Programmation non linéaire

Jean-Pierre Dussault[1]

4 septembre 2001

[1]Professeur titulaire, département de Mathématiques et Informatique, Université de Sherbrooke, Sherbrooke , Canada J1K 2R1

Table des matières

1 Préliminaires **11**
1.1 Optimisation . . . 12
1.2 Types d'optima . . . 13
1.2.1 Optima locaux et globaux . . . 13
1.2.2 Optima stricts . . . 14
1.2.3 Optima isolés . . . 14
1.2.4 Remarques . . . 15
1.3 Conditions d'optimalité . . . 16
1.3.1 Conditions pour un point stationnaire . . . 16
1.3.2 Conditions pour un optimum . . . 17
1.4 Algorithmes de descente . . . 18
1.4.1 Réduction d'intervalle avec f' . . . 20
1.4.2 Réduction d'intervalles sans utiliser la dérivée . . . 22
1.5 Algorithmes d'approximation polynômiale . . . 24
1.5.1 Approximation quadratique de la fonction f . . . 25
1.5.2 Approximation linéaire de la fonction f' . . . 26
1.6 Vitesse de convergence d'algorithmes itératifs . . . 27
1.7 Convergence locale quadratique de l'itération de Newton . . . 28
1.7.1 Notation "grand O" et "petit o" . . . 28
1.7.2 Analyse de l'itération de Newton . . . 29
1.7.3 Preuve concise de la convergence quadratique . . . 30
1.7.4 Preuve détaillée de la convergence quadratique . . . 31
1.8 Algorithme combiné . . . 32
1.9 Organisation des chapitres . . . 33

2 Optimisation locale de fonctions différentiables sans contrainte **35**
2.1 Formulation du problème . . . 36
2.2 Conditions d'optimalité . . . 37
2.2.1 Analyse du problème à une seule variable . . . 37
2.2.2 Conditions d'optimalité pour un problème à n variables . . . 39
2.3 Déduction d'une classe d'algorithmes itératifs . . . 42
2.3.1 Recherche de points stationnaires . . . 42
2.3.2 Conditions pour la convergence . . . 45
2.3.3 Recherche de minima locaux faibles . . . 52

2.3.4 Méthodes de descente pour les systèmes d'équations 55
2.4 Itération de Newton modifiée pour l'optimisation 56
2.4.1 Modification de la direction de Newton 56
2.4.2 Analyse du pas θ pour la direction de Newton 56
2.4.3 Convergence de la méthode de Newton modifiée 57
2.4.4 Convergence superlinéaire de la méthode de Newton avec régions de confiance 59
2.5 Fonctions objectif quadratiques 60
2.5.1 Décomposition de Cholesky 60
2.5.2 Algorithme du gradient conjugué 61
2.6 Implantation de la direction de Newton modifiée 63
2.6.1 Adaptation de la décomposition de Cholesky 63
2.6.2 Adaptation du gradient conjugué 64
2.6.3 Adaptation du gradient conjugué : régions de confiance 65
2.7 Implantation d'un pas admissible 67
2.7.1 Interpolation polynômiale avec garde-fous 67
2.8 Résumé 67
2.9 Extensions et références 68

3 Programmation linéaire 71
3.1 Formulation du problème 72
3.2 Solutions de base réalisables 72
3.3 Condition d'optimalité 73
3.4 Algorithme du simplexe 74
3.5 Dualité 77
3.6 Dégénérescence 79
3.6.1 Un exemple de cyclage dans l'algorithme du simplexe 79
3.6.2 Règles d'anti-cyclage 80
3.6.3 Convergence finie de l'algorithme du simplexe 82
3.7 Implantation numérique de l'algorithme du simplexe 83
3.7.1 Utilisation de la décomposition LU 84
3.7.2 Utilisation de la décomposition QR 85
3.7.3 Dégénérescence numérique 85
3.8 Comment obtenir la première solution de base réalisable 85
3.9 Extensions et références 86

4 Optimisation différentiable avec contraintes linéaires 87
4.1 Énoncé du problème 88
4.2 Conditions d'optimalité pour les points stationnaires 88
4.2.1 Formulation inspirée de la programmation linéaire 89
4.2.2 Formulation sous forme de projection 90
4.3 Conditions de Karush-Khun-Tucker 91
4.3.1 Conditions pour les contraintes d'égalité 91
4.3.2 Conditions pour les contraintes d'inégalité 92

4.3.3 Exemples . . . 94
4.4 Remarques générales sur les algorithmes . . . 97
4.5 Algorithmes inspirés de la programmation linéaire . . . 97
4.5.1 Algorithme de Frank&Wolfe . . . 98
4.5.2 Algorithme du gradient réduit . . . 99
4.5.3 Problèmes de complémentarité linéaire . . . 101
4.6 Algorithmes de projection . . . 102
4.6.1 Exemple de calculs de projection . . . 103
4.7 Algorithmes de contraintes actives . . . 104
4.7.1 Un exemple d'algorithme globalement convergent . . . 105
4.8 Résumé . . . 106
4.9 Extensions et références . . . 108

5 Optimisation différentiable avec contraintes d'égalité 109
5.1 Conditions d'optimalité . . . 110
5.1.1 Conditions de premier ordre . . . 110
5.1.2 Conditions de second ordre . . . 112
5.2 Construction d'une famille d'algorithmes de pénalité . . . 115
5.2.1 Solution des conditions de premier ordre . . . 115
5.2.2 Perturbation des conditions d'optimalité . . . 115
5.2.3 Limites de la famille de méthode . . . 116
5.2.4 Propriétés de convergence globales de la pénalisation . . . 117
5.2.5 Propriétés numériques de la pénalisation . . . 118
5.3 Implantation de la pénalisation . . . 118
5.3.1 Élimination du mauvais conditionnement . . . 118
5.3.2 Convergence locale superlinéaire de la pénalisation . . . 122
5.3.3 Convergence globale et localement superlinéaire . . . 127
5.4 Lagrangiens et pénalités augmentés . . . 131
5.4.1 Lagrangiens augmentés . . . 133
5.4.2 Pénalités augmentées . . . 135
5.5 Fonctions de pénalité exacte . . . 137
5.6 Extensions et références . . . 138
5.7 Résumé . . . 138

6 Optimisation différentiable avec contraintes d'inégalité 141
6.1 Conditions d'optimalité . . . 142
6.1.1 Conditions de premier ordre . . . 142
6.1.2 Conditions de second ordre . . . 144
6.1.3 Une analyse basée sur les pénalités . . . 144
6.2 Construction de plusieurs familles d'algorithmes de pénalité . . . 146
6.2.1 Perturbation des conditions d'optimalité I : $g(x) \leq 0$ . . . 147
6.2.2 Perturbation des conditions d'optimalité II : $\lambda g(x) = 0$ . . . 147
6.2.3 Perturbation des conditions d'optimalité III : méthodes semi-intérieures 148
6.3 Propriétés générales des algorithmes de pénalisation . . . 148

6.3.1 Propriété de convergence globale . . . 148
6.3.2 Propriété d'identification des contraintes actives . . . 150
6.3.3 Propriété numériques . . . 150
6.4 Propriétés asymptotiques des algorithmes . . . 150
6.4.1 Élimination du mauvais conditionnement . . . 151
6.4.2 Ordre (superlinéaire) de convergence locale . . . 153
6.4.3 Algorithmes globalement et superlinéairement convergents . . . 154
6.5 Pénalités et Lagrangiens augmentés . . . 154
6.5.1 Pénalités extérieures . . . 154
6.5.2 Pénalités intérieures . . . 155
6.5.3 Pénalités primales-extérieures . . . 155
6.6 Extensions et références . . . 156
6.7 Résumé . . . 156

Table des figures

1.1 Minimisation globale difficile . . . 13
1.2 Optima locaux . . . 14
1.3 Optima stricts . . . 15
1.4 Points stationnaires . . . 16
1.5 Algorithme de Fibonacci . . . 24
1.6 Approximation quadratique . . . 25
1.7 Approximation linéaire . . . 26
1.8 Fonction $1 - \frac{1}{5x^2-6x+5}$ . . . 34

2.1 Points stationnaires . . . 40
2.2 Point de selle . . . 41
2.3 Exemple de l'algorithme . . . 44
2.4 Direction admissible . . . 46
2.5 Pas admissible . . . 47

4.1 Géométrie de la projection . . . 90

5.1 Pénalisation . . . 119

Table des algorithmes

1.1 Trouve intervalle. 19
1.2 Réduit intervalle—bissection. 20
1.3 Réduit intervalle—Fibonacci. 23
1.4 Réduit intervalle accéléré—bissection–Newton. 33
2.1 Descente simplifiée. 43
2.2 Newton avec région de confiance. 53
2.3 Newton modifié. 63
2.4 CG—direction de Newton modifée 66
2.5 CG—Région de confiance 66
2.6 Armijo 67
3.1 Simplexe révisé. 76
4.1 Gradient réduit. 100
4.2 Contraintes actives. 105
4.3 Gradient réduit amélioré. 107
5.1 Pénalité 128
5.2 Lagrangien augmenté 134
5.3 Pénalité augmentée 136

Chapitre 1

Préliminaires

Sujets du chapitre.

- Vocabulaire des problèmes d'optimisation.
- Notion de conditions d'optimalité.
- Algorithmes de descente.
- Convergence globale des algorithmes.
- Vitesse de convergence des algorithmes.

Introduction

Ce texte est consacré à l'étude de *problèmes d'optimisation*. Ce chapitre de préliminaires vise à fixer les éléments de vocabulaire qui permettent de décrire, de parler de ce type de problèmes, de leurs solutions, des algorithmes de résolution. On profite aussi de ces préliminaires pour effectuer certains rappels d'analyse mathématique, d'algèbre linéaire, et d'analyse numérique.

1.1 Optimisation

Avant d'entreprendre l'étude de problèmes d'optimisation, il est bon de bien définir ce qu'est un problème d'optimisation. Dans toute sa généralité, le problème d'optimisation consiste à déterminer la plus petite (grande) valeur possible qu'une fonction réelle $f : E \to I\!R$ nommée *fonction objectif* puisse prendre dans l'ensemble E nommé *ensemble réalisable*. Sous forme mathématique, ceci s'exprime (pour le cas de la minimisation)

$$f^* = \inf_{x \in E} f(x),$$

et signifie :

$$f(x) \geq f^*, \forall x \in E \text{ et } \forall \epsilon > 0 \; \exists x_\epsilon \in E : f(x_\epsilon) < f^* + \epsilon;$$

Une autre façon d'écrire la signification de *inf* est d'introduire l'ensemble $\mathcal{F}$ des valeurs qui bornent inférieurement $f(x)$ sur l'ensemble E : $\mathcal{F} = \{v : f(x) \leq v, \forall x \in E\}$. Cet ensemble est une demi-droite réelle, possiblement ouverte ; sa fermeture,$\overline{\mathcal{F}}$ est une demi-droite fermée. Alors, $f^* = \max\{v \in \overline{\mathcal{F}}\}$.

Pour le cas de maximisation, l'expression est $f^* = \sup_{x \in E} f(x)$, et signifie : $f(x) \leq f^*, \forall x \in E$ et $\forall \epsilon > 0 \; \exists x_\epsilon \in E : f(x_\epsilon) > f^* - \epsilon$. En fait, maximiser f revient à minimiser $-f$. Dans la suite de ce texte, arbitrairement, on considère les problèmes de minimisation.

Sans ajouter d'hypothèse sur la fonction f et l'ensemble E, il n'est pas certain que l'on puisse trouver un élément x^* de l'ensemble E pour lequel $f(x^*) = f^*$. Lorsque c'est le cas, la formulation mathématique devient (toujours pour le cas de la minimisation)

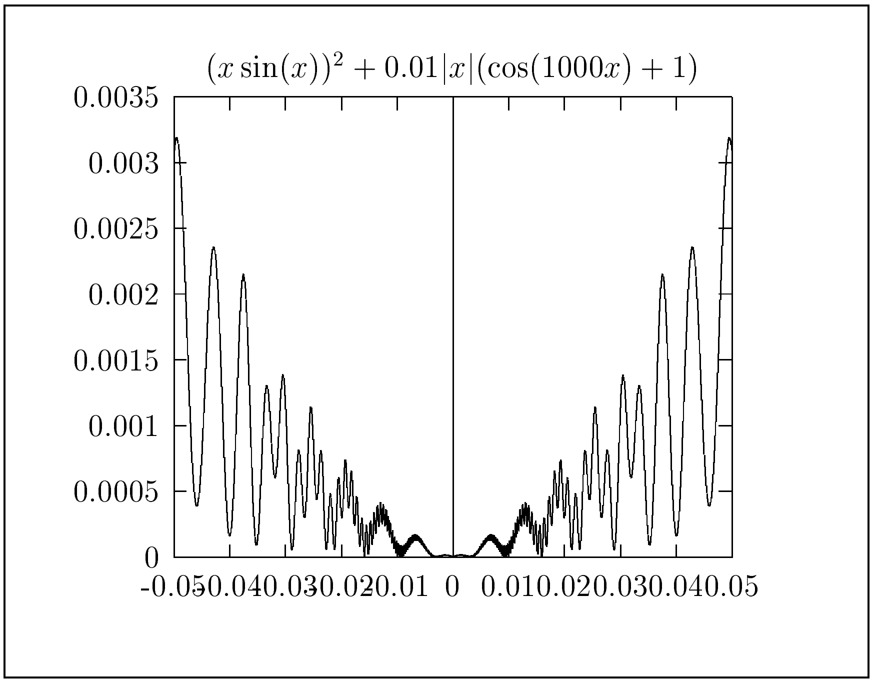
$$f(x^*) = f^* = \min_{x \in E} f(x),$$

et signifie : $f(x) \geq f(x^*), \forall x \in E$. La notation *min* est réservée aux problèmes pour lesquels il existe un $x^* \in E$ qui minimise f.

Exemple 1.1.1 Soit $E = I\!R$ et $f(x) = e^x$. On a $0 = \inf e^x$, mais $e^x > 0$ pour toute valeur réelle x. Par contre, si $f(x) = x^2$, on a $0 = f(0) = \min x^2$. □

Exercice 1.1.1 *[]* Vérifiez que $0 = \inf e^x$ et $0 = f(0) = \min x^2$ satisfont à la définition de inf et min. □

Nous nous intéressons plutôt aux problèmes pour lesquels il existe un élément de l'ensemble tel que $f(x^*) = f^*$. Un tel point de l'ensemble E est nommé un *optimum*, qui peut être un *minimum* ou encore un *maximum*.



Cette fonction comporte une infinité de minima locaux près de l'origine, mais il est très difficile d'identifier son(ses) minima globaux

FIG. 1.1 – Minimisation globale difficile

Les chapitres qui suivent considèrent différentes hypothèses sur les données du problème (la fonction f et l'ensemble E). La suite de ce chapitre de préliminaires propose une classification des *solutions*, ou optima ainsi que quelques notions élémentaires sur les conditions d'optimalité et les algorithmes.

1.2 Types d'optima

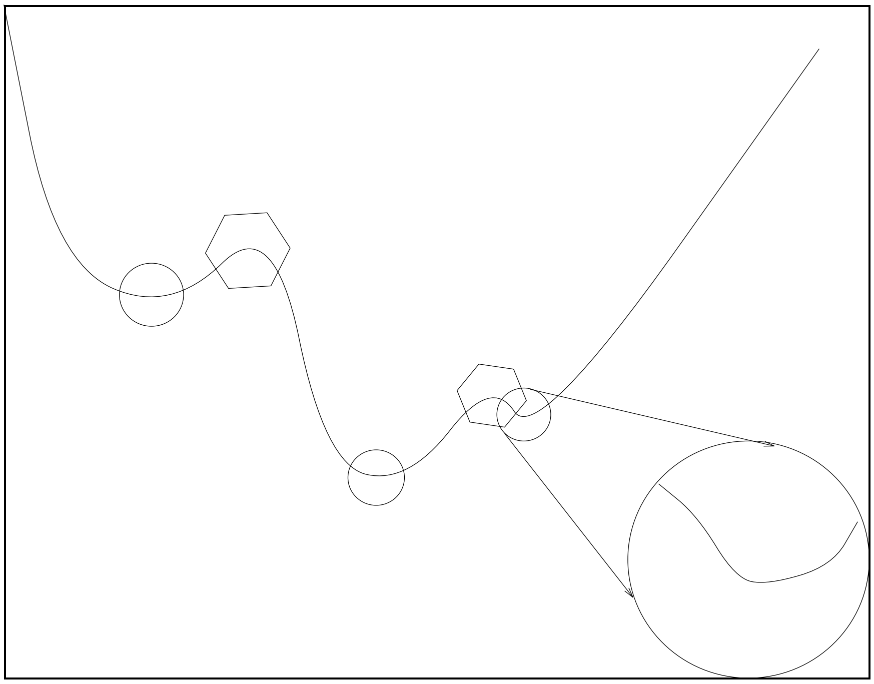
Sans hypothèse additionnelle, les problèmes tels que décrits plus haut sont en général impossible à résoudre. En effet, considérons la figure 1.1. Où se situe le minimum de la fonction ? De quel outil peut-on disposer pour identifier celui (ou ceux) des nombreux (infinis) minima apparents qui est le plus petit ?

Pour arriver à étudier les problèmes d'optimisation, nous allons classifier les optima selon différents critères. Par exemple, sur la figure 1.1, l'origine est un minimum local, mais ce minimum local n'est pas isolé alors que les autres minima locaux sont des minima locaux isolés. Voyons ces notions en détail.

1.2.1 Optima locaux et globaux

Jusqu'à maintenant, nous avons défini les optima en comparant la valeur de la fonction f à l'optimum avec sa valeur en *tout* autre point de E. Ce type de problème est connu sous le nom d'*optimisation globale*.

Si l'on affaiblit ses attentes, et que l'on compare les valeurs de la fonction f dans un voisinage d'un candidat optimum x^*, alors x^* est un *minimum local* de f s'il existe un $\epsilon > 0$



Sur cette fonction, les deux maxima locaux sont entourés d'hexagones alors que les minima sont encerclés. Un des minima est explosé, et on peut constater que dans le voisinage que décrit le cercle, la fonction atteint son point le plus bas

FIG. 1.2 – Optima locaux

tel que $f(x) \geq f(x^*), \forall x \in E \cap V_\epsilon(x^*)$, où $V_\epsilon(x^*)$ dénote un voisinage de diamètre ϵ centré en x^*. [1] La raison principale de se restreindre aux optima locaux est de pouvoir utiliser les outils de l'analyse mathématique. Un optimum global correspond au plus petit des optima locaux.

1.2.2 Optima stricts

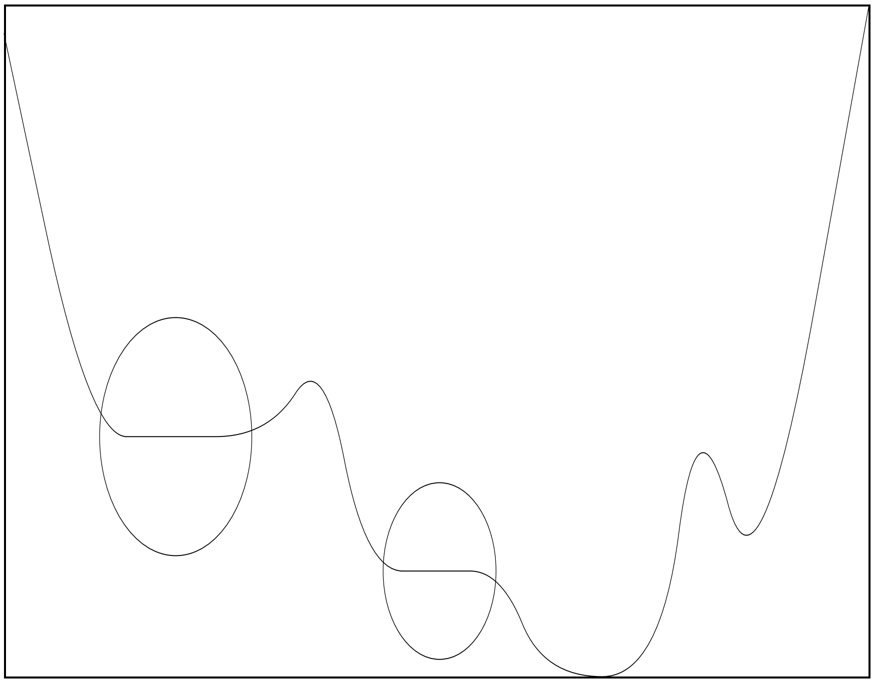
La notion d'optimum strict concerne le fait que d'autres points du voisinage puissent avoir la même valeur de fonction que x^*. Par exemple, sur la figure 1.3, les zones encerclées contiennent des minima locaux qui ne sont pas stricts alors que les autres optima locaux sont stricts.

Définition 1.2.1 *Un minimum local x^* est dit* strict *s'il existe une valeur $\epsilon > 0$ telle que quel que soit $x \in V_\epsilon(x^*)$, $x \neq x^*$, $f(x) > f(x^*)$; l'inégalité est stricte.*

1.2.3 Optima isolés

Sur la figure 1.3, les optima encerclés ne sont pas stricts, mais chaque ensemble d'optima est séparé des autres. La notion d'*optimum isolé* formalise cette situation. Par exemple, la fonction $x \sin(\frac{1}{x})$ comporte une accumulation de minima locaux à l'origine. L'origine n'est donc pas un optimum isolé.

[1] Nous effectuons notre étude dans des espaces euclidiens, et donc munis d'une norme. Alors, on peut définir $V_\epsilon(x^*)$ explicitement par $V_\epsilon(x^*) \stackrel{\text{def}}{=} \{x : |x - x^*| \leq \epsilon\}$.



Sur cette fonction, les zones encerclées constituent des minima *et* des maxima locaux, mais ne sont pas des optima stricts : dans un voisinage, la fonction est constante

FIG. 1.3 – Optima stricts

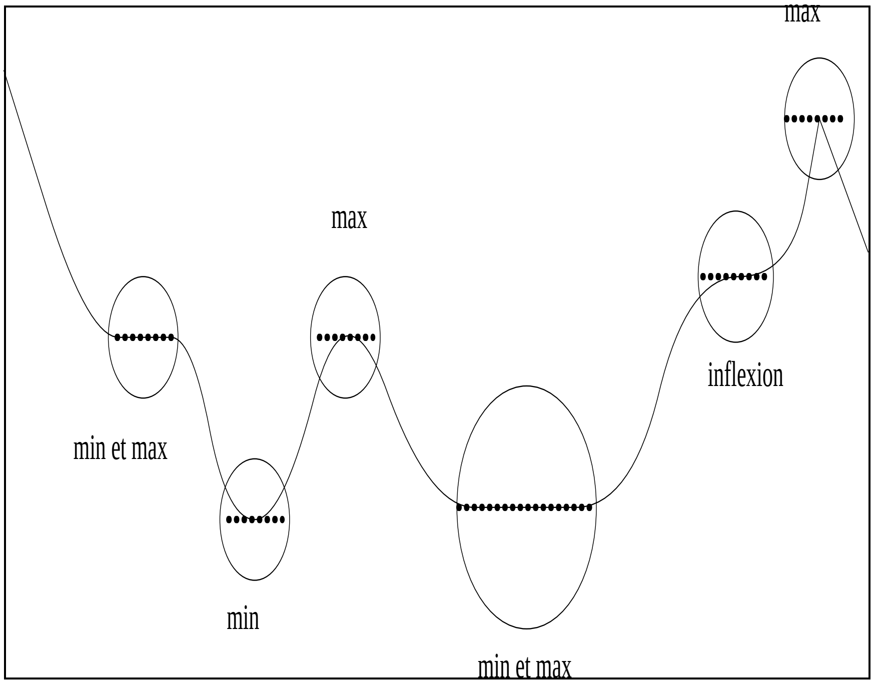
Définition 1.2.2 *Un ensemble connexe d'optima O est dit* isolé *si quelque soit un optimum $x \notin O$, la distance de x à O est bornée inférieurement par une constante positive.*

Exercice 1.2.1 *[Optima isolés]* Trouvez un exemple de fonction qui admet une accumulation de minima locaux (autre que la fonction $x \sin(1/x)$ ou la fonction de la figure 1.1). □

1.2.4 Remarques

Chacun des minima locaux de la fonction $(x \sin(1/x))^2$ lorsque x s'approche de zéro est un minimum satisfait à la définition 1.2.1. Cependant, plus le minimum s'approche de zéro, plus la valeur de ϵ diminue de sorte que l'origine ne satisfait plus à la définition. La notion de minimum isolé est semblable. Bien que l'origine ne satisfasse pas à la définition d'un minimum local strict pour la fonction $(x \sin(1/x))^2$, on peut trouver des valeurs de x *aussi proches de zéro* que l'on veut pour lesquelles $(x \sin(1/x))^2 > 0$. Proche de zéro, on a donc une infinité de minima locaux, ainsi qu'une infinité de points qui ne sont pas des minima locaux. L'origine n'est pas un minimum local isolé.

La notion de minimum isolé s'applique à des ensembles connexes de minima locaux. Par exemple, la fonction $f(x) = \max(\sin(x), 0)$ possède plusieurs intervalles $([0, \pi], [2\pi, 3\pi], ...)$ de minima locaux. Ces intervalles sont tous à une distance de π, et donc sont isolés. Par opposition, la fonction $f(x) = x^2 \max(\sin(1/x), 0)$ possède également de nombreux intervalles de minima locaux. Cependant, près de l'origine, on retrouve le phénomène que les intervalles sont de plus en plus rapprochés les un des autres, de sorte que l'on retrouve une infinité de petits intervalles dans tout voisinage de l'origine.



Cette fonction illustre différents types de points stationnaires. Le dernier est un maximum atteint en un point où la fonction n'est pas différentiable

FIG. 1.4 – Points stationnaires

1.3 Conditions d'optimalité

Malgré les classifications que nous venons d'effectuer, il serait difficile, à ce stade, d'identifier ou encore de vérifier qu'un point x^* constitue bel et bien un optimum d'un problème. En effet, les définitions que nous avons exigeraient de comparer la valeur d'une fonction en un point x^* avec sa valeur en une infinité de points voisins de x^*. L'analyse mathématique vient alors à notre secours, et lorsque la fonction possède certaines propriétés de différentiabilité, on peut établir des équations que tout optimum doit satisfaire. Il est alors facile de vérifier qu'un candidat est un optimum : il suffit de vérifier qu'il satisfait aux équations, qui sont nommées *conditions d'optimalité*.

Le but de cette section est d'illustrer le principe des conditions d'optimalité sur des exemples simples de fonctions réelles.

1.3.1 Conditions pour un point stationnaire

On apprend dans les premiers cours de calcul différentiel qu'une fonction réelle possède des minima locaux en des points qui annulent sa dérivée. De tels points sont nommés *points stationnaires*. Si l'on dépose une bille infiniment petite en un point stationnaire du graphe d'une fonction, celle-ci demeure immobile.

On observe donc que les points stationnaires, pour une fonction réelle, peuvent être de trois types : minima locaux, maxima locaux ou points d'inflexion. Par conséquent, tout minimum local est un point stationnaire, mais la contraposée n'est pas vraie.

Les points stationnaires satisfont donc à la *condition* nécessaire *d'optimalité de premier ordre*.

1.3.2 Conditions pour un optimum

On apprend aussi dans les cours de calcul que si, en un point stationnaire, la dérivée seconde d'une fonction est positive, il s'agit d'un minimum local ; si elle est négative, il s'agit d'un maximum local ; si elle s'annule, on ne peut rien conclure.

Les points stationnaires qui satisfont à la *condition suffisante d'optimalité de second ordre* ($f''(x^*) > 0$) sont donc des minima.

Par ailleurs, si x^* est un minimum local de la fonction f, alors *nécessairement* on a $f'(x^*) = 0$ et $f''(x^*) \geq 0$. En effet, considérons la fonction $f(x) = x^4$. $x^* = 0$, $f'(0) = f''(0) = 0$.

Exercice 1.3.1 *[Condition d'optimalité]*

a) L'origine est un point stationnaire, c'est-à-dire un point pour lequel $f'(x) = 0$, de la fonction $f(x) = x^2 \cos x$. Déterminez s'il s'agit d'un minimum (local), maximum (local) ou point d'inflexion.

b) Montrez que la fonction
$$f(x) = (2 - x^2)^2 + x^5$$
possède deux points stationnaires, un maximum et un minimum local.

c) Trouvez un exemple de fonction qui possède trois points stationnaires, un minimum local, un maximum local et un point d'inflexion.

□

Théorème 1.3.1 *Soit f de classe $\mathcal{C}^3$. Un minimum local qui satisfait à la condition suffisante d'optimalité ($f'(x^*) = 0$ et $f''(x^*) > 0$) est un minimum local strict et isolé.*

Preuve Montrons que x^* est un minimum local strict. Exprimons f à l'aide d'un développement de Taylor d'ordre 3 autour de x^* :

$$f(x) = f(x^*) + f'(x^*)(x - x^*) + \frac{1}{2}f''(x^*)(x - x^*)^2 + \frac{1}{6}f'''(\xi)(x - x^*)^3,$$

avec ξ compris entre x et x^*. Les conditions suffisantes nous instruisent que $f'(x^*) = 0$ alors que $f''(x^*) > 0$. Choisissons un voisinage V de x^* $V = \{x : |x - x^*| \leq \epsilon\}$ tel que quel que soit $\xi \in V$, $|f'''(\xi)| < A = 2|f'''(x^*)|$. Alors, dans ce voisinage,

$$\begin{aligned} f(x) - f(x^*) &= \frac{1}{2}f''(x^*)(x - x^*)^2 + \frac{1}{6}f'''(\xi)(x - x^*)^3 \\ &= \frac{(x - x^*)^2}{2}\left(f''(x^*) + \frac{1}{3}f'''(\xi)(x - x^*)\right) \\ &\geq \frac{(x - x^*)^2}{2}\left(f''(x^*) - \frac{1}{3}A|(x - x^*)|\right) \end{aligned}$$

Maintenant, la parenthèse de la dernière inégalité est positive dès que $f''(x^*) - \frac{1}{3}A|(x - x^*)| > 0$, c'est-à-dire
$$|x - x^*| < \frac{3f''(x^*)}{A},$$
de sorte que si $|x - x^*| < \min\left\{\epsilon, \frac{3f''(x^*)}{A}\right\}$, et $x \neq x^*$, alors $f(x) < f(x^*)$.

Montrons maintenant que x^* est un minimum isolé. Pour ce faire, nous allons vérifier que $f'(x) \neq 0$ dans un voisinage de x^*. Exprimons f' à l'aide d'un développement de Taylor d'ordre 2 :

$$f'(x) = f'(x^*) + f''(x^*)(x - x^*) + \frac{1}{2}f'''(\xi)(x - x^*)^2.$$

Les conditions suffisantes nous informent que $f'(x^*) = 0$ alors que $f''(x^*) > 0$. Donc, si $|x - x^*| \leq \epsilon$,

$$f'(x) = (x - x^*)\left(f''(x^*) + \frac{1}{2}f'''(\xi)(x - x^*)\right),$$

et donc

$$\begin{aligned} |f'(x)| &= \left|(x - x^*)\left(f''(x^*) + \frac{1}{2}f'''(\xi)(x - x^*)\right)\right| \\ &= |(x - x^*)|\left|f''(x^*) + \frac{1}{2}f'''(\xi)(x - x^*)\right| \\ &\geq |(x - x^*)|\left(|f''(x^*)| - \frac{1}{2}|f'''(\xi)||(x - x^*)|\right) \end{aligned}$$

La suite de la preuve est similaire à celle de la première partie. □

Exercice 1.3.2 *[Minimum strict]* Complétez les détails de la seconde partie de la preuve □

Exercice 1.3.3 *[Fonctions $\mathcal{C}^2$]* Le théorème 1.3.1 demeure vrai si l'on suppose que f est de classe $\mathcal{C}^2$ plutôt que $\mathcal{C}^3$; adaptez la démonstration à cette hypothèse plus faible □

1.4 Algorithmes de descente

Maintenant que nous savons caractériser des optima locaux par les conditions d'optimalité, penchons-nous sur le problème de calculer, d'identifier un optimum local d'une fonction. Le principe du calcul est d'utiliser un *algorithme itératif* : à partir d'un point qui ne satisfait pas aux conditions d'optimalité, on va produire un autre point, meilleur que le précédent. Puis, on va recommencer, et engendrer une suite de points qui s'approche d'un optimum.

Nous avons vu que la recherche de minima locaux d'une fonction peut passer par la recherche de points stationnaires, c'est-à-dire, de racines de la dérivée de la fonction $f'(x) = 0$. Cependant, ce qui distingue les problèmes d'optimisation des problèmes d'équations, c'est justement la fonction f, qui permet de comparer la qualité de deux points : puisque l'on veut minimiser f, il est intuitif de préférer un point x_1 à un point x_2 lorsque $f(x_1) < f(x_2)$. Nous allons tenter de formaliser cette intuition en engendrant une suite de points $\{x_n\}$ telle que pour tous les membres de la suite, $f(x_{i+1}) < f(x_i)$. La méthode qui produit une telle suite de points est nommée *algorithme de descente*.

Si un point n'est pas stationnaire, alors le signe de la dérivée de la fonction objectif permet de savoir de quel côté la fonction diminue. On peut donc envisager de rechercher un minimum local en effectuant de petits déplacements dans la direction de diminution de la fonction. Deux indices nous permettent de savoir que nous avons cerné un minimum local :

– la fonction se met à augmenter, c'est-à-dire $f(x_k) > f(x_{k-1})$ pour la première fois, donc $f(x_i) < f(x_{i-1})\ \forall i < k$;

– la pente de la fonction change de signe pour la première fois : $f'(x_k)f'(x_{k-1}) < 0$ et $f'(x_i)f'(x_{i-1}) > 0 \; \forall i < k$.

Si jamais on n'a pas cerné de minimum local après un très grand nombre d'essais, c'est un indice que la fonction f est *peut-être* non-bornée inférieurement, ou encore qu'il n'existe pas de x^* tel que $f(x^*) = \inf f(x)$, puisque nous avons identifié de nombreux termes d'une suite présumée monotone décroissante.

```
Trouve_intervalle
{ Donnés : x_0, δ > 0 et 0 < M < ∞ }
cerné ← faux
borné ← vrai
x⁺ ← x_0
tantque ( ¬ cerné ∧ borné )
    x ← x⁺
    x⁺ ← x − signe(f'(x)) ∗ δ
    cerné ← f(x⁺) > f(x) ∨ f'(x) ∗ f'(x⁺) ≤ 0
    borné ← |x| < M
```

Algorithme 1.1: Trouve intervalle.

Cette idée simple est à la base des algorithmes de descente. On voit que les algorithmes de descente sont construits à partir de points qui *ne satisfont pas* aux conditions d'optimalité. On démontre habituellement un théorème de convergence globale dont l'énoncé ressemble au théorème 1.4.1. Observons que l'algorithme Trouve_intervalle est codé à l'aide de variables informatiques : les variables x et x^+ représentent deux points successifs dans la suite $\{x_k\}$, c'est-à-dire x_k et x_{k+1} pour une certaine valeur de k.

Théorème 1.4.1 *Supposons que l'on choisisse $M = +\infty$ dans l'algorithme* Trouve_intervalle. *Alors, si l'algorithme termine, l'intervalle d'extrémités x_{k-2} et x_k contient un minimum local de la fonction f, supposée de classe $\mathcal{C}^1$.*

Preuve Si l'algorithme termine, alors au moins une des deux conditions de terminaison est satisfaite. Vérifions que chacune de ces conditions assure que l'intervalle contient un minimum local.

– $f(x_k) > f(x_{k-1})$ et $f(x_{k-1}) < f(x_{k-2})$. Puisque f est continue, elle atteint son minimum sur l'intervalle d'extrémités x_{k-2} et x_k ; de plus, puisque son minimum est de valeur au moins aussi petite que $f(x_{k-1})$, il existe un point à l'*intérieur* de l'intervalle qui minimise f sur l'intervalle. Ce minimum global sur l'intervalle est en particulier un minimum local.

– $f'(x_k)f'(x_{k-1}) < 0$. Supposons sans perte de généralité que $x_k < x_{k-1}$ et que $f'(x_k) < 0$. Alors, $f'(x_{k-1}) > 0$. Donc, puisque f' est une fonction continue, elle doit *traverser* l'axe des X en au moins un point x appartenant à l'intervalle, et ce de manière croissante.

□

Remarquons que cet algorithme possède la propriété de *convergence globale*, c'est-à-dire qu'aucune hypothèse n'est requise sur les propriétés du point initial. Cependant, il ne cerne que des *minima locaux*. Il est donc important de différencier la notion de minimum local et global qui précise la nature du minimum, et la notion de convergence globale d'un algorithme qui garanti la convergence peu importe l'estimé initial.

1.4.1 Réduction d'intervalle avec f'

Une fois qu'un intervalle contenant un minimum local est cerné, on utilise des techniques de *réduction d'intervalle* pour améliorer notre estimé de la solution. Par exemple, l'*algorithme de bissection* évalue la dérivée de f au point milieu de l'intervalle, et conserve la moitié de l'intervalle pour laquelle les dérivées des extrémités sont de signe opposé. Il serait dangereux d'interrompre l'algorithme lorsque la dérivée au point milieu s'annule : rien ne garantirait alors que ce point stationnaire est un minimum local. Tel qu'énoncé, l'algorithme est assuré de cerner soit un minimum local ou un point d'inflexion (mais pas un maximum local) car les conditions assurent que $f'(x_1)$ demeure toujours positif alors que $f'(x_0)$ demeure toujours négatif ou nul.

```
Bissection
  { Donnés : x_0 < x_1 tels que f'(x_0) < 0 et f'(x_1) > 0 ; }
  {    un critère d'arrêt, ε. }
  répéter
     x_m ← (x_0 + x_1)/2
     si (f'(x_m) ≤ 0 ) alors x_0 ← x_m
     sinon x_1 ← x_m
  jusqu'à ( |x_1 − x_0| < ε)
```

Algorithme 1.2: Réduit intervalle—bissection.

Théorème 1.4.2 *L'algorithme bissection cerne un point stationnaire qui ne peux pas être un maximum local.*

L'algorithme *Bissection* assure la convergence vers un point stationnaire pour lequel la dérivée n'est pas décroissante. C'est un point qui satisfait aux conditions nécessaires d'optimalité. Il est en général impossible de concevoir un algorithme convergeant vers un point satisfaisant aux conditions suffisantes d'optimalité.

Exercice 1.4.1 *[Point milieu stationnaire]*

a) Trouvez un exemple (une fonction f et un intervalle $[a, b]$) pour lequel les dérivées de f sont de signe opposé aux extrémités de l'intervalle ($f'(a) < 0$ et $f'(b) > 0$) pour lequel le point milieu de l'intervalle est un maximum local.

b) Trouvez un pareil exemple pour lequel le point milieu de l'intervalle est un point d'inflexion.

c) Vérifiez que l'algorithme bissection ne converge pas vers un maximum local pour vos deux exemples.

□

iter	x_0	x_1	$f'(x_0)$	$f'(x_1)$
0	-2.0000000000	1.8349000000	-2017.1439674636	12.3537736269
1	-0.0825500000	1.8349000000	-1.2740125688	12.3537736269
2	-0.0825500000	0.8761749999	-1.2740125688	0.6749976921
3	0.3968125000	0.8761749999	-0.1624455002	0.6749976921
4	0.3968125000	0.6364937500	-0.1624455002	0.2166009131
5	0.3968125000	0.5166531249	-0.1624455002	0.0259461281
6	0.4567328125	0.5166531249	-0.0675193202	0.0259461281
7	0.4866929687	0.5166531249	-0.0207307233	0.0259461281
8	0.4866929687	0.5016730468	-0.0207307233	0.0026059477
9	0.4941830078	0.5016730468	-0.0090608627	0.0026059477
10	0.4979280273	0.5016730468	-0.0032273217	0.0026059477
11	0.4998005371	0.5016730468	-0.0003106837	0.0026059477
12	0.4998005371	0.5007367919	-0.0003106837	0.0011476289
13	0.4998005371	0.5002686645	-0.0003106837	0.0004184723
14	0.4998005371	0.5000346008	-0.0003106837	0.0000538943
15	0.4999175689	0.5000346008	-0.0001283947	0.0000538943
16	0.4999760848	0.5000346008	-0.0000372501	0.0000538943
17	0.4999760848	0.5000053428	-0.0000372501	0.0000083220
18	0.4999907138	0.5000053428	-0.0000144640	0.0000083220
19	0.4999980283	0.5000053428	-0.0000030710	0.0000083220
20	0.4999980283	0.5000016856	-0.0000030710	0.0000026255

TAB. 1.1 – Itérations de bissection sur la fonction $e^{x(x-1)}$.

Le tableau 1.1 illustre l'application de l'algorithme avec la fonction $e^{x(x-1)}$ et l'intervalle initial $[-2, 1.8349]$.

L'algorithme de bissection requiert un intervalle de départ qui satisfait à la seconde condition de terminaison de l'algorithme qui identifie un intervalle cernant un minimum local. Si jamais l'algorithme 1.1 avait terminé avec un intervalle satisfaisant à la première condition, c'est-à-dire $f(x_k) > f(x_{k-1})$ et $(f(x_{k-1}) < f(x_i)\ \forall i < k-2)$, alors il faudrait d'abord identifier un point de l'intervalle de bornes x_k et x_{k-1} dont la dérivée est de signe opposé à la dérivée en x_0.

Exercice 1.4.2 *[Initialisation de bissection]*

a) Dans le contexte de la première condition de terminaison de l'algorithme de recherche d'intervalle, justifiez qu'il existe un point de l'intervalle de bornes x_k et x_{k-1} dont la dérivée est de signe opposé à la dérivée en x_0 ;

b) Proposez une technique de réduction d'intervalle pour identifier un tel point.

□

1.4.2 Réduction d'intervalles sans utiliser la dérivée

Dans certaines applications, la dérivée de f peut-être difficile à calculer. Il est possible de concevoir une méthode de réduction d'intervalle qui n'utilise que la fonction f. Il faut cependant accepter une hypothèse additionnelle assez restrictive : que la fonction soit *unimodulaire*

Définition 1.4.1 *Une fonction unimodulaire sur un intervalle ne possède qu'un seul minimum local dans l'intervalle, et de plus elle est monotone decroissante à gauche de celui-ci, et monotone croissante à droite.*

Avec ces conditions, si l'on considère quatre points distincts, $x_0 < x_1 < x_2 < x_3$, il est certain que l'on peut satisfaire la seconde condition du théorème 1.4.1 en éliminant un des points x_0 ou x_3. Un algorithme de réduction d'intervalle peut donc être construit de la façon suivante : à partir d'un intervalle initial $[x_0, x_3]$, on ajoute deux points intermédiaires x_1 et x_2. Puisque f est unimodulaire, on a $f(x_1) < f(x_0)$, $f(x_1) < f(x_3)$, $f(x_2) < f(x_0)$, et $f(x_2) < f(x_3)$. Si $f(x_1) < f(x_2)$, alors le triplet de points x_0, x_1, x_2 satisfait à la seconde condition du théorème 1.4.1. L'algorithme de Fibonacci utilise ce principe, et de plus, choisit x_1 et x_2 de telle sorte qu'après avoir conserve le triplet x_0, x_1, x_2 (ou encore le triplet x_1, x_2, x_3 si $f(x_2) < f(x_1)$), le point x_1 joue le rôle d'un des deux points intermédiaires dans le nouvel intervalle $[x_0, x_2]$. Ceci évite de recalculer $f(x_1)$.

Exercice 1.4.3 *[Hypothèse nécessaire]* Trouvez un exemple de fonction qui n'est pas unimodulaire pour laquelle l'algorithme de Fibonacci ne fonctionne pas. □

Exercice 1.4.4 *[Applications d'algorithmes]*

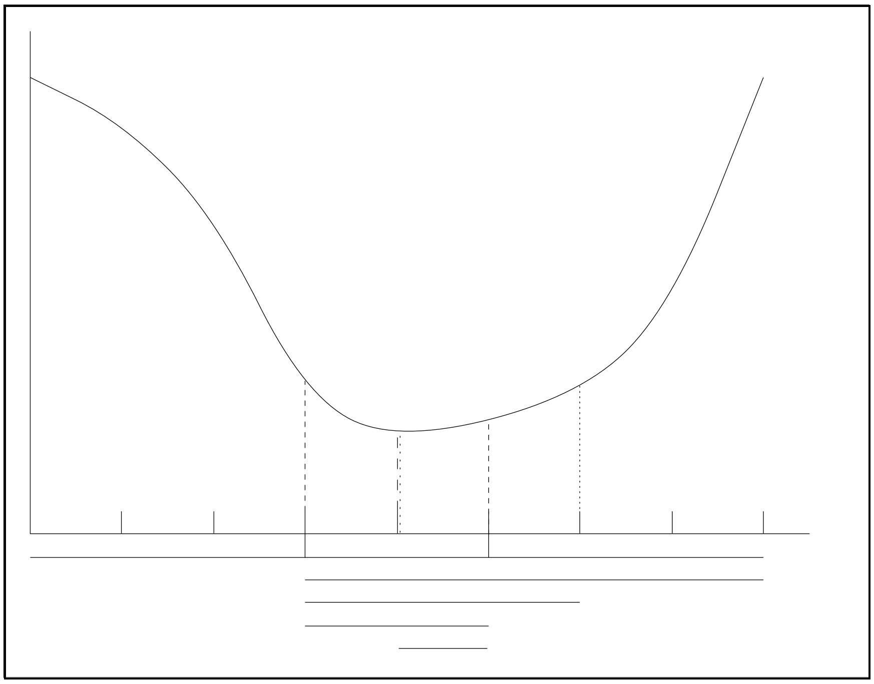
a) Pour la fonction $f(x) = e^{x(x-1)}$, effectuez trois itérations des deux algorithmes (bissection, Fibonacci) à partir de l'intervalle initial $[-1, 1]$.

b) Même question pour la fonction $f(x) = x^2(1 - \cos x)$ sur l'intervalle $[-1, 0.5]$.

$\boxed{\text{Fibonacci}}$

{ Donnés : a et b tels que $f'(a) * f'(b) < 0$; }
{ un nombre de réductions de l'intervalle, n ; }
{ la suite des nombres de Fibonacci $F_0..F_{n+2}$. }
{ Hypothèse : la fonction f est unimodulaire. }
$L \leftarrow b - a$
$x_1 \leftarrow a + \frac{F_n}{F_{n+2}} * L$
$x_2 \leftarrow a + \frac{F_{n+1}}{F_{n+2}} * L$
$f_1 \leftarrow f(x_1); \quad f_2 \leftarrow f(x_2)$
$k \leftarrow 1$
répéter
  si $(f_1 < f_2)$ **alors**
    $b \leftarrow x_2; \quad x_2 \leftarrow x_1$
    $x_1 \leftarrow (a * F_{n-k+1} + b * F_{n-k})/F_{n-k+2}$
    $f_2 \leftarrow f_1; \quad f_1 \leftarrow f(x_1)$
  sinon
    $a \leftarrow x_1; \quad x_1 \leftarrow x_2$
    $x_2 \leftarrow (b * F_{n-k+1} + a * F_{n-k})/F_{n-k+2}$
    $f_1 \leftarrow f_2; \quad f_2 \leftarrow f(x_2)$
  $k \leftarrow k + 1$
jusqu'à $(k = n)$
$x_2 \leftarrow x_2 + \epsilon$
$f_2 \leftarrow f(x_2)$
si $(f_1 < f_2)$ **alors**
  résultat $\leftarrow [a, x_2]$
sinon
  résultat $\leftarrow [x_1, b]$

Algorithme 1.3: Réduit intervalle—Fibonacci.



Les premiers nombres de Fibonacci sont 1, 1, 2, 3, 5, 8. Sur cet exemple, on voit que l'intervalle initial est découpé en 3/8 et 5/8. L'intervalle restant, $[3/8, 1]$ est à son tour découpé en 2/5 et 3/5. L'intervalle restant est maintenant de longueur 3/8, et est découpé en son 1/3 et 2/3. Finalement, l'avant dernier intervalle, de longueur 1/4, est découpé en deux ; on choisit le point milieu, et un point $1/2 + \epsilon$.

