRNG1	10000	0	10000	9604	396
JNLBRNG2	10000	0	10000	9604	396
JNLBRNGA	10000	0	10000	9604	396
JNLBRNGB	10000	0	10000	9604	396
KISSING2	100	625	100	625	6
KISSING	127	903	127	861	42
... la suite sur la page suivante...					

... suite de la page précédente...					
KOWOSB	4	0	4	0	0
LAKES	90	78	90	18	78
LAUNCH	25	28	25	70	9
LEAKNET	156	153	156	82	153
LIARWHD	5000	0	5000	0	0
LIN	4	2	4	8	2
LINSPANH	97	33	97	162	49
LINVERSE	1999	0	1999	1000	0
LISWET10	2002	2000	2002	2000	0
LMINSURF	5625	0	5625	0	296
LOADBAL	31	31	31	62	11
LOGHAIRY	2	0	2	0	0
LOGROS	2	0	2	2	0
LOTSCHD	12	7	12	12	7
LSNNODOC	5	4	5	6	4
LSQFIT	2	1	2	2	0
MANCINO	100	0	100	0	0
MARATOSB	2	0	2	0	0
MARATOS	2	1	2	0	1
MATRIX2	6	2	6	6	0
MAXLIKA	8	0	8	16	0
MCCORMCK	5000	0	5000	10000	0
MCONCON	15	11	15	5	11
MDHOLE	2	0	2	1	0
... la suite sur la page suivante...					

... suite de la page précédente...					
MESH	41	48	41	72	32
METHANB8	31	31	31	0	31
METHANL8	31	31	31	0	31
MEXHAT	2	0	2	0	0
MEYER3	3	0	3	0	0
MIFFLIN1	3	2	3	2	0
MIFFLIN2	3	2	3	2	0
MINC44	1113	1032	1113	1193	1042
MINMAXBD	5	20	5	20	0
MINMAXRB	3	4	3	4	0
MINPERM	1113	1033	1113	1213	1033
MINSURFO	5306	0	5306	5002	304
MINSURF	64	0	64	0	28
MISTAKE	9	13	9	14	0
MODEL	1542	38	1542	99	1523
MOREBV	5000	0	5000	0	0
MOSARQP1	2500	700	2500	3200	0
MOSARQP2	2500	700	2500	3200	0
MRIBASIS	36	55	36	94	21
MSQRTALS	1024	0	1024	0	0
MSQRTA	1024	1024	1024	0	1024
MSQRTBLS	1024	0	1024	0	0
MSQRTB	1024	1024	1024	0	1024
MSS3	2070	1981	2070	0	1981
... la suite sur la page suivante...					

... suite de la page précédente...					
MWRIGHT	5	3	5	0	3
NASH	72	24	72	6	78
NCVXQP1	1000	500	1000	2000	500
NCVXQP1	1000	500	1000	2000	500
NET1	48	57	48	65	43
NET2	144	160	144	173	134
NLMSURF	5625	0	5625	0	296
NOBNDTOR	5476	0	5476	5184	292
NONCVXU2	5000	0	5000	0	0
NONCVXUN	5000	0	5000	0	0
NONDIA	5000	0	5000	0	0
NONDQUAR	5000	0	5000	0	0
NONSCOMP	5000	0	5000	10000	0
NUFFIELD	940	5000	940	5000	1
ODC	5184	0	5184	0	284
ODFITS	10	6	10	10	6
OPTCDEG2	4502	3000	4502	4499	3003
OPTCDEG3	4502	3000	4502	4499	3003
OPTCNTRL	32	20	32	30	23
OPTCTRL3	4502	3000	4502	0	3003
OPTCTRL6	4502	3000	4502	0	3003
OPTMASS	3010	2505	3010	501	2008
OPTPRLOC	30	30	30	90	0
ORBIT2	2698	2097	2698	2095	1803
... la suite sur la page suivante...					