FIG. 1.5 – Algorithme de Fibonacci

c) Un inconvénient de la méthode de Fibonacci est de devoir fixer *a priori* le nombre d'itérations ; si on se fixait une infinité d'itérations, en sachant que

$$\lim_{i \to \infty} \frac{F_{i+1}}{F_i} = \tau,$$

le nombre d'or $\tau = 1 + \frac{1}{\tau} = \frac{(1+\sqrt{5})}{2} = 1.618...$, on utiliserait ce rapport à chaque itération. Cette méthode, nommée *section dorée*, consiste donc à utiliser

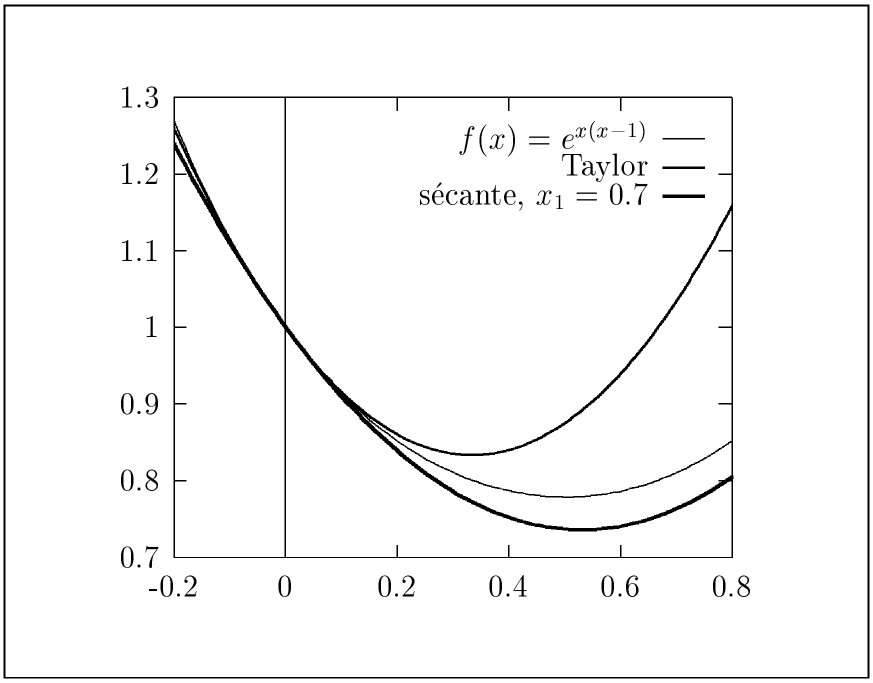
$$a, \qquad a + (2 - \tau)(b - a), \qquad a + (\tau - 1)(b - a), \qquad b$$

comme quatre points initiaux, et à conserver ce rapport à chaque itération. Effectuez trois itérations de cet algorithme sur les fonctions de a) et b).

□

1.5 Algorithmes d'approximation polynômiale

Les algorithmes de la section précédente sont intéressants car ils assurent de cerner un minimum local. Cependant, comme nous allons l'étudier plus loin, ces algorithmes ne convergent pas vite. Il est donc d'usage de changer d'algorithme lorsque l'intervalle contenant un minimum local est assez petit. Nous verrons plus loin ce que nous entendons par assez petit. Penchons nous maintenant sur une famille d'algorithmes consistant à approcher la fonction f ou la fonction f' par un polynôme.



On observe deux des possibilités d'approximation quadratique exactes en $x = 0$. Remarquons que l'approximation sécante possède la même pente que f en $x = 0.7$. L'approximation sécante est meilleure car on utilise l'information du point $x = 0.7$ qui est proche du minimum de f. En général, le point auxiliaire n'est pas aussi bon, et l'approximation de Taylor est meilleure que l'approximation sécante.

FIG. 1.6 – Approximation quadratique

1.5.1 Approximation quadratique de la fonction f

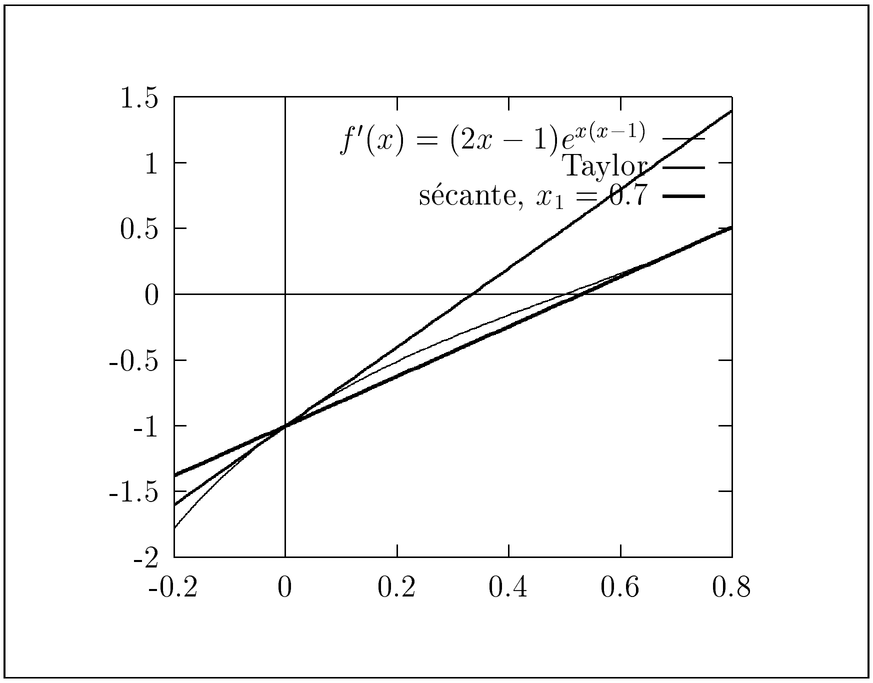
Plusieurs stratégies permettent d'approcher (localement) la fonction f par un polynôme p de degré deux. La plus naturelle, lorsque la fonction f est de classe $\mathcal{C}^2$ est d'utiliser un développement de Taylor d'ordre 2 autour d'un point x_0 :

$$p(x) \stackrel{\text{def}}{=} f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2} f''(x_0)(x - x_0)^2.$$

Nous verrons plus loin que cette approximation correspond à la méthode de Newton pour calculer une racine de $f'(x)$. Une fois que nous avons déterminé une approximation quadratique de f, il suffit de calculer son minimum pour obtenir un estimé (que l'on espère meilleur) du minimum de f. Utilisons la condition d'optimalité sur la fonction $p(x)$: $p'(x) = 0$, c'est-à-dire $f''(x_0)(x - x_0) + f'(x_0) = 0$. Pour définir le prochain point x_N, la méthode nommée méthode de Newton utilise donc la formule suivante : $x_N = x_0 - \frac{f'(x_0)}{f''(x_0)}$.

En fait, un polynôme quadratique p est défini par trois coefficients $p(x) = a + b(x - x_0) + \frac{c}{2}(x - x_0)^2$, et il suffit de fournir trois équations linéaires permettant de définir ces trois coefficients inconnus. Pour l'approximation de Taylor, les trois équations sont les suivantes : $p(x_0) = f(x_0)$, $p'(x_0) = f'(x_0)$, et $p''(x_0) = f''(x_0)$. Une autre possibilité consiste à choisir trois points, x_0, x_1, x_2 et utiliser les équations $p(x_0) = f(x_0)$, $p(x_1) = f(x_1)$, et $p(x_2) = f(x_2)$.

Exercice 1.5.1 *[Approximation quadratique]* Fournissez les coefficients du polynôme quadratique issu des conditions $p(x_0) = f(x_0)$, $p(x_1) = f(x_1)$, et $p(x_2) = f(x_2)$. Fournissez aussi une expression pour le minimum de la fonction quadratique. □



On observe deux des possibilités d'approximation linéaire de f' exactes en $x = 0$. Remarquons que l'approximation sécante coïncide avec f' en $x = 0.7$. L'approximation sécante est meilleure car on utilise l'information du point $x = 0.7$ qui est proche de la racine de f'. Les racines de f' et des deux approximations linéaires correspondent aux minima de la figure 1.6.

FIG. 1.7 – Approximation linéaire

D'autres possibilités sont envisageables, comme d'utiliser deux points x_0 et x_1 et les trois équations suivantes : $p(x_0) = f(x_0)$, $p'(x_0) = f'(x_0)$, $p'(x_1) = f'(x_1)$. Dans ce cas, les coefficients du polynôme sont : $a = f(x_0)$, $b = f'(x_0)$ et $c = \frac{f'(x_1)-f'(x_0)}{x_1-x_0}$. Nous verrons plus loin que cette approximation correspond à la méthode de sécante pour calculer une racine de $f'(x)$. Comme toutes les approximations quadratiques de f, un estimé (que l'on espère meilleur) du minimum de f est donné par $x^+ = x_0 - \frac{b}{c}$, à condition bien sur que $p''(x) = c > 0$.

1.5.2 Approximation linéaire de la fonction f'

Puisque nous recherchons une racine de $f'(x)$, nous pouvons envisager utiliser une approximation linéaire de $g \stackrel{\text{def}}{=} f'$ pour identifier un meilleur estimé. Si l'on approche g par un développement de Taylor d'ordre un, $g(x) \sim g(x_0) + g'(x_0)(x - x_0)$, on obtient la méthode de Newton. Le prochain estimé est une racine de l'approximation linéaire, c'est-à-dire $x^+ = x_0 - \frac{g(x_0)}{g'(x_0)}$ qui correspond au même point que d'utiliser une approximation quadratique de f.

Si l'on utilise plutôt deux points pour calculer une approximation sécante de g, on obtient $g(x) \sim g(x_0) + \frac{g(x_1)-g(x_0)}{x_1-x_0}(x - x_0)$ et le prochain point, toujours donné par une racine de l'approximation linéaire, est $x_S = x_0 - \frac{g(x_0)(x_1-x_0)}{g(x_1)-g(x_0)}$.

Exercice 1.5.2 *[Applications d'algorithmes]*

a) Pour la fonction $f(x) = e^{x(x-1)}$, effectuez trois itérations des trois algorithmes (Newton, sécante, Approximation quadratique passant par trois points) à partir de l'intervalle

Iter	x	$f'(x)$	$\|x - x^*\|$
0	-2.0000000000000000	-2017.1439674636756081	-2.5000000000000000
1	-1.8148148148148149	-765.7413872532831647	-2.3148148148148149
2	-1.6172499842815362	-291.7836916974232508	-2.1172499842815364
3	-1.4047918990578092	-111.6927862529728941	-1.9047918990578092
4	-1.1740888018727405	-42.9909451447428594	-1.6740888018727405
5	-0.9206366236151708	-16.6508404819880234	-1.4206366236151708
6	-0.6385637391018000	-6.4833811628303968	-1.1385637391018000
7	-0.3216494208351671	-2.5138450692553556	-0.8216494208351671
8	0.0279565647948271	-0.9187767726223022	-0.4720434352051729
9	0.3544833396867886	-0.2315076371696912	-0.1455166603132114
10	0.4940877267267001	-0.0092092880147647	-0.0059122732732999
11	0.4999995867021617	-0.0000006437533603	-0.0000004132978383
12	0.5000000000000000	0.0000000000000000	0.0000000000000000

TAB. 1.2 – Itérations de Newton sur la fonction $e^{x(x-1)}$, depuis un point de départ éloigné

initial $[-1, 1]$. Pour l'approximation utilisant trois points, choisir le troisième point en 0.

b) Même question pour la fonction $f(x) = x^2(1 - \cos x)$ sur l'intervalle $[-1, 0.5]$.

□

1.6 Vitesse de convergence d'algorithmes itératifs

Les deux prochaines sections permettent de préciser ce que nous entendons par efficacité d'un algorithme, et analysent l'efficacité des méthodes de bissection et de Newton.

Nous introduisons maintenant le vocabulaire permettant de comparer l'efficacité asymptotique de divers algorithmes.

Lorsqu'on parle de vitesse de convergence, il faut supposer que toute la suite $\{x_k\}$ converge, c'est-à-dire que la suite ne possède qu'un seul point d'accumulation, $\bar{x}$.

Définition 1.6.1 *La vitesse de convergence de la suite $\{x_k\}$ vers le point $\bar{x}$ s'exprime à l'aide des scalaires p et γ dans l'expression suivante*

$$\limsup_{k\to\infty} \frac{|x_{k+1} - \bar{x}|}{|x_k - \bar{x}|^p} = \gamma < \infty.$$

*L'*ordre *de convergence de la suite est donné par la plus grande valeur que p puisse prendre pour que la limite ci-haut demeure finie. Lorsque $p = 1$, γ est nommée le* taux *de convergence.*

Les cas importants (fréquents) sont le cas $p = 1$, dite *convergence linéaire*, et le cas $p = 2$, dite *convergence quadratique*. Un cas particulier de convergence linéaire est également important, soit le cas où $p = 1$ et $\gamma = 0$; on nomme ce type de comportement *convergence superlinéaire*.

Évidemment, puisque l'on s'intéresse au comportement limite de la suite $\{x_k\}$ qui converge vers $\bar{x}$, ce type d'analyse ne qualifie que le comportement local d'un algorithme, et n'est valable que lorsque x_k devient proche de $\bar{x}$.

Exemple 1.6.1 Dans l'algorithme de bissection, on a la garantie que l'intervalle contient un point stationnaire. La largeur de l'intervalle borne donc l'erreur que l'on commet. Puisque l'intervalle est réduit de moitié à chaque itération, nommant δ_k cette largeur à l'itération k, on a (puisque δ_k converge vers zéro)

$$\limsup_{k\to\infty} \frac{|\delta_{k+1}|}{|\delta_k|^1} = \frac{1}{2} < \infty,$$

et donc l'ordre de convergence de l'algorithme de bissection est un, et on parle de convergence linéaire de taux 1/2. □

En observant le tableau 1.1, on constate que l'algorithme identifie une décimale additionnelle de la solution environ à toutes les trois itérations. C'est typique d'un algorithme qui converge linéairement. Par opposition, sur le tableau 1.2, bien que les premières itérations ne soient pas très efficaces, on constate que chacune des trois dernières itérations *double* le nombre de décimales exactes de la solution, ce qui est typique des algorithmes qui convergent quadratiquement. Nous abordons maintenant l'analyse de la convergence quadratique de la méthode de Newton.

1.7 Convergence locale quadratique de l'itération de Newton

L'analyse de convergence locale peut très bien être réalisée dans le contexte de la recherche d'une racine de l'équation $g(x) = 0$. Pour les problèmes d'optimisation, $g = f'$, mais cette propriété n'est pas utile pour l'analyse locale de la convergence comme elle l'a été jusqu'ici pour l'analyse de la convergence globale des algorithmes.

Il est facile de se convaincre que l'ordre de convergence de l'itération de Newton, *localement*, est quadratique. Il est souvent plus concis d'utiliser la notation "grand O", permettant d'escamoter des détails de preuves fastidieux. Cependant, l'utilisation de cette notation nécessite une grande prudence. Nous analysons donc ici la vitesse de convergence locale de la méthode de Newton en utilisant les techniques d'analyse classiques, ainsi que, pour fins de comparaison, les formulations de ces analyses avec la notation "grand O".

1.7.1 Notation "grand O" et "petit o"

Soient deux fonctions, f et g, de $\mathbb{R}$ dans $\mathbb{R}$, d'un même paramètre x telles que $f(\bar{x}) = g(\bar{x}) = 0$.

Définition 1.7.1 *Nous notons $f \sim \mathcal{O}(g)$ si et seulement si il existe $\gamma > 0$ tel que*

$$\limsup_{x\to\bar{x}} \left|\frac{f(x)}{g(x)}\right| = \gamma < \infty.$$

En prose, f est "grand O" de g lorsque f s'approche de zéro au moins aussi vite que g s'approche de zéro quand x tend vers $\bar{x}$. On laisse à montrer, en exercices, les propriétés suivantes.

Exercice 1.7.1 *[Notation $\mathcal{O}$]*

a) Démontrez les propriétés de la notation "grand O"

  i) $f \sim \mathcal{O}(g)$ entraîne que $\alpha f \sim \mathcal{O}(g)$, où α est une constante finie ;

  ii) $f + g \sim \mathcal{O}(\max(f, g))$;

  iii) $fg \sim \mathcal{O}(f)\mathcal{O}(g)$;

b) que peut-on dire de $\frac{1}{f} \sim \mathcal{O}(\cdot)$?

□

Par exemple, deux fonctions linéaires $f(x) \stackrel{\text{def}}{=} a_f x$ et $g(x) \stackrel{\text{def}}{=} a_g x$ sont mutuellement "grand O" l'une de l'autre si $a_f \neq 0$ et $a_g \neq 0$ car, selon la définition, $f \sim \mathcal{O}(g)$ et inversement, $g \sim \mathcal{O}(f)$ puisque $\frac{f}{g} = \frac{a_f}{a_g}$. Ces deux fonctions s'approchent de zéro *linéairement.*

Dans un second exemple, supposons f comme ci-haut, et $g \stackrel{\text{def}}{=} a_g x^2$. Maintenant, selon la définition, $g \sim \mathcal{O}(f)$ car $\frac{g}{f} = \frac{a_g}{a_f} x \to 0$ mais pas l'inverse car $\frac{f}{g} = \frac{a_f}{a_g x} \to \infty$. g s'approche de zéro quadratiquement, ce qui est plus vite que f qui s'approche de zéro linéairement.

Exercice 1.7.2 *[Ordre de convergence à zéro]* Soit f une fonction réelle de classe $\mathcal{C}^2(I\!\!R)$ avec $f(\bar{x}) = 0$ et $f'(\bar{x}) \neq 0$. Montrez qu'il existe un voisinage de $\bar{x}$ (défini comme $\{x : |x - \bar{x}| \leq \epsilon\}$) et des constantes appropriées $K_1, K_2, K_3, K_4 > 0$ telles que

$$\left\{ \begin{array}{c} K_1|x - \bar{x}| \leq |f(x)| \leq K_2|x - \bar{x}| \\ K_3|f(x)| \leq |x - \bar{x}| \leq K_4|f(x)| \end{array} \right\} \quad \forall x : |x - \bar{x}| \leq \epsilon.$$

Ceci démontre, selon la notation $\mathcal{O}$, que $f(x) \sim \mathcal{O}(x - \bar{x})$ et que $(x - \bar{x}) \sim \mathcal{O}(f(x))$ si $f(\bar{x}) = 0$ et $f'(\bar{x}) \neq 0$. □

Similairement, lorsque f est négligeable par rapport à g, on peut utiliser la notation "petit o".

Définition 1.7.2 *Nous notons $f \sim o(g)$ si et seulement si* $\limsup_{x \to \bar{x}} \left| \frac{f(x)}{g(x)} \right| = 0$.

1.7.2 Analyse de l'itération de Newton

Plaçons-nous dans le contexte d'identifier une racine de la dérivée de f, $g(x) \stackrel{\text{def}}{=} f'(x)$. On veut donc résoudre l'équation $g(x) = 0$. Dans ce contexte de solution d'une équation non-linéaire, rappelons que l'itération de Newton s'écrit :

$$x_{k+1} \leftarrow x_k - \frac{g(x_k)}{g'(x_k)}.$$

Nous nous intéressons ici à la convergence locale de l'algorithme, c'est-à-dire à la convergence de l'algorithme lorsque x_k est devenu très proche de x^*, une racine de l'équation $g(x) = 0$ pour laquelle $g'(x^*) \neq 0$.

Cette itération de Newton peut se reformuler ainsi :

$$\begin{aligned} x_{k+1} &\leftarrow x_k + d_k; \\ d_k &\leftarrow -\frac{g(x_k)}{g'(x_k)}. \end{aligned}$$

On voit donc que d_k satisfait à la relation

$$g'(x_k)d_k + g(x_k) = 0.$$

Introduisons la notion d'erreur à l'itération k ; $\epsilon_k \stackrel{\text{def}}{=} x^* - x_k$. Selon la notation "grand O", nous allons montrer que $|\epsilon_{k+1}| \sim \mathcal{O}(|\epsilon_k|^2)$, c'est à dire que lorsque $\epsilon_k \to 0$, la quantité $|\epsilon_{k+1}|$ (vue comme fonction de ϵ_k) s'approche de zéro au moins aussi vite que la quantité $|\epsilon_k|^2$. Selon la définition 1.6.1,

$$\limsup_{\epsilon_k \to 0} \frac{|\epsilon_{k+1}|}{|\epsilon_k|^2} = \gamma < \infty,$$

ce qui peut se réécrire

$$|\epsilon_{k+1}| \leq \gamma |\epsilon_k|^2.$$

Théorème 1.7.1 *Soit x^* une racine isolée de la fonction g telle que $g'(x^*) \neq 0$. Alors, il existe un voisinage de x^* tel que si la méthode de Newton est initialisée dans ce voisinage, elle produit une suite convergeant vers x^* et la vitesse de convergence asymptotique est quadratique.*

1.7.3 Preuve concise de la convergence quadratique

Nous allons donc donner une première preuve de l'ordre quadratique de convergence pour l'itération de Newton en utilisant la notation "grand O", et ensuite, fournirons les détails d'une preuve qui manipule explicitement les constantes mises en jeu par le processus, mais cachées dans la première preuve.

D'abord, exprimons $g(x^*)$ en utilisant une série de Taylor autour du point x_k :

$$\begin{aligned} 0 = g(x^*) &= g(x_k) + g'(x_k)\epsilon_k + \mathcal{O}(|\epsilon_k|^2) \\ &= (g(x_k) + g'(x_k)d_k) + g'(x_k)(\epsilon_k - d_k) + \mathcal{O}(|\epsilon_k|^2) \\ &= 0 + g'(x_k)(\epsilon_{k+1}) + \mathcal{O}(|\epsilon_k|^2). \end{aligned}$$

De cette dernière équation, on tire que ϵ_{k+1} satisfait à la relation

$$g'(x_k)(\epsilon_{k+1}) \sim \mathcal{O}(|\epsilon_k|^2).$$

Utilisons maintenant les faits que
- $g'(x^*) > 0$, et
- g' est une fonction Lipschitz, c'est-à-dire que $|g'(x_k) - g'(x^*)| \leq L \cdot |x_k - x^*|$,

pour conclure que $\epsilon_{k+1} \sim \mathcal{O}(|\epsilon_k|^2)$. On voit donc que la convergence quadratique de l'itération de Newton est assurée.

1.7.4 Preuve détaillée de la convergence quadratique

Maintenant, examinons une preuve détaillée du même résultat, illustrant la puissance de la notation “grand O”.

D'abord, exprimons $g(x^*)$ en utilisant une série de Taylor autour du point x_k :

$$\begin{aligned} 0 = g(x^*) &= g(x_k) + g'(x_k)\epsilon_k + R_2(\epsilon_k) \\ &= (g(x_k) + g'(x_k)d_k) + g'(x_k)(\epsilon_k - d_k) + R_2(\epsilon_k) \\ &= 0 + g'(x_k)(\epsilon_{k+1}) + R_2(\epsilon_k), \end{aligned}$$

où R_2 est le reste du développement de Taylor d'ordre un ; la notation R_2 souligne le fait que ce reste est, si g est $\mathcal{C}^2$, quadratiquement petit en ϵ, propriété explicitement utilisée dans la preuve abrégée.

Comme dans la preuve abrégée, de cette dernière équation, on tire que ϵ_{k+1} est solution du système linéaire

$$g'(x_k)(\epsilon_{k+1}) = R_2(\epsilon_k).$$

Gardons en tête la propriété de Lipschitz de g' et écrivons

$$g'(x_k)(\epsilon_{k+1}) = g'(x^*)(\epsilon_{k+1}) + (g'(x_k) - g'(x^*))(\epsilon_{k+1}),$$

ce qui se transforme en prenant les normes de chaque côté en

$$\begin{aligned} |g'(x_k)(\epsilon_{k+1})| &= |g'(x^*)(\epsilon_{k+1}) + (g'(x_k) - g'(x^*))(\epsilon_{k+1})| \\ &\geq |g'(x^*)(\epsilon_{k+1})| - |(g'(x_k) - g'(x^*))(\epsilon_{k+1})| \\ &\geq |g'(x^*)(\epsilon_{k+1})| - L \cdot |\epsilon_k||\epsilon_{k+1}|. \end{aligned}$$

Maintenant, $|g'(x^*)| = K_1 > 0$, mais, plus important, $g'(x^*)$ est bornée, d'où l'on tire qu'il existe un certain K_2 tel que

$$K_1|\epsilon_{k+1}| \leq |g'(x^*)(\epsilon_{k+1})| \leq K_2|\epsilon_{k+1}|.$$

Les précédents développements nous permettent maintenant décrire :

$$(K_1 - L|\epsilon_k|)|\epsilon_{k+1}| \leq |g'(x_k)(\epsilon_{k+1})| = |R_2(\epsilon_k)| \leq |R_2(\epsilon_k)|.$$

Cependant, le théorème de Taylor nous assure qu'il existe un $K_3 > 0$ tel que $|R_2(\epsilon_k)| \leq K_3|\epsilon_k|^2$, et donc, on a

$$(K_1 - L|\epsilon_k|)|\epsilon_{k+1}| \leq K_3|\epsilon_k|^2,$$

et donc

$$|\epsilon_{k+1}| \leq \frac{K_3}{(K_1 - L|\epsilon_k|)}|\epsilon_k|^2,$$

ce qui complète la preuve puisque $K_1 > 0$ est une constante indépendante de ϵ_k alors que $L|\epsilon_k|$ converge vers zéro lorsque $|\epsilon_k|$ s'approche de zéro. Notons que nous pouvons maintenant spécifier rigoureusement ce que nous entendons par *localement*. En effet, nous avons l'assurance que $|\epsilon_{k+1}| < |\epsilon_k|$ dès que

$$\frac{K_3}{(K_1 - L|\epsilon_k|)}|\epsilon_k| < 1,$$

qui est équivalent à

$$|\epsilon_k| < \left(\frac{K_1}{K_3 + L}\right).$$

Commentaires

La preuve concise n'est pas beaucoup plus brève pour ce résultat, puisque on ne nécessite que trois constantes pour exprimer le raisonnement.

Exercice 1.7.3 *[Méthode de Newton]*

a) Fournissez un exemple (une fonction et un point de départ) pour lequel l'algorithme de Newton diverge.

b) Fournissez un exemple (une fonction et un point de départ) pour lequel l'algorithme de Newton oscille entre deux points.

c) Fournissez un exemple pour lequel l'algorithme de Newton converge vers une racine, mais pour laquelle $g'(x^*) = 0$; vérifiez que dans ce cas, la vitesse de convergence asymptotique est seulement linéaire.

□

1.8 Algorithme combiné

Nous avons étudié deux familles d'algorithmes : des algorithmes qui possèdent la propriété de convergence globale, et des algorithmes qui possèdent la propriété de convergence locale rapide. Nous avons évoqué que l'on devrait utiliser les algorithmes de type Newton dès que l'intervalle contenant un minimum local est assez petit. Nous allons préciser ce point.

L'idée est la suivante : nous allons retenir l'itération de Newton seulement lorsqu'elle appartient à l'intervalle spécifié par l'algorithme de réduction d'intervalle. Il est possible que l'intervalle ne soit pas réduit d'autant que par la méthode de réduction d'intervalle, mais la vitesse de convergence asymptotique de la méthode de Newton nous assure que ses estimés seront bien meilleurs que ceux de la méthode de réduction d'intervalle.

Pour être vraiment intéressant, l'algorithme combiné devrait posséder les propriétés de convergence globale de la méthode de bissection, et la vitesse de convergence asymptotique de la méthode de Newton.

Théorème 1.8.1 *L'algorithme combiné est assuré d'approcher soit un minimum local de f ou encore un point d'inflexion. Si f de classe $\mathcal{C}^3$ et si le minimum local cerné par l'algorithme combiné est tel que $f''(x^*) > 0$, alors, en posant $\epsilon = 0$, à partir d'un certain point, x^+ sera toujours choisi égal à x_N.*

Preuve L'algorithme est assuré d'approcher un minimum local ou un point d'inflexion car $f'(\max(x_0, x_1))$ demeure positif alors que $f'(\min(x_0, x_1))$ demeure négatif ou nul.

Supposons maintenant que l'algorithme engendre des intervalles qui cernent un minimum local isolé strict (théorème 1.3.1). Chaque extrémité des intervalles converge vers x^*. En général, soit x_0 ou x_1 est plus proche de x^*. S'ils sont à égale distance, et que le point x_N n'est pas choisi, alors l'algorithme termine à la prochaine itération avec $f'(x^*) = 0$.

Bissection-Newton

{ Donnés : x_0 et x_1 tels que $f'(x_0) * f'(x_1) < 0$ et $(f'(x_1) - f'(x_0))(x_1 - x_0) > 0$; }
{ un critère d'arrêt, $\epsilon > 0$; }
{ un seuil de réduction d'intervalle, $\frac{1}{2} < \gamma < 1$. }
répéter
    $d_N \leftarrow -f'(x_1)/f''(x_1)$
    si $(((x_0 - x_1)d_N > 0) \wedge (d_N/(x_0 - x_1) < \gamma))$ **alors** $x^+ \leftarrow x_1 + d_N$
    sinon $x^+ \leftarrow (x_0 + x_1)/2$
    si $(x_1 > x_0)$ **alors**
        si $(f'(x^+) \leq 0)$ **alors** $x_0 \leftarrow x_1$; $x_1 \leftarrow x^+$
        sinon $x_1 \leftarrow x^+$
    sinon
        si $(f'(x^+) \leq 0)$ **alors** $x_1 \leftarrow x^+$
        sinon $x_0 \leftarrow x_1$; $x_1 \leftarrow x^+$
jusqu'à $(|x_1 - x_0| < \epsilon \vee (|f'(x_1)| < \epsilon \wedge f''(x_1) > 0))$

Algorithme 1.4: Réduit intervalle accéléré—bissection–Newton.

Iter	x_0	x_1	$f'(x_1)$
0	-2.0000000000000000	2.0000000000000000	0.0828402366863905
1	2.0000000000000000	0.0000000000000000	-0.2400000000000000
2	0.0000000000000000	1.0000000000000000	0.2500000000000000
3	1.0000000000000000	0.5000000000000000	-0.0946745562130177
4	0.5000000000000000	0.6065573770491803	0.0064028281317895
5	0.6065573770491803	0.5999982373826895	-0.0000017213059672
6	0.6065573770491803	0.6000000000000000	0.0000000000000000

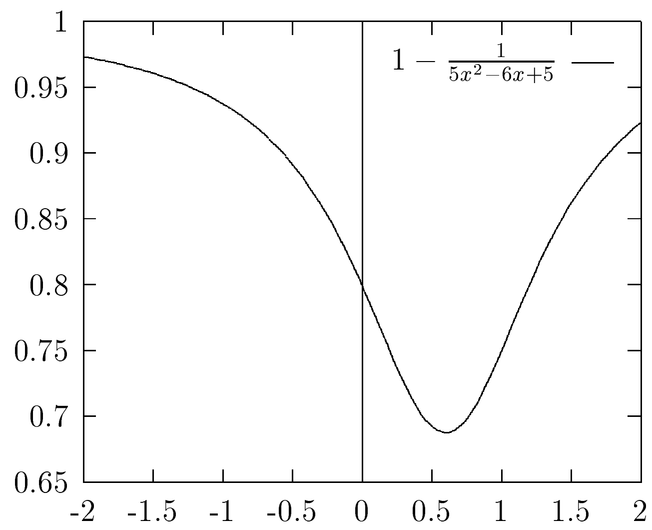
TAB. 1.3 – Itérations de l'algorithme combiné sur la fonction $1 - \frac{1}{5x^2-6x+5}$.

Puisque les intervalles diminuent, éventuellement, ce sera x_1 qui sera le plus proche de x^*. Puisque $x_N - x^* \sim \mathcal{O}(x_1 - x^*)^2$, si x_1 est suffisamment proche de x^*, $(x_1 - x_N)/(x_1 - x_0) < \frac{1}{2} < \gamma$ et le point x_N est retenu. Dans ces conditions, à la prochaine itération, le point x_N devient le point x_1, qui sera le point le plus proche de x^*, et donc le point x_N sera encore retenu, et ainsi de suite. □

Sur le tableau 1.3, on peut observer la réduction de l'intervalle initial par des pas de bissection, ainsi que la convergence quadratique des dernières itérations. En fait, même en $x = 0$, $f''(x) < 0$ et donc l'itération de Newton n'a pas de sens. Cependant, dès que x atteint 0.5, la combinaison Newton/bissection change automatiquement au profit de l'itération de Newton, et seulement trois autres itérations sont requises pour identifier le minimum local à la précision de la machine, 10^{-16}.

1.9 Organisation des chapitres

La structure générale des chapitres qui suivent prends donc la forme suivante :



On observe que la fonction possède un minimum isolé en $x = 0.6$; cependant, l'algorithme de Newton ne peut pas être utilisé loin de $x = 0.6$ car la fonction est *courbée vers le bas*, condition dûe à une dérivée seconde négative.

FIG. 1.8 – Fonction $1 - \frac{1}{5x^2-6x+5}$

1. Formulation du problème.
2. Énoncé de conditions d'optimalité.
3. Déduction d'algorithmes à partir de points qui ne satisfont pas aux conditions d'optimalité.
4. Étude détaillée de quelques algorithmes spécifiques.
5. Solution des calculs non-triviaux des algorithmes spécifiques.

Chapitre 2

Optimisation locale de fonctions différentiables sans contrainte

Sujets du chapitre.

- Analyse des conditions d'optimalité (équations ou inéquations satisfaites pour toute solution du problème d'optimisation).
- Déduction d'une famille d'algorithmes itératifs globalement convergents à partir de la violation des conditions d'optimalité.
- Présentation de l'algorithme de Newton modifié.
- Algorithmes spécialisés aux fonctions quadratiques.

Introduction

Les problèmes d'optimisation les plus simples sont ceux qui consistent à minimiser (ou maximiser) une fonction $f : \mathbb{R}^n \to \mathbb{R}$ où la fonction f est supposée définie dans $\mathbb{R}^n$ en entier, et de plus de classe $\mathcal{C}^3(\mathbb{R}^n)$. En fait, on suppose que f est suffisamment différentiable pour éviter les difficultés dans l'analyse des algorithmes de solution. De plus, seuls les *minima locaux* retiennent notre attention dans ce premier contact.

Les vecteurs de $\mathbb{R}^n$ sont des vecteurs *colonnes*. Les gradients des fonctions $f : \mathbb{R}^n \to \mathbb{R}$, notés $\nabla f(x)$ sont des vecteurs *lignes*. Une motivation pour cette notation est que le gradient est en fait une application linéaire de $\mathbb{R}^n$ dans $\mathbb{R}^n$, donc située dans le dual de $\mathbb{R}^n$. Bien que les espaces euclidiens soient auto-duaux, il est commode de considérer le dual comme un espace ligne.

La classe de tels problèmes (aussi appelés *programmes non-linéaires*) peut sembler trop simple pour être vraiment utile et/ou intéressante à étudier. Cependant, l'étude de cette classe de problèmes permet d'introduire les notions fondamentales qui seront utilisées constamment dans l'étude de problèmes plus généraux, comme par exemple les problèmes où la fonction f n'est plus assez différentiable pour permettre une analyse simple telle que nous présentons dans ce chapitre, ou encore la classe de problèmes de minimisation d'une fonction dans un domaine restreint de $\mathbb{R}^n$.

Comme dans les chapitres suivants, nous analysons d'abord les équations et inéquations nommées *conditions d'optimalité* qui caractérisent une solution au problème étudié, et ensuite nous déduisons des algorithmes itératifs à partir de la non-réalisation de ces conditions d'optimalité.

2.1 Formulation du problème

Énonçons le problème que nous résolvons dans ce chapitre. Il consiste à trouver un point x^* qui *minimise* la fonction f dans $\mathbb{R}^n$:

$$f(x^*) = \min_{x \in \mathbb{R}^n} f(x),$$

ou encore trouver x^* tel que

$$f(x^*) \leq f(x), \quad \forall x \in \mathbb{R}^n.$$

Comme nous l'avons expliqué au chapitre précédent, nous nous limitons à une version *locale* de ce problème, soit qu'il existe un voisinage $V(x^*)$ de x^* tel que

$$f(x^*) \leq f(x), \quad \forall x \in V(x^*).$$

Puisque la notion de voisinage entre de manière très intime dans la définition du problème, on doit s'attendre à ce que la notion de proximité soit utilisée dans l'analyse de sa solution.

En général, l'analyse du problème de minimiser localement une fonction de n variables ne peut pas se réduire à l'analyse de la minimisation locale d'une fonction d'une seule variable. On pourrait récrire le problème ainsi :

$$f(x^*) \leq f(x^* + d), \quad \forall d \in V(0),$$

où d représente le déplacement de x^* à un point voisin, ou encore, en séparant la notion de *direction* et la notion de *distance* du déplacement,

$$f(x^*) \leq f(x^* + \theta d), \quad \forall d \in I\!R^n, \|d\| = 1, \text{ et } \forall \theta < \epsilon,$$

où ϵ représente le diamètre d'une boule centrée en x^* contenue dans $V(x^*)$. Cependant, nous ne pouvons pas établir d'équivalence entre les restrictions unidimensionnelles et les problèmes originaux.

En fait, si x^* est un minimum local de f, alors 0 est un minimum local de la fonction d'une seule variable $h_{x^*,d}(\theta) \stackrel{\text{def}}{=} f(x^* + \theta d)$, et ce quelle que soit la valeur de $d \in I\!R^n$ telle que $\|d\| = 1$. Cependant, la contraposée n'est pas vraie, comme l'illustre le programme suivant :

$$\min(x^2 - y)(x^2 - 4y),$$

pour lequel l'origine est un point stationnaire correspondant à un point de selle. Pourtant, le long de toute droite passant par l'origine, cette fonction possède un minimum local à l'origine!

2.2 Conditions d'optimalité

Comme nous venons de le voir, en général, l'analyse du problème de minimiser localement une fonction de n variables ne peut pas se réduire à l'analyse de la minimisation locale d'une fonction d'une seule variable. Malheureusement, dans la pratique, il est très difficile de traiter systématiquement des exemples comme la fonction $(x^2 - y)(x^2 - 4y)$, et on se contente souvent de caractérisations issues de l'analyse de la fonction univariée $h_{x^*,d}$.

2.2.1 Analyse du problème à une seule variable

Penchons-nous sur le problème de minimiser une fonction réelle $h(\theta)$. Supposons sans perte de généralité que nous étudions un minimum local de h situé en $\theta = 0$. Exprimons h en série de Taylor autour de $\theta = 0$:

$$h(\theta) = h(0) + h'(0)\theta + \frac{h''(0)\theta^2}{2!} + \frac{h'''(0)\theta^3}{3!} + \ldots + \frac{h^{(j)}(0)\theta^j}{j!} + \ldots$$

Supposons maintenant que les $j - 1$ premières dérivées de h au point 0 sont toutes nulles, et que la première non-nulle est $h^{(j)}(0)$. Alors, en utilisant le théorème de Taylor, on a

$$h(\theta) = h(0) + \frac{h^{(j)}(0)\theta^j}{j!} + R_{j+1}(\theta),$$

où $R_{j+1}(\theta)$ est le reste d'ordre $j + 1$. On voit donc que 0 est un minimum local si $\frac{h^{(j)}(0)\theta^j}{j!} + R_{j+1}(\theta)$ est positif pour des valeurs suffisamment petites de $|\theta|$. Détaillons ce raisonnement.

Maintenant, toujours supposant que h est suffisamment ($j + 1$ fois) différentiable, le reste $R_{j+1}(\theta)$ satisfait à la borne

$$\exists\, K_1 \;:\; |R_{j+1}(\theta)| \leq K_1 |\theta^{j+1}|.$$

Plus précisément, rappelons que lorsque les fonctions sont suffisamment différentiables, le reste s'écrit

$$R_{j+1}(\theta) = \frac{h^{(j+1)}(\xi)\theta^{j+1}}{(j+1)!},$$

où comme à l'accoutumée, $\xi = \alpha\theta$ avec $0 \le \alpha \le 1$. Puisque h est suffisamment différentiable, $h^{(j+1)}$ est une fonction continue, et donc, bornée ; en fait, quelle que soit la borne que l'on ait en tête de la forme $|h^{(j+1)}(0)| + \delta$, il existe un $\epsilon_\delta > 0$ tel que si $|\xi| < \epsilon$, alors $|h^{(j+1)}(\xi)| < |h^{(j+1)}(0)| + \delta$. En d'autres termes, si une fonction ϕ est continue à l'origine, alors, pas trop loin de l'origine, $\phi(\xi)$ ne sera pas beaucoup plus grand que $\phi(0)$. Supposons donc maintenant $\delta = 1$ de sorte que si $|\xi| < \epsilon_1$, on a la garantie que $|h^{(j+1)}(\xi)| < |h^{(j+1)}(0)| + 1$. Donc, on peut prendre $K_1 = \frac{1}{(j+1)!}(|h^{(j+1)}(0)| + 1)$.

Intéressons-nous maintenant au signe de la quantité $\frac{h^{(j)}(0)\theta^j}{j!} + R_{j+1}(\theta)$ qui peut se récrire

$$\frac{\theta^j}{j!}\left(h^{(j)}(0) + \frac{h^{(j+1)}(\xi)\theta}{(j+1)}\right). \tag{1}$$

Puisque nous avons fait l'hypothèse que les $j-1$ premières dérivées de h étaient nulles et que la $j^{\text{ième}}$ est non-nulle, vérifions que le signe de $(h^{(j)}(0) + \frac{h^{(j+1)}(\xi)\theta}{(j+1)})$ est donné par le signe de $h^{(j)}(0)$ lorsque θ est assez petit, et précisons ce que nous entendons par *assez petit*.

Pour ce faire, argumentons que si $h^{(j)}(0)$ domine $\frac{h^{(j+1)}(\xi)\theta}{(j+1)}$ en valeur absolue, ce dernier terme ne peut pas faire changer le signe de la somme, donné par le signe du premier terme. Écrivons donc

$$|h^{(j)}(0)| > \left|\frac{h^{(j+1)}(\xi)\theta}{(j+1)}\right| = \left|\frac{h^{(j+1)}(\xi)}{(j+1)}\right| |\theta|.$$

Donc, si

$$|\theta| < \frac{(j+1)|h^{(j)}(0)|}{|h^{(j+1)}(\xi)|},$$

la somme des deux termes est du même signe que le premier terme $h^{(j)}(0)$. Utilisons maintenant la borne que nous avons trouvée, qui assure que si $\theta < \epsilon_1$, $|h^{(j+1)}(\xi)| < |h^{(j)}(0)| + 1$ pour écrire que si

$$|\theta| < \frac{(j+1)|h^{(j)}(0)|}{|h^{(j+1)}(0)| + 1},$$

alors forcément

$$|\theta| < \frac{(j+1)|h^{(j)}(0)|}{|h^{(j+1)}(\xi)|},$$

à condition bien sûr que $|\theta| < \epsilon_1$. Par conséquent, le signe de la somme est donné par le signe de $h^{(j)}(0)$ dès que

$$|\theta| < \min\left(\epsilon_1,\ \frac{(j+1)|h^{(j)}(0)|}{|h^{(j+1)}(0)| + 1}\right).$$

Dans ces conditions, le signe que nous voulions étudier, de l'équation (1), est le même que le signe de $h^{(j)}(0)\theta^j$.

Maintenant, observons que j doit être pair, car autrement, $h^{(j)}(0)\theta^j$ serait négatif si θ était de signe opposé à $h^{(j)}(0)$. Nous venons donc de déduire que la première dérivée de h en un minimum local est *d'ordre pair*. Si j est pair, alors $h^{(j)}(0)\theta^j$ est positif à condition que $h^{(j)}(0)$ soit positif. Nous venons donc de déduire que cette dérivée d'ordre pair doit avoir une valeur positive.

Théorème 2.2.1 *Une condition nécessaire et suffisante pour qu'une fonction $h : \mathbb{R} \to \mathbb{R}$ possède un minimum local en θ est que la première dérivée non nulle de h en θ soit d'ordre pair, et de valeur positive.*

2.2.2 Conditions d'optimalité pour un problème à n variables

Comme nous l'avons exprimé plus haut, on se contente fréquemment de conditions d'optimalités déduites de l'analyse de problèmes univariés. Nous explorons dans cette section les implications du théorème 2.2.1 lorsqu'appliqué aux fonctions $h_{x^*,d}$, quelle que soit la valeur de $d \in \mathbb{R}^n$.

Exercice 2.2.1 *[Dérivées de h]* Obtenez les expressions des deux premières dérivées de $h_{x,d}(\theta)$ par rapport à θ. □

Dans tous les cas, le théorème 2.2.1 nous informe que $h'_{x^*,d}(0)$ doit être nulle, et ce pour tout d dans $\mathbb{R}^n$. Voyons ce que ça implique.

$$h'_{x^*,d}(0) = \nabla f(x^*)d.$$

En fait, $h'_{x^*,d}(0)$ représente la dérivée directionnelle de f en x^* dans la direction d. Si $\nabla f(x^*)d$ doit s'annuler pour tout $d \in \mathbb{R}^n$, en particulier, il doit s'annuler pour $d = \nabla f(x)^t$, et donc $\|\nabla f(x)\| = 0$, donc il faut que $\nabla f(x^*) = 0$, ce qui constitue la condition d'optimalité de premier ordre.

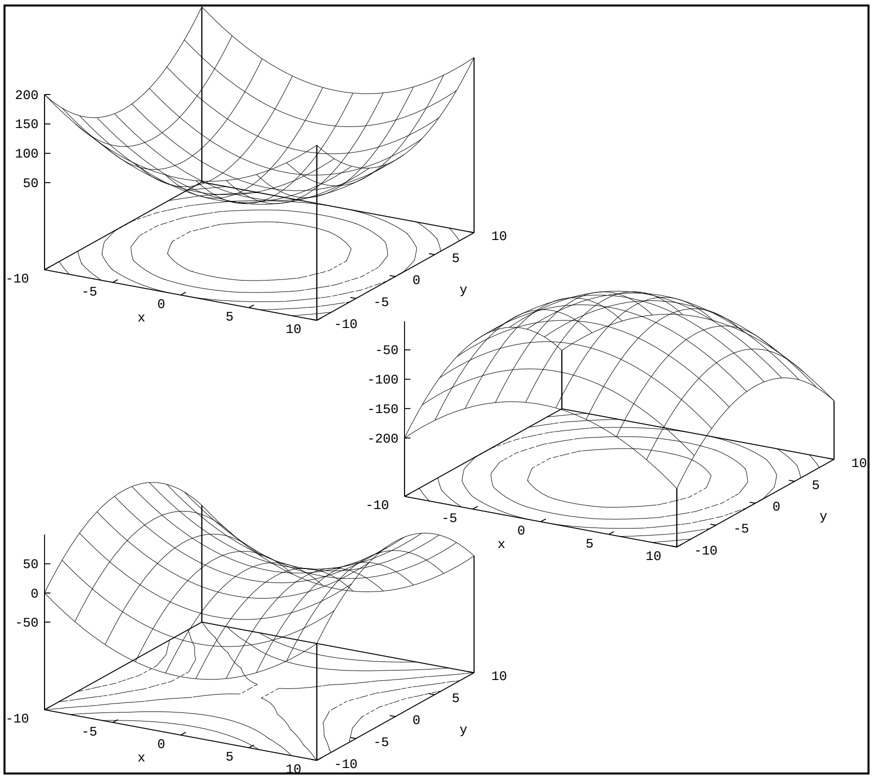
Remarquons que la recherche de minima locaux de f peut débuter par la recherche de racines de l'équation $\nabla f(x) = 0$; comme dans le cas d'une fonction réelle d'une seule variable réelle, de telles racines sont nommées *points stationnaires* pour le problème $\min f(x)$. Nous discuterons plus loin des gains que nous pouvons effectuer en sachant que l'équation $\nabla f(x) = 0$ provient d'un problème d'optimisation.

En plus que $h'(0)$ soit nulle, le théorème nous informe qu'il faut que la première dérivée non nulle de h soit d'ordre pair. Interprétons cette notion pour la *seconde* dérivée de $h_{x^*,d}$.

$$h''_{x^*,d}(0) = d^t\nabla^2 f(x^*)d.$$

C'est la dérivée seconde directionnelle de f en x^* dans la direction d. Si x^* est un minimum local de f, d'après le théorème 2.2.1, cette dérivée est soit nulle, ou encore positive. Par ailleurs, si cette dérivée est positive quel que soit $d \in \mathbb{R}^n$, d'après le théorème 2.2.1, nous sommes assurés que x^* est un minimum local de f. Dans tous les cas, si cette dérivée est négative, x^* n'est certainement pas un minimum local de f.

Il découle des propriétés (définitions) des matrices symétriques que $d^t\nabla^2 f(x^*)d$ est positive (non-négative) quel que soit $d \in \mathbb{R}^n$ si et seulement si la matrice symétrique $\nabla^2 f(x^*)$ est une matrice définie positive (semi-définie positive).



Remarquez les lignes de niveau constant des fonctions : les lignes de niveau sont les mêmes qu'il s'agisse d'un maximum ou d'un minimum, et décrivent des ellipses concentriques ; on distingue les asymptotes de la selle.

FIG. 2.1 – Points stationnaires

Pour des problèmes à n variables, il est difficile de fournir soit des interprétations, soit des formules algébriques pour les dérivées d'ordre supérieur à deux de la fonction $h_{x^*,d}$. Par conséquent, nous nous contenterons de la caractérisation (incomplète) suivante des minima locaux de f :

Théorème 2.2.2 *Soit une fonction $f : I\!R^n \to I\!R$ de classe $\mathcal{C}^3$; si x^* est un minimum local de f, alors $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice semi-définie positive ; à l'opposé, si $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice définie positive, alors x^* est un minimum local de f.*

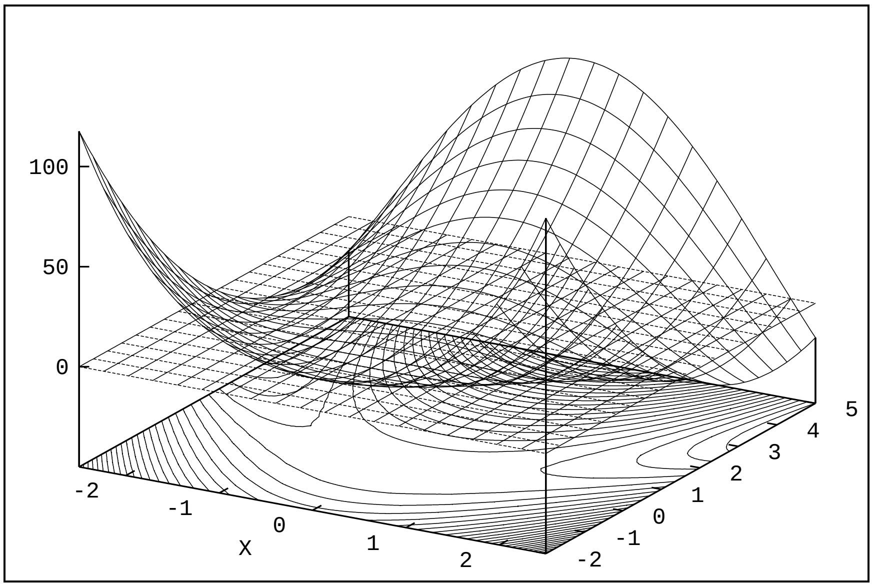
Preuve La condition nécessaire découle de la discussion précédente concernant les deux premières dérivées de la fonction $h_{x^*,d}$ en 0.