... suite de la page précédente...					
ORTHRDS2	5003	2500	5003	0	2500
ORTHREGB	27	6	27	0	6
ORTHREGC	5005	2500	5005	0	2500
ORTHREGD	5003	2500	5003	0	2500
ORTHREGF	4805	1600	4805	2	1600
ORTHRGDS	5003	2500	5003	0	2500
OSBORNEA	5	0	5	0	0
OSBORNEB	11	0	11	0	0
OSLBQP	8	0	8	11	0
PALMER1A	6	0	6	2	0
PALMER1B	4	0	4	2	0
PALMER1C	8	0	8	0	0
PALMER1D	7	0	7	0	0
PALMER1E	8	0	8	1	0
PALMER1	4	0	4	3	0
PALMER2A	6	0	6	2	0
PALMER2B	4	0	4	2	0
PALMER2C	8	0	8	0	0
PALMER2E	8	0	8	1	0
PALMER2	4	0	4	3	0
PALMER3A	6	0	6	2	0
PALMER3B	4	0	4	2	0
PALMER3C	8	0	8	0	0
PALMER3E	8	0	8	1	0
... la suite sur la page suivante...					

... suite de la page précédente...					
PALMER3	4	0	4	3	0
PALMER4A	6	0	6	2	0
PALMER4B	4	0	4	2	0
PALMER4C	8	0	8	0	0
PALMER4E	8	0	8	1	0
PALMER4	4	0	4	3	0
PALMER5A	8	0	8	2	0
PALMER5B	9	0	9	2	0
PALMER5C	6	0	6	0	0
PALMER5D	8	0	8	4	0
PALMER5E	8	0	8	1	0
PALMER6A	6	0	6	2	0
PALMER6C	8	0	8	0	0
PALMER6E	8	0	8	1	0
PALMER7A	6	0	6	2	0
PALMER7C	8	0	8	0	0
PALMER7E	8	0	8	1	0
PALMER8A	6	0	6	2	0
PALMER8C	8	0	8	0	0
PALMER8E	8	0	8	1	0
PENALTY1	1000	0	1000	0	0
PENALTY2	200	0	200	0	0
PENTAGON	6	15	6	15	0
PENTDI	5000	0	5000	5000	0
... la suite sur la page suivante...					

... suite de la page précédente...					
PFIT1LS	3	0	3	1	0
PFIT1	3	3	3	1	3
PFIT2LS	3	0	3	1	0
PFIT2	3	3	3	1	3
PFIT3LS	3	0	3	1	0
PFIT3	3	3	3	1	3
PFIT4LS	3	0	3	1	0
PFIT4	3	3	3	1	3
POLAK3	12	10	12	10	0
PORTFL1	12	1	12	24	1
PORTFL2	12	1	12	24	1
PORTFL3	12	1	12	24	1
PORTFL4	12	1	12	24	1
PORTFL6	12	1	12	24	1
POWELLBS	2	2	2	0	2
POWELLSG	5000	0	5000	0	0
POWELLSQ	2	2	2	0	2
POWER	10000	0	10000	0	0
PRIMALC1	230	9	230	224	0
PRIMALC2	231	7	231	236	0
PRIMALC5	287	8	287	286	0
PROBPENL	500	0	500	1000	0
PRODPL0	60	29	60	69	20
PRODPL1	60	29	60	69	20
... la suite sur la page suivante...					

... suite de la page précédente...					
PSPDOC	4	0	4	1	0
QPCBLEND	83	74	83	114	43
QPCBOEI2	143	166	143	378	4
QPNBLEND	83	74	83	114	43
QPNBOEI2	143	166	143	378	4
QR3DLS	610	0	610	20	0
QR3D	610	610	610	20	610
QRTQUAD	5000	0	5000	2200	0
QUARTC	5000	0	5000	0	0
QUDLIN	5000	0	5000	10000	0
RAYBENDL	2050	0	2050	0	4
RAYBENDS	2050	0	2050	0	4
READING1	4002	2000	4002	8002	2001
READING3	4002	2001	4002	8004	2001
READING6	102	50	102	102	51
READING7	1002	500	1002	1002	501
RECIPE	3	3	3	0	3
RES	20	14	20	42	12
RK23	17	11	17	6	11
ROBOTARM	4412	3202	4412	4801	3214
ROBOT	14	2	14	14	9
ROCKET	2407	2002	2407	2401	2006
ROSENBR	2	0	2	0	0
ROTDISC	905	1081	905	1604	733
... la suite sur la page suivante...					

... suite de la page précédente...					
RSNBRNE	2	2	2	0	2
S308	2	0	2	0	0
S316-322	2	1	2	0	1
S368	8	0	8	16	0
SAWPATH	583	774	583	196	580
SBRYBND	5000	0	5000	0	0
SCHMVETT	5000	0	5000	0	0
SCOND1LS	5002	0	5002	10000	2
SCOSINE	5000	0	5000	0	0
SCURLY10	10000	0	10000	0	0
SCURLY20	10000	0	10000	0	0
SENSORS	100	0	100	0	0
SIM2BQP	2	0	2	2	1
SIMBQP	2	0	2	2	0
SIMPLLPA	2	2	2	4	0
SINEALI	1000	0	1000	2000	0
SINEVAL	2	0	2	0	0
SINQUAD	5000	0	5000	0	0
SINROSNB	1000	999	1000	2000	0
SINVALNE	2	2	2	0	2
SISSER	2	0	2	0	0
SMBANK	117	64	117	234	64
SMMPFS	720	263	720	743	240
SNAIL	2	0	2	0	0
... la suite sur la page suivante...					

... suite de la page précédente...					
SOSQP1	5000	2501	5000	10000	2501
SOSQP2	5000	2501	5000	10000	2501
SPARSINE	5000	0	5000	0	0
SPARSQUR	10000	0	10000	0	0
SPECAN	9	0	9	18	0
SPMSRTLS	4999	0	4999	0	0
SREADIN3	4002	2001	4002	8004	2001
SROSENBR	5000	0	5000	0	0
SSC	5184	0	5184	0	284
SSEBLIN	194	72	194	384	50
SSEBNLN	194	96	194	384	74
SSNLBEAM	3003	2000	3003	4000	2002
STEENBRA	432	108	432	432	108
STEENBRB	468	108	468	468	108
STEENBRC	540	126	540	540	126
STEENBRD	468	108	468	468	108
STEENBRE	540	126	540	540	126
STEENBRF	468	108	468	468	108
STEENBRG	540	126	540	540	126
STEERING	2006	1600	2006	803	1607
SUPERSIM	2	2	2	1	2
SWOPF	83	92	83	34	78
SYNTHE1	6	6	6	18	0
SYNTHE2	11	14	11	34	1
... la suite sur la page suivante...					

... suite de la page précédente...					
SYNTHE3	17	23	17	55	2
TAME	2	1	2	2	1
TENBARS2	18	8	18	14	8
TENBARS3	18	8	18	12	8
TESTQUAD	5000	0	5000	0	0
TFI1	3	101	3	101	0
TFI2	3	101	3	101	0
TFI3	3	101	3	101	0
TOINTGOR	50	0	50	0	0
TOINTGSS	5000	0	5000	0	0
TOINTPSP	50	0	50	0	0
TOINTQOR	50	0	50	0	0
TORSION1	5476	0	5476	10368	292
TORSION2	5476	0	5476	10368	292
TORSION3	5476	0	5476	10368	292
TORSION4	5476	0	5476	10368	292
TORSION5	5476	0	5476	10368	292
TORSION6	5476	0	5476	10368	292
TORSIONA	5476	0	5476	10368	292
TORSIONB	5476	0	5476	10368	292
TORSIONC	5476	0	5476	10368	292
TORSIOND	5476	0	5476	10368	292
TORSIONE	5476	0	5476	10368	292
TORSIONF	5476	0	5476	10368	292
... la suite sur la page suivante...					

... suite de la page précédente...					
TQUARTIC	5000	0	5000	0	0
TRAINF	4008	2002	4008	4000	2010
TRAINH	4008	2002	4008	4000	2010
TRIDIA	5000	0	5000	0	0
TRIGGER	7	6	7	0	7
TRIMLOSS	142	75	142	319	20
TRUSPYR2	11	11	11	16	3
TRY-B	2	1	2	2	1
TWIRISM1	343	313	343	775	224
VANDERM1	100	199	100	99	100
VANDERM2	100	199	100	99	100
VANDERM3	100	199	100	99	100
VANDERM4	100	199	100	99	100
VARDIM	200	0	200	0	0
VAREIGVL	50	0	50	0	0
VIBRBEAM	8	0	8	0	0
WATER	31	10	31	62	10
WATSON	12	0	12	0	0
WEEDS	3	0	3	4	0
WOODSNE	4000	0	4000	0	0
WOODS	2002	2000	2002	2001	2
YORKNET	312	256	312	288	256
ZAMB2-10	270	96	270	528	102
ZAMB2-11	270	96	270	528	102
... la suite sur la page suivante...					