Si $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice définie positive, alors, puisque f est de classe $\mathcal{C}^3$, on peut écrire un développement de Taylor avec reste de la forme, où d est unitaire, $\|d\| = 1$:

$$\begin{aligned} f(x^* + \theta d) - f(x^*) &= \theta \nabla f(x^*) d + \frac{1}{2}\theta^2 d^t \nabla^2 (x^*) d + \mathcal{O}(\theta^3) \\ &= \frac{1}{2}\theta^2 d^t \nabla^2 (x^*) d + \mathcal{O}(\theta^3); \end{aligned}$$

Puisque $\nabla^2 f(x^*)$ est définie positive, il existe une constante positive K telle que $d^t \nabla^2 (x^*) d > K$, d'où

$$f(x^* + \theta d) - f(x^*) > K\frac{1}{2}\theta^2 + \mathcal{O}(\theta^3) = \theta^2 (\frac{K}{2} + \mathcal{O}(\theta)).$$



L'origine est un point de selle de la fonction

$$(x^2 - y)(x^2 - 4y).$$

Cependant, l'origine est un minimum local de la restriction de cette fonction à toute droite passant par l'origine ; sur la figure, on voit bien qu'aucune droite passant par l'origine ne pénètre localement dans la zone négative de la fonction

FIG. 2.2 – Point de selle

Cette dernière quantité est positive si θ est assez petit. □

Corollaire 2.2.1 *Soit une fonction $f : \mathbb{R}^n \to \mathbb{R}$; si x^* est un maximum local de f, alors $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice semi-définie négative ; à l'opposé, si $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice définie négative, alors x^* est un maximum local de f.*

Remarquons que le théorème 2.2.2 n'est pas aussi fort que le théorème 2.2.1. En effet, il ne permet pas de distinguer le problème $\min x^4 + y^4$ du problème $\min x^4 - y^4$ car en 0, point stationnaire des deux problèmes, le hessien de chacun des problèmes s'annule, et il faudrait examiner leur dérivées d'ordre supérieur.

Exercice 2.2.2 *[]* Montrez qu'un point qui satisfait aux conditions suffisantes de second ordre est un minimum local strict et isolé. □

Exercice 2.2.3 *[Points stationnaires]* Considérez la fonction $f(x) = (x_1 + x_2)^2 + (2(x_1^2 + x_2^2 - 1) - \frac{1}{3})^2$.

a) Vérifiez que l'origine ainsi que le point $(-\sqrt{\frac{7}{12}}, \sqrt{\frac{7}{12}})$ constituent des points stationnaires de f.

b) Déterminez la nature de chacun de ces points stationnaires.

□

Exercice 2.2.4 *[Point de selle]* Nous avons présenté plus haut (figure 2.2) la fonction $(x^2 - y)(x^2 - 4y)$ pour laquelle l'origine n'est qu'un point de selle, mais constitue un minimum local de la fonction objectif le long de toute droite passant par l'origine. Cet exercice approfondi l'étude de ce problème.

a) Vérifiez que l'origine est un point stationnaire de la fonction $f(x, y) = (x^2 - y)(x^2 - 4y)$: $\nabla f(0, 0) = 0$.

b) Considérez la fonction réelle $h_d(\theta) \stackrel{\text{def}}{=} f(\theta d_x, \theta d_y)$. Vérifiez que quel que soit d, $h'_d(0) = 0$ et $h''_d(0) > 0$.

c) Pour un vecteur d quelconque, identifiez une valeur $\epsilon_d > 0$ pour laquelle $h_d(\theta) \geq 0$ pour toute valeur de θ satisfaisant $|\theta| \leq \epsilon_d$.

d) Vérifiez que $f(x, y) \leq 0$ pour tous les points satisfaisant $x^2 = 2y$.

□

2.3 Déduction d'une classe d'algorithmes itératifs

Dans cette section, nous construisons une famille d'algorithmes itératifs pour calculer un minimum local d'une fonction f. La base de notre étude est la suivante : si un point x ne satisfait pas aux conditions d'optimalité, il est possible de produire un autre point, $\bar{x}$ pour lequel $f(\bar{x}) < f(x)$. Cette famille d'algorithmes porte le nom de *méthodes de descente*. La disponibilité de méthodes de descentes pour les problèmes d'optimisation en fait des problèmes privilégiés, permettant d'obtenir des propriétés de convergence plus fortes que pour des problèmes d'équations non-linéaires $F(x) = 0$ lorsque $F \neq \nabla f$, pour une certaine fonction réelle f. En effet, à moins d'accepter des hypothèses assez fortes sur une application F, il est difficile de construire un algorithme dont la convergence vers un point stationnaire soit assurée. De tels algorithmes prennent la formes d'itérations de points fixes, et seront explorés plus loin. Pour l'instant, nous exploitons la structure d'optimisation de nos problèmes pour construire une classe d'algorithmes assurés de converger vers un point stationnaire quel que soit le point de départ x_0 utilisé.

Supposons que nous disposions d'un point x qui n'est pas un minimum local de la fonction f. Alors, au moins une des deux conditions suivantes n'est pas vérifiée :

1. $\nabla f(x) = 0$, ou
2. $\nabla^2 f(x)$ est une matrice semi-définie positive.

Dans les deux sous-section suivante, nous exploitons la première des conditions précédentes.

2.3.1 Recherche de points stationnaires

Supposons pour l'instant que nous nous contentons de calculer un point stationnaire, satisfaisant $\nabla f(x) = 0$; donc, nous ne nous préocuppons pas de la condition 2 ci haut. En fait, cette seconde condition sera étudiée plus loin, à la section 2.3.3.

Donc, $\nabla f(x) \neq 0$. En se référant au théorème 2.2.1, on constate que l'ensemble des $d \in \mathbb{R}^n$ pour lesquels $h'_{x,d} < 0$ satisfait à la relation

$$\nabla f(x) d < 0,$$

et cet ensemble est nommé $DED(x)$, c'est-à-dire le *Demi Espace de Diminution* de la fonction f en x. La famille des algorithmes de descente utilisera donc des directions de déplacement $d \in DED(x)$. En fait, nous n'avons pas encore spécifié la forme générale des algorithmes que nous étudierons ci-après. Il s'agit d'algorithmes de descentes qui ont tous la même forme, consistant à calculer d'abord une direction dans le $DED(x)$, et ensuite à minimiser (approximativement) la fonction $h_{x,d}(\theta)$ pour passer au point suivant $x^+ = x+\theta d$. Nommons ce passage de x à x^+ $A(x)$, de telle sorte que $x^+ = A(x)$. Les propriétés élémentaires de d et de la minimisation (approximative) de $h_{x,d}$ nous assurent que $f(x^+) = f(A(x)) < f(x)$. Cependant, en analysant l'application A qui engendre une suite $x_{k+1} = A(x_k)$, il faudra introduire d'autres notions pour garantir que la suite $\{x_k\}$ possède bel et bien des points d'accumulation qui sont stationnaires (qui satisfont $\nabla f(x) = 0$). Avant d'approfondir cette étude, énonçons le schéma algorithmique que nous analyserons, et observons un exemple simple d'un algorithme typique.

```
Itération de descente
  { Donnés : x0, ϵ > 0 et 0 < M < ∞ }
  terminé ← faux
  borné ← vrai
  x ← x0
  tantque ( ¬ terminé ∧ borné )
    d ← direction(f, x)
    { h'x,d < 0. }
    θ* ← arg min hx,d(θ)
    x ← x + θ*d
    terminé ← ∇f(x) ≤ ϵ
    borné ← ||x|| < M
```

Algorithme 2.1: Descente simplifiée.

Examinons un exemple : minimisons la fonction quadratique $f(x,y) = 2x^2 - 2xy + y^2 + 2x - 2y$ en utilisant le choix $d = -\nabla f(x)^t$ comme direction de descente. Puisqu'il s'agit d'une fonction quadratique, on peut calculer le pas θ^* exactement : notons $f(x) = 0.5x^tQx + cx$, où dans l'exemple, $Q = \begin{pmatrix} 4 & -2 \\ -2 & 2 \end{pmatrix}$ et $c = \begin{pmatrix} 2 & -2 \end{pmatrix}$; alors, $\nabla f(x) = x^tQ + c$. Les conditions d'optimalité pour $\min_{\theta>0} f(x+\theta d)$ sont données par $\nabla f(x+\theta d)d = 0$, ce qui se simplifie à $\nabla f(x)d + \theta d^tQd = 0$, ou encore

$$\theta^* = -\frac{\nabla f(x)d}{d^tQd}.$$

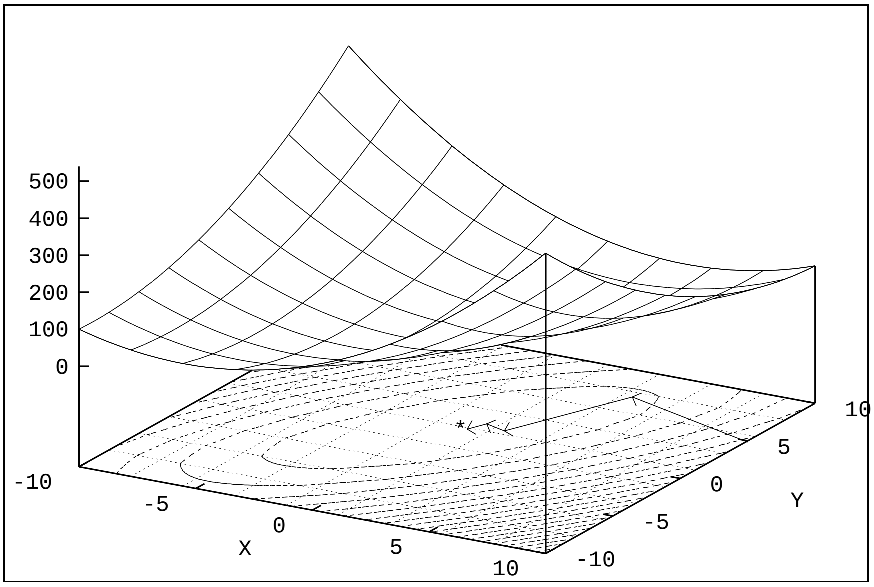
Pour notre exemple, on a de plus $d = -\nabla f(x)^t$ de sorte que

$$\theta^* = \frac{\|\nabla f(x)\|^2}{\nabla f(x)Q\nabla f(x)^t}.$$

Appliquons maintenant l'algorithme à partir du point initial $(10,5)$. En ce point, $\nabla f(10,5) = \begin{pmatrix} 32 & -12 \end{pmatrix}$ et le calcul de θ donne $\theta^* = 0.197297$. Le prochain point engendré est donc le point $\begin{pmatrix} 10 \\ 5 \end{pmatrix} + 0.197297 \begin{pmatrix} -32 \\ 12 \end{pmatrix} = \begin{pmatrix} 3.686486 \\ 7.367568 \end{pmatrix}$. La suite des itérations est donnée dans le tableau 2.1. On peut observer la nature cyclique de cet algorithme, connu sous le nom d'algorithme du gradient, ou encore de la pente la plus forte. À toutes les itérations paires les déplacements sont parallèles et les valeurs du pas θ^* sont les mêmes. Cette propriété sera exploitée à la section 2.5.2.

Iter	$\begin{pmatrix} x \\ y \end{pmatrix}$	$f(x,y)$	θ	$\nabla f(x,y)$
0	$\begin{pmatrix} 10.000000 \\ 5.000000 \end{pmatrix}$	135.000000	0.197297	$(32.000000 \quad -12.000000)$
1	$\begin{pmatrix} 3.686486 \\ 7.367568 \end{pmatrix}$	19.778378	1.073529	$(2.010811 \quad 5.362162)$
2	$\begin{pmatrix} 1.527822 \\ 1.611129 \end{pmatrix}$	2.174566	0.197297	$(4.889030 \quad -1.833386)$
3	$\begin{pmatrix} 0.563229 \\ 1.972851 \end{pmatrix}$	-0.514983	1.073529	$(0.307216 \quad 0.819243)$
4	$\begin{pmatrix} 0.233424 \\ 1.093370 \end{pmatrix}$	-0.925898	0.197297	$(0.746957 \quad -0.280109)$

TAB. 2.1 – Itérations de l'algorithme du gradient sur la fonction $f(x,y) = 2x^2 - 2xy + y^2 + 2x - 2y$.



Le point (0,1) est un minimum local de la fonction $f(x,y) = 2x^2 - 2xy + y^2 + 2x - 2y$. L'algorithme 2.1 est amorcé en $(10,5)$ et engendre les points suivants :

$$(3.69, 7.37)$$
$$(1.53, 1.61)$$
$$(0.56, 1.97)$$
$$(0.23, 1.09)$$
$$\vdots$$

FIG. 2.3 – Exemple de l'algorithme

Exercice 2.3.1 *[Algorithme du gradient]*

a) Pour la fonction $f(x,y) = 2x^2 - 2xy + y^2 + 2x - 2y$, vérifiez que si à une certaine étape k l'algorithme atteint le point $\begin{pmatrix} 0 \\ 1 - 1/5^k \end{pmatrix}$, alors à l'étape $k+2$, il aura atteint le point $\begin{pmatrix} 0 \\ 1 - 1/5^{k+1} \end{pmatrix}$.

b) Pour la fonction $x^2 + 2y^2 + 4x + 4y$, vérifiez par induction que l'algorithme du gradient amorcé à l'origine engendre la suite $\begin{pmatrix} \frac{2}{3^k} - 2 \\ \left(-\frac{1}{3}\right)^k - 1 \end{pmatrix}$. Déduisez-en le minimum de la fonction.

□

2.3.2 Conditions pour la convergence

Notre but est maintenant de produire une suite de points $\{x_k\}$ dont les points d'accumulation sont stationnaires, et ce quel que soit le point de départ x_0 de la suite. Nous utiliserons pour ce faire le fait que nous tentons de minimiser une fonction f, ce qui nous aide à évaluer la qualité relative de deux points consécutifs de la suite.

Plusieurs auteurs ont proposé des schémas d'analyse pour des algorithmes de descente. Le moins restrictif de ceux-ci a été proposé par Zangwill. Simplifié pour notre contexte de minimisation sans contrainte, il s'énonce ainsi :

> Si l'application A qui décrit le passage de x_k à x_{k+1} est continue, et si $f(x_{k+1}) < f(x_k)$, alors les points d'accumulation de la suite $\{x_k\}$, s'il y en a, sont stationnaires pour le problème $\min f(x)$.

Bien qu'étant une caractérisation favorite de l'auteur, nous ne l'approfondirons pas maintenant, car nous donnerons la préférence à un schéma d'analyse moins abstrait, plus facilement applicable directement, et dû à Wolfe.

Pour énoncer ce schéma d'analyse, nous avons besoin des conditions suivantes qui caractérisent les directions d et les pas θ conjointement qui assurent que les points d'accumulation de la suite engendrée par un algorithme de descente les utilisant sont tous stationnaires.

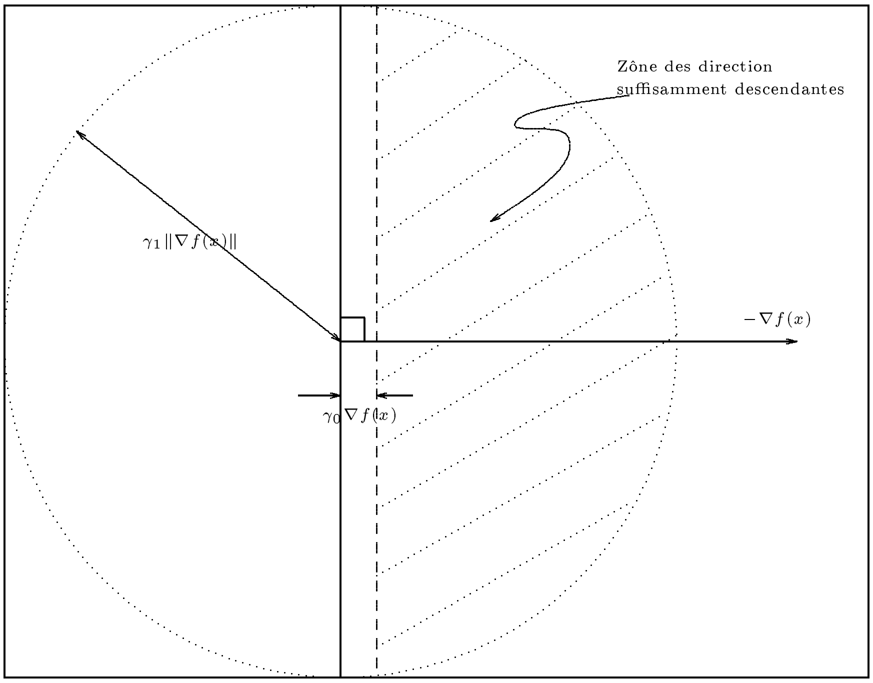
Définition 2.3.1 *Une direction d est considérée* suffisamment descendante *s'il existe deux constantes positives γ_0 et γ_1 indépendantes de x telles que d satisfasse aux inégalités suivantes :*

$$\nabla f(x) d \leq -\gamma_0 \|\nabla f(x)\|^2;$$

$$\|d\| \leq \gamma_1 \|\nabla f(x)\|.$$

La définition précédente nous assure que toute direction d utilisée par un algorithme de descente est un vecteur assez "long", et fait un angle assez aigu avec l'opposé de ∇f. Il est clair que $d = -\nabla f(x)^t$ satisfait à la définition précédente.

Exercice 2.3.2 *[Direction de l'opposé du gradient]* Justifiez que la direction $d = -\nabla f(x)^t$ satisfait aux conditions de la définition 2.3.1 ; pour quelles valeurs de γ_0 et γ_1 ? □



Nous voyons une interprétation géométrique des deux conditions de la définition 1.1. La première nous assure que d nous conduit à réduire l'approximation linéaire de f au moins autant que $-\gamma_0 \nabla f$; la seconde nous assure que d n'est pas plus que γ_1 fois plus longue que ∇f

FIG. 2.4 – Direction admissible

Exercice 2.3.3 *[Géométrie des directions]* Quel est l'angle le plus grand entre une direction d et l'opposé du gradient tel que la direction satisfasse la définition 2.3.1 ? L'angle est fonction de γ_0 et γ_1. □

Définition 2.3.2 *Un pas θ est dit* admissible *pour une direction suffisamment descendante d lorsqu'il satisfait aux deux inégalités suivantes, nommées critère d'Armijo et de Wolfe respectivement :*

$$f(x+\theta d) - f(x) \leq \tau_0 \theta \nabla f(x) d, \quad \tau_0 \in (0, \frac{1}{2}), \qquad \text{(Armijo)}$$

$$\nabla f(x+\theta d) d \geq \tau_1 \nabla f(x) d, \quad \tau_1 \in (\tau_0, 1), \qquad \text{(Wolfe)}$$

où τ_0 et τ_1 sont des paramètres fixés pour déterminer l'admissibilité d'un pas.

L'existence de pas admissibles est cruciale pour l'obtention d'un algorithme bien défini, et fait l'objet du

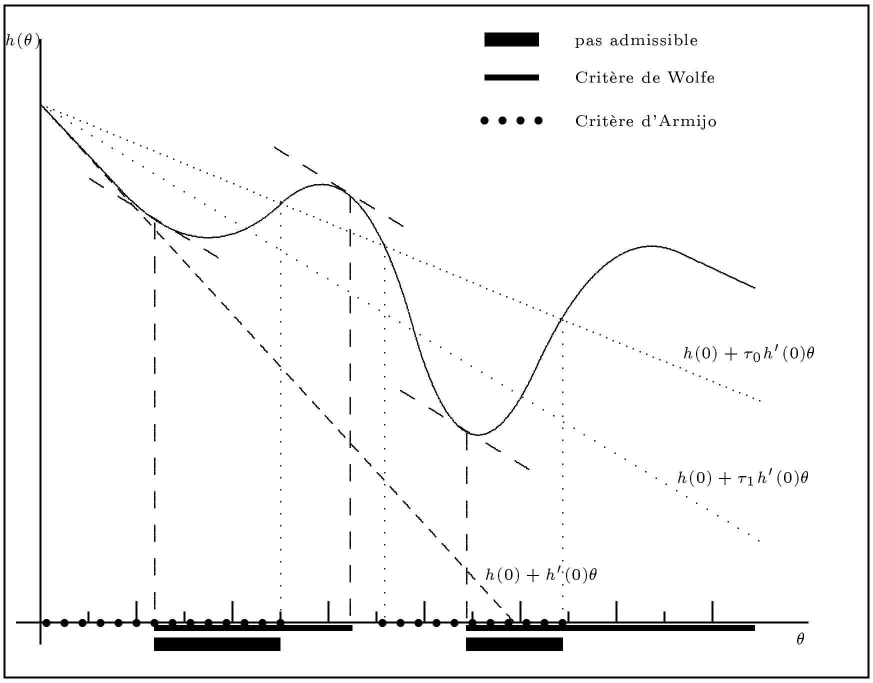
Théorème 2.3.1 *Soit $f \in \mathcal{C}^2(I\!R^n)$, et d satisfaisant à la définition 2.3.1. Alors, si la fonction f est bornée inférieurement dans la direction d, il existe au moins une valeur de θ satisfaisant à la définition 2.3.2.*

La preuve de ce théorème sera présentée un peu plus loin.

Avec ces deux définitions, nous sommes en mesure de démontrer le théorème de convergence globale suivant, sachant qu'un algorithme utilisant des directions suffisamment descendantes et des pas admissibles est bien défini.

Théorème 2.3.2 *Soit un algorithme de descente appliqué au problème*

$$\min_{x \in I\!R^n} f(x), \quad f \in \mathcal{C}^1(I\!R^n);$$



On voit que les pas qui satisfont aux conditions de la définition 2.3.2 constituent un ensemble d'intervalles dans lesquels f est assez petite alors que ses dérivées directionnelles sont assez grandes

FIG. 2.5 – Pas admissible

supposons qu'à chaque itération la direction utilisée d_k est une direction suffisamment descendante, et pour lequel le pas utilisé dans cette direction est un pas admissible ; alors, tous les points d'accumulation de la suite $\{x_k\}$ engendrée par l'algorithme sont des points stationnaires pour le problème $\min f(x)$.

Preuve Supposons que la suite $\{x_k\}$ possède au moins un point d'accumulation $\bar{x}$. Alors, par l'admissibilité des pas θ_k utilisés dans l'algorithme, $f(x_k) > f(\bar{x})$, et ce pour tout k. Donc, $f(x_k)$ est une suite de nombres réels monotone décroissante, et bornée inférieurement par $f(\bar{x})$. Par le théorème de Cauchy, $f(x_{k+1}) - f(x_k)$ converge vers zéro. Donc,

$$\lim_{k\to\infty} f(x_{k+1}) - f(x_k) = f(x_k + \theta_k d_k) - f(x_k) = 0,$$

et en utilisant la condition d'Armijo, ceci entraîne que

$$\lim_{k\to\infty} \tau_0 \theta_k \nabla f(x_k) d_k = 0.$$

Alors, la première des conditions de la définition 2.3.1 nous assure que $\tau_0\theta_k\nabla f(x_k)d_k \leq -\tau_0\theta_k\|\nabla f(x_k)\| < 0$. Ainsi, il se peut que θ_k converge vers 0, ou encore que $\|\nabla f(x_k)\|$ converge vers 0. Supposons que θ_k converge vers 0. Alors, la seconde condition de la définition 2.3.1 nous assure que $0 < \|\theta_k d_k\| \leq \gamma_1\theta_k\|\nabla f(x_k)\|$, et donc que $\|\theta_k d_k\|$ converge vers zéro. Donc, par la condition de Wolfe, on a

$$\lim_{k\to\infty} \nabla f(x_k + \theta_k d_k) d_k = \nabla f(x_k) d_k \geq \lim_{k\to\infty} \tau_1 \nabla f(x_k) d_k,$$

ce qui entraîne que $\lim_{k\to\infty} \nabla f(x_k) d_k = 0$.

Maintenant, puisque $\nabla f(x_k)d_k < -\gamma_0\|\nabla f(x_k)\|^2$, d'après la définition 2.3.1, on retrouve

$$0 = \lim_{k\to\infty} \nabla f(x_k)d_k < -\gamma_0 \lim_{k\to\infty} \|\nabla f(x_k)\|^2,$$

et donc que $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$. Finalement, puisque ∇f est une fonction continue, $\nabla f(\bar{x}) = 0$. □

Un autre variante utilise le *procédé* d'Armijo qui consiste à rechercher un pas θ qui satisfait à la première condition d'admissibilité parmi les puissances de $\frac{1}{2}$; la première de ces puissances est éloignée de zéro par le procédé plutôt que par la condition d'admissibilité.

Théorème 2.3.3 *Soit un algorithme de descente appliqué au problème*

$$\min_{x\in\mathbb{R}^n} f(x), \quad f \in \mathcal{C}^1(\mathbb{R}^n);$$

supposons qu'à chaque itération la direction utilisée d_k est une direction suffisamment descendante, et pour lequel le pas utilisé dans cette direction est la première valeur de la suite $\{1, 1/2, 1/4, \ldots, (1/2)^m \ldots\}$ qui satisfasse au critère d'Armijo ; alors, tous les points d'accumulation de la suite $\{x_k\}$ engendrée par l'algorithme sont des points stationnaires pour le problème $\min f(x)$.

Preuve Imitons la preuve du théorème 2.3.2. Il est clair que nous aurons encore

$$\lim_{k\to\infty} \tau_0\theta_k\nabla f(x_k)d_k = 0,$$

et que nous devons justifier que si θ_k converge vers zéro (pour une certaine sous-suite), alors nécessairement $\nabla f(x_k)d_k$ converge vers 0. Supposons que $\nabla f(x_k)d_k$ ne converge pas vers zéro. Alors, θ_k converge vers zéro, et donc, à partir d'un certain rang, on ne prend plus le pas unitaire. Donc,

$$f(x_k + 2\theta_k d_k) - f(x) > 2\tau_0\theta_k\nabla f(x_k)d_k. \tag{2.1}$$

En dénotant

$$p_k = \frac{d_k}{\|d_k\|} \quad , \quad \bar{\theta}_k = 2\theta_k\|d_k\|,$$

on a toujours que $\bar{\theta}_k$ converge vers zéro. Puisque p_k est normalisé, la suite p_k possède des points d'accumulation, et on peut extraire une sous-suite telle que p_k converge vers $\bar{p}$. De la relation (2.1), on tire que

$$\frac{f(x_k + \bar{\theta}_k p_k) - f(x)}{\bar{\theta}_k} > \tau_0\nabla f(x_k)p_k, \tag{2.2}$$

et en prenant la limite dans (2.2), on retrouve

$$0 \le (1 - \tau_0)\nabla f(\bar{x})\bar{p},$$

et puisque $\tau_0 < 1$,

$$0 \le \nabla f(\bar{x})\bar{p},$$

ce qui contredit le fait que les directions d_k sont suffisamment descendantes, puisque $\bar{p}$ est la normalisation de la limite d'une sous-suite de directions d_k. □

Si l'objectif f est supposé de classe $\mathcal{C}^2$, une preuve plus simple permet de conclure.

Corollaire 2.3.1 *Soit un algorithme de descente appliqué au problème*

$$\min_{x\in\mathbb{R}^n} f(x), \quad f \in \mathcal{C}^2(\mathbb{R}^n);$$

supposons qu'à chaque itération la direction utilisée d_k *est une direction suffisamment descendante, et pour lequel le pas utilisé dans cette direction est la première valeur de la suite* $\{1, 1/2, 1/4, \ldots, (1/2)^m \ldots\}$ *qui satisfasse au critère d'Armijo ; alors, tous les points d'accumulation de la suite* $\{x_k\}$ *engendrée par l'algorithme sont des points stationnaires pour le problème* $\min f(x)$.

Preuve Imitons la preuve du théorème 2.3.3. Le début est semblable, mais on réexprime le critère d'Armijo en utilisant un développement de Taylor d'ordre deux : $f(x+\theta d) = f(x) + \theta\nabla f(x)d + \frac{\theta^2}{2}d^t\nabla^2 f(\tilde{x})d$ pour obtenir

$$\theta_k(1-\tau_0)\nabla f(x_k)d_k + \frac{1}{2}\theta_k^2 d_k^t \nabla^2 f(\tilde{x})d_k < 0.$$

Maintenant, selon la définition 2.3.1, on a

$$\nabla f(x_k)d_k < \frac{-\gamma_0}{\gamma_1^2}\|d_k\|^2,$$

et puisque f est de classe $\mathcal{C}^2(\mathbb{R}^n)$, il existe une constante positive A telle que, dès que x_k est suffisamment proche d'un point stationnaire $\bar{x}$,

$$\frac{1}{2}d_k^t\nabla^2 f(\tilde{x})d_k < A\|d_k\|^2.$$

En combinant ces deux dernières inégalités dans la réécriture du critère d'Armijo, on a que

$$\theta_k(1-\tau_0)\frac{-\gamma_0}{\gamma_1^2}\|d_k\|^2 + \theta_k^2 A\|d_k\|^2 < 0$$

entraîne la condition d'Armijo. Mais $\theta_k > 0$, et donc dès que $\theta_k < \frac{(1-\tau_0)\gamma_0}{A\gamma_1^2}$, le critère d'Armijo est satisfait, et donc si $\nabla f(x_k)d_k$ ne converge pas vers zéro, θ_k non plus puisqu'il satisfait $\theta_k > \frac{(1-\tau_0)\gamma_0}{2A\gamma_1^2}$ par construction. □

Preuve (du théorème 2.3.1)
Il est clair de la discussion précédente qu'il existe des valeurs de θ qui satisfont au critère d'Armijo. Vérifions que parmi celles-ci, au moins une satisfait au critère de Wolfe.

Considérons la plus petite valeur de θ pour laquelle le critère d'Armijo est satisfait avec égalité : $\bar{\theta} = \inf_\theta : f(x+\theta d) - f(x) > \tau_0\theta\nabla f(x)d$. Si le critère d'Armijo est satisfait pour tout $\theta > 0$, c'est que la fonction f n'est pas bornée inférieurement dans la direction d.

En $\bar{\theta}$, la dérivée directionnelle de f dans la direction d est plus grande que $\tau_0\nabla f(x)d$ puisqu'immédiatement avant ce point, la fonction devient plus grande que la droite de pente $\tau_0\nabla f(x)d$. Or, cette dérivée directionnelle est justement donnée par l'expression $\nabla f(x+\theta d)d$. On a donc

$$\nabla f(x+\theta d)d > \tau_0\nabla f(x)d > \tau_1\nabla f(x)d$$

puisque $\tau_1 > \tau_0$. □

Direction de la pente la plus forte

Des conditions de descente suffisante, il devrait maintenant être clair que la direction dite de *la pente la plus forte*, $d = -\nabla f(x)^t$ serait une direction candidat privilégiée. Pour mieux qualifier la nature optimale de cette direction, revenons aux conditions d'optimalité présentées plus haut. Puisque nous ne cherchons seulement qu'un point stationnaire, tentons de trouver d qui minimise $h'_{x,d}(0) = \nabla f(x)d$. Évidemment, un tel d serait non-borné, égal à $-\infty\nabla f(x)$. Pour se sortir d'une impasse apparente, considérons l'inégalité de Cauchy-Schwartz :

$$-\|d\|\|\nabla f\| \leq \nabla f(x)d \leq \|d\|\|\nabla f\|.$$

Alors, on voit bien que la valeur de d qui atteint la borne inférieure des précédentes inégalités est précisément $d = -\nabla f(x)^t$.

Malheureusement, cette direction $d = -\nabla f(x)^t$ conduit à un algorithme qui converge très lentement.

Exercice 2.3.4 *[Vitesse de convergence de l'algorithme du gradient]* On peut montrer que l'ordre de convergence de l'algorithme de la pente la plus forte est linéaire, et que le taux de convergence est donné par le rapport $\frac{1-r}{1+r}$ où $r = \lambda_{min}/\lambda_{max}$ et λ_{min} et λ_{max} sont les valeurs propres extrêmes de la matrice $\nabla^2 f(x^*)$. Considérez la fonction quadratique $f(x, y) = 10x^2 + y^2$, et vérifiez numériquement que ce taux est atteint si l'on amorce l'algorithme du gradient en $\begin{pmatrix} 0.1 \\ 1 \end{pmatrix}$. Remarquons que ce taux peut être arbitrairement proche de un si le rapport des valeurs propres extrêmes est proche de zéro. □

Direction de Newton

Plutôt que de "minimiser" $h'_{x,d}(0)$, minimisons par rapport à d la fonction $h'_{x,d}(0) + \frac{1}{2}h''_{x,d}(0)$. Un examen sommaire des expressions impliquées permet de reformuler cette minimisation par rapport à d comme

$$\min_d \nabla f(x)d + \frac{1}{2}d^t\nabla^2 f(x)d,$$

dont la solution, en autant que $\nabla^2 f(x)$ soit une matrice définie positive (et donc inversible), peut s'écrire

$$d_N = -\nabla^2 f(x)^{-1}\nabla f(x)^t.$$

Le fait qu'elle satisfasse aux critères de la définition 2.3.1 est démontré à la section 2.4.

Cette direction d_N est importante, et correspond à la direction dite de Newton. Pour comprendre cette terminologie de *direction de Newton*, rappelons la méthode de Newton pour la solution de systèmes d'équations non-linéaires $F(x) = 0$:

$$x_{k+1} \leftarrow x_k - \nabla F(x_k)^{-1}F(x_k).$$

Pour les problèmes de minimisation, nous retrouvons les identités $\nabla f \equiv F^t$ et $\nabla^2 f \equiv \nabla F$. Selon cette notation, la direction de Newton serait $x_{k+1} - x_k = \nabla F(x_k)^{-1}F(x_k)$.

Nous proposons à la section 2.4.3 une analyse qui illustre que cette direction de Newton conduit, au moins localement, à un algorithme efficace.

Exercice 2.3.5 *[Méthode de Newton]* Considérez à nouveau (exercice 2.2.3 page 41) la fonction $f(x) = (x_1 + x_2)^2 + (2(x_1^2 + x_2^2 - 1) - \frac{1}{3})^2$. Vérifiez que la méthode de Newton peut être appliquée à partir de $x^0 = (\sqrt{7/6}, 0)^t$ et que le prochain point, en utilisant $\theta = 1$ est situé sur la droite reliant les deux points stationnaires. □

Une famille de directions de suffisamment descendantes

La direction de Newton peut être vue comme une transformation linéaire de la direction de la plus forte pente. La matrice représentant cette transformation linéaire pour la direction de Newton est l'inverse de la matrice hessienne de f.

Considérons le cas général d'une direction $\bar{d} \stackrel{\text{def}}{=} -H\nabla f(x)^t$. En supposant que H est une matrice symétrique définie positive, il est facile de vérifier que $\bar{d}$ satisfait aux conditions de la définition 2.3.1. En effet, on peut écrire

$$\nabla f(x)\bar{d} = -\nabla f(x) H \nabla f(x)^t,$$

et si $\lambda_{min} > 0$ dénote la valeur propre la plus petite de H, on a

$$\nabla f(x) H \nabla f(x)^t \geq \lambda_{min} \|\nabla f(x)\|^2.$$

Ceci assure que

$$\nabla f(x)\bar{d} \leq -\lambda_{min} \|\nabla f(x)\|^2.$$

Pour vérifier la seconde inégalité de la définition 2.3.1, nous utilisons plutôt la plus grande valeur propre de H, λ_{max} pour écrire

$$\|\bar{d}\| \leq \lambda_{max} \|\nabla f(x)\|.$$

Enfin, remarquons que si H est obtenue par l'inversion d'une matrice (inversible) B, qui approche probablement le hessien de f, alors H est définie positive si B l'est aussi. Ceci se vérifie par les manipulations suivantes :

$$d^T H d = d^t(B^{-1})d = d^t(B^{-1}BB^{-1})d = (B^{-1}d)^t B (B^{-1}d) > \lambda_{min} \|B^{-1}d\|^2.$$

Maintenant, $\|B^{-1}d\| \geq \frac{1}{\lambda_{max}} \|d\|$, où λ_{min} et λ_{max} sont les valeurs propres extrêmes de B, supposées toutes deux positives. Par conséquent,

$$d^T H d > \frac{\lambda_{min}}{\lambda_{max}^2} \|d\|^2.$$

Exercice 2.3.6 *[Direction de descente]* Complétez l'analyse des directions $d = -B^{-1}\nabla f(x)^t$ en fournissant des expressions pour γ_0 et γ_1. □

2.3.3 Recherche de minima locaux faibles

Si la condition que $\nabla^2 f(x)$ soit semi-définie positive n'est pas vérifiée, même si x est un point stationnaire, il n'en est pas un minimum local pour autant. En fait il est essentiel que $\nabla^2 f(x)$ soit semi-définie positive pour qu'un point stationnaire x soit un minimum local, mais ce n'est toutefois pas suffisant. Cependant, comme nous en avons discuté à la section 2.2, nous nous contentons de ces conditions.

Dans la suite de cette sous-section, nous développons un schéma algorithmique qui assure que les points d'accumulation des algorithmes, en plus d'être stationnaires, possèdent un hessien $\nabla^2(x)$ semi-défini positif. Ce schéma algorithmique nous permet également d'introduire une technique qui n'utilise pas un pas admissible dans une direction suffisamment descendante, mais plutôt une restriction ensembliste sur la direction, dans laquelle on effectue un pas unitaire. Ce type d'algorithme n'est pas sans rappeler les techniques de réduction d'intervalle utilisées au chapitre 1.

L'idée sous-jacente aux méthodes de *région de confiance* est que le développement de Taylor d'ordre un ou deux utilisé pour calculer une direction constitue une bonne approximation de f dans un certain voisinage du point courant. Le développement de Taylor

$$q_x(d) = f(x) + \nabla f(x)d + \frac{1}{2}d^t\nabla^2 f(x)d$$

sera nommé *fonction modèle* de f, et la direction sera calculée en résolvant un problème de la forme

$$\min_{\|d\|\leq\Delta_k} q_{x_k}(d).$$

La valeur de Δ_k est ajustée à l'aide de la quantité r_k ainsi définie :

$$\begin{aligned} \Delta f_{x_k} &\stackrel{\text{def}}{=} f(x_k) - f(x_{k+1}) \\ \Delta q_{x_k} &\stackrel{\text{def}}{=} f(x_k) - q(d_k) \\ r_k &\stackrel{\text{def}}{=} \frac{\Delta f_{x_k}}{\Delta q_{x_k}}. \end{aligned}$$

Ainsi, r_k représente le rapport entre la réelle diminution de f en comparaison avec la diminution prévue par la fonction modèle q. Pour la méthode de Newton, on s'attend à ce que r_k s'approche de l'unité lorsque x_k converge vers un minimum local de f.

Pour formuler un algorithme utilisant ces quantités, remarquons que si r_k est petit, il est vraisemblable que Δ_k soit trop grand, alors que si r_k s'approche de un, il faudrait augmenter Δ_k pour que l'algorithme se comporte exactement comme l'itération de Newton asymptotiquement. Ceci conduit aux règles suivantes, où des choix typiques des paramètres $0 < \eta_1 < \eta_2 < 1$ et $0 < \gamma_1 < 1 < \gamma_2$ sont $\eta_1 = 0.25$, $\eta 2 = 0.75$, $\gamma_1 = 0.25$ et $\gamma_2 = 2$.

- si $r_k < \eta_1$ alors $\Delta_{k+1} \leftarrow \gamma_1\Delta_k$;
- si $r_k > \eta_2$ alors
  - si $\|d_k\| = \Delta_k$ alors $\Delta_{k+1} \leftarrow \gamma_2\Delta_k$
  - sinon $\Delta_{k+1} \leftarrow \Delta_k$;
- si $r_k > \eta_1$ alors $x_{k+1} \leftarrow x_k + d_k$
- sinon $x_{k+1} \leftarrow x_k$.

```
Newton—région de confiance(x, Δ, ε, f)
{ Donnés : x ; }
{    un critère d'arrêt, ε > 0 ; }
{    une fonction deux fois différentiable f ; }
{    une taille initiale Δ de la région de confiance. }
répéter
    d_R ← arg min_{||d||≤Δ} q_x(d)
    r ← (f(x) − f(x + d_R)) / (q(0) − q(d_R))
    si (r < η1 ) alors Δ ← γ1Δ
    sinon
        x ← x + d_R
    si (r > η2 ) alors
        si (||d_R|| = Δ ) alors
            Δ ← γ2Δ
jusqu'à ( ||∇f(x)|| < ε)
Résultat ← x
```

Algorithme 2.2: Newton avec région de confiance.

Cet algorithme possède des propriétés de convergence globales très fortes, au prix de sous-problèmes plus difficiles à résoudre, la minimisation *globale* $\min_{\|d\|\leq\Delta_k} q_{x_k}(d)$.

Théorème 2.3.4 *Si les points engendrés par l'algorithme demeurent dans un compact K, avec f de classe $\mathcal{C}^2$ dans K, et $\nabla^2 f$ uniformément borné dans K, alors il existe un point d'accumulation $\bar{x}$ de la suite $\{x_k\}$, et tout point d'accumulation satisfait que $\nabla f(\bar{x}) = 0$ et $\nabla^2 f(\bar{x})$ est une matrice semi-définie positive.*

Preuve Selon que $\inf \Delta_k = 0$ dans l'algorithme, il existe une sous-suite notée $\{x_k\}$ qui converge vers $\bar{x}$ pour laquelle soit

1. $r_k \geq \eta_1$, et donc $\inf \Delta_k > 0$, ou
2. $r_k < \eta_1$, entraînant $\Delta_{k+1} \to 0$ et $\|d_k\| \to 0$.

cas 1
Ce cas devrait se produire le plus souvent car on espère que d_k sera localement identique à la direction de Newton, et donc que la borne $\|d_k\| \leq \Delta_k$ ne sera pas saturée. Puisque $f(\bar{x})$ est une quantité bornée (les x_k sont contenus dans un compact), on a que la somme $\sum \Delta f_{x_k} < \infty$, ce qui entraîne que $\Delta f_{x_k} \to 0$, et donc, puisque $r_k \geq \eta_1$, $\Delta q_{x_k} \to 0$.

Maintenant, en définissant $\bar{q}(d) \stackrel{\text{def}}{=} f(\bar{x}) + \nabla f(\bar{x})d + \frac{1}{2}d^t\nabla^2 f(\bar{x})d$, et $0 < \bar{\Delta} < \inf \Delta_k$, considérons $\bar{d}$ la solution de $\min_{\|d\|\leq\bar{\Delta}} \bar{q}(d)$. Dénotons par $\hat{x} \stackrel{\text{def}}{=} \bar{x} + \bar{d}$. Observons que si k est assez grand, $\|\hat{x} - x_k\| \leq \Delta_k$, ce qui entraîne que

$$q_{x_k}(\hat{x} - x_k) \geq q_{x_k}(d_k) = f(x_k) - \Delta q_{x_k}.$$

En passant à la limite dans les dernières relations, on retrouve

$$\bar{q}(\bar{d}) \geq f(\bar{x}) = \bar{q}(0),$$

ce qui nous indique que 0 est solution de $\min_{\|d\|\leq\bar{\Delta}} \bar{q}(d)$, et puisque $0 < \bar{\Delta}$, ceci entraîne que $\nabla f(\bar{x}) = 0$ et $\nabla^2 f(\bar{x})$ est une matrice semi-définie positive.

cas 2

Séparons cette portion de la preuve en deux : établissons d'abord une contradiction au fait que $\nabla f(\bar{x}) \neq 0$, et ensuite établissons une contradiction au fait que $\nabla^2 f(\bar{x})$ possède une valeur propre négative.

1. Supposons que $\nabla f(\bar{x}) \neq 0$. À partir d'un x_k quelconque, considérons un déplacement le long de la direction opposée au gradient $\hat{d}_k = -\nabla f(x_k)^t$. Alors, en un point le long de cette direction,

$$q_{x_k}(\alpha\hat{d}_k) = f(x_k) - \alpha\|\nabla f(x_k)\|^2 + \frac{\alpha^2}{2}\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k$$

est minimisé (si $\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k$ est positif) pour $\alpha^* = \|\nabla f(x_k)\|^2/(\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k)$, et $q^* = q_{x_k}(\alpha^*\hat{d}_k) = \frac{1}{2}\|\nabla f(x_k)\|^4/(\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k)$. Par ailleurs, pour demeurer dans la région de confiance, on doit avoir $\alpha \leq \Delta_k/\|\hat{d}_k\| = \alpha_{max}$, qui est également le minimum lorsque $\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k$ est négatif. Puisque nous sommes dans le cas où $\Delta_k \to 0$, $\Delta q_{x_k} \geq q_{x_k}(0) - q_{x_k}(\alpha_{max}\hat{d}_k)$. Si $\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k$ est positif, $q_{x_k}(\alpha\hat{d}_k)$ est convexe en α, et on obtient $\Delta q_{x_k} \geq (q_{x_k}(0) - q^*)\frac{\Delta_k}{\alpha^*} = \frac{\Delta_k}{2}\|\nabla f(x_k)\|$. On retrouve cette dernière inégalité directement si $\hat{d}_k^t\nabla^2 f(x_k)\hat{d}_k$ est négatif.
Un développement de Taylor de f près de x_k donne

$$f(x_k + d_k) = q_{x_k}(d_k) + \mathcal{O}(\|d_k\|^3), \qquad (**)$$

et donc

$$\Delta f_{x_k} = \Delta q_{x_k} + \mathcal{O}(\|d_k\|^3).$$

Or, $\|d_k\| < \Delta_k$ et on vient de voir que $\Delta q_{x_k} \geq \frac{\Delta_k}{2}\|\nabla f(x_k)\| \sim \mathcal{O}(\Delta_k)$. Par conséquent, $r_k = 1 + \mathcal{O}(\|d_k\|)$, contredisant l'hypothèse que $r_k < \eta_1$.

2. Supposons maintenant qu'il existe une direction de courbure négative unitaire $\bar{d}$ telle que

$$\bar{d}^t\nabla^2 f(\bar{x})\bar{d} = -\delta < 0,$$

et en un point de la sous-suite, effectuons un déplacement de longueur Δ_k dans la direction $\sigma\bar{d}$ avec $\sigma = \pm 1$ choisi pour que $\nabla f(x_k)\bar{d} \leq 0$. Encore une fois, on écrit

$$\Delta q_{x_k} \geq q_{x_k}(0) - q_{x_k}(\Delta_k\sigma\bar{d}) \geq -\frac{1}{2}\Delta_k^2\bar{d}^t\nabla^2 f(x_k)\bar{d} = \frac{1}{2}\Delta_k^2\delta + \mathcal{O}(\Delta_k^3).$$

Ceci implique $r_k = 1 + \mathcal{O}(\|d_k\|)$, contredisant encore une fois l'hypothèse que $r_k < \eta_1$.

□

La preuve du théorème est importante car elle suggère des critères plus faibles que la minimisation globale de $q_x(d)$ dans la région $\|d\| \leq \Delta$. En particulier, si l'on veut garantir que les points d'accumulation sont stationnaires, il suffit que q décroisse autant qu'un pas optimal dans la direction de la plus forte pente $-\nabla f(x)^t$; un tel pas est habituellement nommé *point de Cauchy*. Pour la recherche de points stationnaires, il n'est pas nécessaire que le modèle q soit défini à partir de $\nabla^2 f$, et une matrice M_k peut fort bien être utilisée.

Corollaire 2.3.2 *Si l'algorithme 2.2 utilise un modèle* $q = f(x) + \nabla f(x)d + \frac{1}{2}d^t M_x d$ *avec des matrices* M_x *uniformément bornées, et si* q *est minimisée approximativement au moins pour obtenir la réduction du point de Cauchy, alors les points d'accumulation sont stationnaires.*

Exercice 2.3.7 *[Preuve du corollaire 2.3.2]* Adaptez la preuve du théorème 2.3.4 ; indice : le cas 2 n'a pas besoin de modification, et le cas 1 doit tenir compte du fait que d n'est plus un minimum global, mais seulement une réduction au moins aussi bonne que le point de Cauchy. □

D'un autre côté, pour garantir que les points d'accumulation satisfont aux conditions nécessaires de second ordre, il faut bien utiliser $\nabla^2 f$, mais il suffit d'assurer que q_x décroit au moins autant qu'une direction de courbure négative associée à la valeur propre la plus petite, et que le point de Cauchy.