... suite de la page précédente...					
ZAMB2-8	138	48	138	264	54
ZAMB2-9	138	48	138	264	54
ZAMB2	3966	1440	3966	7920	1446
ZANGWIL2	2	0	2	0	0
ZANGWIL3	3	3	3	0	3
ZECEVIC2	2	2	2	6	0
ZECEVIC3	2	2	2	6	0
ZECEVIC4	2	2	2	6	0
ZIGZAG	3004	2500	3004	4000	2006
ZY2	3	2	3	6	0

TAB. C.2: Index des problèmes de la base CUTEr utilisées dans les expériences numériques.

C.2 Le langage de modélisation AMPL

Le langage AMPL (Langage de Modélisation Algébrique pour la Programmation mathématique) a été proposé et implémenté vers 1985 par R. Fourer, D. M. Gay et B. W. Kernighan [25]. Il est proche du langage humain, d'où l'aisance de modéliser ou de lire un programme écrit en AMPL. Malheureusement les fichiers de données AMPL sont plus grands que ceux écrits dans l'environnement CUTEr. Pour faciliter le traitement de gros fichiers, on peut modéliser les problèmes en deux fichiers, le premier `.mod` contient les déclarations des données : ensembles, variables, paramètres, contraintes, commentaires,... et le deuxième (`.dat`) regroupe les données : valeurs des paramètres, désignations des variables ou des ensembles. Outre les fichiers de données (`.mod` et `.dat`), AMPL

offre un environnement de commandes interactif pour reprendre un problème, changer un point initial, ou charger un autre solveur. Une panoplie d'options est proposée pour permettre à l'utilisateur plus de choix dans la résolution et l'analyse des données et des résultats.

Soit `problem1` un problème modélisé par le langage AMPL dans les fichiers `problem1.mod` et `problem1.dat`, alors la résolution de ce problème dans l'environnement AMPL se fait grâce aux commandes suivantes :

```
>ampl
ampl>model problem1.mod ;
ampl>data problem1.dat ;
ampl>solve ;
```

Algorithme 21: Les commandes de base pour le langage AMPL

Le solveur par défaut est le programme MINOS [58]. Ce dernier sera appelé pour résoudre le problème `problem1` et produira des messages relatifs à son exécution, par exemple :

```
MINOS 5.5 : optimal solution found.
```

```
12 iterations, objective 12032
```

Pour travailler avec un autre solveur, en prenant par exemple le solveur SDC, l'utilisateur doit exécuter la commande

```
ampl>option solver sdc ;
ampl>solve ;
```

Algorithme 22: Utiliser le code SDC en langage AMPL

Sans choisir un autre problème à résoudre, c'est le dernier problème chargé qui sera résout, dans notre cas il s'agit de `problem1`.

La commande

```
ampl>quit ;
```

termine l'exécution et renvoi le curseur vers le système d'opération ou le shell d'origine. D'autres commandes et options sont disponibles (consulter le livre [25] et le site [95]). Les méthodes de résolutions LOQO, SNOPT et KNITRO que nous avons présentés au chapitre 4 sont utilisables, sous certaines conditions, avec la programmation AMPL.

BIBLIOGRAPHIE

- [1] C. M. Ablow et G. Brigham (1955), An Analog Solution of Programming Problems, *Operation Res.*, 3, pp 388-394.
- [2] M. Anitescu (2002), A superlinearly convergent sequential quadratically constrained quadratic programming algorithm for degenerate nonlinear programming, *SIAM J. Optim.*, Vol 12, pp. 949-978.
- [3] K. J. Arrow (1951), Agradient Method for Approximating Points and constrained Maxim, RAND Corp., Santa Monica, California, Paper P 223, 15 pp.
- [4] K. M. Austreicher et J. P. Vial (1994), on the convergence of an infeasible primal-dual interior point method for convex programming. *Optimization Methods ans software*, 3 :273-283.
- [5] L. T. Biegler et A. Wächter (2000), Failure of a class of interior Point Methods for nonlinear Programming. *Math. Program.*, 88, pp. 565-587.
- [6] L. T. Biegler et A. Wachter (2001). Global and local convergence of line search filter methods or nonlinear programming, *Mathematical Programming* 88(3), pp 565-587,2002.
- [7] M. Bouhtou. Méthodes de points intérieurs pour l'optimisation des systèmes de grande taille. Thèse de Doctorat. Université Paris Dauphine. 1993.
- [8] R. H. Byrd, J. C. Gilbert, et J. Nocedal (2000), A trust region method based on interior point techniques for nonlinear programming. *Math. program.*, 89, pp. 149-185.
- [9] R. H. Byrd, M. E. Hribar, et J. Nocedal (1999), An interior point method for large scale nonlinear programming, *SIAM J. OPIM.* V.9, No.4, pp.877-900.

- [10] R. H. Byrd, J. Nocedal, et R. A. Waltz (2000), Feasible Interior Method Using Slacks for Nonlinear Optimization, Report OTC-00/04, Optimization Technology Center, Northwestern University.
- [11] C. W. Carroll (1959), An Operations Research Approach to the Economic Optimization of a Kraft Pulping Process, Ph.D. Dissertation, Institute of Paper Chemistry, Appleton, Wis.
- [12] L. Chauvier, A. Fudilli, et J-C Gilbert (2001), A Truncated SQP algorithm for solving nonconvex equality optimization problems, Rapport de recherche n 4346, Thème 4, INRIA Rocquencourt.
- [13] A. R. Conn, N. I. M. Gould et Ph. L. Toint (1995), CUTE :Constrained and Unconstrained Testing Environment. ACM Transactions on Mathematical Software, 21(1) : 123-160.
- [14] A. R. Conn, N. I. M. Gould et Ph. L. Toint (1997), A globally convergent Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds, Math. Comp., 66, pp. 261-288, S1-S11.
- [15] R. Courant (1943), Variational Methods for Solution of Problems of Equilibrium and Vibrations, Bull. AM. Math. Soc., 49 : 1-23.
- [16] I. I. Dikin (1967) : iterative solutions of linear and quadratic programming, Soviet Math. Doklady 8, 674-675.
- [17] E. D. Dolan et J. J. Moré ((2000). Benchmarking optimization software with COPS. Mathematics and Computer Science Technical Report ANL/MCS-246, Argonne National Laboratory, Argonne, Illinois, USA.
- [18] A. S. El-Bakry, R. A. Tapia (1999), T. Tsuchiga, and Y. Zhang. on the formulation and theory of Newton interior-point method for nonlinear programming. Technical report, Department of computational and applied Mathematics Rice University.

- [19] A. V. Fiacco et G. P. McCormik, *Nonlinear Programming, Sequential Unconstrained Minimization Techniques*, Classics in Appl. Math., SIAM, Philadelphia, PA, 1990. Rééditer de l'originale de 1968.
- [20] R. Fletcher, N. I. M. Gould, S. Leyffer et Ph. L. Toint (1999), *Global Convergence of Trust-region Nonlinear Programming*. Report 99/03 Rutherford Appleton Laboratory, Computational science and Engineering Departement Chilton, Oxfordshire. England.
- [21] R. Fletcher et S. Leyffer (1997), *Nonlinear Programming without a penalty function*. Numerical Analysis Report NA/171, Departement of Mathematics, University of Dundee, Dundee, Scotland.
- [22] R. Fletcher et S. Leyffer et Ph. L. Toint (2002), *On the global convergence of a filter-SQP Algorithm*. SIAM J. Optim. vol.13, n 1, pp 44-59.
- [23] A. Forsgren, P. E. Gill and M. H. Wright (2002). *Interior methods for Nonlinear Optimization*. SIAM review, Vol.44,No4, pp. 525-597.
- [24] R. Fletcher et T. Johnsson (1996), *On the stability of the null-space methods for KKT systems*. SIAM Journal on Matrix Anal. Appl., Octobre 1997, vol. 18, n° 4, pp. 938-958.
- [25] R. Fourer, D. M. Gay et B. W. Kernighan (2003). *AMPL : A Modeling language For Mathematical Programming*. Brooks/Cole-Thomson Learning. Pacific Grove USA.
- [26] K. R. Frisch (1954) *Priciples of Linear Programming-With Particular Reference to the Double Gradient Form of the Logarithmic Potential Method*, Memorandum of October 18, 1954, University of Oslo.
- [27] M. Fukushima, Z-Q. Luo et P. Tseng (2003), *A Sequential Quadratically constrained Quadratic programming method for differentiable convex minimization*, SIAM J. Optim., Vol 13, number 4, pp. 1098-1119.