Corollaire 2.3.3 *Si l'algorithme 2.2 utilise un modèle* $q = f(x) + \nabla f(x)d + \frac{1}{2}d^t \nabla^2 f(x)d$*, et si* q *est minimisée approximativement au moins pour obtenir la réduction du point de Cauchy et une réduction donnée par la direction de courbure la plus négative, alors les points d'accumulation satisfont aux conditions nécessaires de second ordre.*

Exercice 2.3.8 *[Preuve du corollaire 2.3.3]* Adaptez la preuve du théorème 2.3.4 ; indice : le cas 2 n'a pas besoin de modification, et le cas 1 doit tenir compte du fait que d n'est plus un minimum global, mais seulement une réduction au moins aussi bonne que le point de Cauchy et la direction de courbure la plus négative ; par le corollaire 2.3.2, les points d'accumulation sont stationnaires. □

Si un point d'accumulation satisfait aux conditions suffisantes de second ordre, alors la suite engendrée par l'algorithme converge vers ce point, et ce avec un ordre asymptotique quadratique, comme nous le démontrerons un peu plus loin.

Dans la suite de ce chapitre, nous approfondirons surtout l'étude de méthodes utilisant des directions suffisamment descendantes et des pas admissibles. Une des raisons est que la solution des sous-problèmes du calcul de d_k dans les méthodes de région de confiance nécessite des techniques d'optimisation avec contraintes d'inégalités, que nous étudierons au chapitre 4. Nous y reviendrons à ce moment.

2.3.4 Méthodes de descente pour les systèmes d'équations

Sans hypothèses additionnelles, il est impossible de résoudre un problème de trouver la racine de F, $F(x) = 0$ par un algorithme globalement convergent utilisant une méthode de descente. Cependant, sous certaines hypothèses (F est une application monotone), on peut appliquer un algorithme de minimisation au problème $\min \|F(x)\|^2$, qui sera alors globalement convergent. Cependant, dès que ∇F n'est pas inversible en un x qui n'est pas une racine de F, la fonction $\|F(x)\|^2$ peut posséder un minimum local en x si $F(x) \in \ker(\nabla F(x))$, et donc un algorithme de descente échoue à résoudre le problème.

Exercice 2.3.9 *[Fausses racines]* Vérifiez à l'aide d'un exemple de fonction f de $\mathbb{R}$ dans $\mathbb{R}$ qu'il est possible que f^2 possède un minimum local en x tel que $f(x) \neq 0$. □

2.4 Itération de Newton modifiée pour l'optimisation

Nous avons vu à la section 2.3.2 que la direction de Newton est une direction suffisamment descendante si $\nabla^2 f(x)$ est une matrice définie positive. Qu'en est-t-il sinon, ou encore si au fil des itérations, cette matrice se rapproche d'une matrice singulière ? Il en est que la méthode de Newton n'est peut-être plus globalement convergente. Par ailleurs, l'utilisation d'un pas θ est susceptible de détruire les propriétés de convergence locale quadratique de l'itération de Newton, dont la preuve utilisait toujours un plein pas $\theta \equiv 1$. Nous modifions dans cette section la méthode de Newton pour la rendre globalement convergente, et, en supposant que toute la suite $\{x_k\}$ converge vers un point stationnaire x^* pour lequel $\nabla^2 f(x^*)$ est une matrice définie positive, la modification conserve la propriété de convergence locale quadratique.

2.4.1 Modification de la direction de Newton

Pour éviter les problèmes associés à une matrice hessienne $\nabla^2 f(x_k)$ qui ne serait pas uniformément définie positive, nous allons la remplacer par une approximation $\bar{B}_k$, uniformément définie positive, et donc inversible, d'inverse $\bar{H}_k$, également définie positive. Conceptuellement, il suffit de prendre

$$\bar{B}_k = \nabla^2 f(x) + \max(-\lambda_{min} + \epsilon, 0)I$$

où λ_{min} est la plus petite valeur propre de $\nabla^2 f(x)$ et ϵ est une petite quantité positive. En effet, pour un tel choix, $\bar{B}_k$ sera égal au hessien de f si celui-ci est une matrice définie positive, et se verra ajouter un multiple de l'identité adéquat pour le rendre défini positif sinon.

Si x_k est proche d'un point stationnaire x^* pour lequel $\nabla^2 f(x^*)$ est une matrice définie positive, alors $\nabla^2 f(x_k)$ est également une matrice définie positive, et donc $\bar{B}_k = \nabla^2 f(x_k)$. Autrement, la valeur propre la plus petite de $\bar{B}_k$ vaut au moins ϵ, et donc est uniformément éloignée de zéro, garantissant que la direction satisfait aux conditions de la définition 2.3.1.

Il serait prématuré de présenter ici un algorithme de calcul pour $\bar{B}_k$. Nous reportons l'étude de l'implantation de la direction de Newton modifiée à la section qui traite du cas particulier de fonctions objectif quadratiques. En effet, une direction $\bar{d} = -\bar{H}_k \nabla f(x)^t$ pour une matrice $\bar{H}_k = \bar{B}_k^{-1}$ définie positive est solution du sous-problème quadratique suivant :

$$\min_d \frac{1}{2} d^t \bar{B}_k d + \nabla f(x) d.$$

Exercice 2.4.1 *[Propriété de descente de la méthode de Newton]* Trouvez la plus grande boule centrée sur l'origine pour laquelle le hessien de la fonction $f(x) = 2x_1^2 + x_2^2 - 2x_1x_2 + 2x_1^3 + x_1^4$ est une matrice définie positive. Si l'on démarre en un point $x^0 = (x, x)$, pour quelles valeurs dans cette boule l'algorithme de Newton (avec $\theta = 1$) converge-t-il ? □

2.4.2 Analyse du pas θ pour la direction de Newton

Il apparaît qu'un pas $\theta = 1$ est admissible pour la direction de Newton lorsque x_k devient assez proche de x^*, un point stationnaire pour lequel $\nabla^2 f$ est défini positif. En fait, ceci découle d'un résultat beaucoup plus général qui stipule que si une direction est de longueur "optimale" pour l'approximation de Taylor d'ordre deux de f, alors un pas unitaire est localement admissible.

Théorème 2.4.1 *Soit f une fonction de classe $\mathcal{C}^3$, d une direction suffisamment descendante, telle que le minimum de la fonction quadratique*

$$h_{x,d}(\theta) \stackrel{\text{def}}{=} f(x) + \theta \nabla f(x) d + \frac{\theta^2}{2} d^t \nabla f^2(x) d$$

soit atteint pour $\theta = 1$. Alors, le pas $\theta = 1$ est admissible pour la fonction f dans la direction d lorsque $\|d\|$ est suffisamment petite.

Preuve Puisque $\theta = 1$ minimise $h_{x,d}(\theta)$, alors $h'_{x,d}(1) = 0$. Donc,

$$\nabla f(x) d + d^t \nabla f^2(x) d = 0.$$

Écrivons maintenant la condition d'Armijo :

$$f(x+d) - f(x) - \tau_0 \nabla f(x) d < 0$$

en exprimant $f(x+d) = f(x) + \nabla f(x) d + \frac{1}{2} d^t \nabla^2 f(x) d + R_3(d)$:

$$\begin{aligned} f(x+d) - f(x) - \tau_0 \nabla f(x) d &= (1-\tau_0) \nabla f(x) d + \frac{1}{2} d^t \nabla^2 f(x) d + \mathcal{O}(\|d\|^3) \\ &= (\frac{1}{2} - \tau_0) \nabla f(x) d + \mathcal{O}(\|d\|^3), \end{aligned}$$

la dernière égalité provenant de la condition $h'_{x,d}(1) = 0$. Maintenant, selon la définition 2.3.1, $\nabla f(x) d < -\frac{\gamma_0}{\gamma_1^2} \|d\|^2$ et donc la condition d'Armijo est impliquée par l'inégalité suivante :

$$(\tau_0 - \frac{1}{2}) \frac{\gamma_0}{\gamma_1^2} \|d\|^2 + \mathcal{O}(\|d\|^3) < 0$$

qui est vérifiée lorsque $\|d\|$ est suffisamment petite.

Quant au critère de Wolfe, il suffit d'écrire que $\nabla f(x+d) = \nabla f(x) + d^t \nabla^2 f(x) + R_2(d)$ pour réexprimer le critère de Wolfe comme

$$(1-\tau_1) \nabla f(x) d + d^t \nabla^2 f(x) d + R_2(d) = -\tau_1 \nabla f(x) d + d R_2(d),$$

qui demeure positif lorsque d est assez petite car il existe une constante positive λ telle que $d R_2(d) \leq \lambda \|d\|^3$ lorsque $\|d\|$ est assez petit alors que $-\nabla f(x) d > \frac{\gamma_0}{\gamma_1^2} \|d\|^2$. $\square$

2.4.3 Convergence de la méthode de Newton modifiée

Nous avons maintenant tous les ingrédients pour démontrer la convergence globale ainsi que la convergence locale quadratique de la méthode de Newton modifiée pour l'optimisation. En fait, nous allons démontrer la convergence d'une version approchée de la méthode de Newton.

Rappelons que l'itération de Newton modifiée s'écrit :

$$\begin{aligned} x_{k+1} &\leftarrow x_k - \theta_k d_k; \\ d_k &\leftarrow -\bar{B}_k^{-1} \nabla f(x_k)^t. \end{aligned}$$

On voit donc que d_k satisfait au système

$$d_k^t \bar{B}_k + \nabla f(x_k) = r_k,$$

où, dans la méthode classique, $r_k \equiv 0$. Cette dernière formulation constitue en fait un sous-problème linéaire, dont la solution d_k n'est qu'approximative, à r_k près, et donc exacte pour la méthode de Newton usuelle, où $r_k \equiv 0$. L'analyse à venir n'utilisera pas cette propriété que $r_k \equiv 0$, et nous pourrons déduire les propriétés requises par la suite r_k pour qu'une méthode se comporte “bien”.

Théorème 2.4.2 *Soit une fonction f de classe $\mathcal{C}^3$. Les points d'accumulation de la méthode de Newton modifiée pour minimiser la fonction f sont tous des points stationnaires de la fonction f. De plus, si un des points d'accumulation satisfait à la condition suffisante de second ordre, alors toute la suite $\{x_k\}$ y converge, et la vitesse de convergence asymptotique est quadratique*

Preuve Puisque la méthode de Newton modifiée assure que les direction sont suffisamment descendantes, et que les pas utilisés sont admissibles, tous les points d'accumulation sont stationnaires.

Supposons maintenant (nous justifierons plus loin cette supposition) que toute la suite $\{x_k\}$ converge vers un point limite x^* qui satisfait aux conditions suffisantes de second ordre. Alors, $\nabla^2 f(x^*)$ est une matrice définie positive et donc, asymptotiquement, $\bar{B}_k \equiv \nabla^2 f(x_k)$. Par conséquent, les hypothèses du théorème 2.3.2 sont satisfaites, et le pas $\theta = 1$ est admissible asymptotiquement comme le montre le théorème 2.4.1. Enfin, complétons la preuve de convergence quadratique.

D'abord, exprimons $\nabla f(x^*)$ en utilisant une série de Taylor autour du point x_k en notant $\epsilon_k \stackrel{\text{def}}{=} (x_k - x^*)$:

$$\begin{aligned} 0 = \nabla f(x^*) &= \nabla f(x_k) - \epsilon_k^t \nabla^2 f(x_k) + \mathcal{O}(\|\epsilon_k\|^2) \\ &= (\nabla f(x_k) + d_k^t \nabla^2 f(x_k)) + (d_k - \epsilon_k)^t \nabla^2 f(x_k) + \mathcal{O}(\|\epsilon_k\|^2) \\ &= r_k - (\epsilon_{k+1})^t \nabla^2 f(x_k) + \mathcal{O}(\|\epsilon_k\|^2). \end{aligned}$$

De cette dernière équation, on tire que ϵ_{k+1} est solution du système linéaire

$$(\epsilon_{k+1})^t \nabla^2 f(x_k) = -r_k + \mathcal{O}(\|\epsilon_k\|^2).$$

Utilisons maintenant les faits que

– $\nabla^2 f(x^*)$ est une matrice inversible, et

– $\nabla^2 f$ est une fonction Lipschitz, c'est-à-dire que $\|\nabla^2 f(x_k) - \nabla^2 f(x^*)\| \leq L \cdot \|x_k - x^*\|$,

pour conclure que

$$\epsilon_{k+1} \sim \mathcal{O}(\|r_k\| + \|\epsilon_k\|^2) \sim \mathcal{O}(\max(\|r_k\|, \|\epsilon_k\|^2).$$

On voit donc que si $\|r_k\| \sim \mathcal{O}(\|\epsilon_k\|^2)$, la convergence quadratique locale de l'itération de Newton modifiée est assurée. En particulier, pour le cas $r_k \equiv 0$ que nous traitons maintenant, la convergence quadratique locale de l'itération de Newton modifiée est assurée.

Finalement, si un des points d'accumulation, disons $\hat{x}$ satisfait aux conditions suffisantes de second ordre, alors, il existe une sous-suite de $\{x_k\}$, notée $\{x_{k_i}\}$ telle que, pour des valeurs assez grande de k_i, les raisonnements locaux précédents s'appliquent au point x_{k_i}. Par conséquent, à partir d'un tel k_i, le reste de la suite converge vers $\hat{x}$. □

Comme nous l'avons vu dans la preuve, il n'est pas plus difficile de démontrer la convergence quadratique de l'itération de Newton si les sous-problèmes linéaires ne sont résolus qu'approximativement. Cependant, l'admissibilité du pas unitaire n'est plus assurée par le théorème 2.4.1. Si l'on choisit $\|r_k\| \sim \|\epsilon_k\|^2$, on conserve la propriété car la preuve du théorème 2.4.1 s'adapte facilement.

Exercice 2.4.2 *[Admissibilité du pas unitaire]* Justifiez les affirmations suivantes. Si $\nabla^2 f(x^*)$ est une matrice définie positive, et si x_k est proche de x^*, alors

a) $\|\epsilon_k\| \sim \|\nabla f(x_k)\|$; il s'agit de montrer que $\|\epsilon_k\| \sim \mathcal{O}(\|\nabla f(x_k)\|)$ et $\|\nabla f(x_k)\| \sim \mathcal{O}(\|\epsilon_k\|)$

b) si $\|r_k\| \sim \mathcal{O}(\|\epsilon_k\|^2)$, alors un pas unitaire demeure localement admissible. Modifiez légèrement la preuve du théorème 2.4.1 pour y incorporer r_k, puis constater que le même raisonnement demeure valable.

c) sous quelles conditions sur $\|r_k\|$ l'adaptation de la preuve du théorème 2.4.1 en b) demeure-t-elle valable ?

□

2.4.4 Convergence superlinéaire de la méthode de Newton avec régions de confiance

Théorème 2.4.3 *Si un point d'accumulation de l'algorithme 2.2 satisfait aux conditions suffisantes de second ordre, alors toute la suite engendrée y converge, et $r_k \to 1$, la suite Δ_k demeure éloignée de 0 et l'ordre de convergence est superlinéaire.*

Preuve Pour fin de contradiction, supposons qu'une sous-suite se produit avec le cas 2 du théorème 2.3.4. En un point x_k, considérons un déplacement dans la direction de Newton, bien définie au voisinage du point d'accumulation satisfaisant aux conditions suffisantes d'optimalité. Si la direction de Newton $d_N = (\nabla^2 f(x_k)^{-1})\nabla f(x_k)^t \leq \Delta_k$, alors $d_R = d_N$ et $\Delta q_{x_k} = \frac{1}{2} d_R^t \nabla^2 f(x_k) d_R \geq \frac{1}{2}\lambda_k \|d_R\|^2$, avec λ_k la plus petite valeur propre de $\nabla^2 f(x_k)$. Si au contraire la direction de Newton $d_N > \Delta_k$, alors l'optimalité de q nous permet d'établir que $\Delta q_{x_k} \geq \frac{1}{2}\lambda_k \|d_R\|^2$ également. Comme $\Delta f_{x_k} = \Delta q_{x_k} + \mathcal{O}(\|d_R\|^3)$, on obtient que $r_k \to 1$, contredisant que la sous-suite provient du cas 2.

Donc, Δ_k est forcément éloigné de 0, et comme les directions de Newton proche du point d'accumulation satisfaisant aux conditions de second ordre s'approchent de 0, l'algorithme 2.2 se ramène à une itération de Newton pure, d'où le résultat de convergence superlinéaire. □

2.5 Fonctions objectif quadratiques

Examinons maintenant deux algorithmes, une méthode dite directe, et une autre dite itérative pour solutionner un problème de minimiser une fonction quadratique $q(x) \stackrel{\text{def}}{=} \frac{1}{2}x^tQx + cx$ sans contrainte ; c est un vecteur ligne. Il est clair que la matrice Q, qui est symétrique, doit être au moins semi-définie positive, sans quoi la fonction q n'est pas bornée inférieurement. En fait, si la matrice Q n'est pas définie positive, il est également possible que q ne soit pas bornée inférieurement. Par exemple, considérons la fonction $q(x, y) = x^2$. Pour cette fonction, la matrice $Q = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ est semi-définie positive, et l'axe des y en entier est solution du problème. Cependant, la fonction $q(x, y) = x^2 + y$ n'est pas bornée inférieurement, bien qu'elle partage le même hessien que la fonction $q(x, y) = x^2$.

Supposons donc que la matrice Q qui définit la forme quadratique q est une matrice définie positive. Alors, la solution au problème $\min_x q(x)$ peut s'écrire $x^* = Q^{-1}c^t$. Nous présentons ci-après deux techniques pour calculer x^*. La première est une technique d'algèbre linéaire numérique, une méthode directe de solution d'un système linéaire symétrique défini positif alors que la seconde est un algorithme d'optimisation itératif spécialisé aux formes quadratiques.

2.5.1 Décomposition de Cholesky

Plaçons-nous dans le contexte de solutionner un système d'équations linéaires $Qx + c^t = 0$, où Q est une matrice définie positive. Il est connu que toute matrice définie positive peut s'exprimer comme $Q = LL^t$, où L est une matrice triangulaire inférieure (de l'anglais, *Lower triangular*). Cette expression de Q est connue sous le nom de décomposition de Cholesky de Q. Une fois Q ainsi exprimée, la solution du système linéaire $Qx + c^t = 0$ est facile, puisqu'elle se réduit à deux résolution de systèmes triangulaires, que l'on peut obtenir par les méthodes de substitution ascendantes ou descendantes :

$$\begin{aligned} Lv &= -c^t \\ L^tx &= v. \end{aligned}$$

On peut calculer la décomposition de Cholesky en analysant l'équation $Q = LL^t$:
- $L_{11} = \sqrt{Q_{11}}$;
- $L_{21} = \frac{Q_{21}}{L_{11}}$;
- $L_{22} = \sqrt{Q_{22} - L_{21}^2}$;
- $\vdots$
- $L_{ii} = \sqrt{Q_{ii} - \sum_{k=1}^{i-1} L_{ik}^2}$;
- $L_{ij} = \frac{Q_{ij} - \sum_{k=1}^{i-1} L_{ik}L_{jk}}{L_{ii}}$, $j < i$.

Si Q est une matrice définie positive, les racines carrées de l'algorithme sont toutes bien définies.

2.5.2 Algorithme du gradient conjugué

Toujours pour trouver la racine d'équations linéaires issues de matrices Q définies positives, construisons l'algorithme du gradient conjugué à partir d'observations élémentaires. En fait, il est plus commode pour cette interprétation de considérer que nous tentons de minimiser la fonction $q(x) = \frac{1}{2}x^tQx + cx$.

Pour un point x arbitraire, exprimons le minimum de q le long d'un axe de direction v originant en x :

$$\min_\theta q(x + \theta v) = \min_\theta \frac{1}{2}(x + \theta v)^t Q(x + \theta v) + c(x + \theta v).$$

En utilisant les conditions d'optimalité de la section 2.2, on retrouve que

$$((x + \theta v)^t Q + c)v = 0,$$

d'où l'on tire que les points minima $y = x + \theta v$ satisfont tous la relation $(y^tQ + c)v = 0$, ou encore $y^t(Qv) = -cv$. Par conséquent, en notant $a = (Qv)$ et $b = -cv$, on voit que les minima y satisfont à la relation $y^ta = b$, et donc appartiennent tous au même hyperplan, qui ne dépend pas de x! De plus, pour se déplacer d'un point à un autre sur cet hyperplan, il suffit d'utiliser un déplacement d satisfaisant $d^ta = 0$, ou, remplaçant a par sa valeur, $d^tQv = 0$. Deux vecteurs v et d qui satisfont à une telle relation sont nommés *conjugués* par rapport à la matrice Q. Ces observations formalisent l'intuition fournie par l'exemple du tableau 2.1 où nous observions que les directions des itérations paires étaient parallèles, et utilisaient les mêmes pas θ^*.

D'après les précédents développements, si l'on pouvait restreindre un algorithme à n'utiliser que des directions conjuguées après sa première étape, il se comporterait comme un algorithme en $n-1$ variables. Si on pouvait restreindre l'algorithme restreint à n'utiliser que des directions conjuguées aux deux premières directions de l'algorithme, il se comporterait comme un algorithme en $n - 2$ variables, et ainsi de suite. Les algorithmes de directions conjuguées constituent justement une réalisation de ce “rêve”.

Algorithme des directions conjuguées

Supposons que nous ayons à notre disposition n vecteurs linéairement indépendants v_1, $v_1, \ldots, v_n$. Il est possible les convertir en n vecteurs $u_1, \ldots, u_n$ mutuellement conjugués par rapport à une matrice Q symétrique et définie positive. En effet, il suffit d'appliquer le procédé d'orthogonalisation de Gramm-Schmidt aux vecteurs v. Une fois ces vecteurs u en main, il suffit d'appliquer l'algorithme

$$\begin{aligned} x_0 &\leftarrow \text{donné} \\ x_i &\leftarrow x_{i-1} + \left(\arg\min_\theta (q(x_{i-1} + \theta u_i)\right) u_i \end{aligned}$$

pour identifier x_n, la solution au problème $\min q(x)$. En effet, ce procédé séquentiel assure qu'à une étape donnée, la suite des itérations sont toutes dans l'intersection des hyperplans de directions conjuguées aux itérations précédentes.

Algorithme du gradient conjugué

Lorsque les vecteurs v utilisés pour produire les vecteurs mutuellement conjugués u proviennent de gradients de la fonction q, forme quadratique associée à la matrice Q, justement la matrice par rapport à laquelle les vecteurs u sont conjugués, alors le procédé de Gramm-Schmidt se réduit à la récurrence de deux termes suivante :

$$d_k = -\nabla f(x_k)^t + \beta_{k-1} d_{k-1},$$

où

$$\beta_k \stackrel{\text{def}}{=} \frac{\nabla f(x_{k+1}) Q d_k}{d_k^t Q d_k}.$$

Exercice 2.5.1 *[Rappels de Gramm-Schmidt]* Quelles sont les formules utilisées dans le procédé d'orthogonalisation de Gramm-Schmidt ? □

Exercice 2.5.2 *[Simplification de Gramm-Schmidt]* Vérifiez que le procédé de Gramm-Schmidt se simplifie bien aux formules du gradient conjugué lorsque les directions linéairement indépendantes utilisées pour produire les directions conjuguées sont les gradients des points x_k. □

L'algorithme du gradient conjugué est donc exprimé comme suit :

$$\begin{aligned}
d_k &= -\nabla f(x_k)^t + \beta_{k-1} d_{k-1} \\
\theta_k &= \frac{-\nabla f(x_k) d_k}{d_k^t Q d_k} \\
x_{k+1} &= x_k + \theta_k d_k \\
\beta_k &= \frac{\nabla f(x_{k+1}) Q d_k}{d_k^t Q d_k}.
\end{aligned}$$

On retrouve la formule utilisée dans l'exemple du tableau 2.1 pour le calcul du pas θ^*.

Exercice 2.5.3 *[Application de l'algorithme du gradient conjugué]* Appliquez l'algorithme du gradient conjugué à la fonction $f(x, y) = 2x^2 - 2xy + y^2 + 2x - 2y$ à partir du point $x_0 = (10, 5)^t$; comparez vos résultats avec l'exemple du tableau 2.1. □

Exercice 2.5.4 *[Formules équivalentes du gradient conjugué]* Démontrez que les formules suivantes constituent bel et bien une implantation de l'algorithme du gradient conjugué :

$$\begin{aligned}
d_k &= -\nabla f(x_k)^t + \beta_{k-1} d_{k-1} \\
\theta_k &= \frac{\|\nabla f(x_k)\|^2}{d_k^t Q d_k} \\
x_{k+1} &= x_k + \theta_k d_k \\
\beta_k &= \frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2}.
\end{aligned}$$

□

2.6 Implantation de la direction de Newton modifiée

En adaptant adéquatement les algorithmes de la section précédente, nous pourrons proposer des algorithmes pour calculer la direction de Newton modifiée utilisée à la section 2.4 pour construire un algorithme globalement convergent possédant la propriété de convergence locale quadratique. L'algorithme 2.3 résume les modifications, avec en plus une précaution qui limite le nombre total d'itérations.

```
Newton modifié(x, ε, f)
  { Donnés : x ; }
  {    un critère d'arrêt, ε > 0 ; }
  {    une fonction deux fois différentiable f ; }
  répéter
    d_N ← Direction_Newton_Modifiée(∇f(x), ∇²f(x))
    θ ← Pas_admissible(x, d_N, f)
    x ← x + θ d_N
  jusqu'à ( ‖∇f(x)‖ < ε)
  Résultat ← x
```

Algorithme 2.3: Newton modifié.

2.6.1 Adaptation de la décomposition de Cholesky

Nous avons vu plus haut comment calculer la décomposition de Cholesky d'une matrice symétrique définie positive. Si la matrice d'intérêt n'est pas définie positive, l'algorithme ne pourra pas terminer, car certaines racines carrées qu'il doit obtenir porteront sur des nombres négatifs.

En modifiant la procédure de calcul de la décomposition de Cholesky d'une matrice symétrique Q pour remplacer *toute racine carrée d'un nombre négatif* par un petit nombre positif (ϵ), nous obtenons

$$Q = LL^t + E,$$

où E représente l'"erreur" de la décomposition dûe à la modification de la procédure de calcul. Ainsi, si Q est véritablement définie positive, $E \equiv 0$ alors que si Q possède une ou des valeurs propres négatives, ou encore très petites, E assurera que LL^t est une matrice uniformément définie positive.

C'ette idée simple n'est toutefois pas suffisante pour obtenir un algorithme performant. En effet, comme le calcul des facteurs de Cholesky fait intervenir une division par les éléments diagonaux, s'ils valent ϵ, les facteurs de Cholesky deviendront énormes. Gill *et al* [15] ont proposé une stratégie beaucoup plus subtile et efficace. Il s'agit de déceler que Q n'est pas définie positive bien avant de rencontrer un pivot négatif. On peut récrire les formules de Cholesky comme suit, en dénotant A_j la sous-matrice qu'il reste à traiter à l'étape j, et donc $A = A_1$,

$$A_j = \begin{pmatrix} \alpha_j & a_j^t \\ a_j & \hat{A}_j \end{pmatrix}, \tag{2.3}$$

on peut calculer $L_{jj} = \sqrt{\alpha_j + \delta_j}$, $L_{ij} = (a_j)_i / L_{ii}$ pour $i = j+1, = \dots, n$, et

$$A_{j+1} = \hat{A}_j - \frac{a_j a_j^t}{\alpha_j + \delta_j}. \tag{2.4}$$

δ_j constitue la perturbation diagonale que l'on applique. La stratégie simple introduite ci-haut s'écrit $\delta_j = -\min(\alpha_j - \epsilon, 0)$. La stratégie préconisée dans Gill *et al* choisit δ_j comme étant le plus petit nombre non-négatif tel que

$$0 \leq \frac{\|a_j\|_\infty^2}{\alpha_j + \delta_j} \leq \beta^2, \tag{2.5}$$

avec un minimum de $\delta_j = -2\alpha_j$ si $\alpha_j < 0$, où β est une borne a priori choisie pour minimiser un pire cas sur la norme $\|E\|$ tout en assurant que $E = 0$ si A est suffisamment définie positive. Si on dénote ϵ_M comme étant la précision machine,

$$\beta^2 = \max\{\gamma, \xi/\sqrt{n^2 - 1}, \epsilon_M\}, \tag{2.6}$$

où $\gamma = \max_i |A_{ii}|$ et $\xi = \max_{j<i} |A_{ij}|$. La portion $\beta^2 \geq \gamma$ assure que $E = 0$ lorsque A est définie positive.

Bien entendu, ces choix de δ_j à chaque pas de l'algorithme de décomposition de Cholesky influencera le comportement global de la modification de l'itération de Newton. Cependant, le succès de logiciels comme MINOS confirme le potentiel de cette stratégie.

Exercice 2.6.1 *[Direction de Newton modifiée]* Considérez la fonction $f(x) = x_1^4 + x_1 x_2 + (1 + x_2)^2$.

a) Montrez qu'à partir de l'origine, ni d_N ni $-d_N$ ne constitue une direction de descente.

b) Calculez la direction de Newton modifiée à l'aide de la décomposition de Cholesky.

c) Vérifiez que la direction obtenue est bien une direction de descente.

□

2.6.2 Adaptation du gradient conjugué

Nous adaptons maintenant l'algorithme du gradient conjugué pour résoudre les sous-problèmes linéaires issus de la méthode de Newton, même si la matrice $\nabla^2 f(x)$ n'est pas définie positive. L'idée est la suivante : nous allons engendrer une suite $0 = \delta_0, \delta_1, \dots$ par l'algorithme itératif du gradient conjugué appliqué au problème

$$\min_d q(d) \stackrel{\text{def}}{=} \nabla f(x_k) d + \frac{1}{2} d^t \nabla^2 f(x_k) d = g_k d + \frac{1}{2} d^t H_k d.$$

Pour ces sous-problèmes, nous noterons p_i la suite des directions conjuguées, et ϑ_i les pas utilisés à l'intérieur de l'algorithme du gradient conjugué, de sorte que $\delta_{i+1} = \delta_i + \vartheta_i p_i$. δ_i dénote la suite qui convergerait vers d_N si $\nabla^2 f(x_k)$ était définie positive. Nous allons interrompre l'algorithme du gradient conjugué aussitôt qu'il devient capable de déceler que H_k n'est pas définie positive. Pour ce faire, remarquons que

$$\delta_l = \sum_{i=0}^{l} \vartheta_i p_i,$$

et, puisque les directions p_i sont mutuellement conjuguées par rapport à la matrice H_k,

$$\delta_l^t H_k \delta_l = \sum_{i=0}^{l} \vartheta_i (p_i^t H_k p_i).$$

Par conséquent, il est certain que $\delta_l^t H_k \delta_l > 0$ si tous les p_j qui entre dans la somme satisfont tous $p_i^t H_k p_i > 0$. Nous interromprons donc l'algorithme du gradient conjugué aussitôt qu'une des directions $p_{\bar{\jmath}}$ ne satisfait pas $p_{\bar{\jmath}}^t H_k p_{\bar{\jmath}} > \epsilon$. Si jamais $\bar{\jmath} = 0$, c'est que la direction $p_0 = -\nabla f(x_k)^t$ est une direction de courbure négative, mais c'est néanmoins une direction de descente forte, et nous l'utiliserons : $d = p_0$. Autrement, $\bar{\jmath} > 0$ et nous utiliserons

$$d = \sum_{i=0}^{\bar{\jmath}-1} \vartheta_i p_i,$$

et la définition de d assure que c'est une direction suffisamment descendante.

Finalement, on peut implanter une stratégie de résolution inexacte des problèmes linéaires en terminant l'algorithme du gradient conjugué aussitôt que $\|\nabla q\| < \|\nabla f\|^2$, ce qui ne nuit pas à la convergence locale quadratique de la méthode (voir section 2.4.3). L'algorithme 2.4 incorpore ces stratégies.

Exercice 2.6.2 *[Direction de Newton modifiée]* Considérez à nouveau la fonction $f(x) = x_1^4 + x_1 x_2 + (1 + x_2)^2$.

a) À partir de l'origine, calculez la direction de Newton modifiée à l'aide de l'algorithme du gradient conjugué.

b) Vérifiez que la direction obtenue est bien une direction de descente.

□

2.6.3 Adaptation du gradient conjugué : régions de confiance

Il est remarquable que la technique de la section précédente peut également être utilisée pour résoudre approximativement le sous-problème introduit dans la section 2.3.3. Rappelons qu'il s'agit de résoudre un problème quadratique $\min_{\|d\| \leq \Delta} q(d)$. Une légère modification de l'algorithme 2.4 nous fournit l'algorithme 2.5. Plutôt que d'interrompre les itérations lorsqu'une direction de courbure négative est rencontrée, il suffit de suivre cette direction jusqu'à la frontière de la région de confiance.

Direction_Newton_Modifiée(c,Q)

{ Version du gradient conjugué }
{ $q(\delta) \stackrel{\text{def}}{=} (1/2)\delta^t Q\delta + c\delta$ }
$\delta \leftarrow 0$
$p \leftarrow -\nabla q(\delta)^t$
si $((p^t Qp < \epsilon_2 \|p\|^2))$ **alors**
  Résultat $\leftarrow p$
sinon
  répéter
    $\vartheta \leftarrow \frac{-\nabla q(\delta)p}{p^t Qp}$
    $\delta \leftarrow \delta + \vartheta p$
    assez_précis $\leftarrow (\|\nabla q(\delta)\| < \min(\epsilon_1, \|\nabla q(0)\|^2))$
    si ($\neg$ assez_précis) **alors**
      $\beta \leftarrow \frac{\nabla q(\delta)Qp}{p^t Qp}$
      $p \leftarrow -\nabla q(\delta)^t + \beta p$
      courbure_négative $\leftarrow (p^t Qp < \epsilon_2 \|p\|^2)$
  jusqu'à (assez_précis $\vee$ courbure_négative)
  Résultat $\leftarrow \delta$

Algorithme 2.4: CG—direction de Newton modifée

Région_de_confiance(c,Q,Δ,$\|\cdot\|$)

{ Version du gradient conjugué }
{ $q(\delta) \stackrel{\text{def}}{=} (1/2)\delta^t Q\delta + c\delta$ }
$\delta \leftarrow 0$
$p \leftarrow -\nabla q(\delta)^t$
répéter
  $\alpha \leftarrow \max\{\alpha > 0 : \|\delta + \alpha p\| \le \Delta\}$
  sortie $\leftarrow p^t Qp \le 0$
  si ($\neg$ sortie) **alors**
    $\vartheta \leftarrow \frac{-\nabla q(\delta)p}{p^t Qp}$
    sortie $\leftarrow \vartheta > \alpha$
    si ($\neg$ sortie) **alors**
      $\delta \leftarrow \delta + \vartheta p$
      $\beta \leftarrow \frac{\nabla q(\delta)Qp}{p^t Qp}$
      $p \leftarrow -\nabla q(\delta)^t + \beta p$
jusqu'à (sortie $\vee \nabla q(\delta) = 0$)
si (sortie) **alors** $\delta \leftarrow \delta + \alpha p$
Résultat $\leftarrow \delta$

Algorithme 2.5: CG—Région de confiance

2.7 Implantation d'un pas admissible

Nous avons déjà vu que le procédé d'Armijo conduit à un algorithme globalement convergent. Cependant, il peut parfois être avantageux d'utiliser un algorithme plus efficace pour identifier un pas admissible, donc satisfaisant aux conditions de la définition 2.3.2 :

$$\begin{aligned} f(x+\theta d) - f(x) &< \tau_0 \theta \nabla f(x) d, \quad \tau_0 \in (0, \frac{1}{2}), \\ \nabla f(x+\theta d) d &> \tau_1 \nabla f(x) d, \quad \tau_1 \in (\tau_0, 1). \end{aligned}$$

Un tel pas est une approximation d'un minimum local de la fonction $h_{x_k,d}(\theta) = f(x_k + \theta d)$.

Comme nous l'avons vu au chapitre précédent, le type d'algorithme le plus avantageux combine la réduction d'intervalle à une technique d'interpolation polynômiale.

```
Pas_admissible(x,d,f)
  { Version de l'algorithme d'Armijo }
  { 0 < τ0 < 0.5. }
  θ ← 1
  tantque (f(x + θd) − f(x) > −θτ0∇f(x)d )
    θ ← θ/2
  Résultat ← θ
```

Algorithme 2.6: Armijo

2.7.1 Interpolation polynômiale avec garde-fous

Tout comme dans le cas multidimensionnel, où la méthode d'interpolation polynômiale (Newton) a été modifiée pour lui procurer la propriété de convergence globale, nous combinerons maintenant un algorithme de type réduction d'intervalle à une méthode d'interpolation polynômiale pour rendre la méthode modifiée globalement convergente. Il s'agit donc de la même démarche qu'aux sections 1.8 et 2.4, mais les outils de modification sont un peu différents.

2.8 Résumé

Dans ce chapitre, nous avons développé des outils d'analyse de problèmes d'optimisation différentiable sans contrainte qui conduisent à l'énoncé de conditions nécessaires, légèrement différentes des conditions suffisantes d'optimalité : un point x^* qui est un minimum local de f satisfait nécessairement $\nabla f(x^*) = 0$ et $\nabla^2 f(x^*)$ est une matrice semi-définie positive. Par ailleurs, pour avoir l'assurance qu'un point est bien un minimum local de f, il faut exiger de plus que la matrice $\nabla^2 f(x^*)$ soit définie positive. L'écart vient du fait que nous n'utilisons que les dérivées d'ordre un et deux de f, alors que ce sont possiblement ses dérivées d'ordre supérieur qui décrivent adéquatement son comportement près de x^*.

Ensuite, nous avons construit des schémas algorithmiques itératifs abstraits qui possèdent la propriété de convergence globale, c'est-à-dire que peu importe le point de départ des itérations, les points d'accumulation de la suite engendrée par l'algorithmes satisfont à une condition d'optimalité. L'existence d'algorithmes d'une telle famille, nommés algorithmes de descente, provient de la structure du problème d'optimisation ; la solution de systèmes d'équations quelconques ne peut pas, en général, être obtenue par des algorithmes de descente.

Nous avons ensuite modifié l'itération de Newton en assurant une propriété de descente suffisante de la direction, et en utilisant un pas admissible dans la direction. Ces modifications préservent la convergence locale quadratique de la méthode, mais la dotent en plus de la propriété de convergence globale.

L'implantation de la méthode de Newton modifiée est réalisée en adaptant judicieusement les algorithmes de minimisation d'une fonction quadratique, et nous avons illustré ce point par l'analyse de deux techniques : la décomposition de Choleski, et l'algorithme du gradient conjugué.

Finalement, nous avons étudié quelques algorithmes unidimensionnels, que nous avons combinés en un algorithme robuste tout en étant efficace, en applicant encore une fois le principe de modification d'un algorithme d'interpolation polynômiale, comme l'algorithme de Newton, pour le rendre globalement convergent.

2.9 Extensions et références

L'optimisation différentiable sans contrainte ainsi que les problèmes de racines d'équations non-linéaires ont été beaucoup étudiées. Il est évidemment impossible de rendre justice à la richesse de méthodes, d'idées publiées sur ce sujet. Le livre de Dennis et Schnabel [10] est consacré dans son entier à de tels problèmes.

Les méthodes de Newton modifiées "qui marchent" ont été proposées assez tard, vers 1972 [21, 14]. Gill, Murray et leurs collaborateurs sont à l'origine des méthodes utilisant la décomposition de Cholesky ; avant leurs travaux, les méthodes de Newton modifiées étaient considérées trop délicates à mettre en œuvre pour être compétitives. L'implantation à l'aide du gradient conjugué modifié ainsi que la notion de méthode de Newton inexacte sont dues à Dembo, Eisenstat et Steihaug [8] alors que l'implantation à l'aide du gradient conjugué se retrouve dans [9].

L'algorithme du gradient conjugué est dû à Hestenes et Stiefel [19]. La dérivation présentée à la section 2.5.2 est inspirée d'un article de Beale [1]. De nombreux travaux, dont ceux de Polak et Ribiere [23], ainsi que ceux de Fletcher et Reeves [13] ont porté sur la généralisation de l'algorithme du gradient conjugué à des fonctions non-quadratiques. Cependant, il semble que l'implantation de l'itération de Newton tronquée utilisant des différences finies pour évaluer les gradients soit pour le moins compétitive avec les diverses versions du gradient conjugué non-linéaire.

Exercice 2.9.1 *[Projet de programmation]* Implantez quelques variantes de l'algorithme de Newton modifié : comparez les résultats d'une variante avec une recherche de pas admissible simple, l'algorithme d'Armijo qui prend la première puissance de $\frac{1}{2}$ satisfaisant au critère d'Armijo, une recherche linéaire plus sophistiquée combinant les idées des algorithmes 1.1 et 1.4, et enfin une version avec régions de confiance. Vous pouvez utiliser la modification issue de la décomposition de Cholesky, ou encore l'algorithme du gradient conjugué (comparez le comportement de la version région de confiance avec celle de la recherche linéaire dans ce cas). Votre programme utilisera trois fonctions qui retournent respectivement $f(x)$, $\nabla f(x)$ et $\nabla^2 f(x)$. Testez votre programme (entre autres) avec la fonction de Rosenbrook : $f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ à partir du point de départ $(-1.2, 1.0)^t$. Cette fonction a longtemps constitué un défi pour les programmes de minimisation. Les modifications de l'algorithme de Newton traitent habituellement cette fonction facilement. $\square$

Chapitre 3

Programmation linéaire

Sujets du chapitre.

- Observations sur la géométrie du problème.
- Condition d'optimalité.
- Déduction de l'algorithme du simplexe.
- Théorie de la dualité linéaire.
- Dégénérescence.
- Aspects numériques.

Introduction

Dans ce chapitre, nous abordons l'étude *directe* des problèmes de minimisation de fonctions linéaires dans un domaine défini par des contraintes linéaires. Par une suite d'observations de nature géométriques ou encore algébriques, nous déduirons les conditions d'optimalité de la programmation linéaire, l'algorithme du simplexe révisé, les notions de dualité, et les variantes duales et primales-duales de l'algorithme du simplexe.

3.1 Formulation du problème

Pour simplifier l'exposé, nous considérons que le problème est formulé sous la forme dite *standard*, c'est-à-dire

$$\begin{aligned} &\min cx \\ &\text{s.à. } Ax = b \\ &x \geq 0, \end{aligned} \tag{3.1}$$

où c et x sont des vecteurs de $I\!R^n$, b un vecteur de $I\!R^m$ et A une matrice $m \times n$. Les lignes de la matrice A sont notées $a_i, i = 1, 2, \ldots, m$ alors que ses colonnes sont notées $A_j, j = 1, 2, \ldots, n$. De plus, nous faisons l'hypothèse que la matrice A est de rang m, c'est-à-dire que ses lignes sont linéairement indépendantes. Ainsi, les contraintes définissent l'intersection d'un sous-espace de $I\!R^n$ de dimension $n - m$ avec l'orthant positif. Le vecteur c constitue le gradient de la fonction linéaire cx, et donc est un vecteur-ligne.

Nous verrons plus tard que l'hypothèse que les lignes de A sont linéairement indépendantes n'est pas très restrictive.

3.2 Solutions de base réalisables

Attardons-nous un instant sur la structure du domaine réalisable, cette intersection d'un sous-espace avec l'orthant positif. Plus particulièrement, essayons de caractériser les points d'intersection avec la frontière de l'orthant. Dans $I\!R^3$, par exemple, s'il n'y a qu'une seule contrainte (en plus des bornes qui définissent l'orthant), le domaine réalisable est l'intersection d'un plan avec l'orthant. Selon la position de ce plan, l'intersection peut être vide, ou encore constituée seulement de l'origine, ou encore d'un demi-axe, ou d'un cône, ou finalement d'un triangle. Dans tous les cas où le domaine réalisable n'est pas vide, au moins un point de la partie positive des axes canoniques en fait partie. C'est une propriété importante. Nous verrons qu'un tel point appartenant à un axe canonique est un *point extrême* du domaine réalisable, correspondant à ce que nous nommerons une *solution de base*. Complétons notre observation de $I\!R^3$ en examinant l'intersection d'un sous-espace de dimension deux (une droite) avec l'orthant. Dans ce cas, si l'intersection n'est pas vide, elle est constituée de l'origine, ou encore d'une demi-droite, ou encore d'un segment de droite. Dans ce cas, les points extrêmes appartiennent à un plan canonique (extrémités de la demi-droite ou du segment).

Ces illustrations géométriques sont à la base de la caractérisation des solutions de base en programmation linéaire. En effet, en supposant que le domaine n'est pas vide, un sous-espace de $I\!R^n$ de dimension $n-m$ intersecte au moins un hyperplan canonique de dimension m. Dans l'exemple de $I\!R^3$, le plan intersecte au moins un axe canonique (hyperplan de dimension un) alors que la droite intersecte au moins un des plans canoniques (xy, yz ou zx). Supposons donc que les composantes d'un vecteur de $I\!R^n$ sont ordonnées de telle sorte que l'hyperplan canonique qui contient un tel point d'intersection est formé des m premières composantes. Ce point est donc de la forme $(x_1, x_2, \ldots, x_m, 0, 0, \ldots, 0)^t$. Ordonnons conséquemment les colonnes de la matrice A ainsi que les composantes du vecteur c, et écrivons :

$$\begin{aligned}
\min cx &= [c_B, c_N]\begin{pmatrix} x_B \\ x_N \end{pmatrix} \\
\text{s.à. } Ax &= [B, N]\begin{pmatrix} x_B \\ x_N \end{pmatrix} = Bx_B + Nx_N = b \\
x_B &\geq 0 \\
x_N &= 0.
\end{aligned}$$

Maintenant, supposons que la matrice B est inversible et donc $x_B = B^{-1}b$. Les colonnes de la matrice B forment une base du sous-espace canonique de dimension m alors que le vecteur x_B constitue les coordonnées du vecteur b dans cette base. La partition $[\![B \mid N]\!]$, ou encore le vecteur $x = [x_B, 0]$ est nommé *solution de base réalisable.*

Cette notation est universelle, malgré l'ambigüité de la signification des symboles B et N, qui réfèrent à la partition des composantes de $I\!R^n$ mais qui réfèrent aussi aux sous-matrices correspondantes de la matrice A.

Avant de terminer cette introduction, revenons sur l'hypothèse que la matrice B soit inversible. Puisque nous supposons que $Bx_B = b$, même si la matrice B est singulière, le système d'équations $Bx = b$ possède tout de même une solution. En fait, si la matrice B est singulière, c'est que le système $Bx = b$ possède plusieurs solutions, dont certaines comportent des composantes nulles. C'est une manifestation du phénomène de dégénérescence, que nous traiterons plus tard.

Exercice 3.2.1 *[Systèmes d'équations linéaires]*

a) Montrez que si B est une matrice $m \times m$ singulière, et si le système $Bx = b$ possède une solution, alors l'ensemble des solutions constitue un ensemble affine.

b) Dans les conditions de a), montrez que si le système possède une solution telle que $x \geq 0$, alors il possède une solution avec une ou des composantes nulles.

□

3.3 Condition d'optimalité

Nous examinons dans cette section les conditions sous lesquelles une solution $[x_B, 0]$ est optimale pour le programme linéaire (3.1). Il semble clair que pour qu'une telle solution soit optimale, il suffit de s'assurer qu'il est impossible de réduire la fonction objectif $c_B x_B + c_N 0$ en augmentant les composantes de x_N. Profitons du fait que B est inversible pour écrire

$x_B = B^{-1}(b - Nx_N)$. Cette dernière relation permet de mettre x_B en fonction de x_N. De même, la fonction objectif peut se récrire comme

$$\begin{aligned} c_B x_B + c_N x_N &= c_B B^{-1}(b - Nx_N) + c_N x_N \\ &= (c_N - c_B B^{-1} N)x_N + c_B B^{-1} b. \end{aligned}$$

En fait, il est maintenant possible de reformuler le programme (3.1) en utilisant seulement les composantes x_N, donnant le programme réduit :

$$\begin{aligned} \min\ & (c_N - c_B B^{-1} N)x_N \\ \text{s.à. } & B^{-1}(b - Nx_N) \geq 0 \\ & x_N \geq 0. \end{aligned} \tag{3.2}$$

On voit donc que si $(c_N - c_B B^{-1} N) \geq 0$, alors $x_N = 0$ est solution du programme (3.2), et donc x_B, x_N est solution du programme (3.1). Si de plus $x_B > 0$ et $x_N = 0$ est solution du programme (3.2), alors $(c_N - c_B B^{-1} N) \geq 0$. En effet, autrement, il serait possible d'augmenter légèrement une composante de x_N associée à une composante négative du vecteur $(c_N - c_B B^{-1} N)$ tout en assurant que x_B demeure non-négatif, ce qui ferait décroître la fonction objectif, contredisant l'optimalité de la solution.

Cependant, si une composante de x_B est nulle, il est concevable qu'aucune augmentation d'une certaine composante de x_N ne maintienne x_B non-négatif. Ceci est une manifestation du phénomène de dégénérescence, que nous étudierons en détail plus tard. Mentionnons seulement que dans un tel cas, la décomposition $[\![B \mid N]\!]$ des composantes positives n'est pas unique (puisqu'une valeur de x_B est nulle), et il est possible que le vecteur $(c_N - c_B B^{-1} N)$ soit non-négatif pour une autre décomposition.

En terminant cette section, présentons la notation usuelle en programmation linéaire. Le vecteur ligne $\pi \stackrel{\text{def}}{=} c_B B^{-1}$ est nommé *multiplicateurs* alors que le vecteur $c_N - \pi N$ est nommé vecteur des *coûts réduits*. Souvent, on dénote $\pi N = z$, de telle sorte que le $j^{\text{ième}}$ coût réduit s'exprime comme $c_j - z_j$.