- [28] P. E. Gill, W. Murray, et M. A. Saunders (2002), SNOPT : An SQP Algorithm for Large-scale Constrained Optimization. *SIAM J. Optim.* Vol 12 No. 4, pp 979-1006.
- [29] P. E. Gill, W. Murray, et M. A. Saunders (1997), User's guide for SQOPT 5.3 : A Fortran Package for Large-Scale Linear and Quadratic Programming, Numerical Analysis Report 97-4, Departement of Mathematics, University of California, San Diego, La Jolla, CA.
- [30] P. E. Gill, W. Murray, et M. H. Wright (1981), Practical optimization. Academic Press.
- [31] I. M. Gould, D. Orban, et P. L. Toint (2002), CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, revisited. Rutherford Appleton Laboratory.
- [32] N. I. M. Gould (1985), On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem, *Math. Program.*, 32 , pp. 90-99.
- [33] N. I. M. Gould (1986), On the accurate determination of search directions for simple differentiable penalty functions, *IMA J. Numer. Anal.*, 6, pp357-372
- [34] N. I. M. Gould, et Ph. L. Toint (2000), A quadratic programming Bibliography. Numerical Analysis Group Internal Report 2000-1, Rutherford Appleton Laboratory, Chilton, England.
- [35] N. I. M. Gould, et Ph. L. Toint (2001), Numerical methods for large-scale non-convex quadratic programming. TR RAL-TR-2001-017 (2001), Rutherford Appleton Laboratory, Chilton, England.
- [36] M.E. Hallabi et R. A. Tapia (1993), A global convergence theory for arbitrary norm trust-region methods for non linear equations, Tech. Rep.

- TR93-41, Departement of Computational and Applied Mathematics, Rice University, Houston, TX.
- [37] M.E. Hallabi (2003), A Hybrid algorithm for nonlinear equality constrained optimization problems : Global and local convergence theory. un préprint sur le site www.optimization-online.org. Nonlinear Programming.
- [38] S. P. Han et O. L. Mangasarian (1979). Exact penalty functions in nonlinear programming. *Mathematical Programming*, vol. 17, pp. 251-269.
- [39] C. R. Hargraves et S. W. Paris (1997). Direct trajectory optimization using nonlinear programming problems, *Math. Programming*, 11, pp. 263-282.
- [40] N. J. Higham (1961), Accuracy and stability of numerical algorithms. Library of congress Cataloging-in-Publication Data. Rééditer en 1996 par The Society for Industrial and Applied Mathematics.
- [41] J. B. Hiriart-Urruty (1982). Subdifferential calculus in convex analysis and optimization (J. P. Aubin and R. Vinter, eds). Pitman, pp. 43-92.
- [42] J. B. Hiriart-Urruty (1985), Generalized differentiability, duality and optimization for problems dealing with differences of convex functions. *Lecture Notes in Economics and Mathematical System*, 256, pp. 37-70.
- [43] J. B. Hiriart-Urruty and C. Lemarechal (1993), *Convex analysis and minimization algorithms I*. Springer-Verlag Berlin Heidelberg.
- [44] M. E. Hribar (1996), Large-Scale constrained optimization. PHD Thesis. Northwestern University Evanston, Illinois.
- [45] P. Huard (1967) : Resolution of mathematical programming with nonlinear constraints by the method of centers. *Nonlinear programming*, J. Abadie (ed). John Wiley and Sons, New York 1967, 207-219.

- [46] X. Jonsson (2002), MPI avec RC en optimisation non linéaire.à la conception de verres ophtalmique progressif. Thèse de Doctorat, Paris 6.
- [47] A. Kadiri (2001), Analyse numérique des méthodes de points intérieurs pour les problèmes de complémentarité linéaire et la programmation quadratique convexe. Thèse de Doctorat, INSA de Rouen.
- [48] N. K. Karmarkar (1984), A Polynomial-time algorithm for linear programming, *combinatorica*, 4 :373-395.
- [49] S. Kruk et H. Wolkowicz (1998), SQ2P, sequential quadratic constrained quadratic programming, in *Advances in Nonlinear Programming*, Y. X. Yuan, ed., Kluwer Academic Publishers, Dorderchet, The Netherlands, pp. 177-204.
- [50] S. Kruk et H. Wolkowicz (1998), Sequential quadratic constrained quadratic programming for general nonlinear programming, in *Handbook of semidefinite Programming*, H. Wolkowicz, R. Saigal, and L. Vandenberghe, eds., Kluwer Academic Publishers, Bonton, MA, pp. 563-575.
- [51] H. W. Kuhn et A. W. Tucker (1951), Non-linear Programming. in J. Neyman (Ed.), *Proceedings of the Second Berkeley Symposium On Mathematical Statistics and Probability*, University of California Press, Berkeley, pp. 481-493.
- [52] K. Levenberg (1944), A method for the solution of certain problems in least squares, *Quart. Appli. Math.*, 2, pp. 164-168.
- [53] O. L. Lootsama (1969), Hessian matrices of penalty functions for solving constrained-optimization problems, *Philips Res. Rep.*, 24, pp. 322-330.
- [54] O. L. Mangasarian et S. Fromowitz (1967), The Fritz John Necessary Optimality Conditions in the Presence of Equality and Inequality Constraints, *J. Math. Anal. Appl.*, 17(1) : 37-47.

- [55] N. Maratos (1978), Exact penalty function algorithms for finite dimensional and control optimization problems, Ph.D. Dissertation, University of London.
- [56] D.W. Marquardt (1963), An algorithm for least-squares estimation of nonlinear parameters, *J. Optim. SIAM*, 11, pp. 431-441.
- [57] M. Minoux (1983), *Programmation Mathématique, théorie et algorithmes. Tome 1. Collection Technique et Scientifique de Télécommunications.*
- [58] B. A. Murtagh et M. A. Saunders (1995), MINOS 5.4 user's guide. Technical report, SOL 83-20 R, Systems Optimization Laboratory, Stanford University, 1983. Revised 1995.
- [59] J. L. Morales, J. Nocedal, R. A. Waltz, G. Liu et J-P. Goux (2003). Assessing the potential of interior methods for nonlinear optimization.
- [60] W. Murray (1967), Ill-Conditioning in Barrier and Penalty Functions Arising in Constrained Non-linear Programming, paper presented at Princeton Mathematical Programming Symposium, August 14-18, 1967.
- [61] S. G. Nash, R. Polyak et A. Sofer (1994), Numerical comparaison of barrier and modified-barrier methods for large-scale bound-constrained optimization, in *Large-scale Optimization : State of the Art*, D. W. Hearn and P. M. Pardalos, eds., Kluwer Academics, Dordrecht, The Netherlands, pp. 319-338.
- [62] J. Nocedal, S.J.Wright (1999), *Numerical Optimization. Springer Series in operations research.* Springer, New York.
- [63] E. Omojoun (1989), *Trust Region Algorithms for Optimization with nonlinear Equality and Inequality constraints.* Phd Thesis, Univeristy of colorado, 1989.
- [64] M. Ouriemchi and A. Yassine (2003), An Interior Point Algorithm for Nonlinear Programming. soumis à *Control & Cybernetics* review.

- [65] V. M. Panin (1979), A Second-order method for the discrete min-max problem, U.S.S.R. Comput. Math. and Math. Phys., 19 (1), pp. 90-100.
- [66] V. M. Panin (1981), A SomeS methods of solving convex programming problems, U.S.S.R. Comput. Math. and Math. Phys., 21 (2), pp. 90-100.
- [67] T. Pham Dinh (1988) Duality in d.c. (difference of convex functions) optimization. Subgradient methods. Trends in Mathematical Optimization, International Series of Numer Math., vol. 84, Birkhauser, pp. 277-293.
- [68] M. J. D. Powell (1970), A Hybrid method for nonlinear equations. I P. Rabinowitz, editor, Numerical Methods for Nonlinear Algebraic Equations. Gordon and Breach.
- [69] M. J. D. Powell (1970), A new algorithm for unconstrained optimization, in Nonlinear Programming, O. Mangasarian, R. Meyer, and S. Robinson, eds., Academic Press, New York, pp. 31-65.
- [70] M. J. D. Powell (1975), Convergence properties of a class of minimization algorithm, in Nonlinear Programming 2, O. Mangasarian, R. Meyer, and S. Robinson, eds., Academic Press, New York, pp. 1-27.
- [71] M. J. D. Powell (1976), Some global coverage properties of a variable metric algorithm for minimization without exact linesearch, in Nonlinear Programming, SIAM-AMS Proceeding, vol. 9, AMS, Providence, R.I.
- [72] M. J. D. Powell (1977), The coverage of variable metric methods for nonlinearly constrained optimization calculations, in Nonlinear Programming 3 (Proceeding of the Symposium, Special Interest Group Mathematical Programming. University of Wisconsin, Madison, WI, 1977), Academic Press, New York, pp. 27-63.
- [73] R. T. Rocckafellar (1970), Convex Analysis. Princeton University, Princeton.