3.4 Algorithme du simplexe

Encore une fois, nous allons déduire l'algorithme du simplexe en supposant que la condition d'optimalité énoncée plus haut n'est pas satisfaite en un certain point qui s'exprime comme $[x_B, 0] \geq 0$.

Examinons donc un cas où la décomposition $[x_B, 0]$ donne lieu à des coûts réduits négatifs. En d'autres termes, examinons la situation lorsque les conditions d'optimalité discutées plus haut ne sont pas satisfaites. Nous allons proposer une autre décomposition $[x_{B'}, 0]$ pour laquelle la valeur de la fonction objectif $c_{B'} x_{B'} < c_B x_B$. Tout comme dans la discussion au sujet des conditions d'optimalité, supposons que $x_B > 0$. Nous étudierons en détail le cas dégénéré plus loin.

Comme nous l'avions discuté plus haut, il est alors possible d'augmenter une composante de x_N tout en diminuant la fonction objectif. Supposons que nous ayons une composante $\bar{\jmath}$ pour laquelle $c_{\bar{\jmath}} - z_{\bar{\jmath}} < 0$. Alors, nous allons augmenter la composante $x_{\bar{\jmath}}$. S'il est possible d'augmenter $x_{\bar{\jmath}}$ sans qu'aucune composante de x_B ne diminue, alors il est possible de produire une solution réalisable de coût aussi petit que l'on veut, et le problème est non-borné inférieurement.

Autrement, il nous faut déterminer l'augmentation maximale que nous pouvons réaliser à la composante $x_{\bar{\jmath}}$ tout en assurant $x_B \geq 0$. Examinons l'expression que nous connaîssons, $x_B = B^{-1}(b - Nx_N)$. Puisque nous augmentons seulement la composante $x_{\bar{\jmath}}$, écrivons $x_B = B^{-1}(b - N_{\bar{\jmath}}x_{\bar{\jmath}})$. C'est donc le vecteur $B^{-1}N_{\bar{\jmath}}$ qui est responsable de l'augmentation ou de la diminution des composantes de x_B en fonction de $x_{\bar{\jmath}}$. Si $B^{-1}N_{\bar{\jmath}} \leq 0$, alors aucune composante de x_B ne diminue lorsque $x_{\bar{\jmath}}$ augmente, et le problème n'est pas borné. Autrement, une augmentation unitaire de $x_{\bar{\jmath}}$ provoque un changement de $-\delta x_B = -B^{-1}N_{\bar{\jmath}}$, ou encore, pour une composante i donnée, $\delta x_i = (B^{-1}N_{\bar{\jmath}})_i$. Puisque pour x_N=0 on a $x_B = B^{-1}b$, on a que la valeur de $x_{\bar{\jmath}}$ qui annule x_i est précisément $\frac{x_i}{\delta x_i}$. On garanti donc que x_B demeure non-négatif en posant

$$x_{\bar{\jmath}} = \min_{i:\delta x_i > 0}(\frac{x_i}{\delta x_i}).$$

Après ce changement, la partition $[\![B \mid N]\!]$ est modifiée en ajoutant la variable $\bar{\jmath}$ à B et en enlevant de B (pour l'ajouter dans N) un des x_i qui devient zéro, c'est-à-dire un qui atteint le $\min_{i:\delta x_i>0}(\frac{x_i}{\delta x_i})$. Alors, on passe de la partition $[\![B \mid N]\!]$ à la nouvelle partition $[\![B' \mid N']\!]$ définie comme suit : $B' = B \cup \{\bar{\jmath}\} \backslash \{\bar{\imath}\}$ et $N' = N \cup \{\bar{\imath}\} \backslash \{\bar{\jmath}\}$, où $\bar{\imath}$ est un des indices qui atteignent le $\min_{i:\delta x_i>0}(\frac{x_i}{\delta x_i})$. Nous supposons pour l'instant que ce min n'est atteint que pour un seul indice $\bar{\imath}$ et de plus que $x_{\bar{\imath}} > 0$. Lorsque ce n'est pas le cas, c'est que le problème comporte de la dégénérescence, et ce phénomène sera étudié plus tard.

Nous examinerons plus tard diverses techniques permettant de *mettre-à-jour* l'information que représente la matrice B^{-1} plutôt que de recalculer celle-ci à chaque modification de la partition $[\![B \mid N]\!]$. Pour l'instant, contentons-nous d'observer que si on sous la main une représentation de B^{-1} alors $B^{-1}(B, N_{\bar{\jmath}}) = (I, \delta x)$, donc, dans l'ordre des variables de la nouvelle partition, cette dernière matrice devient

$$B^{-1}(B_1, \dots, B_{i-1}, N_{\bar{\jmath}}, B_{i+1}, \dots, B_m, B_i) = (e_1, \dots, e_{i-1}, \delta x, e_{i+1}, \dots, e_m, e_i)$$

et en utilisant des opérations élémentaires sur les matrices, il est possible de transformer cette dernière en $(I, N'_{\bar{\imath}})$.

Sans préciser davantage comment s'effectuent les calculs, nous pouvons énoncer l'algorithme dit du simplexe révisé. Si le choix de la variable sortante est unique de même que $x_B > 0$, il est clair que cet algorithme termine en un nombre fini de calculs car le nombre de décomposition $[\![B \mid N]\!]$ est grand, mais fini et à chaque nouvelle décomposition examinée, la fonction objectif décroît, empêchant d'en examiner aucune plus d'une fois. Cependant, nous verrons plus tard qu'il faut préciser davantage l'algorithme pour garantir sa terminaison pour des problèmes dégénérés.

Remarquons ici que l'algorithme du simplexe original (non révisé) consiste à manipuler un *tableau* donné par la matrice $\bar{A} = B^{-1}[b, B, N] = [x_B, I, B^{-1}N]$ en calculant explicitement toute la matrice transformée. La version révisée de l'algorithme présentée ici consiste à calculer au besoin seulement la colonne $B^{-1}N_{\bar{\jmath}}$ à partir d'une représentation de la matrice B^{-1}.

```
Simplexe révisé
  { Données : une partition ⟦B | N⟧ de ℝ^n, une représentation }
  { de la matrice B^{-1} ainsi qu'une solution réalisable [x_B, 0]. }
  optimal ← faux
  non_borné ← faux
  répéter
    { Production d'un vecteur de coûts réduits }
        π ← SOLUTION(πB = c_B)
        z_N ← πN
        d ← c_N − z_N
    { Choix d'une variable entrante }
    si (d > 0 ) alors optimal ← vrai
    sinon
      j̄ ← Calcule_j̄
      δx ← SOLUTION(Bδx = N_j̄)
      { Choix d'une variable sortante }
      si (δx < 0 ) alors non_borné ← vrai
      sinon
        ī ← arg min_{i:δx_i>0}(x_i / δx_i)
        { Mises-à-jour diverses }
            ⟦B | N⟧ ← ⟦B ∪ {j̄}\{ī} | N ∪ {ī}\{j̄}⟧
            B^{-1} ← Mise_à_jour(B^{-1}, ⟦B | N⟧, ī, j̄)
  jusqu'à ( optimal ∨ non_borné)
```

En pratique, les deux résolutions de systèmes linéaires sont efficaces car la matrice B est représentées par une décomposition appropriée à la solution. Cette décomposition est donc mise à jour en même temps que la partition des composantes ; dans l'algorithme, nous avons nommé cette représentation B^{-1}, mais nous verrons qu'il s'agit plutôt d'une décomposition de la matrice $B = LU$

Algorithme 3.1: Simplexe révisé.

Exercice 3.4.1 *[Programmes non bornés]* Est-il possible que le programme (3.1) ainsi que le programme $\max cx$ sous les mêmes contraintes soient tous deux non-bornés ? Justifiez. □

3.5 Dualité

Examinons maintenant les relations que nous avons obtenues jusqu'à date pour en déduire une propriété importante à laquelle satisfont les multiplicateurs π lorsque la base est optimale. Soit donc $[x_B, 0]$ une solution réalisable optimale associée à la décomposition $[\![B \mid N]\!]$. Nous avons la définition $\pi B = c_B$. Par ailleurs, l'optimalité de la base nous assure que $c_N - z_N \geq 0$, ou encore $\pi N \leq c_N$. Donc, les multiplicateurs associés à une base optimale satisfont à la relation $\binom{B^t}{N^t}\pi = \pi A \leq c$. Ceci justifie la proposition suivante.

Proposition 3.5.1 *π est une solution réalisable au programme (nommé dual)*

$$\begin{aligned} &\max \pi b \\ &s.à.\ \pi A \leq c \end{aligned} \tag{3.3}$$

Cette proposition constitue la première moitié du théorème de dualité. En fait, puisque nous supposons qu'il existe une solution optimale $[x_B, 0]$, nous avons également

Proposition 3.5.2 *π est une solution optimale au programme (3.3)*

Preuve Nous allons démontrer cette proposition en ramenant ce dernier programme à la forme standard. Cet exercice nous permet d'utiliser des manipulations classiques et d'en introduire la notation habituelle. Pour transformer en égalités les contraintes d'inégalité, nous introduisons des *variables d'écart* $\alpha \geq 0$ et récrivons $\pi A \leq c$ sous la forme $\pi A + \alpha = c$. Pour que toutes les variables qui définissent le programme soient non-négatives, nous remplaçons les variables *libres* π par une différence de nouvelles variables non-négatives : $\pi = \beta^+ - \beta^-$ avec $\beta^+ \geq 0$ et $\beta^- \geq 0$. Ainsi, le programme (3.3) se récrit sous forme standard

$$\begin{aligned} \max\ & (\beta^+ - \beta^-)b \\ \text{s.à.}\ & (\beta^+ - \beta^-)A + \alpha = c \\ & \beta^+ \geq 0 \\ & \beta^- \geq 0 \\ & \alpha \geq 0. \end{aligned} \tag{3.4}$$

Si l'on décompose α selon la partition $[\![B \mid N]\!]$, on retrouve $\alpha_B = 0$ et $\alpha_N \geq 0$. En conséquence, une base du programme (3.4) est donnée par $\begin{pmatrix} B_1^t & -B_2^t & 0 \\ N_1^t & -N_2^t & I \end{pmatrix}$. En écrivant la définition des multiplicateurs pour le problème (3.4), on constate qu'il s'agit précisément du vecteur $[x_B, 0]$ et en écrivant les conditions d'optimalité pour le programme (3.4), on constate qu'il ne s'agit de rien d'autre que la condition $x_B \geq 0$, ce qui confirme que π est solution optimale pour le problème (3.4). □

Corollaire 3.5.1 *[Écarts complémentaires] Ajoutons des variables d'écart s de telle sorte que les contraintes du programme (3.3) deviennent $yA + s = c$ avec $s \geq 0$; alors, si x et $[y, s]$ sont des solutions optimales pour les programmes (3.1) et (3.3) respectivement, $sx = 0$.*

Théorème 3.5.3 *Considérons les problèmes (3.1) et (3.3).*

1. *si les deux programmes possèdent une solution réalisable x et π, alors $cx \geq \pi b$; de plus, si $cx = \pi b$, alors x et π sont optimaux pour les programmes (3.1) et (3.3) respectivement.*
2. *si un des problèmes n'est pas borné, alors l'autre n'est pas réalisable.*
3. *Si un des programmes n'est pas réalisable, alors l'autre est soit non-borné, soit non-réalisable.*

Preuve La partie a) découle des propositions précédentes. De même, pour la partie c), si un programme n'est pas réalisable, l'autre ne peut pas posséder de solution optimale car sinon, de la contredirait la partie a). On laisse en exercice le soin de trouver un exemple où ni le primal, ni le dual ne possèdent de solution réalisable.

Pour justifier la parties b), examinons le problème (3.3). Ses contraintes sont $\pi A \leq c$, et donc $\pi A x \leq cx$ pour tout vecteur π réalisable pour le problème (3.3). Maintenant, en supposant que x est réalisable pour le problème (3.1), on retrouve $Ax = b$ de sorte que l'on obtient $\pi b \leq cx$, c'est-à-dire que toute solution réalisable pour le dual possède une valeur de fonction objectif à la valeur de l'objectif primal pour une solution réalisable quelconque. Si le primal n'est pas borné, c'est qu'aucun π ne satisfait $\pi b > -\infty$. □

Les problèmes (3.1) et (3.3) sont liés entre eux par une relation de dualité. D'autres formulations de programmes linéaires conduisent à d'autres paires de problèmes duaux.

Exercice 3.5.1 *[Programmes irréalisables]* Fournissez un exemple de programme pour lequel ni le primal, ni le dual ne possède de solution réalisable. □

Exercice 3.5.2 *[Dualité]* Considérez le programme linéaire suivant

$$\begin{aligned} &\min cx \\ &\text{s.à. } Ax \leq b \\ &\qquad x \geq 0, \end{aligned}$$

et vérifiez que son dual est

$$\begin{aligned} &\max yb \\ &\text{s.à. } yA \geq c \\ &\qquad y \geq 0 \end{aligned}$$

en le ramenant à la forme standard, puis en simplifiant le dual ainsi obtenu. □

Exercice 3.5.3 *[Dualité]* Considérez le programme linéaire le plus général envisageable

$$\begin{aligned} &\min c_1x_1 + c_2x_2 \\ &\text{s.à. } A_{11}x_1 + A_{12}x_2 \leq b_1 \\ &\text{s.à. } A_{21}x_1 + A_{22}x_2 = b_2 \\ &\qquad x_1 \geq 0, \end{aligned}$$

où A est une matrice $(m_1 + m_2) \times (n_1 + n_2)$ et $c, x \in I\!R^{n_1+n_2}$ et $b \in I\!R^{m_1+m_2}$. Caractérisez le dual de ce problème. □

3.6 Dégénérescence

Jusqu'à maintenant, nous avons escamoté les difficultés causées par la dégénérescence en supposant systématiquement que $x_B > 0$, et que le choix de la variable sortante était unique dans l'algorithme 1.1. Penchons-nous maintenant sur le problème de la dégénérescence.

Définition 3.6.1 *[Solution de base dégénérée] Une solution de base x_B est dite dégénérée lorsque une ou plusieurs composantes de x_B sont nulles.*

Ce phénomène peut se manifester pour le programme (3.1) tout comme pour son dual (3.3), où il s'énonce comme suit : le vecteur de coût réduit $c_N - \pi N$ associé à une solution réalisable de base $[x_B, 0]$ possède une composante nulle.

Dans le cas de dégénérescence du programme primal (3.1), il est probable que l'on puisse trouver une autre décomposition $[\![B \mid N]\!]$ décrivant la même solution réalisable x_B puisque sa composante nulle n'appartient pas naturellement aux composantes B. Dans le second cas, il est possible de modifier la partition $[\![B \mid N]\!]$ en changeant la solution réalisable, mais sans modifier la valeur de la fonction objectif. Dans chacun de ces cas, le fait de changer de partition $[\![B \mid N]\!]$ n'entraîne pas de diminution de la fonction objectif, et il faut raffiner l'algorithme pour pouvoir démontrer qu'il ne produit pas de cycle de partitions. Nous illustrerons un tel cycle à la section prochaine, et proposerons une analyse d'une règle permettant d'éviter ce phénomène de cyclage.

Exercice 3.6.1 *[Dégénérescence et unicité]*

a) Démontrez que si aucune solution optimale du programme (3.1) n'est dégénérée, alors le programme (3.3) possède une solution unique.

b) Est-il vrai que le programme (3.3) possède toujours plus d'une solution lorsque le primal (3.1) possède une solution dégénérée ?

□

Exercice 3.6.2 *[Dégénérescence]* Considérez les contraintes linéaires

$$\sum_{i=1}^{n} x_{ij} = 1, j = 1, 2, \dots n$$
$$\sum_{j=1}^{n} x_{ij} = 1, i = 1, 2, \dots n$$
$$x_{ij} \geq 0, i = 1, 2, \dots n, j = 1, 2, \dots n.$$

Trouvez un vecteur c pour lequel ce problème possède des bases dégénérée simultanément pour les programmes (3.1) et (3.3). □

3.6.1 Un exemple de cyclage dans l'algorithme du simplexe

Pour présenter cet exemple, nous utiliserons la forme usuelle de *tableau* du simplexe. Il s'agit de donner sous forme condensée toutes les quantités utiles dans l'algorithme 1.1. Le tableau comporte donc les quantités relatives à la partition $[\![B \mid N]\!]$, la matrice A multipliée par l'inverse de la matrice B, le vecteur $B^{-1}b$ ainsi que les quantités $c_N - z_N$. La structure du tableau est la suivante :

	x
$-c_Bx_B$	$c - z$
x_B	$B^{-1}A$

Évidemment, $c_B - z_B = 0$ et, à moins que la séparation $[\![B \mid N]\!]$ sous la main soit optimale, au moins un $c_j - z_j$ est négatif. Dans l'exemple, nous choisissons systématiquement la variable entrante correspondant à l'unique indice du $c_j - z_j$ le plus négatif. Pour ce qui est de la variable sortante, nous choisissons systématiquement la première (dans l'ordre x_1,x_2,...) qui atteint le $\min_{i:\delta x_i>0}(\frac{x_i}{\delta x_i})$. Pour l'exemple, on a quelquefois le choix entre deux variables de bases qui sont nulles. On choisit alors celle d'indice minimum.

Il devrait être clair que les règles utilisées pour déterminer les variables entrantes et sortantes influencent le comportement de l'algorithme du simplexe, et que l'exemple de cyclage suivant ne vaut que pour notre règle d'entrer la variable de coût réduit minimum et de sortir la première variable (si plusieurs peuvent sortir).

Nous avons déjà mentionné qu'il est possible de mettre-à-jour la représentation de la matrice B^{-1} en utilisant des opérations élémentaires sur les matrices. Cette opération de mise-à-jour se nomme *opération de pivot* du simplexe. Dans les tableaux qui suivent, le pivot est encadré. Le passage d'un tableau au suivant peut s'effectuer sans connaître la matrice A originale, simplement en *zérotant* la colonne du pivot (sauf le pivot lui-même) et en divisant la ligne du pivot par le pivot. Dans ce processus, la colonne du pivot devient une un vecteur unitaire alors qu'un des vecteurs unitaires de la matrice identité $B^{-1}B$ est modifié.

Exercice 3.6.3 *[]* Quelle est la valeur optimale du programme linéaire sur l'exemple de cyclage ? □

Ce dernier tableau, ou encore cette dernière partition $[\![B \mid N]\!]$ est exactement la même que la première, et l'algorithme ne terminera donc jamais, étant prisonnier de ce cycle.

Cet exemple produit un cycle parce que deux variables de base sont nulles, et selon les règles d'entrée et de sorties, aucun pivot ne change de valeur de x, mais les pivots changent uniquement la décomposition $[\![B \mid N]\!]$.

3.6.2 Règles d'anti-cyclage

Pour remédier au phénomène de cyclage de l'algorithme du simplexe, plusieurs auteurs ont présenté des règles pour choisir les variables entrantes et sortantes.

Nous présentons deux telles règles. La première conserve la même stratégie pour choisir la variable sortante, mais impose une stratégie nouvelle pour choisir la variable entrante (règle de Bland) ; son analyse est remarquablement simple, et elle s'applique très bien à l'algorithme du simplexe révisé tel que présenté. La seconde permet n'importe quelle stratégie pour choisir la variable entrante, mais impose un choix précis pour choisir la variable sortante (règle lexicographique) ; d'analyse plus complexe, cette règle semble plus utile en pratique, malgré qu'elle s'applique plus naturellement au simplexe dit non-révisé.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
3	$-\frac{3}{4}$	20	$-\frac{1}{2}$	6	0	0	0	
0	$\boxed{\frac{1}{4}}$	-8	-1	9	1	0	0	$B = \{5, 6, 7\}$
0	$\frac{1}{2}$	-12	$-\frac{1}{2}$	3	0	1	0	$N = \{1, 2, 3, 4\}$
1	0	0	1	0	0	0	1	
				$\Downarrow$				
3	0	-4	$-\frac{7}{2}$	33	3	0	0	
0	1	-32	-4	36	4	0	0	$B = \{1, 6, 7\}$
0	0	$\boxed{4}$	$\frac{3}{2}$	-15	2	1	0	$N = \{2, 3, 4, 5\}$
1	0	0	1	0	0	0	1	
				$\Downarrow$				
3	0	0	-2	18	1	1	0	
0	1	0	$\boxed{8}$	-84	-12	8	0	$B = \{1, 2, 7\}$
0	0	1	$\frac{3}{8}$	$-\frac{15}{4}$	$-\frac{1}{2}$	$\frac{1}{4}$	0	$N = \{3, 4, 5, 6\}$
1	0	0	1	0	0	0	1	
				$\Downarrow$				
3	$\frac{1}{4}$	0	0	-3	-2	3	0	
0	$\frac{1}{8}$	0	1	$-\frac{21}{2}$	$-\frac{3}{2}$	1	0	$B = \{2, 3, 7\}$
0	$-\frac{3}{64}$	1	0	$\boxed{\frac{3}{16}}$	$\frac{1}{16}$	$-\frac{1}{8}$	0	$N = \{1, 4, 5, 6\}$
1	$-\frac{1}{8}$	0	0	$\frac{21}{2}$	$\frac{3}{2}$	-1	1	
				$\Downarrow$				
3	$-\frac{1}{2}$	16	0	0	-1	1	0	
0	$-\frac{5}{2}$	56	1	0	$\boxed{2}$	-6	0	$B = \{3, 4, 7\}$
0	$-\frac{1}{4}$	$\frac{16}{3}$	0	1	$\frac{1}{3}$	$-\frac{2}{3}$	0	$N = \{1, 2, 5, 6\}$
1	$\frac{5}{2}$	-56	0	0	-2	6	1	
				$\Downarrow$				
3	$-\frac{7}{4}$	44	$\frac{1}{2}$	0	0	-2	0	
0	$-\frac{5}{4}$	28	$\frac{1}{2}$	0	1	-3	0	$B = \{4, 5, 7\}$
0	$-\frac{1}{6}$	-4	$-\frac{1}{6}$	1	0	$\boxed{\frac{1}{3}}$	0	$N = \{1, 2, 3, 6\}$
1	0	0	1	0	0	0	1	
				$\Downarrow$				
3	$-\frac{3}{4}$	20	$-\frac{1}{2}$	6	0	0	0	
0	$\boxed{\frac{1}{4}}$	-8	-1	9	1	0	0	$B = \{5, 6, 7\}$
0	$\frac{1}{2}$	-12	$-\frac{1}{2}$	3	0	1	0	$N = \{1, 2, 3, 4\}$
1	0	0	1	0	0	0	1	

TAB. 3.1 – Cyclage. Cet exemple montre qu'avec les règles usuelles, l'algorithme du simplexe peut ne pas converger. Le tableau correspond au vecteur b et à la matrice A multipliés par l'inverse de la sous-matrice des colonnes d'indice B ; dans le tableau, on reconnaît donc la matrice identité dans les colonnes d'indice B

Définition 3.6.2 *[Règle de Bland] La variable entrante est la première pour laquelle $c_j - z_j < 0$: $\bar{\jmath} = \min\{j : c_j - z_j < 0\}$. La variable sortante est la première à atteindre le* $\min_{i:\delta x_i > 0}(\frac{x_i}{\delta x_i})$: $\bar{\imath} = \min\{i : (\frac{x_i}{\delta x_i}) \leq (\frac{x_k}{\delta x_k}) \forall k : \delta x_k > 0\}$.

L'autre règle que nous présentons fait intervenir la notion d'ordre lexicographique de vecteurs.

Définition 3.6.3 *[Ordre lexicographique] Un vecteur $v \in I\!\!R^n$ est dit* lex-positif*, noté $v \overset{L}{\geq} 0$ si sa première composante non-nulle est positive ; $v = 0$ est* lex-zéro*, noté $v \overset{L}{=} 0$.*

Utilisant cette définition, on peut comparer deux vecteurs par l'équivalence $v \overset{L}{\geq} u \iff v - u \overset{L}{\geq} 0$, et définir le lex-min d'un ensemble de vecteurs $\{v_i\}$ comme celui (ou ceux) pour lesquels $v_i \overset{L}{\geq} v_{\bar{\imath}}, \forall i$.

La règle de pivot lexicographique consiste à choisir parmi les indices i qui atteignent le $\min_{i:\delta x_i > 0}(\frac{x_i}{\delta x_i})$ celle dont la ligne complète $(B^{-1}N)_i$ est lex-min de toutes les lignes telles que $\delta x_i = (B^{-1}N_{\bar{\jmath}})_i > 0$. Remarquons que l'ordre des composantes de $I\!\!R^n$ est important, et que malgré que nous regroupions en deux ensembles les variables, les comparaisons lexicographiques *doivent s'effectuer* selon l'ordre des variables du programme original (3.1).

Définition 3.6.4 *[Règle lexicographique d'anti-cyclage] Supposons que toutes les lignes de la matrice A soient lex-positives : $a_i \overset{L}{\geq} 0$. On peut choisir toute colonne $\bar{\jmath}$ telle que $c_{\bar{\jmath}} - z_{\bar{\jmath}} < 0$; la variable sortante est choisie selon le critère lex-min comme suit : soit $\bar{A} = B^{-1}[b : A]$, la matrice transformée dont les lignes sont $\bar{a}_i$. $\bar{\imath}$ satisfait*

$$\frac{\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}} \overset{L}{<} \frac{\bar{a}_i}{\bar{a}_{i\bar{\jmath}}}, \forall i : \bar{a}_{i\bar{\jmath}} > 0.$$

Remarquons que sous l'hypothèse que A est de rang m, aucune paire de lignes de A n'est proportionnelle, et donc l'indice $\bar{\imath}$ ainsi défini est unique. Remarquons également qu'il est toujours possible de démarrer l'algorithme du simplexe de sorte que les lignes de la matrice $\bar{A}$ soient lex-positives.

Exercice 3.6.4 [Anti-cyclage] Considérez le tableau initial de l'exemple de la section 3.6.1 ; avec chacune des deux règles d'anti-cyclage présentées à la section 3.6.2, effectuez suffisamment d'itérations de l'algorithme du simplexe pour changer de point réalisable. □

Exercice 3.6.5 [Deux phases] Proposez une méthode, utilisant deux phases, permettant que pour chacune des deux phases, le tableau initial comporte des lignes lex-positives. □

3.6.3 Convergence finie de l'algorithme du simplexe

Nous allons montrer que l'algorithme du simplexe modifié avec la règle lexicographique ne peut pas cycler.

Théorème 3.6.1 *Supposons que l'on démarre l'algorithme du simplexe avec un tableau dont les lignes sont lex-positives. En appliquant la règle lexicographique, (i) les lignes de la matrice $\bar{A} = B^{-1}[b, A]$ demeurent lex-positives et (ii) la première ligne du tableau est strictement lex-croissante.*

Preuve Montrons d'abord que les lignes demeurent lex-positives. Après un pivot, la $\bar{\imath}^{\text{ème}}$ ligne devient

$$\bar{a}'_{\bar{\imath}} = \frac{\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}}$$

avec $\bar{a}_{\bar{\imath}\bar{\jmath}} > 0$ de sorte que $\bar{a}_{\bar{\imath}}$ demeure lex-positive. Lorsque $i \neq \bar{\imath}$, distinguons deux cas.

- $\bar{a}_{i\bar{\jmath}} > 0$; dans ce cas,

$$\begin{aligned}\bar{a}'_i &= \frac{\bar{a}_i - \bar{a}_{i\bar{\jmath}}\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}}\\ &= \bar{a}_{i\bar{\jmath}}\left(\frac{\bar{a}_i}{\bar{a}_{i\bar{\jmath}}} - \frac{\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}}\right) \geq 0\end{aligned}$$

  par choix lexicographique.
- $\bar{a}_{i\bar{\jmath}} \leq 0$; dans ce cas,

$$\begin{aligned}\bar{a}'_i &= \frac{\bar{a}_i - \bar{a}_{i\bar{\jmath}}\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}}\\ &= \bar{a}_i + \frac{|\bar{a}_{i\bar{\jmath}}|\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}} \overset{L}{\geq} \bar{a}_i\end{aligned}$$

Enfin, examinons l'effet d'un pivot sur la "ligne 0" d'un tableau. Cette ligne est composée de $[-c_B x_B, c - c_b B^{-1} A]$. Notons cette ligne par $\bar{a}_0$. On peut montrer qu'un pivot transforme la ligne 0 en

$$\begin{aligned}\bar{a}'_0 &= \bar{a}_0 - \frac{\bar{a}_{0\bar{\jmath}}\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}}\\ &= \bar{a}_0 + \frac{|\bar{a}_{0\bar{\jmath}}|\bar{a}_{\bar{\imath}}}{\bar{a}_{\bar{\imath}\bar{\jmath}}} \overset{L}{>} \bar{a}_0\end{aligned}$$

puisque par hypothèse $\bar{a}_{0\bar{\jmath}} < 0$ (critère d'entrée) et toutes les lignes sont lex-positives. Par conséquent l'algorithme ne peut pas cycler, car il ne peut pas réexaminer deux fois une même base. □

Exercice 3.6.6 *[]* Démontrez la formule utilisée dans la preuve du théorème 3.6.1 donnant la mise à jour de la ligne 0 après un pivot. □

3.7 Implantation numérique de l'algorithme du simplexe

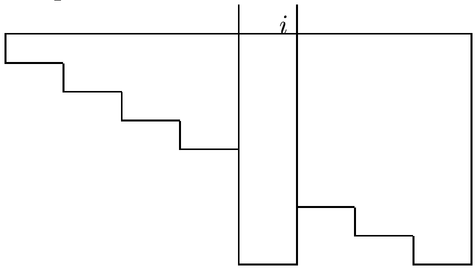
Jusqu'à maintenant, nous avons évité de discuter de l'implantation de l'opération de pivotage, permettant d'effectuer une itération de l'algorithme du simplexe de manière efficace. En général, le calcul de l'inverse d'une matrice B $m \times m$ non-singulière est une opération qui nécessite $\mathcal{O}(m^3)$ opérations. Comme nous allons le voir, il est possible d'implanter l'algorithme du simplexe en utilisant seulement $\mathcal{O}(mn)$ opérations par itération.

La matrice B^{-1} est requise dans deux sous-calculs de l'algorithme 1.1 : il s'agit de $\texttt{SOLUTION}(\pi B = c_B)$ et de $\texttt{SOLUTION}(B\delta x = N_{\bar{\jmath}})$. Nous avons choisi d'écrire ces opérations sous forme de solutions de systèmes linéaires pour atténuer le réflexe naturel de parler de B^{-1}. En effet, l'utilisation de la matrice inverse est un bien mauvais choix pour obtenir la solution de systèmes linéaires, tant du point de vue de l'efficacité des calculs que du point de vue de la précision numérique de la solution. Dans les ouvrages de méthodes numériques, on recommande plutôt d'utiliser une *mise en facteurs*, aussi nommée *décomposition* de la matrice B ; les décompositions recommandées sont la décomposition LU, correspondant à la méthode de Gauss, et la décomposition QR, pouvant être calculée avec les algorithmes de Givens, Householder ou Gramm-Schmidt.

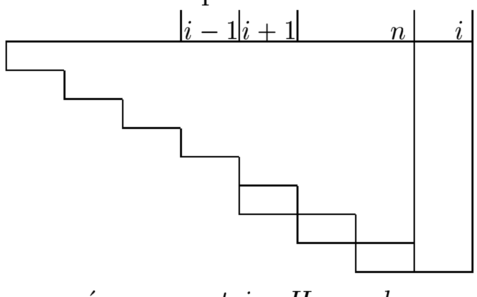
Dans la suite de cette section, nous présentons l'implantation de l'opération de pivotage de l'algorithme du simplexe en utilisant ces deux décompositions matricielles. Ensuite, nous discutons brièvement des difficultés numériques causées par le phénomène de dégénérescence, ou encore de dégénérescence numérique.

3.7.1 Utilisation de la décomposition LU

La décomposition LU d'une matrice carrée est obtenue par le procédé d'élimination de Gauss. L'appendice A résume les variantes de cette méthode. Supposons donc que la matrice B soit décomposée $B = LU$, où U est obtenue par le procédé d'élimination de Gauss, et L représente les opérations élémentaires utilisées lors de l'élimination. Supposons maintenant que la matrice B se fasse substituer une nouvelle colonne B'_i à la place de l'ancienne. Alors, de la relation $U = L^{-1}B$, on tire que $H = L^{-1}B'$ sera de la forme suivante :



Il suffit maintenant, à l'aide d'une permutation de colonne, de déplacer la colonne trouble à la dernière position en décalant les autres pour obtenir la forme suivante :



Cette dernière matrice est nommée une *matrice Hessenberg supérieure*, et est presque déjà triangularisée, à l'exception de quelques éléments sur la sous-diagonale. On voit donc qu'il est possible de triangulariser cette matrice Hessenberg supérieure en utilisant $n - i$ pivots de la méthode d'élimination de Gauss. De plus, on peut utiliser la stratégie de pivot partiel de manière très simple en examinant seulement l'élément diagonal, et l'élément sous-diagonal car les autres éléments d'une colonne sont tous nuls. Soit donc $M = M_n M_{m-1} \ldots M_{i+1}$ la matrice qui représente les $n - i$ pivots. On a $ML^{-1}B' = U'$, de sorte que $L' = LM^{-1}$.

Une autre stratégie de choix de pivot permet de profiter de la structure de zéros de la matrice.

3.7.2 Utilisation de la décomposition QR

Il est possible de triangulariser une matrice non-singulière à l'aide d'une matrice orthogonale Q.

3.7.3 Dégénérescence numérique

Les problèmes causés par le phénomène de dégénérescence ne se limitent pas aux troubles de cyclage décrits plus haut. En fait, même en présence de problèmes dégénérés, lors des calculs, les variables x_B ne seront pas nulles, mais prendront des valeurs numériquement très petites.

Un autre problème numérique relié à la dégénérescence est le suivant. Considérons l'exemple où dans $\mathbb{R}^3$, trois contraintes se rencontrent selon une droite. Bien entendu, un sommet des contraintes impliquera au moins une quatrième contrainte. Cependant, il est fort possible que, *numériquement, les trois contraintes définissent un point.* Ce point n'est pas, en général, un point extrême des contraintes au sens mathématique, mais peut fort bien, dû aux erreurs numériques, être accepté comme solution optimale.

Il faut retenir que le phénomène de dégénérescence engendre des problèmes au niveau théorique, et en engendre d'autres, encore plus difficile à résoudre, au niveau numérique.

3.8 Comment obtenir la première solution de base réalisable

Parmi les techniques usuelles pour initialiser l'algorithme du simplexe, la plus répandue consiste à résoudre un problème en *deux phases*. La première phase résout le problème

$$\begin{aligned} &\min ez \\ &\text{s.à. } Iz + Ax = b \\ &\quad x, z \geq 0, \end{aligned} \tag{3.5}$$

où les variables z sont nommées variables artificielles et le vecteur $e \in \mathbb{R}^m$ est tel que $e_i = 1, i = 1, 2, \ldots, m$. En prenant le soin de multiplier par -1 les contraintes associées à un membre de droite b_i négatif, le vecteur $z = b, x = 0$ constitue une solution de base réalisable pour le problème (3.5).

Si le programme (3.1) possède une solution réalisable, alors le programme (3.5) possède une solution optimale de valeur nulle, et $z = 0$. La phase I consiste donc à appliquer l'algorithme du simplexe au problème (3.5). À sa terminaison, trois cas peuvent se produire :

1. $z = 0$, et toutes les composantes de z sont dans la partition N ;
2. $z = 0$, mais certaines composantes de z sont dans B ;
3. $z \neq 0$.

Dans le premier cas, il est possible de résoudre le programme (3.1) directement. Dans le troisième cas, le programme (3.1) ne possède pas de solution réalisable. Enfin, dans le second cas, il faut effectuer certaines opérations de pivotage pour éliminer les composantes de z de B. Cependant, pour ce faire, les critères de pivot sont simplifiées puisque notre unique but est de compléter la base partielle (éléments de B correspondant à des variables x) avec d'autres composantes x, tout en les laissant nulles. Soit donc une colonne associée à une variable de base $z_i = 0$. Il est alors possible d'effectuer un pivot sur tout élément non-nul de la ligne $\bar{a}_i$ correspondant à une variable x. Ceci a pour objet de changer la partition sans changer le point (z, x), qui constitue une solution dégénérée. Si jamais une ligne de la matrice $\bar{A}$ devenait nulle, c'est le signe que la matrice A n'est pas de rang m, et il est alors possible d'éliminer cette ligne.

3.9 Extensions et références

Le matériel de ce chapitre est bien connu. Deux bonnes références quant aux aspects combinatoires de la programmation linéaire sont les livres de Chvátal et de Papadimitriou et Steiglitz [22]. Bien sûr, le livre classique de Dantzig [7], qui est à l'origine de toute cette théorie demeure un ouvrage de référence précieux.

Nous avons gardé sous silence plusieurs aspects par ailleurs intéressants de la programmation linéaire. Essayons de réparer ces omissions.

- *[Algorithme dual du simplexe]* Considérons un tableau de la forme de ceux utilisés à la section 3.6. Cependant, remplaçons la propriété que la première colonne, $B^{-1}b \geq 0$ par la propriété que la première ligne du tableau $c - z \geq 0$. D'après ce que nous avons vu de la dualité, cet échange de propriétés entraîne que nous avons sous la main une solution de base dual-réalisable plutôt que primal-réalisable. Nous pouvons donc imiter le développement de l'algorithme du simplexe en choisissant une composante négative de la première colonne, destinée à devenir non-négative, qui serait la variable sortante, puis en identifiant une variable entrante de telle sorte que l'opération de pivotage conserve la propriété que la ligne $c - z \geq 0$. Cet algorithme, qui manipule de même tableau que l'algorithme du simplexe est nommé *simplexe dual*. On peut démontrer qu'il s'agit en fait de l'algorithme du simplexe appliqué au programme (3.3).
- *[Interprétations économiques]* Une grande partie de la popularité de la programmation linéaire provient d'interprétations économiques aussi instructives qu'utiles. En voici une résumée. Le problème de la diète consiste, pour un client à choisir des quantités d'aliments x_i (variables primales) qui minimisent le coût d'achat cx tout en satisfaisant des contraintes diététiques $a_i x \geq b^i$ (contenu suffisant de chacun de plusieurs nutriments). Le dual s'interprète du point de vue d'un vendeur de pilules de nutriments. Il veut fixer le prix π_j de ses pilules de telle sorte que, compte tenu de la demande en nutriments b, il maximise son profit πb tout en s'assurant que le coût des nutriments contenus dans chacun des l'aliment ne dépasse pas le coût de l'aliment $\pi A_j \leq c_j$.

Chapitre 4

Optimisation différentiable avec contraintes linéaires

Sujets du chapitre.

- Contraintes linéaires d'égalité et d'inégalité.
- Formulations diverses des conditions d'optimalité.
- Problèmes de complémentarité linéaire.
- Déduction de quelques algorithmes à partir des conditions d'optimalité.

Introduction

La programmation linéaire a connu un essor considérable dans les années 60. Il est donc naturel que des chercheurs aient généralisé de diverses manières les idées développées dans l'analyse de programmes linéaires. Nous présentons dans ce chapitre quelques généralisations. Nous abordons également ces problèmes non-linéaires du point de vue de la programmation non-linéaire, constituant ainsi un premier contact avec les sujets des chapitres à venir.

4.1 Énoncé du problème

Nous abordons maintenant la généralisation de problèmes de type (3.1) au cas où la fonction objectif est différentiable non-linéaire. Nous traitons le problème

$$\begin{aligned} \min\ & f(x) \\ \text{s.à. } & Ax = b \\ & x \geq 0, \end{aligned} \tag{4.1}$$

où l'ensemble $E \stackrel{\text{def}}{=} \{x : Ax = b, x \geq 0\}$ est nommé *ensemble réalisable*. Nous allons d'abord déduire les conditions d'optimalité pour le programme (4.1) à partir de la théorie de la programmation linéaire présentée au chapitre 3. Ensuite, nous effectuerons d'autres démarches qui conduisent à des formulations équivalentes des conditions d'optimalité.

Comme dans les chapitres précédents, nous déduirons quelques algorithmes à partir de points qui ne satisfont pas aux conditions d'optimalité.

4.2 Conditions d'optimalité pour les points stationnaires

Remarquons d'abord que si x^* est une solution optimale du programme (4.1), alors la dérivée directionnelle de f en x^* dans toute direction réalisable $d = (x - x^*)$ doit être non-négative, ce qui s'exprime par l'*inégalité variationnelle*

$$\begin{aligned} \nabla f(x^*)(x - x^*) &\geq 0 \\ Ax &= b \\ x &\geq 0. \end{aligned} \tag{4.2}$$

Un tel point x^* est nommé *point stationnaire* pour le programme (4.1). Cette formulation constitue la base des analyses subséquentes.

4.2.1 Formulation inspirée de la programmation linéaire

Puisque x^* satisfait aux équations et inéquations (4.2), x^* est également solution optimale du programme linéaire

$$\begin{aligned} \min\ & \nabla f(x^*)x \\ \text{s.à. } & Ax = b \\ & x \geq 0. \end{aligned} \tag{4.3}$$

Cependant, ce n'est pas nécessairement une solution optimale de base, et nous devons généraliser quelque peu la théorie présentées plus haut.

La partition Base—Hors-base revisitée

Pour le problème (4.1), rien ne permet de supposer que x^* est une solution de base. En fait, puisque l'objectif f n'est pas linéaire, il est fort possible que toutes les composantes de la solution x^* soient positives. Par exemple, le minimum de $f(x) = x^2$ sur l'intervalle $[-1, 1]$ est atteint *à l'intérieur* du domaine réalisable.

Heureusement, il est possible de tirer profit des cas ou elle comporte des composantes nulles. Ces composantes nulles sont toujours nommées composantes hors-base, et notées x_N. Par ailleurs, parmi les composantes non-nulles, nommons x_B les composantes associées à une sous-matrice inversible B, et baptisons les autres composantes *variables de super-base*, notées x_S. Nous obtenons donc la partition $[\![B \mid S \mid N]\!]$ et la matrice A, ainsi que les vecteurs c et x sont partitionnés de telle sorte que nous puissions écrire $x = [x_B, x_S, x_N]$, $A = [B, S, N]$ ainsi que $\nabla f(x^*) = [\nabla f(x^*)_B, \nabla f(x^*)_S, \nabla f(x^*)_N]$.

Tout comme avant, il est possible de mettre x_B en fonction de x_S et de x_N :

$$x_B = B^{-1}(b - Sx_S - Nx_N).$$

Alors, en définissant encore (voir page 74) $\pi \stackrel{\text{def}}{=} \nabla f(x^*)_B B^{-1}$, on arrive à l'expression

$$\nabla f(x^*)x = (\nabla f(x^*)_N - \pi N)x_N + (\nabla f(x^*)_S - \pi S)x_S + Cst.$$

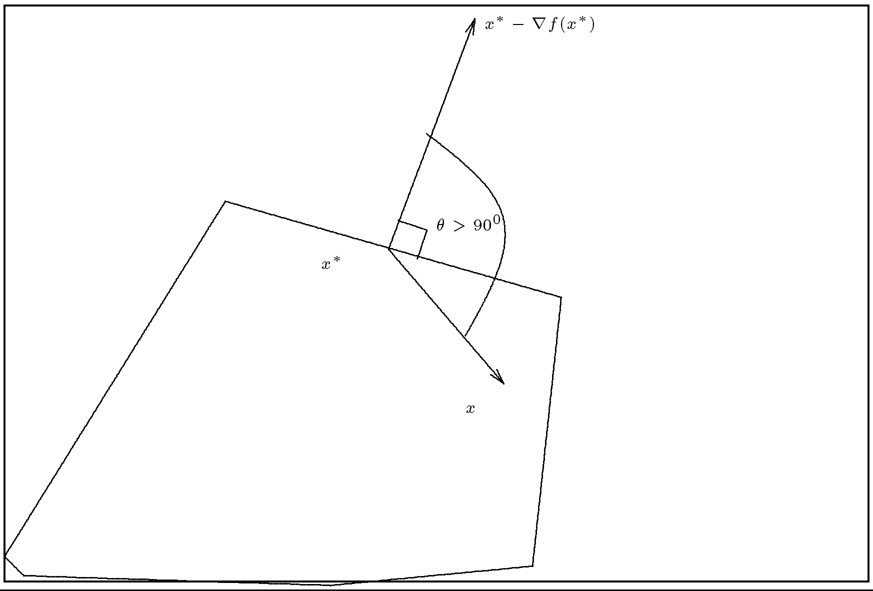
De même les contraintes se réécrivent

$$\begin{aligned} B^{-1}(b - Nx_N - Sx_S) &\geq 0 \\ x_N, x_S &\geq 0. \end{aligned}$$

En adaptant le raisonnement qui nous a conduit aux conditions d'optimalité du programme (3.1), nous obtenons

$$\pi A = \begin{cases} \pi B = \nabla f(x^*)_B \\ \pi S = \nabla f(x^*)_S \\ \pi N \leq \nabla f(x^*)_N. \end{cases}$$

La première égalité (les composantes B) définit les multiplicateurs π ; la seconde égalité vient du fait que les composantes S sont strictement positives, et donc pourraient aussi bien augmenter que diminuer ; leurs coûts relatifs doivent donc s'annuler. L'inégalité sur les composantes hors-base est comme dans le cas linéaire.



On voit que si $-\nabla f(x^*)$ fait un angle obtus avec $x - x^*$ pour tout point réalisable x, alors le point réalisable le plus proche de $x^* - \nabla f(x^*)$ est le point x^* lui-même.

FIG. 4.1 – Géométrie de la projection

En introduisant les variables d'écart $s \geq 0$, on peut reformuler les conditions précédentes comme

$$\pi A + s = \nabla f(x^*)$$
$$sx = 0,$$

le dernier terme étant connu sous le nom de *terme de complémentarité.*

Dégénérescence

Les raisonnements précédents sont valables à condition qu'il soit possible de trouver des composantes positives de x^*, x_B^* associées à une sous-matrice inversible B. Cette limite n'est toutefois seulement apparente. En effet, si la matrice A possède des lignes linéairement indépendantes et si le problème possède au moins une solution réalisable, on peut toujours trouver des composantes $x_B^* \geq 0$ associés à une matrice inversible. À aucun moment nous avons utilisé le fait que les composantes de x_B^* devaient être strictement positives.

Nous verrons un peu plus loin que la dégénérescence de la solution optimale influence l'énoncé de conditions de second ordre.

4.2.2 Formulation sous forme de projection

Revenons aux inéquations variationnelles (4.2). On y fait état que l'angle entre $-\nabla f(x^*)$ et un vecteur allant de x^* à x, un point réalisable quelconque n'est pas un angle aigu car son cosinus est non-positif.

On constate donc que le point le plus proche de $x^* - \nabla f(x^*)^t$ à l'intérieur des contraintes est le point x^* lui-même. Ce point le plus proche se nomme *projection* :

$$Proj_E(x) = \arg\min_{y \in E} \|y - x\|^2.$$

On a donc que $x^* = Proj_{\{x:Ax=b,x\geq 0\}}(x^* - \nabla f(x^*)^t)$. On dit donc que x^* est un point fixe de la fonction $P(x) = Proj_E(x - \nabla f(x)^t)$, c'est-à-dire, $x^* = P(x^*)$.

4.3 Conditions de Karush-Khun-Tucker

Pour les développements qui suivent, nous allons nous éloigner de la formulation des contraintes sous la forme standard du simplexe. Nous allons développer ces conditions en deux étapes : d'abord, nous traiterons seulement les contraintes d'égalité, et ensuite, nous verrons comment adapter la théorie aux inégalités.

4.3.1 Conditions pour les contraintes d'égalité

Donc, momentanément, nous considérons les problèmes de la forme

$$\begin{aligned} &\min f(x) \\ &\text{s.à. } Ax = b. \end{aligned} \tag{4.4}$$

La notion d'orthogonalité est cruciale pour nos développements. Observons d'abord que si on a sous la main un point réalisable x, alors un autre point réalisable peut être obtenu à l'aide d'une direction $d : Ad = 0$, c'est-à-dire une direction appartenant au noyau de la matrice A. Nous savons que le noyau d'une matrice constitue un espace vectoriel. Soit Z une matrice dont les colonnes forment une base de $\ker A$. Alors, $AZ = 0$. De plus, en admettant que les m lignes de la matrice A soient linéairement indépendantes, la matrice Z comportera n lignes et $n - m$ colonnes. Une direction réalisable, donc appartenant à $\ker A$, peut donc s'exprimer comme une combinaison linéaire des colonnes de la matrice Z, qui, rappelons-le, constituent une base de $\ker A$. Notons les $(n - m)$ coefficients de la direction réalisable d dans la base en question $d_z : d = Zd_z$.

Maintenant, supposons que x^* soit un point réalisable et optimal (localement). Nous allons récrire $f(x)$ pour des x réalisables proches de x^* à l'aide de direction réalisables, $f(x) = f(x^* + Zd_z)$. Puisque x^* est un minimum local de f, alors $d_z = 0$ est un minimum local de $\phi(d_z) \stackrel{\text{def}}{=} f(x^* + Zd_z)$. En fait, l'étude de la fonction f dans l'intersection d'un voisinage de x^* avec le domaine réalisable est équivalent à l'étude de la fonction ϕ dans un voisinage de zéro. Les conditions d'optimalité s'exprimeront donc à l'aide des quantités $\nabla\phi(0)$ et $\nabla^2\phi(0)$. Or, $\nabla\phi(d_z) = \nabla f(x^* + Zd_z)Z$ et $\nabla^2\phi(d_z) = Z^t\nabla^2 f(x^* + Zd_z)Z$, et donc $\nabla\phi(0) = \nabla f(x^*)Z$ alors que $\nabla^2\phi(0) = Z^t\nabla^2 f(x^*)Z$. Ces dernières quantités portent les noms de *gradient réduit* et *hessien réduit* de f en x^* respectivement.

Maintenant, en appliquant les conditions d'optimalité du chapitre 2 à la fonction ϕ, on déduit que $\nabla f(x^*)Z = 0$. Examinons de plus près cette condition. Nous avons supposé que les lignes de la matrice A étaient linéairement indépendantes, et puisque les colonnes de la matrice Z forment une base, elles sont également linéairement indépendantes. En fait, les colonnes de la matrice Z constituent une base de $\ker A$ et les lignes de la matrice A constituent une base de $\ker Z^t$. Puisque $\nabla f(x^*)^t \in \ker Z^t$, on a donc que $\nabla f(x^*)$ peut s'écrire comme une combinaison linéaire des lignes de la matrice A, $\nabla f(x^*) = \pi A$.

L'optimisation avec contraintes d'égalité a été étudiée depuis fort longtemps. Lagrange avait formulé les conditions d'optimalité que nous venons de déduire à l'aide d'une fonction auxiliaire appelée *Lagrangien* $L(x, \lambda) \stackrel{\text{def}}{=} f(x) + \lambda(Ax - b)$. Les conditions d'optimalité de premier ordre s'écrivent simplement

$$\begin{aligned} \nabla_x L(x^*, \lambda^*) = 0 = \nabla f(x^*) + \lambda^* A \\ \nabla_\lambda L(x^*, \lambda^*) = 0 = Ax^* - b. \end{aligned}$$

On voit donc que la seconde condition assure que x est réalisable alors que la première est équivalente à celle que nous avons déduite avec $\lambda = -\pi$.

On peut résumer ces développements dans le théorème suivant.

Théorème 4.3.1 *Considérons le problème (4.4) et soit Z une matrice dont les colonnes constituent une base de* $\ker A$.

1. *Si x^* est un minimum local de f, alors il existe un vecteur λ^* tel que $\nabla f(x^*) + \lambda^* A = 0$, ou de façon équivalente, $\nabla f(x^*)Z = 0$. De plus, la matrice $Z^t \nabla^2 f(x^*) Z$ est semi-définie positive.*
2. *S'il existe un vecteur λ^* tel que $\nabla f(x^*) + \lambda^* A = 0$, ou de façon équivalente, $\nabla f(x^*)Z = 0$ et si de plus, la matrice $Z^t \nabla^2 f(x^*) Z$ est définie positive, alors x^* est un minimum local de f.*

Exercice 4.3.1 *[Contraintes d'égalité]* Considérez le problème

$$\begin{aligned} &\min_{x \in \boldsymbol{R}^3} \|x\|^2 \\ &\text{s.à. } x_1 + 2x_2 + 3x_3 = 1. \end{aligned}$$

a) Fournissez une expression pour la matrice Z.

b) Identifiez la solution x^* et λ^* du problème.

c) Vérifiez que $Z^t \nabla f(x^*) = 0$.

d) Vérifiez que $x^* = Proj_E(x^* - \nabla f(x^*))$.

e) Résolvez le problème

$$\begin{aligned} &\min_{x \in \boldsymbol{R}^n} \|x\|^2 \\ &\text{s.à. } \sum_{i=1}^{n} i x_i = 1. \end{aligned}$$

□

4.3.2 Conditions pour les contraintes d'inégalité

Encore ici, nous adoptons une formulation différente de la forme standard du simplexe, et considérons le problème

$$\begin{aligned} &\min f(x) \\ &\text{s.à. } Ax \le b. \end{aligned} \tag{4.5}$$

Soit un point x^* candidat minimum local. Alors, x^* est d'abord un point réalisable. Parmi les contraintes, certaines seront saturées en x^*, c'est-à-dire certaines seront satisfaites avec égalité, et les autres seront satisfaites avec inégalité stricte. Séparons la matrice A ainsi que le vecteur b selon ce critère, de sorte que $A_1x^* = b_1$ et $A_2x^* < b_2$; les premières contraintes sont nommée *contraintes actives* et les autres contraintes inactives. Puisque nous étudions les optima locaux de f, nous limitons notre analyse à un voisinage de x^*, et donc, on peut choisir un voisinage assez petit pour que les contraintes inactives soient toujours satisfaites avec inégalité stricte. Par conséquent, on peut effectuer notre analyse avec les seules contraintes actives.

En particulier, on voit que les conditions présentées plus haut pour les contraintes d'égalités s'appliquent encore ici aux contraintes actives, et donc il devra exister des valeurs λ telles que $\nabla f(x^*) + \lambda A_1 = 0$. Cependant, dans notre cas, les variables λ devront satisfaire à une condition additionnelle : il ne suffit plus que $\nabla f(x^*)^t \in \ker Z_1^t$ mais il faut également qu'il soit orienté *vers l'intérieur* du domaine réalisable. En effet, il faut que les directions réalisables fassent des angles aigus avec $\nabla f(x^*)^t$, selon l'inégalité variationnelle (4.2). Or, cette bonne orientation de $\nabla f(x^*)^t$ s'exprime simplement par $\lambda \geq 0$. Ces observations nous donnent des conditions nécessaire d'optimalité.

Théorème 4.3.2 *Considérons le problème (4.5) et soit x^* un minimum local de f tel que $A_1x^* = b_1$ et $A_2x^* < b_2$; soit également Z une matrice dont les colonnes constituent une base de* $\ker A_1$. *Alors, il existe un vecteur $\lambda^* \geq 0$ tel que $\nabla f(x^*) + A_1^t\lambda^* = 0$, ou de façon équivalente, $\nabla f(x^*)Z = 0$. De plus, la matrice $Z^t\nabla^2 f(x^*)Z$ est semi-définie positive.*

Pour obtenir des conditions suffisantes, il faut observer que même si le vecteur λ^* est non-négatif, certaines de ses composantes pourraient être nulles.

Théorème 4.3.3 *Considérons le problème (4.5) et soit x^* un point tel que $A_1x^* = b_1$ et $A_2x^* \leq b_2$; soit également Z_1 une matrice dont les colonnes constituent une base de* $\ker A_1$. *S'il existe un vecteur $\lambda^* > 0$ tel que $\nabla f(x^*) + \lambda^* A_1 = 0$, ou de façon équivalente, $\nabla f(x^*)Z_1 = 0$ et si de plus, la matrice $Z_1^t\nabla^2 f(x^*)Z_1$ est définie positive, alors x^* est un minimum local de f.*

Ces conditions d'optimalité constituent un cas particulier des conditions de Karush-Khun-Tucker (K–K–T), cas particulier aux contraintes linéaires. Nous verrons dans les chapitres ultérieurs des conditions plus générales.

Encore ici, nous pouvons utiliser le Lagrangien pour exprimer les conditions d'optimalité. Rappelons que $L(x, \lambda) = f(x) + \lambda(Ax - b)$. Les conditions d'optimalité de premier ordre s'écrivent :

$$\begin{aligned} \nabla_x L(x^*, \lambda^*) &= 0 \\ \nabla_\lambda L(x^*, \lambda^*) &\leq 0 \\ \lambda^*(Ax^* - b) &= 0 \\ Ax^* - b &\leq 0 \\ \lambda^* &\geq 0 \end{aligned}$$

et le dernier terme est nommé *terme de complémentarité*. L'hypothèse de stricte complémentarité impose que $\lambda_i^* = 0 \iff A_ix^* > b_i$. Cette condition de stricte complémentarité est reliée au phénomène de dégénérescence dont nous avons parlé plus tôt.

4.3.3 Exemples

Pour illustrer les conditions d'optimalité, voici une série d'exemples. Il s'agit d'abord d'un problème comportant deux contraintes d'égalité. Ensuite, nous analyserons le même problème, mais ne comportant qu'une seule contrainte d'égalité. Finalement, nous analyserons la situation si les deux contraintes du problème initial étaient des inégalités, soit non-négatives, non-positives, ou encore un mélange des deux.

Exemple 1

Considérons le programme

$$\begin{array}{ll} \min & x_1^2 - x_1x_2 + x_2^2 - x_2x_3 + x_3^2 + 2x_1 - x_2 \\ \text{sujet à} & 3x_1 - x_2 + x_3 = 0 \\ & 2x_1 - x_2 - x_3 = 0 \end{array}$$

qui peut également s'exprimer matriciellement :

$$\begin{array}{ll} \min & \frac{1}{2}x^tQx + cx \\ \text{sujet à} & Ax = 0 \end{array}$$

où

$$Q = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}, \quad c = \begin{pmatrix} 2 & -1 & 0 \end{pmatrix}, \quad A = \begin{pmatrix} 3 & -1 & 1 \\ 2 & -1 & -1 \end{pmatrix}.$$

En additionnant la première ligne a la seconde dans la matrice A, on obtient

$$A' = \begin{pmatrix} 3 & -1 & 1 \\ 5 & -2 & 0 \end{pmatrix},$$

d'où l'on déduit que $x_1 = \frac{2}{5}x_2$, et, utilisant la premiere ligne, $x_3 = \frac{-1}{5}x_2$, donc, une expression pour la base du noyau de A est donnée par $Z = \begin{pmatrix} 2 \\ 5 \\ -1 \end{pmatrix}$.

On a donc que tout x réalisable est un multiple de Z, $x = Zd_z$. Par conséquent, on peut solutionner le problème original avec contraintes en minimisant par rapport a d_z la fonction $\frac{1}{2}d_zZ^tQZd_z + c^tZd_z$. La matrice Z^tQZ est réduite au scalaire 50, le vecteur c^tZ est réduit au scalaire -1 et la valeur optimale de $25d_z^2 - d_z$ est $d_z^* = \frac{1}{50}$. Ceci entraîne que $x^* = \begin{pmatrix} \frac{2}{50} \\ \frac{5}{50} \\ \frac{-1}{50} \end{pmatrix}$.

Vérifions que $x^*Q + c + \lambda A$ possède bel et bien une solution λ^*. Remarquons que x^* est réalisable, car il satisfait aux contraintes. En général, pour un point réalisable quelconque $\bar{x}$, le système $\bar{x}Q + c + \lambda A$ ne possède pas de solution en λ. Seuls les points stationnaires partagent cette propriété.

$$x^*Q + c = \begin{pmatrix} \frac{99}{50} & \frac{-41}{50} & \frac{-7}{50} \end{pmatrix}.$$

La solution du système

$$\begin{pmatrix} \frac{99}{50} & \frac{-41}{50} & \frac{-7}{50} \end{pmatrix} + \begin{pmatrix} \lambda_1 & \lambda_2 \end{pmatrix} \begin{pmatrix} 3 & -1 & 1 \\ 2 & -1 & -1 \end{pmatrix} = 0$$

est

$$\lambda = \begin{pmatrix} \frac{-17}{50} & \frac{-24}{50} \end{pmatrix}.$$

Exemple 2

Considérons maintenant la même fonction objectif, mais à minimiser sous seulement la seconde contrainte, $2x_1 - x_2 - x_3 = 0$, donc $A = \begin{pmatrix} 2 & -1 & -1 \end{pmatrix}$. Une base du plan correspondant au domaine réalisable est

$$Z = \begin{pmatrix} 1 & 1 \\ 0 & 2 \\ 2 & 0 \end{pmatrix}$$

car les colonnes de cette matrice sont linéairement indépendantes, et satisfont toutes les deux à la contrainte. Cette fois-ci, la matrice

$$Z^t Q Z = \begin{pmatrix} 10 & -4 \\ -4 & 6 \end{pmatrix}$$

n'est pas réduite à un scalaire, et le vecteur

$$cZ = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$$

non plus. Alors, d_z est ici un vecteur de dimension 2. La solution optimale de $\min_{d_z} \frac{1}{2} d_z^t Z^t Q Z d_z + c^t Z d_z$ est donnée par $d_z^* = \begin{pmatrix} \frac{-3}{11} \\ \frac{-2}{11} \end{pmatrix}$. Par conséquent, $x^* = \begin{pmatrix} \frac{-5}{11} \\ \frac{-4}{11} \\ \frac{-6}{11} \end{pmatrix}$. Calculons

$$x^* Q + c = \begin{pmatrix} \frac{-6}{11} & \frac{3}{11} & \frac{-8}{11} \end{pmatrix} + \begin{pmatrix} 2 & -1 & 0 \end{pmatrix} = \begin{pmatrix} \frac{16}{11} & \frac{-8}{11} & \frac{-8}{11} \end{pmatrix}.$$

On observe bien que $x^* Q + c + \frac{-8}{11} A = 0$, et donc $\lambda^* = \frac{-8}{11}$.

Exemple 3

Les deux contraintes de l'exemple 1 étaient des contraintes d'égalité. Si les deux contraintes avaient été

$$3x_1 - x_2 + x_3 \geq 0$$
$$2x_1 - x_2 - x_3 \geq 0,$$

alors le point x^* trouvé serait un point stationnaire pour le problème. En effet, dans la formulation Lagrangienne, $\nabla f(x^*) + \lambda^* A = 0$, et les multiplicateurs de Lagrange doivent etre du signe opposé aux contraintes $Ax - b$. Inégalités *plus grand ou égal* dans les contraintes entraînent λ *plus petit ou égal* à zéro, et vice-versa. Ici, les contraintes sont *plus grand ou égal* et les λ *plus petit ou égal* à zéro, $\lambda^* = \begin{pmatrix} \frac{-17}{50} & \frac{-24}{50} \end{pmatrix}$.

Exemple 4

Si les deux contraintes avaient été

$$3x_1 - x_2 + x_3 \leq 0$$
$$2x_1 - x_2 - x_3 \geq 0,$$

alors le point x^* trouvé à l'exemple 1 ne serait pas un point stationnaire pour le problème. En effet, la première contrainte étant non-positive, les multiplicateurs doivent être non-négatifs, et ce n'est pas le cas. Il faut supposer qu'au moins une des deux contraintes que nous avons implicitement supposées actives ne l'est pas à la solution. Puisque c'est la première qui permet d'établir la non-optimalité du point, supposons donc que seule la seconde contrainte est active à la solution. Ceci est la situation de l'exemple 2, et le multiplicateur que nous avons calculé est du bon signe, $\lambda^* = \frac{-8}{11}$.

Exemple 5

Si les deux contraintes avaient été

$$3x_1 - x_2 + x_3 \leq 0$$
$$2x_1 - x_2 - x_3 \leq 0,$$

alors le point x^* trouvé à l'exemple 1 ne serait pas un point stationnaire pour le problème. En effet, aucune des deux contraintes se voit associer un multiplicateur du bon signe. Par ailleurs, l'ensemble des contraintes actives à la solution ne peut pas être uniquement la seconde contrainte car l'exemple 2 nous révèle que son multiplicateur n'est pas du bon signe. Il est donc possible que la solution optimale pour ce cinquième exemple ne sature aucune des deux contraintes, ou encore en sature une seule, la première. Les autres possibilités ont été éliminées dans les deux premiers exemples. Supposons donc qu'il n'y ait aucune contrainte saturée. Il faut alors résoudre $xQ + c = 0$. Or, la première ligne de la matrice Q est justement égale à c. La solution est donc $x^* = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$, et par bonheur, satisfait avec inégalité stricte aux deux contraintes. C'est donc la solution de notre problème.

Il aurait pu en être autrement, car même si les deux multiplicateurs sont du mauvais signe cela ne veut pas dire que les multiplicateurs calculés avec moins de contraintes seront toujours du mauvais signe. Il en est de même pour des multiplicateurs du bon signe. Sur de petits exemples, c'est habituellement le cas, mais, comme en programmation linéaire, au cours de l'exploration de l'algorithme du simplexe, qui correspond à la stratégie d'enlever des contraintes actives (variables hors-base) une contrainte associée à un multiplicateur du mauvais signe (coût relatif négatif), il est possible qu'une variable entre dans la base plus d'une fois, i.e. qu'une contrainte devienne supposément active à plusieurs reprises.

Exercice 4.3.2 *[Contraintes d'inégalités]*

a) Vérifiez que la solution obtenue à l'exercice 4.3.1 constitue la solution optimale du problème

$$\min_{x \in \mathbf{R}^3} \|x\|^2$$
$$\text{s.à. } x_1 + 2x_2 + 3x_3 = 1; \qquad x \geq 0.$$

b) Que peux-t-on dire de cette dernière solution par rapport au problème

$$\min_{x \in R^3} \|x\|^2$$
$$\text{s.à. } x_1 + 2x_2 + 3x_3 \leq 1; \qquad x \geq 0.$$

c) Que peux-t-on dire de cette dernière solution par rapport au problème

$$\min_{x \in R^3} \|x\|^2$$
$$\text{s.à. } x_1 + 2x_2 + 3x_3 \geq 1; \qquad x \geq 0.$$

□

Exercice 4.3.3 *[PNL]* Obtenez la solution du problème

$$\min_{x,y,z} 4/3(x^2 - xy + y^2)^{3/4} - z$$
$$\text{s.à. } z \leq 2, \quad x, y, z \geq 0.$$

Est-ce que la condition de stricte complémentarité est satisfaite en x^* ? □

Exercice 4.3.4 *[Projection]* Soit $E = \{x : Ax - b \leq 0\}$ un ensemble réalisable. Vérifiez que $\bar{x}$ est un point fixe de l'opérateur de projection, $\bar{x} = Proj_E(\bar{x} - \nabla f(\bar{x})^t)$, si et seulement si $\bar{x}$ satisfait aux conditions nécessaires de K–K–T. □

4.4 Remarques générales sur les algorithmes

Comme pour les problèmes sans contrainte, nous allons déduire quelques algorithmes à partir de la violation des conditions d'optimalité. Cependant, un élément nouveau nous empêche de présenter une théorie de convergence globale aussi simple que celle du chapitre 2 : les contraintes d'inégalité introduisent un aspect combinatoire, à savoir l'identification des contraintes actives à la solution. Cet aspect se manifeste de deux manières principales : en présence de dégénérescence, les algorithmes inspirés de l'algorithme du simplexe pourront cycler, ou converger vers des points qui ne sont pas stationnaires. Pour d'autres types d'algorithmes, c'est le désir de demeurer dans le domaine réalisable, quitte à réduire le pas θ utilisé, qui provoque le désastre, les pas utilisés ne satisfaisant plus aux conditions d'admissibilité présentées au chapitre 2.

Cependant, tout comme pour l'algorithme du simplexe, ces difficultés théoriques n'ont pas empêché la production d'implantations efficaces qui se sont avérées robustes dans la pratique.

4.5 Algorithmes inspirés de la programmation linéaire

Il est possible de définir un algorithme directement de l'analyse précédente. Nous en déduirons l'algorithme du gradient réduit. Par ailleurs, sans se préoccuper des partitions en variables de base et de super-base, nous pouvons quand même développer un algorithme basé sur le fait que x^* est solution de (4.3) ; c'est l'algorithme de Frank&Wolfe.

4.5.1 Algorithme de Frank&Wolfe

Revenons au problème (4.3). Supposons que l'on résolve ce problème en un point quelconque, qui n'est pas un minimum local du programme (4.1). Alors, le programme linéaire (4.3) possédera une solution y telle que $\nabla f(x)(y-x) < 0$, c'est-à-dire, la direction $(y-x)$ constitue une direction de descente réalisable pour le programme (4.1). Nommant d cette direction, nous allons choisir $x^+ \stackrel{\text{def}}{=} x + \theta d$ de sorte que θ minimise f sur la ligne joignant x et y, tout en demeurant réalisable. Alors, $f(x^+) < f(x)$, et nous pouvons recommencer le processus. Nous en déduisons la suite de quantités suivante :

$$\begin{aligned}
y_k &= \arg\min \nabla f(x_k)y : Ay = b, y \geq 0; \\
d_k &= y_k - x_k; \\
\theta_k &= \arg\min_{0\leq\theta\leq 1} f(x_k + \theta(d_k)); \\
x_{k+1} &= x_k + \theta_k d_k.
\end{aligned}$$

Les observations précédentes montrent que la suite $f(x_k)$ est décroissante. Pour justifier que les points d'accumulation $\bar{x}$ de la suite $\{x_k\}$ produite par l'algorithme sont des solutions satisfaisant $\nabla f(\bar{x})\bar{d} = 0$, nous appliquerons le théorème de Zangwill. Il est bien connu que $\theta = \arg\min_{0\leq\theta\leq 1} f(x + \theta(d))$ est une application fermée en fonction de x et d. De plus, l'exercice suivant montre que d_k est une application fermée en fonction de x_k. Par conséquent, le théorème de Zangwill s'applique.

Exercice 4.5.1 *[]* Une application A qui associe un ensemble $A(x)$ à un point x est dite *fermée* si pour toute suite convergente $x_k \to \bar{x}$, tout point d'accumulation $\bar{y}$ de la suite associée $y_k \in A(x_k)$ appartient à $A(\bar{x})$, $\bar{y} \in A(\bar{x})$. Démontrez que si $\nabla f(x)$ est une fonction continue, l'application A définie comme l'ensemble des solutions optimale du programme linéaire $\min_y \nabla f(x)y : Ay = b, y \geq 0$ est une application fermée. □

Cet algorithme n'est plus beaucoup utilisé car sa vitesse de convergence est très lente. Cependant, il constitue un exemple d'extension de la théorie de la programmation linéaire. De plus, dans certaines applications particulières, notamment dans les problèmes de transport de grande taille, la simplicité et l'efficacité du calcul de la direction compensent sa lente convergence, et en font un algorithme acceptable, voire même l'algorithme de choix.

Exercice 4.5.2 *[Projections]* Considérons le problème de projeter l'origine sur l'enveloppe convexe de points de $\mathbb{R}^n$. On a un ensemble de points $P_1, P_2, \ldots, P_m$; l'enveloppe convexe est définie comme

$$\{x : x = \sum_{i=1}^{m} \alpha_i P_i, \sum_{i=1}^{m} \alpha_i = 1, \alpha \geq 0\}.$$

On peut formuler le problème comme suit :

$$\begin{aligned} \min_{x\in \mathbf{R}^n, \alpha\in \mathbf{R}^m} & \|x\|^2 \\ \text{s.à. } & x = \sum_{i=1}^m \alpha_i P_i = P\alpha \\ & \sum_{i=1}^m \alpha_i = 1 \\ & \alpha \geq 0, \end{aligned}$$

où les colonnes de la matrice P sont les points P_i.

a) En injectant l'expression pour x dans la fonction objectif, reformulez ce problème avec les seules variables α.

b) L'algorithme de Frank&Wolfe appliqué à cette dernière formulation devra calculer des solutions au problème

$$\begin{aligned} \min_{y\in \mathbf{R}^m} & \nabla f(\alpha_k) y \\ \text{s.à. } & \sum_{i=1}^m y_i = 1 \\ & y \geq 0; \end{aligned}$$

Obtenez une expression simple pour la solution de ce sous-problème.

c) Considérez le problème avec quatre points de $\mathbb{R}^2$ $P_1 = (0, 5)^t$, $P_2 = (4, 5)^t$, $P_3 = (4, 1)^t$ et $P_4 = (5, -2)^t$. Obtenez la solution de ce problème.

d) Pour le problème en c), effectuez trois itérations de l'algorithme de Frank&Wolfe ; utilisez comme point de départ celui des quatre points le plus proche de l'origine : $x_0 = \arg\min_{i=1,2,3,4} \|P_i\|$.

□

4.5.2 Algorithme du gradient réduit

Cet algorithme utilise la partition Base—Super-Base—Hors-Base introduite ci haut. À l'aide de cette partition, récrivons (4.1) comme

$$\begin{aligned} \min\ & \bar{f}(x_S, x_N) = f([B^{-1}b - B^{-1}(Nx_N + Sx_S), x_S, x_N]) \\ \text{s.à. } & B^{-1}(b - Nx_N - Sx_S) \geq 0 \\ & x_S > 0 \\ & x_N = 0. \end{aligned} \tag{4.6}$$

Ce dernier programme se nomme *programme réduit*, et tant que ses contraintes demeurent satisfaites, on peut lui appliquer tout algorithme de minimisation d'une fonction sans contrainte. Lorsque l'algorithme de minimisation sans contrainte provoque la violation des contraintes $B^{-1}(b - Nx_N) \geq 0$, il est temps de changer de partition. L'algorithme du gradient réduit utilise comme algorithme sans contrainte l'algorithme du gradient, et formalise cette description intuitive.

Gradient réduit

{ Données : une partition $[\![B \mid S \mid N]\!]$ de $I\!R^n$, une représentation }
{ de la matrice B^{-1} ainsi qu'une solution réalisable $[x_B, x_S, x_N]$. }
$d_N \leftarrow (-\nabla_N \bar{f}(x_S, x_N)^t)^+$
$d_S \leftarrow (-\nabla_S \bar{f}(x_S, x_N)^t)$
tantque ($\|d\| > 0$)
    $\theta_1 \leftarrow \max\{\theta : x_S + \theta d_S \geq 0\}$
    $\theta_2 \leftarrow \max\{\theta : B^{-1}(b - N(x_N + \theta d_N) - S(x_S + \theta d_S) \geq 0\}$
    $\theta_3 \leftarrow \arg \min_{0 \leq \theta \leq \min\{\theta_1, \theta_2\}} \bar{f}(x_S + \theta d_S, x_N + \theta d_N)$
    $x_N \leftarrow x_N + \theta_3 d_N$
    $x_S \leftarrow x_S + \theta_3 d_S$
    $x_B \leftarrow B^{-1}(b - N x_N - S x_S)$
    { Mise-à-jour de la partition }
    si ($d_N \neq 0$) **alors** { certaines composantes passent de N à S }
    si ($\theta_3 = \theta_1$) **alors** { certaines composantes passent de S à N }
    si ($\theta_3 = \theta_2$) **alors**
        { UNE composante passe de B à N }
        { UNE composante passe de N ou S à B }
        { La représentation de la matrice B^{-1} est mise à jour. }
    $d_N \leftarrow (-\nabla_N \bar{f}(x_S, x_N)^t)^+$
    $d_S \leftarrow (-\nabla_S \bar{f}(x_S, x_N)^t)$

Voici l'algorithme classique du gradient réduit. L'opérateur de projection $(-\nabla \bar{f}(x_S, x_N)^t)^+_N$ pour le calcul de d_N consiste à remplacer par 0 les composantes négatives de $\nabla \bar{f}_N$. La mise-à-jour de la partition dans le cas où une composante quitte B s'effectue en utilisant la même mécanique algébrique que l'algorithme du simplexe révisé

Algorithme 4.1: Gradient réduit.

On a pensé longtemps que l'algorithme du gradient réduit possédait la propriété de convergence globale. Puis, Wolfe a fourni ce contre-exemple :

$$\begin{aligned}&\min_{x,y,z} 4/3(x^2 - xy + y^2)^{3/4} - z \\ &\text{s.à. } z \leq 2, \quad x, y, z \geq 0.\end{aligned}$$

Pour cet exemple, on vérifie que l'algorithme, démarré en $(x_0, y_0, z_0)^t = (0, a, 0)^t$, où $0 \leq a \leq \sqrt{2}/4$, engendre une suite qui converge vers $(x_\infty, y_\infty, z_\infty)^t = (0, 0, (1 + \sqrt{2}/2)\sqrt{a})^t$, qui n'est ni la solution (exercice 4.3.3), ni un point stationnaire. C'est un exemple marquant des difficultés que causent la dégénérescence, lorsque la condition de stricte complémentarité n'est pas satisfaite.

Exercice 4.5.3 *[]* Vérifiez que l'exemple de Wolfe converge bel et bien vers le point décrit. Pour ce faire, vérifiez que l'algorithme engendre la suite

$$x_k = \begin{cases} (0, \alpha_k, \beta_k)^t & \text{si } k \text{ est impair} \\ (\alpha_k, 0, \beta_k)^t & \text{si } k \text{ est pair} \end{cases}$$

avec $\alpha_k = (\frac{1}{2})^k$ et $\beta_k = \frac{1}{2}\sum_{j=0}^{k-1}\sqrt{\frac{a}{2^j}}$. □

Il est possible de modifier l'algorithme pour assurer sa convergence globale, notamment en forçant $d_N = 0$ pour un certain nombre d'itérations consécutives. Cependant, une autre difficulté persiste, héritée de l'algorithme du simplexe : en présence de dégénérescence, l'algorithme peut rester emprisonné dans un cycle de partitions ne permettant aucun mouvement.

Exercice 4.5.4 *[GR]* Considérez à nouveau le problème

$$\begin{aligned}&\min_{x \in \boldsymbol{R}^3} \|x\|^2 \\ &\text{s.à. } x_1 + 2x_2 + 3x_3 = 1; \qquad x \geq 0.\end{aligned}$$

Appliquez l'algorithme du gradient réduit à partir du point de départ $x^0 = (1, 0, 0)^t$. □

4.5.3 Problèmes de complémentarité linéaire

L'expression des conditions d'optimalité peut se ramener sous une forme dite standard, qui ressemble à la forme standard pour la programmation linéaire :

$$\begin{aligned}Ax &= b \\ \pi A + s &= \nabla f(x^*) \\ x, s &\geq 0 \\ sx &= 0.\end{aligned}$$

Lorsque de plus, la fonction $f = \frac{1}{2}x^t Q x + cx$ est quadratique, $\nabla f(x) = xQ + c$ constitue une application affine, et les équations et inégalités exprimant les conditions d'optimalité peuvent se reformuler comme

$$\begin{aligned}w - Mz &= q \\ w &\geq 0 \\ z &\geq 0 \\ w^t z &= 0.\end{aligned} \qquad (CL)$$

Or, ce type de problème peut fort bien se résoudre à l'aide de technique de pivotage très semblables à l'algorithme du simplexe. Nous allons en présenter deux. Dans tous les cas, il s'agit de rebaptiser les variables w et z comme le vecteur x, et de considérer la partition $[x_B, x_N = 0]$. Par exemple, si l'on pose $x_B = q$, le problème de complémentarité s'exprime comme

$$\begin{aligned} [I, -M] \begin{pmatrix} x_B \\ x_N \end{pmatrix} &= q \\ x_N &= 0 \\ x_B^t x_N &= 0. \end{aligned}$$

Cette partition fournit la solution lorsque le vecteur q est positif. Autrement, il faut la modifier. Les calculs se simplifient par la mise-à-jour d'une représentation commode de la matrice $[I, -M]$. À une étape donnée, les algorithmes manipulent tous un tableau de la forme $[I, A]$ mais la sous-matrice A ne correspond plus à la matrice $-M$, et les composantes x_B et x_N ne sont plus identifiées aux variables w et z respectivement. Supposions que nous connaissions la partition $[\![B \mid N]\!]$ correspondant à la solution, et supposons de plus que les variables sont ordonnées de sorte que $B = \{1, 2, ...m^*\}$ et $N = \{m^* + 1, ...n\}$. Décomposons la matrice

$$M = \begin{pmatrix} M_{BB} & M_{BN} \\ M_{NB} & M_{NN} \end{pmatrix}$$

ainsi que la matrice

$$I = \begin{pmatrix} I_{BB} & 0 \\ 0 & I_{NN} \end{pmatrix}.$$

Multiplions maintenant le tableau $[I, -M]$ par la matrice

$$C = \begin{pmatrix} M_{BB}^{-1} & 0 \\ -M_{NB}M_{BB}^{-1} & I_{NN} \end{pmatrix}$$

pour obtenir :

$$C[I, -M] = \begin{pmatrix} M_{BB}^{-1} & 0 & I_{BB} & M_{BB}^{-1}M_{BN} \\ -M_{NB}M_{BB}^{-1} & I_{NN} & 0 & \bar{M}_{NN} \end{pmatrix}.$$

De même, notons $Cq = \bar{q} \geq 0$ de sorte que la solution est $z_B = \bar{q}_B$, $z_N = 0$, $w_B = 0$ et $w_N = \bar{q}_N$.

Les algorithmes de pivotage construisent ce tableau final par une suite d'opérations de pivotage du même type que dans l'algorithme du simplexe. Les choix de variables entrantes ou sortantes distinguent les algorithmes.

Pivot principal

Algorithme de Lemke

4.6 Algorithmes de projection

Nous avons vu que les points stationnaires étaient également des points fixes de l'application $Proj_E(x - \nabla f(x)^t)$. Si on a sous la main un point x pour lequel $x \neq Proj_E(x - \nabla f(x)^t)$, il est possible de construire une direction descendante $d = Proj_E(x - \nabla f(x)^t) - x$. L'inconvénient de cette approche vient de la nécessité du calcul de la projection. Si les contraintes sont simples, par exemple des bornes sur les variables, alors il est possible de calculer la projection efficacement, et l'algorithme présente un certain intérêt.

Exercice 4.6.1 *[Direction de projection]* Vérifiez que $d = Proj_E(x - \nabla f(x)^t) - x$ est une direction de descente, c'est-à-dire que $\nabla f(x) d < 0$. □

4.6.1 Exemple de calculs de projection

Si l'ensemble E est constitué de simples bornes $x \geq 0$, alors $Proj_E(x) = \begin{cases} 0 & \text{si } x \leq 0 \\ x & \text{si } x > 0 \end{cases}$. Nous développons maintenant une technique pour calculer la projection pour les contraintes

$$\sum_{i=1}^{n} x_i = 1 \quad x \geq 0.$$

Ceci constitue un exemple de résolution des conditions d'optimalité. Écrivons donc les conditions d'optimalité du programme qui définit la projection :

$$\begin{aligned} \min_x \ & \frac{1}{2}\|x - y\|^2 \\ \text{s.à. } & \sum_{i=1}^{n} x_i = 1 \\ & x \geq 0. \end{aligned} \tag{4.7}$$

Les conditions s'écrivent

$$\begin{aligned} (x - y) + e\lambda - \mu &= 0 \\ ex &= 1 \\ \mu x &= 0 \\ \mu &\geq 0 \end{aligned}$$

Maintenant, de la première équation, on tire que $x_i = y_i - \lambda + \mu_i$. Supposons que l'on connaisse λ. Alors, si $y_i - \lambda < 0$, $x_i = 0$ et $\mu_i = \lambda - y_i$. Autrement, $y_i - \lambda \geq 0$ et $x_i = y_i - \lambda$, $\mu_i = 0$. Nommons $x_i(\lambda) = \max(0, y_i - \lambda)$. Il suffit maintenant d'identifier une valeur de λ pour laquelle $ex(\lambda) = 1$. Or, chaque $x_i(\lambda)$ est une fonction linéaire par morceau, comportant deux morceaux qui se joignent au point $\lambda = y_i$, et non-croissante en fonction de λ. Donc, $e\lambda$ est une fonction non-croissante de λ linéaire par morceaux, dont les points de changement de morceaux sont les valeurs y_i. Il suffit donc de classer les points de changement de morceau y_i par ordre disons croissant $i_1, i_2, \ldots, i_n$, puis d'identifier le morceau tel que $ex(y_{i_j}) \geq 1 \geq ex(y_{i_{j+1}})$, et enfin d'effectuer une interpolation linéaire sur ce morceau.

Exercice 4.6.2 *[Projection]* Considérez un exemple dans $I\!R^4$. Soit $y = \frac{1}{4}(-1, -2, 2, 4)^t$. Calculez la projection décrite par le programme (4.7). □

Exercice 4.6.3 *[Programmes séparables]* La solution de la projection présentée ci-haut peut se généraliser à des programmes quadratiques dont la matrice est une matrice diagonale Δ telle que $\Delta_{ij} = 0$ si $i \neq j$. Considérez le problème

$$\begin{aligned} \min_x \; & \frac{1}{2} x^t \Delta x + cx \\ \text{s.à. } & \sum_{i=1}^n a_i x_i = 1 \\ & l_i \leq x_i \leq u_i. \end{aligned}$$

On peut se limiter à des valeurs $a_i \geq 0$; en effet, si certains $a_i < 0$, proposez un problème équivalent avec $a_i > 0$. (Indice : on peut remplacer x_i par $-x_i$) Imitez la démarche pour proposer une technique de solution ; décrivez les conditions d'optimalité ; obtenez une expression $x_i(\lambda)$; indiquez quels sont les points de changement de morceau ; vérifiez que la fonction $s(\lambda) \stackrel{\text{def}}{=} \sum_{i=1}^n a_i x_i(\lambda)$ est monotone. $\square$

4.7 Algorithmes de contraintes actives

Les conditions d'optimalité de K–K–T distinguent (ne serait-ce que par le terme de complémentarité) les contraintes actives des contraintes inactives. Si l'on connaissait l'ensemble des contraintes actives à la solution, on pourrait résoudre le problème à l'aide d'une méthode sans contrainte appliquée au problème $\phi(d_z) = f(x_0 + Zd_z)$.

L'idée de base de ces méthodes est la suivante. On démarre en un certain point x_0. Nommons W, l'ensemble des contraintes saturées en x_0, ensemble de travail. Nous espérons que l'ensemble de travail deviendra égal à l'ensemble des contraintes actives en x^*. à l'aide de W, nous définissons un sous-problème avec seulement des contraintes d'égalités. Nous appliquons à ce problème un algorithme d'optimisation sans contrainte. L'algorithme d'optimisation sans contrainte est interrompu dans deux circonstances :

- la progression de l'algorithme sans contrainte est empêchée par la rencontre d'une ou plusieurs contraintes ;
- une solution au sous-problème sans contrainte est atteinte.

Dans le premier cas, on doit ajouter des contraintes dans l'ensemble W (choisies parmi celles rencontrées par l'algorithme) alors que dans le second cas, on doit retirer des contraintes de l'ensemble W (choisies parmi celles dont les variables λ_i associées sont négatives).

Plus précisément, on distingue deux types d'itérations : des itérations de *restriction* et de *relaxation*. On parle de restriction lorsque l'on applique un algorithme avec seulement les contraintes d'égalité de W. On parle de relaxation lorsque l'on enlève des éléments de W.

Remarquons que ce schéma d'algorithme peut très bien être implanté à l'aide du schéma de l'algorithme 4.1. Dans ce dernier algorithme, on implante une étape de restriction en forçant $d_N = 0$; on rencontre une nouvelle contrainte et ajoute un ou des éléments à W lorsque $\theta_3 = \min(\theta_1, \theta_2)$ et on atteint une solution au sous-problème sans contrainte lorsque $\|d_S\| \approx 0$ et $\|d_B\| \approx 0$, dans quel cas on enlève une contrainte de W en enlevant un indice de N. On voit donc que les contraintes de travail correspondent en fait aux composante nulles du vecteur x. Une particularité de cette implantation vient du fait que lorsqu'une ou

```
Contraintes actives
{ Données : un point réalisable x un ensemble de travail W, }
{ une matrice Z telle que A_W Z = 0 ; }
{ deux constantes τ_0 et τ_1 ; }
tantque ( ¬ Optimal(x) )
    { Étape de restriction }
        y ← x
        tantque (condition de relaxation NON satisfaite )
            d ← Direction de descente dans ker A_W
            θ_max ← max_θ{A(y + θd) ≤ b}
            θ_adm ← pas admissible (définition 2.3.2)
            si (θ_max < θ_adm ) alors W ← W ∪ {i : [A_i y < b_i] ∧ [A_i(y + θ_max) = b_i]}
            y ← y + min{θ_max, θ_adm}
        x ← y
    { Étape de relaxation }
        W ← W réduit
```

Algorithme 4.2: Contraintes actives.

plusieurs composantes de B devient nulle, la mécanique de l'algorithme du simplexe nous impose d'en introduire une seule dans l'ensemble N. Une autre particularité est que le choix de l'algorithme restreint est celui de la pente la plus forte alors qu'il serait avantageux de calculer une direction de Newton modifiée (section 2.4) plutôt que d'utiliser $d_S = -\nabla \bar{f}_S$.

Le parallèle entre l'algorithme du gradient réduit et l'algorithme de contraintes actives soulève le problème de la convergence globale de ce type d'algorithme. En effet, si la condition de relaxation est trop permissive et la stratégie de réduction de l'ensemble W trop libérale, on peut retrouver un algorithme *équivalent* à l'algorithme 4.1, et donc permettre la possibilité de convergence vers un point non-stationnaire. Néanmoins, la possibilité d'utiliser la mécanique algébrique de l'algorithme du simplexe rends ce type d'algorithmes attrayant.

4.7.1 Un exemple d'algorithme globalement convergent

L'algorithme 4.3 présente une variante possible qui précise une condition de relaxation, un critère d'arrêt basé sur l'optimalité, une direction de Newton sur les contraintes actives et finalement une stratégie de relaxation. Si le problème de cyclage ne se présente pas, alors les points d'accumulations de cette variante sont des points stationnaires.

Condition de relaxation

Parmi les différentes conditions de relaxation proposées dans la littérature, nous avons retenu la plus simple : aussitôt qu'un pas admissible (au sens de la définition 2.3.2) a pu être utilisé, on se permet d'enlever des composantes de W, c'est-à-dire de permettre à des composantes nulles de devenir positives.

Critère d'arrêt

La norme du gradient réduit sert de critère d'arrêt.

Direction de descente

Puisque la réduction rendue possible par la base B transforme le problème original en un problème sans contrainte, et puisque pour ce nouveau problème, les variables S sont libres alors que les variables N sont fixées à zéro dans les étapes de restriction, nous utilisons la direction de Newton dans les composantes S. En fait, il faut utiliser une direction de Newton modifiée (voir la section 2.4) pour s'assurer d'avoir une direction de descente.

Stratégie de relaxation

Comme l'exemple de Wolfe l'illustre, il faut se montrer prudent quant à la stratégie de relaxation. Nous avons retenu la plus simple et la plus usuelle, c'est-à-dire celle de permettre à la variable ayant la composante de $\nabla_N \bar{f}(x_S, x_N)$ la plus négative de devenir positive.

Variantes

En fait, les variantes spécifiques à cet algorithme concernent surtout les aspects reliés à la relaxation : les variantes de la méthode de Newton et les tests d'arrêt ayant déjà été discutées au chapitre 2.

Parmi d'autres conditions de relaxation, mentionnons l'utilisation d'une suite η_k convergeant vers zéro qui sert de critère d'arrêt pour la boucle de restrictions. Avec cette stratégie, on tente de réduire $\|d_S\|$ en deçà du seuil η_k à l'itération k.

Pour ce qui est des stratégies, il est possible de permettre à plusieurs composantes de l'ensemble N de devenir positives. Une possibilité permet aux indices atteignant un certain pourcentage fixé du minimum de $\nabla_N \bar{f}(x_S, x_N)$ d'être considérés pour devenir positifs. Une autre stratégie consiste à utiliser la direction de projection (voir section 4.6) pour déterminer les composantes pouvant être augmentées.

4.8 Résumé

Les problèmes avec contraintes linéaires d'égalité se réduisent aisément à des problèmes sans contrainte en utilisant les propriétés algébriques des contraintes linéaires. Les algorithmes d'optimisation sans contrainte s'adaptent aisément à ce cas.

En présence de contraintes d'inégalité, un phénomène combinatoire se manifeste : identifier lesquelles des inégalités seront saturées à la solution (ou en un point candidat à la solution). Si l'on connaît les contraintes saturées, aussi nommées contraintes actives, le problème se ramène à un problème avec contraintes linéaires d'égalité. Certaines analyses manipulent cet aspect combinatoire de manière explicite (généralisations de la programmation linéaire, conditions de K–K–T, algorithmes de contraintes actives) alors que d'autres dissimulent la difficultés (condition 4.3 et algorithme de Frank&Wolfe, conditions d'optimalité et algorithmes de projection).

Gradient réduit amélioré

{ Données : une partition $[\![B \mid S \mid N]\!]$ de $I\!R^n$, une représentation }
{ de la matrice B^{-1} ainsi qu'une solution réalisable $[x_B, x_S, x_N]$. }
$d_N \leftarrow (-\nabla_N \bar{f}(x_S, x_N)^t)^+$
$d_S \leftarrow -(\nabla^2_{SS} \bar{f}(x_S, x_N))^{-1} \nabla_S \bar{f}(x_S, x_N)^t$
$Pas\_admissible \leftarrow$ **faux**
tantque ($\|d\| > 0$)
- { Restriction et/ou relaxation... }
  - **si** ($\neg$ $Pas\_admissible$) **alors** $d_N \leftarrow 0$
  - **sinon**
    - { Permettre à la composante la plus grande de d_N de quitter N. }
    - $i \leftarrow \arg\min_j \{d_{Nj}\}$
    - $d_N \leftarrow d_{Ni} e_i$
    - $Pas\_admissible \leftarrow$ **faux**
- { Calcul du pas }
  - $\theta_1 \leftarrow \max\{\theta : x_S + \theta d_S \geq 0\}$
  - $\theta_2 \leftarrow \max\{\theta : B^{-1}(b - N(x_N + \theta d_N) - S(x_S + \theta d_S) \geq 0\}$
  - $\theta_{adm} \leftarrow$ pas admissible (définition 2.3.2)
  - $\theta_3 \leftarrow \min\{\theta_1, \theta_2, \theta_{adm}\}$
  - **si** ($\theta_{adm} = \theta_3$) **alors** $Pas\_admissible \leftarrow$ **vrai**
- { Mise-à-jour de x }
  - $x_N \leftarrow x_N + \theta_3 d_N$
  - $x_S \leftarrow x_S + \theta_3 d_S$
  - $x_B \leftarrow B^{-1}(b - Nx_N - Sx_S)$
- { Mise-à-jour de la partition }
  - **si** ($d_N \neq 0$) **alors** { certaines composantes passent de N à S }
  - **si** ($\theta_3 = \theta_1$) **alors** { certaines composantes passent de S à N }
  - **si** ($\theta_3 = \theta_2$) **alors**
    - { UNE composante passe de B à N PV }
    - { UNE composante passe de N ou S à B PV }
    - { La représentation de la matrice B^{-1} est mise à jour. }
- $d_N \leftarrow (-\nabla_N \bar{f}(x_S, x_N)^t)^+$
- $d_S \leftarrow -(\nabla^2_{SS} \bar{f}(x_S, x_N))^{-1} \nabla_S \bar{f}(x_S, x_N)^t$

Algorithme 4.3: Gradient réduit amélioré.

Les algorithmes les plus répandus utilisent la mécanique algébrique du simplexe pour implanter des variantes de la méthode de Newton se restreignant sur un ensemble de contraintes de travail qui, on l'espère, se confondra avec l'ensemble des contraintes actives. La convergence globale de tels algorithmes est délicate, mais des implantations soignées donnent depuis longtemps d'excellents résultats.

4.9 Extensions et références

Chapitre 5

Optimisation différentiable avec contraintes d'égalité

Sujets du chapitre.

- Analyse des conditions d'optimalité (équations ou inéquations satisfaites pour toute solution du problème d'optimisation).
- Hypothèses sur les contraintes.
- Déduction d'une famille d'algorithmes itératifs globalement convergents à partir de la violation des conditions d'optimalité.
- Présentation de l'algorithme de pénalité simple.
- Présentation d'un algorithme de pénalité augmentée.

Introduction

Les problèmes d'optimisation différentiable avec contraintes d'égalité également différentiables constituent un sujet classique, qui a été bien étudié par Lagrange, entre autres. Nous entamerons des sujets plus *modernes* au prochain chapitre.

Comme précédemment, nous utiliserons le concept de DED pour identifier des équations à satisfaire (conditions d'optimalité) pour les solutions locales du problème

$$\begin{aligned} &\min_{x \in I\!R^n} f(x) \\ &\text{sujet à } g(x) = 0, \end{aligned} \tag{5.1}$$

où $g : I\!R^n \to I\!R^m$. g est un vecteur *colonne* de fonctions. Sauf mention contraire, les vecteurs sont toujours des vecteurs colonne. Les fonctions f et g sont différentiables, et en général, supposées de classe $\mathcal{C}^3(I\!R^n)$.

Ensuite, en perturbant les conditions d'optimalité retenues, nous déduirons un algorithme de pénalité. Nous étudierons plusieurs aspects de l'algorithme de pénalité quadratique, et en survolerons d'autres variantes.

5.1 Conditions d'optimalité

Avant d'introduire les conditions d'optimalité, nous devons définir quelques concepts.

Tout d'abord, l'ensemble des points x qui satisfont à la contrainte $g(x) = 0$ est nommé *ensemble réalisable* du problème. Nous devrons aussi définir le concept de *direction réalisable*. Soit x tel que $g(x) = 0$. Considérons une suite y_k, avec $g(y_k) = 0$ telle que $y_k \to x$. Considérons maintenant les vecteurs $d_k = \frac{y_k - x}{\|y_k - x\|}$. Tout point d'accumulation de la suite d_k est nommé *direction réalisable* en x pour les contraintes g. L'ensemble de ces directions, en fait, constitue le *plan tangent* aux contraintes en x.

5.1.1 Conditions de premier ordre

Pour obtenir des conditions d'optimalité de premier ordre en un point x^*, nous devons nous assurer qu'aucune direction réalisable en x^* n'appartient au $DED(x^*)$. Le $DED(x^*)$ est bien défini comme $\{d : \nabla f(x^*)d < 0\}$, et donc l'obtention de conditions d'optimalité (du premier ordre) passe par une caractérisation des directions réalisables.