- [74] G. S. Shultz, R.B. Schnabel and R.H. Byrd (1985), A family of trust-region-based algorithms for unconstrained minimization with strong global convergences properties, *SIAM J. Numer.Anal.*, 22, pp. 47-67.
- [75] M. Sakarovich (1984), *Graphes et programmation lineaire : "Méthodes mathématiques et algorithmiques"* . Hermann, Paris
- [76] A. Sartnear (1997), Automatic Determination of an Initial Trust Region in Nonlinear Programming, *SIAM journal on Scientific Computing*, Vol. 18, n. 6, pp. 1788-1803.
- [77] A. Sartnear (2003), Some Recent Developements in Nonlinear Optimization Algorithms, *ESAIM Proceeding*, December 2003, Vol 13, pp 41-64 J.P. Penot Editor.
- [78] T. Steihaug (1983), The conjugate gradient method and trust regions in large scale optimization, *SIAM Journal on Numerical Analysis*, 20, pp.626-637.J
- [79] P. L. Toint (1981), Towards an efficient sparsity exploiting Newton method for minimiation. in I. Duff, editor, *Sparse Matrices and their Uses*, pp. 57-87. Academic Press, New York.
- [80] J. F. Toland (1978), Duality in nonconvex optimization. *Journal Mathematical Analysis and Applications*, vol. 66, pp. 399-415.
- [81] J. F. Toland (1979), On subdifferential calculus and duality in nonconvex optimization. *Bull. Soc. Math. France*, Mémoire 60, pp. 173-180.
- [82] J. F. Toland (1979) A duality principle for non convex optimization and calculus of variations. *Arch. Rational. Mech. Analysis*, vol. 71.
- [83] M. Ulbrich, S. Ulbrich, et L.N. Vicente (2002). A globally convergente primal-dual interior-point filter method nonlinear programming. Preprint 00-11 Dept of maths, University of Coimbra, Portugal April.

- [84] R. J. Vandrbei (1996), *Linear Programming : Foundations and Extensions*. Kluwer Academic Publishers.
- [85] R. J. Vandrbei et D. F. Shanno (1999), An Interior Point Algorithm fo Nonconvex Nonlinear Programming. *Computational Optimization and Applications*, 13 : 231-252.
- [86] J. H. Wilkinson (1965), *The algebraic Eigenvalue Problem*. Oxford University Press, 1965 xviii pp 662.
- [87] J. H. Wilkinson (1965), Error analysis of transformations based on the use of matrices of the form $I - 2ww^H$. In *Error in Digital Computation*, Louis B. Rall, editor, volume 2, Wiley, New Yrok pages 77-101.
- [88] M. H. Wright (1994), Some properties of the Hessian of the logarithmic barrier function, *Math. Program.*, 67, pp. 265-295.
- [89] M. H. Wright (2005) *The Interior-Point Revolution in Optimization : History, Recent Developments, And Lasting Consequences*. Bulletin (New series) of the AMS, vol. 42, N 1, pp. 93-56.
- [90] A. Yassine (1989), *Études adaptatives et comparatives de certains algorithmes en optimisation. Implimentations iffectives et applications*. Thèse de Doctorat Grenoble I.
- [91] A. Yassine (1997), Sub-gradient algorithms for solving multidimensional analysis problems of dissimilarity data. *Journal of Applied Mathematics and Computer Science*, vol 7, n 3, pp. 521-543.
- [92] A. Yassine (1998), Sub-gradient algorithms for computation of extreme eigenvalues of a real symmetric matrix. *Control and Cybernetics*, vol. 27, n 3, pp. 387-415.
- [93] A. Yassine (1998), *Méthodes de Région de Confiance et d'Optimisation D.C. Théorie, Algorithmes et Applications*. HDR Université Henri Poincaré-Nancy I.

- [94] A. Yassine, N. Alaa and A. Elhilali (2001), Convergence of Tholand's critical points for sequences of D.C. functions and application to the resolution of semilinear elliptic problems. Control and Cybernetics, vol. 30, n 4, pp. 405-417.

Sites sur la toile :

- [95] <http://www.ampl.com>
- [96] <http://cuter.rl.ac.uk/cuter-www/>
- [97] <http://numawww.mathematik.tu-darmstadt.de:8081/opti/select.html>
- [98] <http://www-lmah.univ-lehavre.fr>
- [99] <http://www.ziena.com/knitro.html>
- [100] <http://www.orfe.princeton.edu/loqo>
- [101] <http://www.optimization-online.org>
- [102] http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide/Blurbs/sqopt_snopt.html