Directions réalisables

Soit une suite $d_k = \frac{y_k - x}{\|y_k - x\|}$ convergeant vers la direction réalisable $\bar{d}$ en un point x. Alors, on a que pour chacune des composantes g_i de l'application g, le développement de Taylor limité nous donne

$$g_i(y_k) = g_i(x) + \varphi_k \nabla g_i(x) d_k + \mathcal{O}(\varphi_k),$$

où $\varphi_k = \|y_k - x\|$. Puisque $g_i(y_k) = g_i(x) = 0$, il reste

$$\nabla g_i(x) d_k \sim \mathcal{O}(\varphi_k),$$

et en passant à la limite,

$$\nabla g_i(x)\bar{d} = 0,$$

et ce pour toutes les composantes de l'application g, ce qui s'écrit

$$\nabla g(x)\bar{d} = 0,$$

où ∇g représente le Jacobien de l'application g, c'est-à-dire une matrice dont les lignes sont $\nabla g_i(x)$. Donc, toute direction réalisable d en x satisfait à la relation $\nabla g(x)d = 0$.

Pour aller plus loin, il faut maintenant introduire une hypothèse sur les contraintes g, à savoir une hypothèse de *qualification des contraintes* g. Cette hypothèse stipule que x^* est un point régulier du système $g(x) = 0$, c'est-à-dire, la matrice $\nabla g(x^*)$ est une matrice de plein rang-lignes, $m \leq n$. Cette hypothèse permet de montrer que toute direction satisfaisant $\nabla g(x)d = 0$ est une direction réalisable, complétant ainsi la caractérisation des directions réalisables. Remarquons que cette qualification dépends des fonctions g et non pas de l'ensemble des points réalisables, qui pourrait être exprimé à l'aide d'autres contraintes.

Par exemple, les contraintes $g_1(x,y) = x^2 - y$ et $g_2(x,y) = x^2 + y$ définissent un seul point réalisable, soit l'origine. Pourtant, un mauvais hasard fait que tout l'axe des x est tangent à chacune des deux contraintes alors que le plan tangent tel que défini plus haut se réduit à l'origine, un sous-espace de dimension nulle. L'origine n'est pas un point régulier de ces contraintes. Cependant, les contraintes $\bar{g}(x,y) = x + y$ et $\bar{g}_2(x,y) = x - y$ définissent le même point réalisable, qui est régulier pour les contraintes, cette fois-ci.

Théorème 5.1.1 *Soit x^* un point régulier des contraintes $g(x^*) = 0$. Alors, d est une direction réalisable en x^* si et seulement si $\nabla g(x^*)d = 0$.*

Preuve Cette preuve constitue un exercice d'application du théorème des fonctions implicites. Le même schéma de preuve sera utilisé pour analyser l'algorithme des pénalités présenté plus loin.

Il est clair que si d est une direction réalisable, alors $\nabla g(x^*)d = 0$. En fait, x^* n'a même pas besoin d'être régulier pour cette moitié du théorème.

En revanche, supposons que x^* est un point régulier qui satisfait $\nabla g(x^*)d = 0$. Il faut montrer que, quel que soit d dans le plan tangent aux contraintes en x^*, il existe une courbe réalisable $x(t)$ telle que $x(0) = x^*$ et que, $x'(0) = d$. Pour construire une telle courbe, considérons les équations

$$g(x^* + td + u(t)\nabla g(x^*)) = 0,$$

où u est inconnue, et t est considéré fixé. Ceci constitue un système régulier d'équations paramétrisé par t, avec $u(0) = 0$. On peut exprimer la matrice Jacobienne de ce système en u comme $\nabla g(x^*)\nabla g(x^*)^t$, qui est inversible puisque $\nabla g(x^*)$ est de plein rang, x^* étant un point régulier des contraintes. Les hypothèses du théorème des fonctions implicites sont donc satisfaites, et ce système possède une solution continûment différentiable $u(t)$; lorsque la fonction g est de classe $\mathcal{C}^p$, la courbe u l'est aussi. La courbe $x(t) = x^* + td + u(t)\nabla g(x^*)$ est par construction une courbe réalisable. Dérivons le système $g(x^* + td + u(t)\nabla g(x^*)) = 0$ par rapport à t, et évaluons le tout en $t = 0$:

$$0 = \frac{d}{dt} g(x(t))\Big|_{t=0} = \nabla g d + \nabla g(x^*)\nabla g(x^*)^t u'(0).$$

Nous avions choisi d pour que $\nabla g d = 0$, et comme $\nabla g(x^*)\nabla g(x^*)^t$ est inversible, $u'(0) = 0$, d'où $x'(0) = d$, tel que voulu. $\square$

On voit donc que les directions réalisables sont celles situées dans le noyau de la matrice $\nabla g(x^*)$. Considérons une base du noyau de $\nabla g(x^*)$ formée des vecteurs $b_1, b_2, \ldots, b_{m-n}$, et une matrice Z^* dont les colonnes sont les vecteurs b_i. Alors, toute direction réalisable d peut s'exprimer comme combinaison linéaire de la base du noyau de $\nabla g(x^*)$, $d = Z^* d_z$ où $d_z \in I\!R^{m-n}$ constitue l'expression de d dans la base du noyau.

Condition du premier ordre

En utilisant la caractérisation des directions réalisables, on peut rephraser la condition d'optimalité comme suit : pour toute direction dans le noyau de la matrice $\nabla g(x^*)$, $\nabla f(x^*)d = 0$. Exprimons $d = Z^* d_z$ pour écrire $\nabla f(x^*)(Z^* d_z)$, quel que soit $d_z \in I\!R^{m-n}$. Ceci entraîne que $\nabla f(x^*)Z^* = 0$; cette dernière quantité se nomme *gradient réduit* de f.

Une autre manière de voir ces conditions consiste à exploiter le fait que $\nabla f(x^*)$ est un vecteur orthogonal au noyau de la matrice $\nabla g(x^*)$. Un exercice simple d'algèbre linéaire permet de conclure que ceci est équivalent à l'existence d'un vecteur $\lambda^* \in I\!R^m$ satisfaisant à l'égalité

$$\nabla f(x^*) = \lambda^* \nabla g(x^*).$$

Bon, tout ceci nous donne que $d \in DED(x^*)$ est caractérisé par $\nabla f(x^*)d < 0$ et d est une direction réalisable est caractérisé par $\nabla g(x^*)d = 0$. En observant que si d est réalisable, $-d$ l'est aussi, on retrouve que d ou $-d$ n'appartiennent pas à $DED(x^*)$ seulement si $\nabla f(x^*)d = 0$, ce qui permet d'écrire la condition d'optimalité suivante : si x^* est un minimum local de f dans l'ensemble défini par $g(x) = 0$, alors

$$\nabla g(x^*)d = 0 \Longleftrightarrow \nabla f(x^*)d = 0.$$

Exercice 5.1.1 *[Dualité]* Soit une matrice $n \times m, A$ de rang $m < n$; alors, un vecteur $v \in I\!R^n$ est orthogonal au noyau de A si et seulement s'il existe un vecteur $\lambda \in I\!R^m$ tel que $v = \lambda A$. $\square$

Résumons la précédente discussion dans le

Théorème 5.1.2 *Soit x^* un minimum local de la fonction f sous les contraintes $g(x) = 0$ tel que $\nabla g(x^*)$ est une matrice de plein rang-ligne ; alors, il existe un vecteur $\lambda^* \in I\!R^m$ tel que $\nabla f(x^*) = \lambda^* \nabla g(x^*)$; ceci est équivalent à la condition $Z^{*t}\nabla f(x^*) = 0$, où les colonnes de Z^* forment une base du noyau de la matrice $\nabla g(x^*)$.*

5.1.2 Conditions de second ordre

Nous venons de voir que les conditions de premier ordre expriment le fait que les directions réalisables, donc tangentes à la variété des points satisfaisant aux contraintes $g(x) = 0$, ne sont pas des directions de descente stricte.

Cette propriété n'est pas assez forte pour discriminer des minima locaux, points de selles ou maxima. Revenons à l'exemple de la fonction $f(x, y) = (x^2 - y)(x^2 - 2y)$. Nous avons déjà vérifié que l'origine est un minimum local de cette fonction restreint à toute droite passant par l'origine. En d'autres termes,

$$0 = \min_{ax+by=0} (x^2 - y)(x^2 - 2y),$$

et ce, quelles que soient les valeurs de a et b. Cependant,

$$0 = \max_{x^2-1.5y=0} (x^2 - y)(x^2 - 2y).$$

Pourtant, dans les deux cas, pour le cas particulier où $a = 0$ et $b = -1.5$, le gradient de l'unique contrainte est $\nabla g(x^*, y^*) = \begin{pmatrix} 0 \\ -1.5 \end{pmatrix}$, alors que le gradient de l'objectif est $\nabla f(x^*, y^*) = 0 = 0\nabla g(x^*, y^*)$. Pour le problème sans contrainte, l'origine est un point de selle alors que pour le problème contraint, dépendamment de la contrainte, l'origine est soit un minimum local ou encore un maximum local. Cet exemple illustre de façon convainquante qu'il faut tenir compte de la courbure des contraintes (dérivées d'ordre supérieur) pour caractériser les minima et les maxima locaux contraints.

Il est remarquable que les pondérations relatives des dérivées secondes des contraintes et de la fonction objectif puissent toutes être déduites par l'analyse du *Lagrangien* associé au problème de minimisation avec contraintes. Rappelons que celui-ci est défini comme

$$L(x, \lambda) \stackrel{\text{def}}{=} f(x) + \lambda g(x) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x).$$

Les conditions d'optimalité du premier ordre que nous avons construite plus haut s'expriment de manière concise à l'aide de la fonction L :

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

En effet, $\nabla_x L(x^*, \lambda^*) = \nabla f(x^*) + \lambda^* \nabla g(x^*)$ qui s'annule correspond à la condition de premier ordre alors que $\nabla_\lambda L(x^*, \lambda^*) = g(x^*)$ qui s'annule correspond simplement à la satisfaction des contraintes. Cette formulation est à la source du nom donné aux variables λ, *multiplicateurs de Lagrange* pour le problème (5.1).

La condition de second ordre s'exprime simplement comme suit :

$$\nabla^2_{xx} L(x^*, \lambda^*) = \nabla^2 f(x^*) + \lambda^* \nabla g^2(x^*)$$

est une matrice (semi-)définie positive sur l'espace tangent $\{d : \nabla g(x^*) d = 0\}$, ce qui s'exprime à l'aide de la matrice Z^* introduite plus haut :

$$Z^{*t} \left(\nabla^2_{xx} L(x^*, \lambda^*) \right) Z^* = Z^{*t} \left(\nabla^2 f(x^*) + \lambda^* \nabla g^2(x^*) \right) Z^*$$

est une matrice (semi-)définie positive. Cette dernière matrice porte le nom de *hessien réduit* du Lagrangien.

Pour alléger la notation, nous utiliserons fréquemment la formulation suivante dans l'avenir (tout comme ci-haut!) :

$$\lambda \nabla^2 g(x) \stackrel{\text{def}}{=} \left(\sum_{i=1}^{m} \lambda_i \nabla^2 g_i(x) \right).$$

Grâce à cette formulation, nous pouvons énoncer le

Théorème 5.1.3

1. [Nécessaire] *Si x^*, point régulier des contraintes $g(x) = 0$, est un minimum local du programme (5.1), alors il existe un vecteur $\lambda^* \in I\!R^m$ tel que $\nabla L(x^*, \lambda^*) = 0$, et de plus $d^t \nabla^2_{xx} L(x^*, \lambda^*) d \geq 0$ pour tout d tel que $\nabla g(x^*) d = 0$.*
2. [Suffisante] *S'il existe un vecteur $\lambda^* \in I\!R^m$ tel que $\nabla L(x^*, \lambda^*) = 0$, et de plus $d^t \nabla^2_{xx} L(x^*, \lambda^*) d > 0$ pour tout d tel que $\nabla g(x^*) d = 0$, alors x^* est un minimum local pour le programme (5.1).*

Preuve

– Nécessité.
Considérons une courbe différentiable $x(t)$ de classe $\mathcal{C}^2$ telle que $g(x(t)) = 0$ près de $x(0) = x^*$; pour construire une telle courbe, on procède comme au théorème 5.1.1. Alors, si x^* est un minimum local pour le programme (5.1), on a nécessairement $f(x^*) = f(x(0)) = min_t f(x(t))$, et donc

$$\frac{d^2}{dt^2} f(x(t))\Big|_{t=0} = x'(0)^t \nabla^2 f(x^*) x'(0) + \nabla f(x^*) x''(0) \geq 0.$$

En dérivant deux fois par rapport à t la relation $\lambda g(x(t))$ on obtient :

$$x'(0)^t \left(\sum_{i=1}^m \lambda_i \nabla^2 g_i(x^*) \right) x'(0) + \lambda \nabla g(x^*) x''(0) = 0.$$

La conclusion suit en additionnant les deux dernières relations, et en rappelant que $x'(0)$ est arbitraire.

– Suffisance.
Reprenons les directions $d_k = y_k - x^*$ utilisées plus haut et procédons par contradiction en supposant que x^* n'est pas un minimum local de (5.1) bien qu'il satisfasse aux conditions suffisantes d'optimalité.
Par la caractérisation de directions réalisables, nous savons que $\nabla g(x^*) \bar{d} = 0$. Utilisons le théorème de Taylor pour écrire

$$0 = g_i(y_k) = g_i(x^*) + \varphi_k \nabla g_i(x^*) d_k + \frac{\varphi_k^2}{2} d_k^t \nabla^2 g_i(\eta_i) d_k,$$

et, si y_k est choisi pour que $f(y_k) < f(x^*)$,

$$0 \geq f(y_k) - f(x^*) = \varphi_k \nabla f(x^*) d_k + \frac{\varphi_k^2}{2} d_k^t \nabla^2 f(\eta_0) d_k,$$

où les points η_i sont des points intermédiaires entre y_k et x^*. Maintenant, en multipliant l'avant-dernière relation par λ_i^* et en les ajoutant à la dernière relation, on obtient

$$0 \geq \frac{\varphi_k^2}{2} d_k^t \left(\nabla^2 f(\eta_0) + \sum_{i=1}^m \lambda_i^* \nabla^2 g_i(\eta_i) \right) d_k,$$

ce qui fournit la contradiction désirée lorsque $k \to \infty$.

□

Remarquons que la qualification des contraintes, l'hypothèse que x^* est un point régulier des contraintes, n'est requise que pour établir les conditions nécessaires.

5.2 Construction d'une famille d'algorithmes de pénalité

Contentons-nous pour l'instant de produire un algorithme qui calcule des points stationnaires pour le programme (5.1), c'est-à-dire des points qui satisfont aux conditions d'optimalité de premier ordre : il existe un vecteur $\lambda^* \in I\!R^m$ tel que

$$\begin{aligned} \nabla f(x^*) + \lambda^* \nabla g(x^*) &= 0 \\ g(x^*) &= 0. \end{aligned} \tag{5.2}$$

5.2.1 Solution des conditions de premier ordre

Une approche directe consiste bien sûr à tenter de résoudre le système d'équations (5.2). Par exemple, on pourrait utiliser la méthode de Newton, et espérer obtenir des résultats de convergence *locale* quadratique. Cependant, comme nous l'avons déjà discuté, le voisinage dans lequel la méthode de Newton converge est souvent assez petit. Cette constatation a motivé le développement d'une famille d'algorithmes de descente et les modifications de l'algorithme de Newton. Pour l'optimisation sans contrainte, la modification s'appuie sur le fait que la fonction objectif décroît d'une itération à l'autre (algorithme de descente). Une telle fonction, dans un contexte plus général, est nommée *fonction de mérite*.

Bien qu'il soit possible de construire des fonctions de mérite pour le problème (5.2), souvent ces fonctions de mérite ne sont pas différentiables, et nous en reportons l'étude à plus tard, lorsque nous aborderons l'optimisation de fonctions non-différentiables.

5.2.2 Perturbation des conditions d'optimalité

Dans l'équation (5.2), remplaçons le 0 de la seconde équation par l'approximation $\rho\lambda$, où ρ est un paramètre que nous laisserons tendre vers zéro. Le but de cette manœuvre est de calculer des solutions $x(\rho)$ dépendant du paramètre ρ en espérant pouvoir montrer que $\lim_{\rho \searrow 0} x(\rho) = x^*$. Cette manipulation permet d'écrire

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) &= \rho \lambda^t, \end{aligned}$$

d'où l'on tire que $\lambda = \frac{g(x)^t}{\rho}$, ce qui permet de réduire ces deux équations à une seule :

$$P(x, \rho) \stackrel{\text{def}}{=} \nabla f(x) + \frac{g(x)^t}{\rho} \nabla g(x) = 0,$$

qui sont précisément les conditions d'optimalité du problème

$$\min_{x \in R^n} p(x, \rho) \stackrel{\text{def}}{=} f(x) + \frac{\|g(x)\|^2}{2\rho}.$$

Ce dernier problème est nommé *pénalisation quadratique* du programme (5.1). Comme nous le verrons plus tard, d'autres perturbations des équations (5.2) conduisent à d'autres pénalisations. Dans ce qui suit, nous allons étudier le comportement de la famille de solutions (approximatives) $x(\rho)$ des problèmes pénalisés lorsque ρ s'approche de zéro, et montrer que sous des hypothèses assez faibles, les points d'accumulation de cette famille sont des points stationnaires pour le programme (5.1). Complétons la notation telle que $\nabla_x p = P$ en définissant $\nabla^2_{xx} p = \nabla_x P = H$:

$$H(x,\rho) \stackrel{\text{def}}{=} \nabla^2 f(x) + \left(\sum_{i=1}^m \frac{g_i(x)}{\rho} \nabla^2 g_i(x)\right) + \frac{\nabla g(x)^t \nabla g(x)}{\rho}.$$

Les *algorithmes de pénalité* consistent à définir une suite $\{\rho_k\}$ telle que $\rho_k \searrow 0$ et à calculer une suite $x(\rho_k)$ de minima approximatifs de $p(x, \rho_k)$. L'analyse de ces algorithmes consiste à montrer que $\lim_{k_i \to \infty} x(\rho_{k_i}) = x^*$, un point stationnaire de (5.1), à qualifier la vitesse de la convergence, et à quantifier la quantité de calculs requise pour identifier $x(\rho_{k+1})$ à partir de $x(\rho_k)$.

5.2.3 Limites de la famille de méthode

Avant d'aller plus loin, certaines mises-en-garde sont nécessaires. Une première, fondamentale, vient du fait que le problème $\min p(x,\rho)$ peut être non-borné inférieurement, et ce pour toute valeur positive de ρ. Une seconde vient du fait que sans hypothèse additionnelle, on ne peut résoudre de système d'équations par une méthode de descente.

Pénalisations non-bornées

Considérons le problème trivial suivant : $\min_{x=0} -x^4$. Alors, $p(x,\rho) = \frac{1}{2\rho} x^2 - x^4$ n'est pas bornée inférieurement. Comme nous le verrons ci-après, rien n'assure que les minima *locaux* de $p(x,\rho)$ convergent vers des points réalisables, et les seuls théorèmes de convergence vraiment forts supposent la solution globale des sous-problèmes sans contrainte. L'exemple trivial montre qu'il est parfois irréaliste de supposer des solutions globales aux sous-problèmes.

Difficultés de solution des problèmes pénalisés

Le simple fait de calculer un point réalisable, qui satisfait $g(x) = 0$, constitue un problème difficile. En effet, si $m = n$, il s'agit de solutionner un système d'équations, et nous avons mentionné précédemment que des hypothèses sur g sont requises pour assurer que la minimisation *locale* de l'objectif $\|g(x)\|^2$ conduit bel et bien à la solution de l'équation. Or, la pénalité quadratique consiste en une somme pondérée de l'objectif f et du “pseudo-objectif” $\|g(x)\|^2$. Les mêmes hypothèses (monotonie de g) sont donc requises pour assurer que les points minima sont réalisables. Cette première difficulté est incontournable avec des méthodes de descente. Remarquons que la convergence locale de telles méthodes n'est pas mise en cause, et certains auteurs affirment que si l'algorithme ne converge pas vers un point réalisable, c'est que le modèle et l'estimé initial ne sont pas adéquats.

5.2.4 Propriétés de convergence globales de la pénalisation

Comme nous l'avons dit plus haut, sans hypothèse additionnelle sur les fonctions des contraintes g, rien ne garantit que l'ensemble $x(\rho)$ possède un point limite réalisable. Par conséquent, nous nous contentons de montrer que tout point limite réalisable est stationnaire. En fait, si les $x(\rho)$ sont des minima locaux de l'objectif pénalisé, alors les points limites de la suite $x(\rho_k)$ satisfont aux conditions nécessaires de second ordre.

Si l'on suppose que les minima *globaux* $x(\rho_k)$ des sous-problèmes sont calculés, alors on peut montrer que tout point d'accumulation de la suite $x(\rho_k)$ est un minimum global pour le programme (5.1). Cette preuve est laissée en exercice. Cependant, comme l'indique la remarque plus haut, rien ne garantit l'existence de tels points stationnaires, même si le domaine réalisable est compact (un seul point dans l'exemple trivial).

Exercice 5.2.1 *[Minimisation globale]* Supposez que $x(\rho_k)$ soit un minimum global de $\min_{x\in R^n} f(x) + \frac{\|g(x)\|^2}{2\rho_k}$ pour tout k. Alors, tout point d'accumulation de la suite $x(\rho_k)$ est un minimum global pour le programme (5.1). Indices : (i) la suite des $f(x(\rho_k)) + \frac{\|g(x(\rho_k))\|^2}{2\rho_k}$ est non-décroissante ; (ii) la suite des $\frac{\|g(x(\rho_k))\|^2}{2\rho_k}$ est non-croissante ; (iii) la suite des $f(x(\rho_k))$ est non-décroissante ; (iv) la relation suivante est vérifiée : $f(x^*) \ge f(x(\rho_k)) + \frac{\|g(x(\rho_k))\|^2}{2\rho_k} \ge f(x(\rho_k))$. $\square$

Théorème 5.2.1 *Supposons qu'un point d'accumulation $\bar{x}$ de la suite $\{x(\rho_k)\}$ soit un point régulier des contraintes. Supposons de plus que les points $x(\rho_k)$ sont des minima locaux des sous-problèmes pénalisés. Alors, $\bar{x}$ satisfait aux conditions nécessaires pour un minimum local du problème (5.1).*

Preuve Vérifions d'abord que les propriétés de continuité de f et g assurent que $\bar{x}$ est au moins un point stationnaire. Les points $x(\rho_k)$ satisfont au système d'équations

$$\nabla f(x) + \frac{g(x)^t}{\rho_k}\nabla g(x) = 0,$$

et $\nabla f(x)$ s'approche de $\nabla f(\bar{x})$, un vecteur borné tandis que $\nabla g(x)$ s'approche de $\nabla g(\bar{x})$), une matrice régulière. Par conséquent, il faut que $\frac{g(x)}{\rho_k}$ s'approche d'un vecteur *fini*, sans quoi $\nabla f(x) + \frac{g(x)^t}{\rho_k}\nabla g(x)$ ne pourrait pas s'approcher de zéro. Ceci entraîne que $g(\bar{x}) = 0$, donc que $\bar{x}$ est un point stationnaire.

Pour vérifier que $\bar{x}$ satisfait aux conditions nécessaires pour un minimum local, écrivons

$$H(x(\rho_k), \rho_k) \stackrel{\text{def}}{=} \nabla^2 f(x(\rho_k)) + \left(\sum_{i=1}^m \frac{g_i(x(\rho_k))}{\rho_k}\nabla^2 g_i(x(\rho_k))\right) + \frac{\nabla g(x(\rho_k))^t \nabla g(x(\rho_k))}{\rho_k},$$

et considérons une suite d_k convergeant vers un vecteur $\bar{d}$ tels que $\nabla g(x(\rho_k))d_k = 0$ et $\nabla g(\bar{x})\bar{d} = 0$. Encore par la continuité des fonctions impliquées, $d_k^t H(x(\rho_k))d_k$ qui est non-négative pour chaque k ($x(\rho_k)$ est supposé un minimum local), converge vers une quantité non-négative, et donc $\bar{x}$ satisfait aux conditions nécessaires puisque pour un $\bar{d}$ quelconque dans le plan tangent, le hessien du lagrangien possède une dérivée directionnelle non-négative. $\square$

5.2.5 Propriétés numériques de la pénalisation

L'attrait des méthodes de pénalisation, consistant à remplacer la solution d'un problème avec contraintes par une suite de sous-problèmes sans contrainte, a diminué vers 1970. En effet, l'analyse numérique des sous-problèmes de la suite effectuée par Murray montre que lorsque ρ_k se rapproche de zéro, le sous-problème sans contrainte devient *très mal conditionné*, et donc difficile à résoudre. Ceci se reflète 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. Exprimons donc le Hessien d'un problème pénalisé :

$$H(x,\rho) = \nabla^2 f(x) + \left(\sum_{i=1}^{m} \frac{g_i(x)}{\rho}\nabla^2 g_i(x)\right) + \frac{\nabla g(x)^t \nabla g(x)}{\rho}.$$

Les deux premiers termes de ce Hessien se rapprochent (en autant que x se rapproche de x^*) de $\nabla^2_{xx}L(x^*,\lambda^*)$, que l'on peut supposer bien conditionnée ; si $\nabla^2_{xx}L(x^*,\lambda^*)$ est mal conditionnée, alors le problème est difficile, et le mauvais conditionnement ne provient pas de la méthode de pénalisation. Par conséquent, les valeurs propres extrêmes de $\nabla^2_{xx}L(x^*,\lambda^*)$ sont supposées finies, et éloignées de zéro. Cependant, la matrice $\frac{\nabla g(x)^t \nabla g(x)}{\rho}$ possède m valeurs propres qui divergent lorsque ρ s'approche de zéro. Donc, $H(x,\rho)$ possède $n-m$ valeurs propres bornées, et m valeurs propres divergentes. À moins que $m=0$ ou encore que $m=n$, le rapport des valeurs propres extrêmes de $H(x,\rho)$ diverge, et donc les sous-problèmes deviennent infiniment mal conditionnés lorsque ρ s'approche de zéro.

5.3 Implantation de la pénalisation

5.3.1 Élimination du mauvais conditionnement

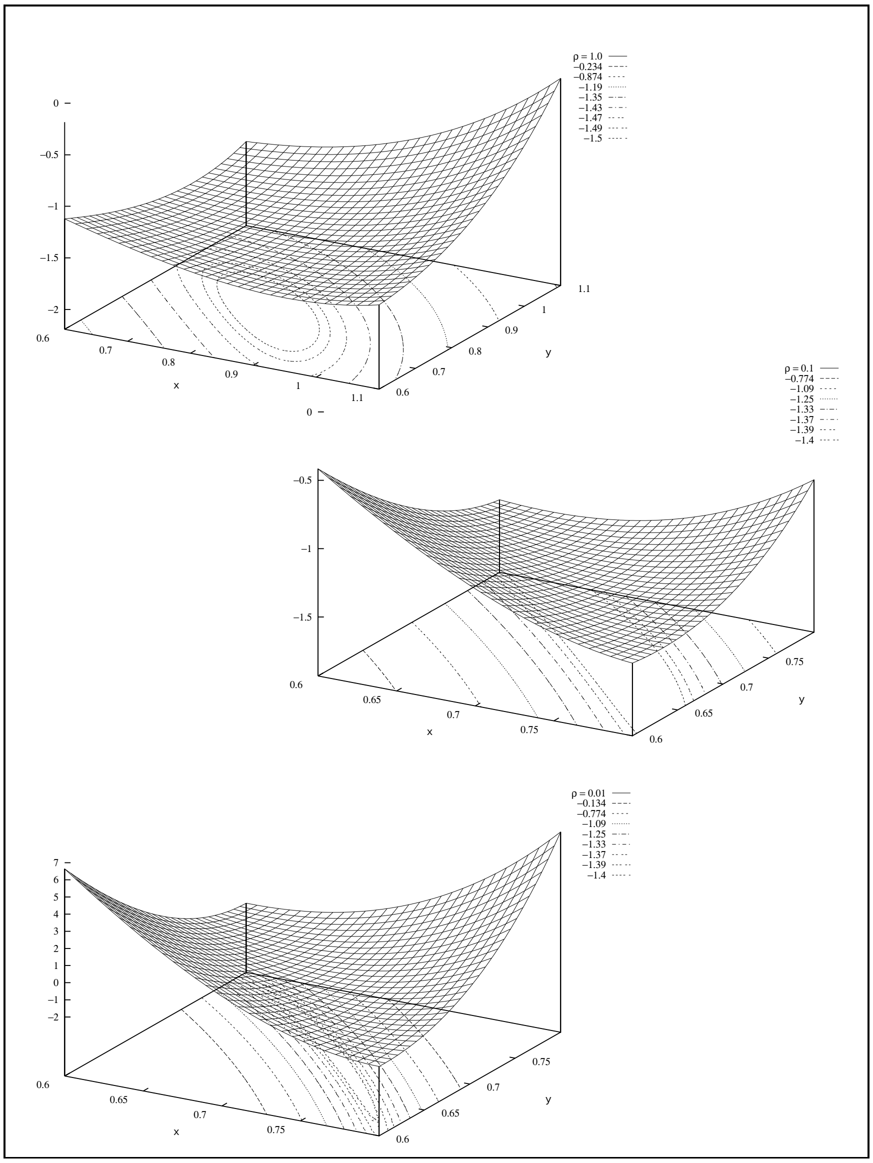
Récemment, Broyden&Attia [3, 4], et Gould [16] ont proposé des techniques qui nous montrent que le problème du mauvais conditionnement des méthodes de pénalisation est seulement apparent, et peut être éliminé par une transformation adéquate des équations mises en jeu. Gould a proposé une transformation qui utilise les variables λ, nommées aussi bien *multiplicateurs de Lagrange* que *variables duales* pour le problème (5.1). Broyden et Attia ont proposé une technique similaire, mais qui ne fait pas intervenir les variables λ.

Méthode primale-duale bien conditionnée

Revenons aux équations

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) - \rho\lambda^t &= 0. \end{aligned}$$

Puisque nous avons déduit le problème de minimisation pénalisé de ces équations en substituant dans la première équation une formule pour λ tirée de la seconde équation, ces deux équations sont équivalentes. Or, le Jacobien de l'équation primale-duale (celle qui utilise les



On observe la pénalisation quadratique pour des valeurs de $\rho = 1$, 0.1 et 0.01. La fonction est $f(x) = -x_1 - x_2$ et la contrainte est $1 - x_1^2 - x_2^2 = 0$. Le mauvais conditionnement s'observe par des lignes de niveau constant de plus en plus "plates", et également de plus en plus rapprochées.

FIG. 5.1 – Pénalisation

variables λ) est donné par la formule

$$\Theta(x,\lambda) = \begin{pmatrix} \nabla^2 f(x) + \lambda \nabla^2 g(x) & \nabla g(x)^t \\ \nabla g(x) & -\rho I_{m\times m} \end{pmatrix},$$

qui converge vers la matrice (lorsque $\rho \to 0$ et $x \to x^*$)

$$\begin{pmatrix} \nabla^2 f(x^*) + \lambda^* \nabla^2 g(x^*) & \nabla g(x^*)^t \\ \nabla g(x^*) & 0 \end{pmatrix},$$

qui est une matrice inversible, comme le stipule le lemme suivant.

Lemme 5.3.1 *Si x^* est un point régulier des contraintes, minimum local du programme (5.1) satisfaisant aux conditions suffisantes de second ordre, alors la matrice*

$$\begin{pmatrix} \nabla^2 f(x^*) + \lambda^* \nabla^2 g(x^*) & \nabla g(x^*)^t \\ \nabla g(x^*) & 0 \end{pmatrix},$$

est inversible.

Exercice 5.3.1 *[Inversibilité de $\nabla^2 L(x^*,\lambda^*)$]* Démontrez le lemme 5.3.1. Utilisez le fait qu'une matrice M est inversible à condition que la seule solution v au système $Mv = 0$ soit $v = 0$. Séparez v selon les deux blocs d'équations, et utilisez le fait que la matrice $\nabla g(x^*)$ est de plein rang. □

Supposons maintenant que nous résolvions les sous-problèmes pénalisés à l'aide de la méthode de Newton. Alors, la solution d'une équation linéaire de Newton pour un problème pénalisé serait, pour d_N la direction de Newton,

$$H(x,\rho)d_N = -P(x,\rho)$$
$$\left(\nabla^2 f(x) + \frac{g(x)^t}{\rho}\nabla^2 g(x) + \frac{\nabla g(x)^t \nabla g(x)}{\rho}\right) d_N = -\left(\nabla f(x) + \frac{g(x)^t}{\rho}\nabla g(x)\right),$$

et ce dernier calcul est de plus en plus mal conditionné lorsque ρ s'approche de zéro. Cependant, d_N satisfait également aux équations suivantes :

$$\begin{pmatrix} \nabla^2 f(x) + \lambda \nabla^2 g(x) & \nabla g(x)^t \\ \nabla g(x) & r \end{pmatrix} \begin{pmatrix} d_N \\ \lambda_N \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \frac{g(x)^t}{\rho}\nabla g(x)) \\ 0 \end{pmatrix}.$$

Puisque la matrice de ce dernier système s'approche d'une matrice inversible, aucun mauvais conditionnement ne peut être imputé à la méthode de pénalisation ainsi modifiée.

Méthode primale bien conditionnée

Reprenons l'équation du hessien de la pénalisation

$$H(x,\rho) = \nabla^2 f(x) + \frac{g(x)^t}{\rho}\nabla^2 g(x) + \frac{\nabla g(x)^t \nabla g(x)}{\rho},$$

et observons que le mauvais conditionnement n'a lieu que dans les sous-espaces image de la matrice ∇g. Puisque cette matrice est supposée de plein rang, il existe une matrice Q_x orthogonale telle que

$$Q_x \nabla g(x)^t = \begin{pmatrix} U \\ 0 \end{pmatrix},$$

avec U une matrice triangulaire supérieure. Par conséquent,

$$Q_x \left(\nabla^2 f(x) + \frac{g(x)^t}{\rho} \nabla^2 g(x) + \frac{\nabla g(x)^t \nabla g(x)}{\rho} \right) Q_x^t$$

se réécrit

$$Q_x \left(\nabla^2 f(x) + \frac{g(x)}{\rho} \nabla^2 g(x) \right) Q_x^t + \frac{1}{\rho} \begin{pmatrix} U \\ 0 \end{pmatrix} \begin{pmatrix} U \\ 0 \end{pmatrix}^t .$$

En dénotant par $G = Q_x \left(\nabla^2 f(x) + \frac{g(x)}{\rho} \nabla^2 g(x) \right) Q_x^t$, on retrouve

$$G + \begin{pmatrix} \frac{UU^t}{\rho} & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} G_{11} + \frac{UU^t}{\rho} & G_{12} \\ G_{21} & G_{22} \end{pmatrix}.$$

Le mauvais conditionnement est ainsi enlevé en multipliant le premier groupe d'équations par ρ :

$$\begin{pmatrix} \rho G_{11} + UU^t & \rho G_{12} \\ G_{21} & G_{22} \end{pmatrix},$$

qui converge vers la matrice inversible

$$\begin{pmatrix} U^* U^{*t} & 0 \\ G_{21}^* & G_{22}^* \end{pmatrix}.$$

Lemme 5.3.2 *Si x^* est un point régulier des contraintes, minimum local du programme (5.1) satisfaisant aux conditions suffisantes de second ordre, alors la matrice*

$$M = \begin{pmatrix} U^* U^{*t} & 0 \\ G_{21}^* & G_{22}^* \end{pmatrix}$$

est inversible.

Exercice 5.3.2 *[Inversibilité de la matrice M]* Démontrez le lemme 5.3.2. □

Si ces transformations sont utilisées pour calculer la direction de Newton, on retrouve les relations

$$\begin{pmatrix} G_{11} + \frac{UU^t}{\rho} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} \bar{d}_N = Q_x \left(\nabla f(x(\rho_k)) + \frac{g(x(\rho_k))^t}{\rho_k} \nabla g(x(\rho_k)) \right),$$

ou encore, en multipliant le premier bloc par ρ_k,

$$\begin{pmatrix} \rho G_{11} + UU^t & \rho G_{12} \\ G_{21} & G_{22} \end{pmatrix} \bar{d}_N = \begin{pmatrix} \rho \Phi_1 \\ \Phi_2 \end{pmatrix},$$

où la véritable direction de Newton satisfait $Q_x d_N = \bar{d}_N$ et où le vecteur

$$\Phi = Q_x \left(\nabla f(x(\rho_k)) + \frac{g(x(\rho_k))^t}{\rho_k} \nabla g(x(\rho_k)) \right).$$

5.3.2 Convergence locale superlinéaire de la pénalisation

Nous avons proposé deux transformations pour lesquelles les matrices jacobiennes des équations définissant les conditions d'optimalité des problèmes pénalisés se rapprochent de matrices inversibles. Nous exploitons dans cette section des formulations pour définir une fonction $x(\rho)$ qui sera différentiable en $\rho = 0$, une technique d'extrapolation pour suivre la trajectoire $x(\rho)$ numériquement, et une analyse de la vitesse de convergence locale du processus d'extrapolation.

Existence de trajectoires différentiables

Nous sommes intéressés à exprimer, proche d'un minimum local satisfaisant aux conditions *suffisantes* de second ordre pour le problème (5.1), les solutions $x(\rho)$ et $\lambda(\rho)$ du sous-problème

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) - \rho\lambda^t &= 0. \end{aligned}$$

Nous avons déjà vu que les solutions de ce système sont les mêmes que les solutions du système

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ \lambda &= \frac{g(x)^t}{\rho}, \end{aligned}$$

qui représentent les conditions d'optimalité de premier ordre des sous-problèmes pénalisés. Cependant, le Jacobien $\Theta(x, \lambda)$ du système équivalent se rapproche de $\nabla^2 L(x^*, \lambda^*)$ qui est une matrice inversible :

$$\lim_{\rho \to 0} \Theta(x(\rho), \lambda(\rho)) = \nabla^2 L(x^*, \lambda^*).$$

Ceci nous permet d'utiliser le théorème des fonctions implicites et de déduire l'existence des fonctions $x(\rho)$ et $\lambda(\rho)$ différentiables solutions du système linéaire transformé, et donc points stationnaires du sous-problème pénalisé. La discussion précédente constitue l'essentiel de la preuve du prochain théorème.

Théorème 5.3.1 *Soit x^* un point régulier des contraintes $g(x) = 0$ satisfaisant aux conditions suffisantes de second ordre pour le programme (5.1). Si les fonctions f et g sont de classe $\mathcal{C}^p(\mathbb{R}^n)$, alors, il existe des trajectoires différentiables $x(\rho)$ et $\lambda(\rho)$ de classe $\mathcal{C}^{p-1}(\mathbb{R}^n)$ telles que*

1. *$x(0) = x^*$ et $\lambda(0) = \lambda^*$;*
2. *si ρ est assez petit, $x(\rho)$ satisfait aux conditions suffisantes de second ordre pour le problème pénalisé $\min f(x) + \sum_{i=1}^m \frac{g_i(x)^2}{2\rho}$, où $x(\rho)$ et $\lambda(\rho)$ sont solutions du système d'équations*

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) - \rho\lambda^t &= 0. \end{aligned}$$

Preuve L'existence de trajectoires différentiables est assurée par le théorème des fonctions implicites puisque la matrice Jacobienne $\Theta(x, \lambda)$ se rapproche d'une matrice inversible.

Pour montrer que les points $x(\rho)$ sont des minima locaux des sous-problèmes pénalisés, il suffit de montrer que les matrices hessiennes des objectifs pénalisés sont définies positives proche de x^*. En effet, les points $x(\rho)$ sont des points stationnaires pour les problèmes pénalisés puisqu'ils sont solutions du système transformé équivalent à leurs conditions d'optimalité.

Pour montrer que les $x(\rho)$ sont des minima locaux de $p(x, \rho)$, considérons une suite quelconque $\rho_k \searrow 0$ et procédons par contradiction en supposant que les matrices $H(x(\rho_k), \rho_k)$ possèdent toutes une valeur propre négative. Soient v_k une suite de vecteurs tels que

$$v_k^t H(x(\rho_k), \rho_k) v_k < 0.$$

Les deux premiers termes dans l'expression de $h(x, \rho)$ s'approchent du hessien en x du Lagrangien, donc d'une matrice bornée. Le dernier terme diverge (vers $+\infty$) à moins que $\nabla g(x(\rho_k))v_k$ s'approche de zéro assez vite. Donc, le seul moyen de rendre éventuellement $v_k^t H(x(\rho_k), \rho_k) v_k$ négatif est que v_k s'approche du plan tangent, et l'hypothèse que x^* satisfait aux conditions suffisantes assure alors que

$$v_k^t \left(\nabla^2 f(x(\rho_k)) + \frac{g(x(\rho_k))^t}{\rho_k} \nabla g(x(\rho_k)) \right) v_k$$

s'approche d'une quantité positive, ce qui établit la contradiction. □

Corollaire 5.3.1 *Sous les hypothèses du théorème 5.3.1,*

1. $\|x(\rho) - x^*\| \sim \mathcal{O}(\rho)$;
2. $\|\lambda(\rho) - \lambda^*\| \sim \mathcal{O}(\rho)$;
3. $\|g(x(\rho))\| \sim \mathcal{O}(\rho)$.

Extrapolation sur la trajectoire

Dans cette section, nous exploitons les résultats d'existence de trajectoires différentiables pour construire une stratégie d'extrapolation afin de suivre les trajectoires. Nous proposons d'abord une stratégie simple utilisant un développement de Taylor d'ordre un, puis explorons plus tard les généralisations qui s'imposent si les sous-problèmes pénalisés ne sont résolus qu'approximativement.

Étude de l'extrapolation Dénotons par $x'(\rho)$ le vecteur dérivée de la trajectoire $x(\rho)$. Par les résultats présentés plus haut, $x'(0)$ existe, tout comme $x'(\rho)$, $\rho > 0$ si ρ est assez petit. Supposons que nous connaissions $x(\rho_k)$; nous pouvons estimer $x(\rho_{k+1})$ par une extrapolation d'ordre un :

$$\hat{x}(\rho_{k+1}) \stackrel{\text{def}}{=} x(\rho_k) + x'(\rho_k)(\rho_{k+1} - \rho_k) \sim x(\rho_{k+1}) + \mathcal{O}((\rho_{k+1} - \rho_k)^2).$$

Puisque ρ_k est une suite décroissante, on déduit que $\hat{x}(\rho_{k+1})$ est un estimé d'ordre $\mathcal{O}(\rho_k^2)$ de $x(\rho_{k+1})$.

Examinons maintenant l'expression de $x'(\rho)$. En dénotant comme avant $P(x,\rho) = 0$ les conditions d'optimalité d'un sous-problème pénalisé, on retrouve

$$\nabla_x P x' + \nabla_\rho P = 0,$$

ce qui se simplifie, compte tenu que $P(x,\rho) = \nabla f(x) + \sum_{i=1}^m \frac{g_i(x)}{\rho} \nabla g_i(x)$ à

$$\left(\nabla^2 f(x) + \frac{g(x)}{\rho}\nabla^2 g(x) + \frac{\nabla g(x)^t \nabla g(x)}{\rho}\right) x' - \frac{g_i(x)}{\rho^2}\nabla g_i(x) = 0,$$

ce qui nous donne que $x'(\rho)$ satisfait à l'équation

$$H(x,\rho)x'(\rho) = \sum_{i=1}^m \frac{g_i(x)}{\rho^2}\nabla g_i(x).$$

Ce vecteur de dérivées de x s'obtient donc en résolvant un système d'équations dont la matrice est *la même que celle définissant la direction de Newton.*

Cette dernière observation nous conduit à interpréter l'itération de Newton comme une technique d'extrapolation. Revenons momentanément au contexte où l'on calcule les racines d'une application $F(x) = 0$, et où $F(x^*)$ est une matrice inversible. Supposons que l'on connaisse un point x_k, proche de x^*, pour lequel $F(x_k) = \Phi_k$ est proche de zéro. Exprimons $x(\Phi_k) - x_k$, et utilisons le théorème des fonctions implicites pour estimer la racine $x(0)$ par $\hat{x}(0) = x(\Phi_k) + x'(\Phi_k)(0 - \Phi_k)$. Attention à la notation, gardée similaire à l'extrapolation selon le paramètre ρ_k, mais qui cache que Φ_k est un *vecteur* de paramètres, et donc x' est une *matrice* des dérivées des composantes de x par rapport aux composantes de Φ. Cette matrice est justement la matrice H^{-1}, et l'extrapolation linéaire de Φ_k à zéro est justement l'itération de Newton.

Enfin, considérons le système

$$\mathcal{P}(x,\rho,\Phi) = \nabla f(x) + \sum_{i=1}^m \frac{g_i(x)}{\rho}\nabla g_i(x) + \Phi = 0.$$

Nous pouvons exprimer deux variantes d'extrapolation :
- l'itération de Newton pour les sous-problèmes pénalisés constitue une extrapolation selon le vecteur de paramètres Φ,
- l'extrapolation selon le paramètre ρ, qui a été déduite pour un système où $\Phi \equiv 0$, constitue une extrapolation pour une valeur de Φ fixée.

Nous analysons dans la section suivante une stratégie mixte où des itérations de Newton pour une valeur de ρ fixée sont utilisées pour décroître Φ de telle sorte que l'on ait $\|\Phi\| \le \rho$, et ensuite une extrapolation *combinée* en ρ et Φ est calculée. Nous dénoterons donc les itérations par la fonction $x(\rho,\Phi)$. La notation précédente $x(\rho)$ correspond au cas particulier $x(\rho, 0)$.

L'extrapolation est utilisée pour estimer le point $x(\rho_{k+1}, 0)$ solution du problème

$$\mathcal{P}(x,\rho_{k+1},0) = 0.$$

Cet estimé, noté $\tilde{x}_{k+1}$ est utilisé comme point de départ de l'itération de Newton qui tente de réduire $\Phi \le \rho_{k+1}$, cette fois.

Un algorithme superlinéairement convergent

Comme nous venons de le voir, nous analysons maintenant un algorithme itératif qui effectue parfois des extrapolations par rapport aux paramètres Φ (méthode de Newton pour ρ fixé), et parfois des extrapolations mixtes par rapport aux paramètres ρ et Φ (extrapolations pour trouver un point de départ pour les itérations de Newton pour la prochaine valeur de ρ). Il s'agit donc d'une méthode *prédicteur-correcteur* où l'extrapolation constitue une prédiction de x_{k+1} alors que les itérations de Newton corrigent et améliorent cet estimé.

Théorème 5.3.2 *Soit x^* un point régulier des contraintes $g(x) = 0$ satisfaisant aux conditions suffisantes de second ordre pour le programme (5.1). Si les fonctions f et g sont de classe $\mathcal{C}^p(\mathbb{R}^n)$, alors, il existe des trajectoires différentiables $x(\rho, \Phi)$ et $\lambda(\rho, \Phi)$ de classe $\mathcal{C}^{p-1}(\mathbb{R}^n)$ telles que*

1. *$x(0,0) = x^*$ et $\lambda(0,0) = \lambda^*$;*
2. *si ρ et $\|\Phi\|$ sont assez petits, $x(\rho, 0)$ satisfait aux conditions suffisantes de second ordre pour le problème pénalisé $\min f(x) + \sum_{i=1}^m \frac{g_i(x)^2}{2\rho}$, où $x(\rho, \Phi)$, $\lambda(\rho, \Phi)$ sont solutions du système d'équations*

$$\nabla f(x) + \lambda \nabla g(x) = \Phi$$
$$g(x) - \rho\lambda^t = 0.$$

De plus, les bornes suivantes sont satisfaites :

(a) $\|x(\rho, \Phi) - x^*\| \sim \mathcal{O}(\max(\rho, \|\Phi\|))$;

(b) $\|\lambda(\rho, \Phi) - \lambda^*\| \sim \mathcal{O}(\max(\rho, \|\Phi\|))$;

(c) $\|g(x(\rho, \Phi))\| \sim \mathcal{O}(\rho)$.

Exercice 5.3.3 *[Preuve du théorème 5.3.2]* Démontrez le théorème 5.3.2. □

Nous étudions ici la vitesse de convergence locale que nous pouvons atteindre par ce processus. En fait, nous démontrons que si la suite $\{\rho_k\}$ est telle qu'il existe un certain R tel que $\frac{\rho_k^2}{\rho_{k+1}} < R, \forall k$, alors, une seule itération de Newton à partir de $\tilde{x}_{k+1}$ fournit une approximation de $x(\rho_{k+1}, 0)$ si bonne que la valeur de $\|\Phi\| < \rho_{k+1}$. La convergence est donc presque quadratique, et nous devons (asymptotiquement) résoudre seulement deux sous-problèmes linéaires pour chaque valeur de ρ_k.

Débutons en écrivant la formule de $\tilde{x}_{k+1}$:

$$\tilde{x}_{k+1} \stackrel{\text{def}}{=} x(\rho_k, \Phi_k) + \frac{\partial x}{\partial \rho}(\rho_k, \Phi_k)(\rho_{k+1} - \rho_k) + \frac{\partial x}{\partial \Phi}(\rho_k, \Phi_k)(0 - \Phi_k).$$

Puisque $x(\rho, \Phi)$ est de classe $\mathcal{C}^2$, on a

$$\|\tilde{x}_{k+1} - x(\rho_{k+1}, 0)\| \sim \mathcal{O}\left((\rho_{k+1} - \rho_k)^2, \|\Phi_k\|^2\right).$$

À partir du dernier terme, on voit que rien ne sert d'exiger que $\|\Phi_k\| < \rho_k$, du moins pour la qualité de $\tilde{x}_{k+1}$. Lorsque $\rho_{k+1} < \rho_k$ et $\|\Phi_k\| < \rho_k$, on a

$$\|\tilde{x}_{k+1} - x(\rho_{k+1}, 0)\| \sim \mathcal{O}(\rho_k^2).$$

Analyse asymptotique Le lemme suivant constitue une étape importante dans notre analyse.

Lemme 5.3.3 *Sous les mêmes hypothèses que le théorème 5.3.2, si f et g sont des fonctions de classe $\mathcal{C}^2(I\!\!R^n)$, alors $\|P(\tilde{x}_{k+1}, \rho_{k+1})\| \sim \mathcal{O}(\frac{\rho_k^2}{\rho_{k+1}})$ et la direction de Newton d_N calculée au point $\tilde{x}_{k+1}$ satisfait à la borne $\|d_N\| \sim \mathcal{O}(\rho_k^2)$.*

Preuve En utilisant les développements de la méthode primale pour enlever le mauvais conditionnement, écrivons $d_N = Q^t_{\hat{x}_{k+1}} \bar{d}$, où $\bar{d}$ est la solution d'un système d'équations de la forme :

$$M(\rho_{k+1})\bar{d} = \begin{pmatrix} \rho_{k+1}\bar{\Phi}_1 \\ \bar{\Phi}_2 \end{pmatrix},$$

avec

$$\bar{\Phi} = \begin{pmatrix} \bar{\Phi}_1 \\ \bar{\Phi}_2 \end{pmatrix} = -Q^t_{\tilde{x}_{k+1}} P(\tilde{x}_{k+1}, \rho_{k+1})$$

et la matrice $M(\rho_{k+1})$ s'approche de la matrice inversible

$$\begin{pmatrix} U^* U^{*t} & 0 \\ G^*_{21} & G^*_{22} \end{pmatrix}.$$

$P(x(\rho_{k+1}, 0), \rho_{k+1}) = 0$, et donc, nous avons

$$\bar{\Phi} = Q^t_{\tilde{x}_{k+1}} \left(P(x(\rho_{k+1}, 0), \rho_{k+1}) - P(\tilde{x}_{k+1}, \rho_{k+1}) \right).$$

En remplaçant $P(x(\rho_{k+1}, 0), \rho_{k+1}) - P(\tilde{x}_{k+1}, \rho_{k+1})$ par son expression :

$$\begin{aligned} \bar{\Phi} = {} & Q^t_{\tilde{x}_{k+1}} \left((\nabla f(x(\rho_{k+1}, 0)) - \nabla f(\tilde{x}_{k+1}) \right) \\ & + \sum_{i=1}^{m} \frac{g_i(x(\rho_{k+1}, 0))}{\rho_{k+1}} Q^t_{\tilde{x}_{k+1}} \left(\nabla g_i(x(\rho_{k+1}, 0)) - \nabla g_i(\tilde{x}_{k+1}) \right) \\ & + Q^t_{\tilde{x}_{k+1}} \left(\begin{pmatrix} \frac{1}{\rho_{k+1}} \left(g_1^t(x(\rho_{k+1}, 0)) - g_1^t(\tilde{x}_{k+1}) \right) \\ \vdots \\ \frac{1}{\rho_{k+1}} \left(g_p^t(x(\rho_{k+1}, 0)) - g_p^t(\tilde{x}_{k+1}) \right) \end{pmatrix} \nabla g(\tilde{x}_{k+1}) \right) \end{aligned} \tag{5.3}$$

et puisque $\|x(\rho_{k+1}, 0) - \tilde{x}_{k+1}\| \sim \mathcal{O}(\rho_k^2)$, le premier terme du côté droit de l'équation (5.3) est également $\sim \mathcal{O}(\rho_k^2)$. Le second terme est également $\sim \mathcal{O}(\rho_k^2)$ puisque $\frac{g_i(x(\rho_{k+1}, 0))}{\rho_{k+1}}$ s'approche de λ_i qui est borné alors que les ∇g_i sont différentiables (les g_i sont de classe $\mathcal{C}^3$), et donc leur différences de l'ordre de ρ_k^2.

Utilisons maintenant la décomposition orthogonale

$$Q^t_{\tilde{x}_{k+1}} \left(\nabla g(\tilde{x}_{k+1}) \right) = \begin{pmatrix} \hat{U} \\ 0 \end{pmatrix},$$

pour réécrire le troisième terme de l'équation (5.3) comme

$$\begin{pmatrix} \frac{1}{\rho_{k+1}} \left(g_1^t(x(\rho_{k+1}, 0)) - g_1^t(\tilde{x}_{k+1}) \right) \\ \vdots \\ \frac{1}{\rho_{k+1}} \left(g_p^t(x(\rho_{k+1}, 0)) - g_p^t(\tilde{x}_{k+1}) \right) \\ \begin{pmatrix} \hat{U} \\ 0 \end{pmatrix} \end{pmatrix}.$$

Maintenant, les bornes précédentes assurent que

$$\frac{1}{\rho_{k+1}} \left|(g_j(x(\rho_{k+1}, 0)) - g_j(\tilde{x}_{k+1}))\right| \sim \frac{1}{\rho_{k+1}} \mathcal{O}(\rho_k^2).$$

Puisque ce bloc d'équations est multiplié par ρ_{k+1} dans les équations modifiées, nous avons

$$\begin{pmatrix} \rho_{k+1}\bar{\Phi}_1 \\ \bar{\Phi}_2 \end{pmatrix} \sim \mathcal{O}(\rho_k^2).$$

Par conséquent, $d_N \sim \mathcal{O}(\rho_k^2)$ car $Q_{\tilde{x}_{k+1}}$ et $M(\rho_{k+1})$ sont des matrices régulières. □

Le prochain lemme permettra de déduire la convergence presque quadratique asymptotique.

Lemme 5.3.4 *Sous les mêmes hypothèses que le théorème 5.3.2, si f et g sont des fonctions de classe $\mathcal{C}^3(I\!R^n)$, soit d_N la direction de Newton calculée à $\tilde{x}_{k+1}$, et $\bar{x}_{k+1} = \tilde{x}_{k+1} + d_N$. Alors,*

$$\|P(\bar{x}_{k+1}, \rho_{k+1})\| \sim \mathcal{O}\left(\frac{\rho_k^4}{\rho_{k+1}}\right)$$

Preuve Développons P en série de Taylor pour obtenir

$$\begin{aligned} P(\bar{x}_{k+1}, \rho_{k+1}) = {} & \nabla f(\tilde{x}_{k+1}) + \frac{1}{\rho_{k+1}} \sum_{i=1}^{m} g_i(\tilde{x}_{k+1}) \nabla g_i(\tilde{x}_{k+1}) \\ & + \frac{G d_N}{\rho_{k+1}} + \frac{1}{\rho_{k+1}} \mathcal{O}(\|d_N\|^2), \end{aligned}$$

où G est le Jacobien du terme $\sum_{i=1}^{m} g_i \nabla g_i$ en $\tilde{x}_{k+1}$. Par la définition de l'itération de Newton, les trois premiers termes de cette dernière équation s'annulent, laissant seulement

$$P(\bar{x}_{k+1}, \rho_{k+1}) \sim \frac{1}{\rho_{k+1}} \mathcal{O}(\|d_N\|^2).$$

Le résultat suit du lemme 5.3.3. □

On constate que localement, $\|P(\tilde{x}_{k+1} + d_N, \rho_{k+1})\| \sim \mathcal{O}(\frac{\rho_k^4}{\rho_{k+1}})$, et donc, pour assurer que $\|P(\tilde{x}_{k+1} + d_N, \rho_{k+1})\| < \rho_{k+1}$, il suffit que $\frac{\rho_k^4}{\rho_{k+1}} < \rho_{k+1}$, ou encore, $\rho_{k+1} > \rho_k^2$, ainsi qu'annoncé au début de la section.

5.3.3 Convergence globale et localement superlinéaire

Pour espérer la convergence globale de l'algorithme, on utilisera une itération de Newton modifiée (chapitre 2). De plus, loin de la solution, il est possible que le point extrapolé ne soit pas bon, et on devra parfois démarrer les itérations de Newton avec la solution approchée du $(k-1)^{\text{ième}}$ sous-problème. La suite de cette section vérifie que asymptotiquement, le point extrapolé est toujours choisi, et que les modifications à l'itération de Newton ne sont plus en vigueur, c'est-à-dire que la direction de Newton est une direction de descente, et le pas unitaire est admissible dans la direction de Newton (asymptotiquement).

Pénalité

{ Donnés : x_0 et ρ_0 tels que $\|P(x_0, \rho_0)\| < \rho_0$. }
$\rho \leftarrow \rho_0$
$x_\rho \leftarrow x_0$
tantque ($\rho > Precision$)
  { Extrapolation }
    $\tilde{x} \leftarrow$ Calcule_x̃ $(x_\rho, r, \text{next}(\rho))$
    $\rho \leftarrow \text{next}(\rho)$
    si ($\|P(x_\rho, \rho)\| < \|P(\tilde{x}, \rho)\|$) **alors** $x \leftarrow x_\rho$
    sinon
      $x \leftarrow \tilde{x}$
  { Corrections de Newton }
    tantque ($\|P(x, \rho)\| > \rho$)
      $d_N \leftarrow$ Direction_de_Newton(x, ρ)
      $\theta \leftarrow$ Pas_d'Armijo(x, d_N)
      $x \leftarrow x + \theta d_N$
    $x_\rho \leftarrow x$

Algorithme 5.1: Pénalité

Choix du point extrapolé

On retient le point extrapolé seulement si la norme du gradient du sous-problème pénalisé y est plus petite qu'à la solution approchée du précédent sous-problème.

Lemme 5.3.5 *Sous les hypothèses du théorème 5.3.2, si $\lambda^* \neq 0$, asymptotiquement, le point extrapolé sera toujours utilisé.*

Preuve Une conséquence du lemme 5.3.3 est que

$$\|P(\tilde{x}_{k+1}, \rho_{k+1})\| \sim \mathcal{O}\left(\frac{\rho_k^2}{\rho_{k+1}}\right)$$

qui converge vers zéro alors que

$$\|P(x(\rho_k, \Phi_k), \rho_{k+1})\| = \|P(x(\rho_k, \Phi_k), \rho_k) + \frac{\rho_k - \rho_{k+1}}{\rho_{k+1}} \sum_{i=1}^{m} \frac{g_i(x_k)}{\rho_k} \nabla g_i(x_k)\|$$

ne converge pas vers zéro car son premier terme s'approche de zéro tandis que son dernier terme diverge puisque $\nabla g_i(x^*)$ sont linéairement indépendants par hypothèse. Donc, le seul moyen que

$$\|P(x(\rho_k, \Phi_k), \rho_{k+1})\|$$

puisse converger vers zéro serait que tous les multiplicateurs de Lagrange soient nuls, possibilité que nous écartons par hypothèse. $\square$

Direction de descente

Dans cette section, nous vérifions que la direction de Newton vérifie la définition 2.3.1. Pour alléger la notation, $\tilde{x}$ dénotera $\tilde{x}_{k+1}$, et le paramètre ρ_k sera omis de la fonction P, de sorte que, par exemple, $P(\tilde{x}+d_N)$ constituera une notation abrégée de $P(\tilde{x}_{k+1}+d_N,\rho_{k+1})$

Lemme 5.3.6 *Sous les mêmes hypothèses que le théorème 5.3.1 et pour des valeurs assez petites de ρ_k, il existe deux constantes positives γ_0 and γ_1 telles que*

$$\begin{aligned} P(\tilde{x})d_N &\leq -\gamma_0\rho_{k+1}\|P(\tilde{x})\|\|d_N\| \\ \|d_N\| &\leq \gamma_1\|P(\tilde{x})\|. \end{aligned}$$

Preuve La seconde borne découle de la preuve du lemme 5.3.3.

Pour dériver la première borne, observons que la preuve du lemme 5.3.3 assure l'existence d'une constante positive η telle que $\rho_{k+1}\|P(\tilde{x})\|\leq\eta\|d_N\|$. Par ailleurs, $\nabla^2 p(\tilde{x})d_N=-P(\tilde{x})^t$. La preuve du théorème 5.3.1 nous assure aussi que $\nabla^2 p(\tilde{x})$ est une matrice définie positive lorsque ρ_{k+1} devient assez petit. Il existe donc une constante positive ζ telle que $-d_N^tP(\tilde{x})=d_N^t\nabla^2p(\tilde{x})d_N\geq\zeta\|d_N\|^2$. En combinant les deux dernières bornes, on retrouve

$$-P(\tilde{x})d_N\geq\zeta\|d_N\|^2\geq\frac{\zeta}{\eta}\rho_{k+1}\|P(\tilde{x})\|\|d_N\|.$$

La première borne dans l'énoncé du théorème découle, avec le choix $\gamma_0=\frac{\zeta}{\eta}$. □

Admissibilité du pas unitaire

Nous montrons dans cette section que le pas unitaire satisfait au critère d'Armijo.

Lemme 5.3.7 *Sous les mêmes hypothèses que le théorème 5.3.1 et pour des valeurs assez petites de ρ_{k+1}, pour une constante $0<\tau_0<\frac{1}{2}$, la relation suivante est satisfaite :*

$$p(\tilde{x}_{k+1}+d_N,\rho_{k+1})-p(\tilde{x}_{k+1},\rho_{k+1})\leq\tau_0P(\tilde{x}_{k+1},\rho_{k+1})d_N.$$

Preuve En utilisant la même notation abrégée que dans la section précédente, on a

$$p(\tilde{x}+d_N)-p(\tilde{x})\leq\tau_0P(\tilde{x})d_N.$$

Exprimons p à l'aide d'un développement de Taylor de second ordre :

$$p(\tilde{x}+d_N)=p(\tilde{x})+P(\tilde{x})d_N+\frac{1}{2}d_N^tH(\hat{x})d_N,$$

où $\hat{x}=\tilde{x}+\sigma d_N$ avec $0\leq\sigma\leq1$ de sorte que $\|\tilde{x}-\hat{x}\|\sim\mathcal{O}(\rho_k^2)$.

Ensuite, utilisant le fait que $H(\tilde{x})d_N=-P(\tilde{x})^t$, réexprimons le critère d'Armijo comme

$$(\tau_0-\frac{1}{2})d_N^tH(\tilde{x})d_N+\frac{1}{2}d_N^t\left(H(\hat{x})-H(\tilde{x})\right)d_N\leq0.$$

Nous montrons maintenant que même si $d_N^t H(\tilde{x}) d_N$ est petit, $d_N^t \left(H(\hat{x}) - H(\tilde{x})\right) d_N$ est encore plus petit, et est toujours négligeable par rapport à $d_N^t H(\tilde{x}) d_N$. Pour ce faire, rappelons l'expression de $\nabla^2 p$.

$$\begin{aligned}\nabla^2 p(\tilde{x}) \stackrel{\text{def}}{=} \nabla^2 f(\tilde{x}) - \sum_{i=1}^{m} \frac{g_i(\tilde{x})}{\rho_{k+1}} \nabla^2 g_i(\tilde{x}) \\ + \sum_{i=1}^{m} \frac{1}{\rho_{k+1}} \nabla g_i^t(\tilde{x}) \nabla g_i(\tilde{x}).\end{aligned}$$

Exprimons maintenant les différences de $\nabla^2 p$ en utilisant la notation $\Delta(f) \stackrel{\text{def}}{=} f(\tilde{x}) - f(\hat{x})$:

$$\begin{aligned}\Delta(\nabla^2 p) =& \Delta(\nabla^2 f) + \sum_{i=1}^{m} \left(\frac{\Delta(g_i)}{\rho_{k+1}} \nabla^2 g_i(\tilde{x}) - \frac{g_i(\tilde{x})}{\rho_{k+1}} \Delta(\nabla^2 g_i) \right) \\ &+ \frac{1}{\rho_{k+1}} \sum_{i=1}^{m} \Delta(\nabla g_i \nabla g_i^t).\end{aligned} \tag{5.4}$$

La suite du raisonnement consiste à comparer $d_N^t \left(H(\hat{x}) - H(\tilde{x})\right) d_N$ et $d_N^t H(\tilde{x}) d_N$ terme à terme. Or, puisque $H(\tilde{x})$ est une matrice définie positive (lorsque ρ est assez petit), $d_N^t H(\tilde{x}) d_N \geq \lambda_{min} \|d_N\|^2$ où $\lambda_{min} > 0$ est la valeur propre la plus petite de $H(\tilde{x})$. Par ailleurs, $\|H(\hat{x}) - H(\tilde{x})\| \sim \mathcal{O}(\frac{\rho_k^2}{\rho_{k+1}})$, et donc $d_N^t \left(H(\hat{x}) - H(\tilde{x})\right) d_N \sim \mathcal{O}(\|d_N\|^2 \frac{\rho_k^2}{\rho_{k+1}})$. Donc, le terme $d_N^t \left(H(\hat{x}) - H(\tilde{x})\right) d_N$ devient asymptotiquement négligeable devant le terme $d_N^t H(\tilde{x}) d_N$, qui est positif. □

Convergence globale et localement superlinéaire

Pour conclure, énonçons un théorème qui résume les propriétés de l'algorithme.

Théorème 5.3.3 *L'algorithme 5.1 est bien défini. Si un point d'accumulation de l'algorithme est un point régulier des contraintes, alors il satisfait aux conditions nécessaires d'optimalité. De plus, s'il s'agit d'un point satisfaisant aux conditions suffisantes d'optimalité, alors pour des valeurs de ρ_k assez petites, on a*

1. *Une seule itération de Newton (modifiée avec une recherche linéaire d'Armijo) depuis $\hat{x}_{k+1}$ fournit directement $x(\rho_{k+1}, \xi_{k+1})$ avec $\|\xi_k\| \leq \epsilon_k$*
2. *de plus, si au moins une composante de λ^* est non-nulle, la suite $x(\rho_{k+1}, \xi_{k+1})$ converge superlinéairement avec un ordre α.*

Preuve

1. L'algorithme est bien défini car pour chaque nouveau sous-problème, il s'agit d'appliquer l'algorithme de Newton modifié pour lequel on a déjà démontré la propriété de convergence globale.
2. Si un point d'accumulation est un point régulier, le théorème 5.2.1 nous assure qu'il satisfait aux conditions nécessaires d'optimalité.
3. Les lemmes précédents nous assurent que si le point d'accumulation satisfait aux conditions suffisantes d'optimalité et si $\lambda^* \neq 0$, asymptotiquement, le point extrapolé est retenu, la direction de Newton est une direction de descente, le pas unitaire est admissible et une seule itération de Newton fournit $x(\rho_{k+1}, \xi_{k+1})$.

4. Si $\lambda^* \neq 0$, il existe au moins une composante non nulle telle que $\frac{g_i(x_k)}{\rho_k} \to \lambda_i^* \neq 0$. Ceci nous assure l'existence de constantes positives γ_1 et γ_2 telles que $\gamma_1 \rho_k \leq |g_i(x_k)| \leq \gamma_2 \rho_k$, et puisque la suite ρ_k converge vers zéro superlinéairement avec un ordre limite $\alpha < 2$, la suite $\|x(\rho_k, \xi_k) - x^*\|$ converge également vers zéro avec un ordre limite $\alpha < 2$.

□

Exercice 5.3.4 *[Convergence globale]* Le théorème 5.2.1 concerne la suite des minima *exacts* des sous-problèmes pénalisés. Adaptez-en la preuve au cas de minima approchés. □

5.4 Lagrangiens et pénalités augmentés

Suite aux analyses de Murray, de nombreux auteurs ont tenté de développer des méthodes inspirées des pénalisations, mais dépourvues du problème du mauvais conditionnement. Un de ces efforts a produit une théorie riche autour du concept de Lagrangien augmenté.

Le Lagrangien augmenté peut se déduire d'une manière similaire à la déduction de la pénalisation quadratique ci haut : les conditions d'optimalité se réécrivent, utilisant une constante $\bar{\lambda}$ censée estimer λ^*,

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) &= \rho(\lambda - \bar{\lambda})^t, \end{aligned}$$

de telle sorte que, puisque $\lambda = \frac{g(x)^t}{\rho} + \bar{\lambda}$, on peut réécrire ce dernier système d'équations en une seule

$$P_A(x, \bar{\lambda}, \rho) \stackrel{\text{def}}{=} \nabla f(x) + (\frac{g(x)^t}{\rho} + \bar{\lambda}) \nabla g(x) = 0,$$

qui constitue les conditions d'optimalité du problème

$$\min p_a(x, \bar{\lambda}, \rho) \stackrel{\text{def}}{=} f(x) + \bar{\lambda} g(x) + \frac{\|g(x)\|^2}{2\rho}, \tag{5.5}$$

qui peut se récrire

$$L(x, \bar{\lambda}) + \frac{\|g(x)\|^2}{2\rho},$$

un Lagrangien augmenté d'un terme de pénalité ou encore

$$p_a(x, \bar{\lambda}, \rho) = p(x, \rho) + \bar{\lambda} g(x),$$

une pénalité augmentée d'un terme issu du Lagrangien.

Une propriété importante de la fonction p_a est donnée dans la proposition suivante.

Proposition 5.4.1 *Soient x^* un minimum local du problème (5.1) et $[x^*, \lambda^*]$ un couple satisfaisant aux conditions nécessaires d'optimalité pour le problème (5.1). Alors, x^* est un point stationnaire pour le problème $\min p_a(x, \lambda^*, \rho)$. De plus, il existe une valeur critique $\bar{\rho}$ telle que x^* est un minimum local de $\min p_a(x, \lambda^*, \rho)$ pour toute valeur $\rho < \bar{\rho}$, et si x^* satisfait aux conditions suffisantes d'optimalité pour le problème (5.1), $\nabla^2_{xx} p_a(x^*, \lambda^*, \rho)$ est une matrice définie positive pour toute valeur $\rho < \bar{\rho}$.*

Exercice 5.4.1 *[]* Démontrez la proposition 5.4.1. □

Les algorithmes de Lagrangien augmenté utilisent cette propriété pour améliorer l'estimé $\bar{\lambda}$ itérativement. Supposons que ρ soit constant, et inférieur à $\bar{\rho}$. Dénotons par $x(\lambda)$ un minimum de la fonction $p_a(x, \lambda, \rho)$. L'algorithme original du Lagrangien augmenté, aussi nommé *méthode des multiplicateurs* prend alors la forme suivante :

$$\lambda_{k+1} \leftarrow \lambda_k + \frac{g(x(\lambda_k))^t}{\rho}.$$

Sous certaines hypothèses (convergence vers un point satisfaisant aux conditions suffisantes d'optimalité), on montre la convergence locale de cet algorithme. Cependant, dans cette forme, l'algorithme n'est pas pratique puisque le seuil $\bar{\rho}$ est inconnu, et de plus il est en général impossible d'obtenir $x(\lambda)$, mais on doit se contenter d'une approximation. L'avantage de cet algorithme vient du fait que les problèmes de mauvais conditionnement associés à la décroissance de ρ vers 0 sont amoindris.

Pour tenir compte des points faibles mentionnés ci-haut, considérons une suite décroissante ρ_k comme dans les méthodes de pénalité, mais qui ne tend pas forcément à zéro. Considérons également des solutions approchées $x(\bar{\lambda}, \rho, \Phi)$ qui satisfont à la condition

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= \Phi \\ g(x) &= \rho(\lambda - \bar{\lambda})^t, \end{aligned}$$

et vérifions que ce système induit des fonctions différentiables $x(\bar{\lambda}, \rho, \Phi)$ et $\tilde{\lambda}(\bar{\lambda}, \rho, \Phi) \stackrel{\text{def}}{=} \bar{\lambda} + \frac{g(x(\bar{\lambda}, \rho, \Phi))^t}{\rho}$. En poursuivant cette démarche, on constatera que les dérivées partielles de ces fonctions par rapport à ρ et à $\bar{\lambda}$ s'annulent en $\bar{\lambda} = \lambda^*$ et $\rho = 0$, et il est plus simple d'effectuer l'analyse en effectuant le changement de variable suivant : $\mu = \rho(\bar{\lambda} - \lambda^*)$, où λ^* est le multiplicateur de Lagrange à la solution optimale. Alors, en imitant la preuve du théorème 5.3.1, on démontre le théorème suivant en vérifiant que les fonctions $x(\rho, \mu, \Phi)$ et $\lambda(\rho, \mu, \Phi)$ sont différentiables.

Théorème 5.4.2 *Soit x^* un point régulier des contraintes satisfaisant aux conditions suffisantes de second ordre pour le programme (5.1). Si f et g sont des fonctions de classe $\mathcal{C}^p(I\!R^n)$, alors, il existe des trajectoires différentiables de classe $\mathcal{C}^{p-1}(I\!R^n)$ $x(\rho, \bar{\lambda}, \Phi)$ et $\tilde{\lambda}(\rho, \bar{\lambda}, \Phi)$ telles que*

1. $x(0, \lambda^*, 0) = x^*$ *et* $\tilde{\lambda}(0, \lambda^*, 0) = \lambda^*$ *;*
2. *Si ρ, $\|\Phi\|$ et $\|\bar{\lambda} - \lambda^*\|$ sont assez petits, alors $x(\rho, \bar{\lambda}, 0)$ satisfait aux conditions suffisantes de second ordre pour le problème pénalisé $f(x) + \bar{\lambda} g(x) + \frac{\|g(x)\|^2}{2\rho}$, et où $x(\rho, \bar{\lambda}, \Phi)$ et $\tilde{\lambda}(\rho, \bar{\lambda}, \Phi)$ sont solutions du système d'équations*

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= \Phi \\ g(x) &= \rho(\lambda - \bar{\lambda})^t; \end{aligned}$$

de plus, les bornes suivantes sont satisfaites :

(a) $\|g(x(\rho, \bar{\lambda}, \Phi))\| \sim \mathcal{O}(\max(\rho\|\bar{\lambda} - \lambda^*\|, \|\Phi\|))$ *;*

(b) $\|\tilde{\lambda}(\rho, \bar{\lambda}, \Phi) - \lambda^*\| \sim \mathcal{O}(\max(\rho\|\bar{\lambda} - \lambda^*\|, \|\Phi\|))$.

Exercice 5.4.2 *[]* Démontrez le théorème 5.4.2. □

Dans la suite, nous utilisons ce théorème pour analyser deux variantes de ces algorithmes que nous nommons *Lagrangien augmenté* et *pénalité augmentée* respectivement. Le Lagrangien augmenté utilise une suite ρ_k éloignée de zéro alors que la pénalité augmentée utilise une suite $\rho_k \searrow 0$.

5.4.1 Lagrangiens augmentés

La notation grand O utilisée dans le théorème 5.4.2 peut se récrire comme suit : il existe une constante positive M et un voisinage de $x^*, \lambda^*, 0$ tels que

$$\|g(x(\rho, \bar{\lambda}, \Phi))\| \leq \max(\rho M\|\bar{\lambda} - \lambda^*\|, M\|\Phi\|)$$

et

$$\|\tilde{\lambda}(\rho, \bar{\lambda}, \Phi) - \lambda^*\| \leq \max(\rho M\|\bar{\lambda} - \lambda^*\|, M\|\Phi\|).$$

De la seconde conditions, on voit que si $\rho < \frac{1}{M}$, $\tilde{\lambda}$ est plus proche de λ^* que $\bar{\lambda}$. La convergence est linéaire, et le taux de convergence est ρM. En particulier, le taux de convergence peut être rendu aussi petit que voulu en diminuant ρ. Par ailleurs, la première relation permet d'évaluer si ρ est assez petit et $\bar{\lambda}$ assez proche de λ^* pour que la convergence linéaire se manifeste : il suffit de comparer $g(x)$ en deux itérations consécutives. En se basant sur cette constatation, il est possible de combiner l'algorithme de pénalité usuel qui réduit ρ avec l'algorithme de Lagrangien augmenté qui modifie λ. L'algorithme résultant possédera la propriété de convergence globale des algorithmes de pénalité et la convergence locale avec ρ éloigné de zéro des algorithmes de Lagrangiens augmentés.

L'algorithme 5.2 diminue ρ tant qu'une réduction de taux η de $\|g(x)\|$ n'est pas observée. Au voisinage d'un point satisfaisant aux hypothèses du théorème (5.4.2), on sait que ρ ne sera plus réduit car il deviendra éventuellement assez petit pour que la convergence linéaire de taux $\eta > 0$ soit observée, et le comportement asymptotique de la méthode est celui du lagrangien augmenté.

Aspects de dualité

Considérons la fonction $\Psi(\lambda) \stackrel{\text{def}}{=} p_a(x(\lambda), \lambda, \rho)$ où x y est vu comme une fonction du paramètre λ et où le paramètre ρ est considéré constant. On voit aisément que $\Psi(\lambda^*) = p_a(x^*, \lambda^*, \rho) = p_a(x^*, \lambda, \rho) \geq p_a(x(\lambda), \lambda, \rho) = \Psi(\lambda)$, et donc λ^* constitue un *maximum* de la fonction Ψ. En appliquant le théorème des fonctions implicites, on peut exprimer les dérivées de la fonction Ψ comme suit : $\nabla\Psi(\lambda) = g(x(\lambda))$ et

$$\nabla^2(x(\lambda)) = \nabla g(x(\lambda))^t H_a(x, \lambda, \rho)^{-1} \nabla g(x(\lambda)),$$

où la matrice H_a est définie comme suit :

$$H_a(x, \lambda, \rho) = \nabla^2 f(x) + \left(\sum(\lambda_i + \frac{g_i(x)}{\rho})\nabla^2 g_i(x)\right) + \frac{1}{\rho}\nabla g(x)^t \nabla g(x).$$

```
Lagrangien_augmenté
  { Donnés : 0 < η < 1, x₀, ρ₀ et λ₀ }
  ρ ← ρ₀
  x ← x₀
  λ ← λ₀
  g⁻ ← g(x)
  tantque (non terminé )
    { Corrections de Newton }
    tantque (‖P_A(x, λ, ρ)‖ > ρ‖g⁻‖ )
      d_N ← Direction_de_Newton(x, λ, ρ)
      θ ← Pas_d'Armijo(x, d_N)
      x ← x + θd_N
    si (‖g(x)‖ < η‖g⁻‖ ) alors
      λ ← λ + g(x)^t/ρ
      g⁻ ← g(x)
    sinon
      ρ ← next(ρ)
```

Algorithme 5.2: Lagrangien augmenté

À l'aide de ces expressions, il est possible d'exprimer un algorithme d'*ascension* pour le problème $\max_\lambda \Psi(\lambda)$.

Évidemment, il semble difficile de mettre en œuvre un algorithme utilisant des directions suffisamment ascendantes conjointement à des pas admissibles. En effet, le calcul d'un pas admissible entraînerait l'évaluation de la fonction Ψ en plusieurs valeurs $\lambda + \theta d$, chacune des évaluations étant effectuée par la minimisation en x de $p_a(x, (\lambda + \theta d), \rho)$. Cependant, la structure de la fonction Ψ nous fournit une autre possibilité : si ρ est assez petit (mais éloigné de zéro), et λ suffisamment proche de λ^*, un pas unitaire est admissible dans la direction de Newton $d_N = (\nabla^2\Psi)^{-1}\nabla\Psi$. En d'autres termes, on peut envisager d'utiliser un algorithme de pénalité qui réduit ρ tant que la direction de Newton avec un pas unitaire n'augmente pas assez Ψ.

De plus, la direction de Newton est reliée à la direction $\frac{g(x)}{\rho}$. On peut montrer que $(\nabla^2\Psi)$ se rapproche de $\frac{1}{\rho}I$ lorsque ρ diminue. Par conséquent, on peut approcher la direction de Newton par la formule $d_N = \frac{1}{\rho}g(x(\lambda))$.

En supposant que la matrice H est inversible et la matrice A est de plein rang et en utilisant la formule de Sherman–Morrison, on vérifie que

$$\begin{pmatrix} H + ASA^t & -A \\ -A^t & 0 \end{pmatrix}^{-1} = \begin{pmatrix} H & -A \\ A^t & 0 \end{pmatrix}^{-1} - \begin{pmatrix} 0 & 0 \\ 0 & S \end{pmatrix}.$$

Par ailleurs, on peut exprimer

$$\begin{pmatrix} H & -A \\ A^t & 0 \end{pmatrix}^{-1} = \begin{pmatrix} W & -T \\ -T^t & U \end{pmatrix},$$

où $U = -(A^t H^{-1} A)^{-1}$. Ainsi, en identifiant H à $\nabla^2 f(x) + \lambda \nabla^2 g(x)$, A à $\nabla g(x)$ et S à $\frac{1}{\rho} I$, l'identité nous fournit

$$\left(\nabla g(x) H_A(x, \lambda, \rho)^{-1} \nabla g(x)^t\right)^{-1} = \left(\nabla g(x) H(x, \lambda)^{-1} \nabla g(x)^t\right)^{-1} + \frac{1}{\rho} I.$$

En appliquant la même identité pour une valeur fixée ρ_0 de ρ, on obtient

$$\left(\nabla g(x) H_A(x, \lambda, \rho)^{-1} \nabla g(x)^t\right)^{-1} = \left(\nabla g(x) H_A(x, \lambda, \rho_0)^{-1} \nabla g(x)^t\right)^{-1} - \frac{1}{\rho_0} I + \frac{1}{\rho} I.$$

Ainsi, lorsque ρ s'approche de zéro, les deux premiers termes du membre de droite de la dernière équation demeurent constants alors que le troisième terme diverge, indiquant que $(\nabla g(x) H_A(x, \lambda, \rho)^{-1} \nabla g(x)^t)^{-1}$ s'approche de $\frac{1}{\rho} I$.

On sait que l'itération de Newton possède la propriété de convergence locale quadratique. Puisque la formule $\lambda^+ = \lambda + \frac{g(x)^t}{\rho}$ approche de mieux en mieux la formule de Newton lorsque ρ s'approche de zéro, on s'attend à ce que la vitesse de convergence de cette formule devienne de plus en plus rapide lorsque ρ diminue, ce que confirme le théorème 5.4.2.

5.4.2 Pénalités augmentées

Appliquons maintenant la démarche du développement des algorithmes de pénalité à un algorithme de pénalité augmentée.

Propriété de convergence globale

Les propriétés de convergence globale des algorithme de Lagrangien augmenté "purs" sont plus faibles que celles des algorithmes de pénalité. Ceci vient du fait que même si un point d'accumulation $\bar{x}$ est tel que $\nabla g(\bar{x})$ soit une matrice régulière, il n'est pas certain que $g(\bar{x}) = 0$. En effet, si l'on dénote x_k une solution approchée du problème (5.5) où $\bar{\lambda} = \lambda_k$, le raisonnement utilisé pour l'analyse des méthodes de pénalité conduit à la conclusion que $\lambda_k + \frac{g(x_k)^t}{\rho_k}$ converge vers un certain $\check{\lambda}$ pour lequel $\nabla f(\bar{x}) + \check{\lambda} \nabla g(\bar{x}) = 0$, mais ceci peut être réalisé même si $g(x_k)$ ne converge pas vers zéro, et dans ce cas, la suite λ_k diverge.

En d'autres termes, si la suite λ_k n'est pas convergente, il est possible que les points d'accumulation $\bar{x}$ ne soient pas réalisables. En s'inspirant de l'algorithme 5.2, on peut proposer un schéma qui combine la stratégie de pénalité (diminuer ρ) avec la stratégie hybride (diminuer ρ et modifier λ).

Convergence locale superlinéaire

Tout comme pour l'analyse des méthodes de pénalité, nous allons proposer un critère d'approximation pour $\|\Phi\|$ et une vitesse de décroissance à zéro pour ρ_k. La différence est que nous allons remplacer $\bar{\lambda}$ à chaque nouvelle valeur de ρ_k.

S'il s'agissait d'un algorithme de pénalité , on s'attendrait à ce que l'on ait sous la main un estimé $\bar{\lambda}$ d'ordre un (en ρ) de λ^*. Dans ce cas, le théorème 5.4.2 nous informe que $g(x(\rho, \bar{\lambda}, \Phi)) \sim \max(\rho^2, \|\Phi\|)$, et donc que le choix naturel est $\|\Phi\| \sim \mathcal{O}(\rho^2)$.

Par ailleurs, puisque $\|x(\rho, \bar{\lambda}, \Phi) - x^*\|$ et $\|\tilde{\lambda}(\rho, \bar{\lambda}, \Phi) - \lambda^*\|$ sont tous deux de l'ordre de ρ^2, l'utilisation d'une extrapolation d'ordre un (comme pour l'algorithme de pénalité) n'apporte rien. Cependant, comme nous le verrons, ces estimé d'ordre deux constituent de bons points de départ pour l'itération de Newton du prochain sous-problème.

Analysons maintenant l'algorithme 5.3.

```
Pénalité_augmentée
{ Donnés : 0 < η < 1, x0, ρ0 et λ0 }
ρ ← ρ0
x ← x0
λ ← λ0
g⁻ ← g(x)
tantque (ρ > Precision )
    { Corrections de Newton }
    tantque (‖P_A(x, λ, ρ)‖ > ρ² )
        d_N ← Direction_de_Newton(x, λ, ρ)
        θ ← Pas_d'Armijo(x, d_N)
        x ← x + θd_N
    si (‖g(x)‖ < η‖g⁻‖ ) alors
        λ ← λ + g(x)^t/ρ
    ρ ← next(ρ)
    g⁻ ← g(x)
```

Algorithme 5.3: Pénalité augmentée

Lemme 5.4.1 *Sous les hypothèses du théorème 5.4.2, si de plus $\|\lambda_k - \lambda^*\| \sim \mathcal{O}(\rho_k)$, on a*

$$\|P_A(x(\rho_k, \lambda_k, \rho_k), \tilde{\lambda}(\rho_k, \lambda_k, \rho_k), \rho_{k+1})\| \sim \mathcal{O}(\frac{\rho_k^2}{\rho_{k+1}}).$$

Exercice 5.4.3 *[]* Démontrez le lemme 5.4.1. □

Ce dernier lemme nous empêche d'envisager une suite ρ_k qui converge vers zéro plus rapidement que quadratiquement. En effet, pour que l'algorithme soit bien défini, il faut s'assurer que $\|P_A(x(\rho_k, \lambda_k, \rho_k), \tilde{\lambda}(\rho_k, \lambda_k, \rho_k), \rho_{k+1})\|$ demeure bornée.

Lemme 5.4.2 *Sous les hypothèses du lemme 5.4.1, la direction de Newton d_N calculée au point $x(\rho_k, \lambda_k, \Phi_k)$ pour le problème $\min p_a(x, \tilde{\lambda}((\rho_k, \lambda_k, \Phi_k), \rho_{k+1})$ satisfait à la borne $\|d_N\| \sim \mathcal{O}(\rho_k^2)$.*

Exercice 5.4.4 *[]* Démontrez le lemme 5.4.2. □

Lemme 5.4.3 *Sous les hypothèses du lemme 5.4.1, au point $\bar{x} = x(\rho_k, \lambda_k, \Phi_k) + d_N$, on a*

$$\|P_A(x(\rho_k, \lambda_k, \Phi_k), \tilde{\lambda}(\rho_k, \lambda_k, \Phi_k), \rho_{k+1})\| \sim \mathcal{O}(\frac{\rho_k^4}{\rho_{k+1}}).$$

Exercice 5.4.5 *[]* Démontrez le lemme 5.4.3. □

Donc, localement, une seule itération de Newton fournit le point désiré à condition que $\frac{\rho_k^4}{\rho_{k+1}^3} < \infty$, et à condition que l'algorithme de Newton modifié soit bien défini et que le pas unitaire soit admissible. Ces deux propriétés font l'objet des deux prochains lemmes.

Lemme 5.4.4 *Sous les hypothèses du lemme 5.4.1 et pour des valeurs assez petites de ρ_k, il existe deux constantes positives γ_0 and γ_1 telles que au point $\bar{x}$ défini au lemme 5.4.3, on ait*

$$P_A(\bar{x})d_N \leq -\gamma_0\rho_k\|P_A(\bar{x})\|\|d_N\|$$
$$\|d_N\| \leq \gamma_1\|P_A(\bar{x})\|.$$

Exercice 5.4.6 *[]* Démontrez le lemme 5.4.4. □

Lemme 5.4.5 *Sous les hypothèses du lemme 5.4.4, pour une constante $0 < \tau_0 < \frac{1}{2}$, la relation suivante est satisfaite :*

$$p_a(\bar{x} + d_N, \tilde{\lambda}(\rho_k, \lambda_k, \Phi_k), \rho_{k+1}) - p_a(\bar{x}, \tilde{\lambda}(\rho_k, \lambda_k, \Phi_k), \rho_{k+1}) \leq \tau_0 P_A(\bar{x}, \tilde{\lambda}(\rho_k, \lambda_k, \Phi_k), \rho_{k+1})d_N.$$

Exercice 5.4.7 *[]* Démontrez le lemme 5.4.5. □

On peut résumer les résultats de convergence locale.

Théorème 5.4.3 *Sous les hypothèses du lemme 5.4.4, une seule itération de Newton est requise pour chaque sous-problème à condition que $\frac{\rho_k^4}{\rho_{k+1}^3} \to 0$. Si $\frac{\rho_k^8}{\rho_{k+1}^5} \to 0$, deux itérations de Newton sont requises, donnant ainsi une convergence quasi-quadratique en deux étapes.*

Puisqu'une convergence quadratique en deux étapes correspond, dans un certain sens, à une convergence d'ordre $\sqrt{2}$ par étape, on voit que le comportement asymptotique de la pénalité augmentée n'améliore pas le comportement asymptotique de l'algorithme de pénalité qui utilise une extrapolation.

5.5 Fonctions de pénalité exacte

L'algorithme de pénalité quadratique produit une suite $x(\rho_k)$ qui converge vers x^* lorsque ρ_k s'approche de zéro. Une des pistes pour éviter le mauvais conditionnement causé par la convergence à zéro de ρ_k est d'utiliser d'autres fonctions de pénalités pour lesquelles une valeur strictement positive de ρ_k permet de définir un sous-problème dont la solution est la même que celle du programme (5.1).

Deux familles de telles fonctions ont été proposées : des fonctions de pénalités *non-différentiables* qui utilisent $\|g(x)\|$ plutôt que $\|g(x)\|^2$, et des fonctions de pénalités différentiables, qui sont inspirées des méthodes de Lagrangiens augmentés, mais qui explicitent le fait que λ est une fonction de x et de ρ.

Les fonctions de pénalité non-différentiables sont plus facilement analysées en utilisant les notions de sous-differentiel, que nous introduirons plus tard. Les fonctions de pénalité différentiables, quant à elles, sont en fait des méthodes duales, et seront traités en plus grand détail ailleurs.

5.6 Extensions et références

La notion de qualification des contraintes a suscité beaucoup d'intérêt, dans le but d'affaiblir l'exigence que x^* soit un point régulier. Cependant, pour l'analyse d'algorithmes, cette condition demeure la plus utile. L'analyse des conditions d'optimalité est inspirée de celle de Luenberger.

L'algorithme de pénalité quadratique semble avoir été proposé pour la première fois par Courant[6]. Fiacco&McCormick[12] ont poussé l'analyse de diverses méthodes de pénalités dans leur livre fameux. L'élimination du mauvais conditionnement est récente, cependant, et dûe à Broyden&Attia[3, 4] et à Gould[16]. Ce dernier [17] a publié une analyse de la pénalisation quadratique dont les résultats sont semblables à ceux de la section 2.5, mais dont les preuves et lemmes intermédiaires sont différents, et plus complexes. Dussault[11] a analysé une famille abstraite d'algorithmes de pénalités, dont la pénalité quadratique est un cas particulier. De plus, comme il le souligne, le calcul de $\frac{g_i(x(\rho))}{\rho}$ conduit, en général à des erreurs numériques causées aux cancellations éventuelles (et souvent inévitables) dans le calcul de g. Il propose donc d'utiliser d'autres estimés des multiplicateurs de Lagrange λ. Les estimateurs des multiplicateurs de Lagrange ont été assez peu étudiés, mais les travaux de Gill, Murray et leurs collaborateurs sur ce sujet, en plus d'être sans concurrence réelle, sont excellents.

Les techniques de Lagrangien augmenté ont été proposées par Hestenes [18] et Powell [24], et analysées en profondeur par Bertsekas[2] et aussi par Rockafellar[25, 26].

5.7 Résumé

Nous avons déduit d'observations élémentaires des conditions d'optimalité pour les problèmes différentiables : $\nabla f(x^*)$ doit être orthogonal au plan tangent aux contraintes en x^*, ce qui, en autant que x^* soit un point régulier des contraintes, est équivalent à ce que $\nabla f(x^*)$ soit orthogonal au noyau de la matrice $\nabla g(x^*)$, dont les colonnes de la matrice Z^* forment une base. Ainsi, $\nabla f(x^*)Z^* = 0$. De plus, le hessien réduit du Lagrangien doit être (semi-)défini positif.

L'idée naturelle de résoudre les équations issues des conditions d'optimalité du premier ordre (équations de Lagrange) à l'aide de la méthode de Newton conduit à une convergence *locale* uniquement. Certaines fonctions de mérite peuvent être utilisées pour procurer à cette approche la propriété de convergence globale, mais puisqu'il s'agit souvent de fonctions non-différentiables, l'étude de cette approche est reportée à plus tard.

Nous avons déduit l'algorithme de pénalité par le biais d'une perturbation des équations de Lagrange, et avons présenté ses propriétés de convergence globale et de convergence locale superlinéaire. De plus, nous avons présenté deux techniques pour contourner le problème du mauvais conditionnement (qui constitue l'objection habituelle face aux algorithmes de pénalité). Il en résulte un algorithme robuste et efficace.

L'algorithme du Lagrangien augmenté a été présenté sous l'angle d'un algorithme de pénalité augmenté d'un terme de Lagrangien. Un attrait important de cet algorithme vient du fait qu'il n'est plus nécessaire que le paramètre de pénalité s'approche de zéro. Cependant, compte tenu des techniques nouvelles permettant de contourner le problème du mauvais conditionnement, cette caractéristique est moins importante. L'algorithme du Lagrangien augmenté est moins attrayant car sa convergence globale est difficile à établir.

Chapitre 6

Optimisation différentiable avec contraintes d'inégalité

Sujets du chapitre.

- Analyse des conditions d'optimalité (équations ou inéquations satisfaites pour toute solution du problème d'optimisation).
- Hypothèses sur les contraintes.
- Déduction d'une famille d'algorithmes itératifs globalement convergents à partir de la violation des conditions d'optimalité.
- Présentation d'algorithmes de pénalités simples et augmentées.

Introduction

Les problèmes d'optimisation différentiable avec contraintes d'inégalité également différentiables constituent un sujet *moderne* en ce qu'il nécessite des outils d'analyse d'inéquations par opposition aux outils d'analyse d'équations utilisés jusqu'à maintenant. En fait, dans une mesure limitée, la programmation linéaire et l'optimisation avec contraintes d'inégalités linéaires constituent également des sujets modernes, mais nous avons pu les traiter avec des outils simples d'algèbre linéaire.

Dans ce chapitre, nous restreignons les problèmes que nous traitons à ne comporter *que des contraintes d'inégalités*, et supposons que les gradients des contraintes satisfaites avec égalités (contraintes actives) sont linéairement indépendants. Ainsi, l'analyse du chapitre précédents est facile à adapter, et nous pouvons aborder rapidement la déduction et l'étude d'algorithmes de résolution. Au prochain chapitre, nous reprendrons l'étude fondamentale de problèmes avec contraintes générales.

Comme au chapitre précédent, nous utilisons le concept de DED pour identifier des équations à satisfaire (conditions d'optimalité) pour les solutions locales du problème

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f(x) \\ &\text{sujet à } g(x) \leq 0, \end{aligned} \tag{6.1}$$

où $g : \mathbb{R}^n \to \mathbb{R}^m$. g est un vecteur *colonne* de fonctions. Les fonctions f et g sont différentiables, et en général, supposées de classe $\mathcal{C}^3(\mathbb{R}^n)$.

Ensuite, en perturbant les conditions d'optimalité retenues, nous déduirons des algorithmes de pénalité.

6.1 Conditions d'optimalité

Avant d'introduire les conditions d'optimalité, revenons sur la notion de direction réalisable. Soit x tel que $g(x) \leq 0$. Considérons une suite y_k, avec $g(y_k) \leq 0$ telle que $y_k \to x$. Considérons maintenant les vecteurs $d_k = \frac{y_k - x}{\|y_k - x\|}$. Tout point d'accumulation de la suite d_k est nommé *direction réalisable* en x pour les contraintes g. Ces directions, en fait, constituent le *cône tangent* aux contraintes en x. Dans ce chapitre, puisque nous supposons que les gradients des contraintes actives sont linéairement indépendants, on peut exprimer sans difficulté le cône tangent à l'aide des linéarisations des contraintes.

6.1.1 Conditions de premier ordre

Comme précédemment, pour obtenir des conditions d'optimalité de premier ordre en un point x^*, nous devons nous assurer qu'aucune direction réalisable en x^* n'appartient au $DED(x^*)$. Le $DED(x^*)$ est bien défini comme $\{d : \nabla f(x^*)d < 0\}$, et donc l'obtention de conditions d'optimalité (du premier ordre) passe par une caractérisation des directions réalisables.

Directions réalisables : cône tangent aux contraintes

Soit une suite $d_k = \frac{y_k - x}{\|y_k - x\|}$ convergeant vers la direction réalisable $\bar{d}$ en un point x. Alors, on a que pour chacune des composantes g_i de l'application g, le développement de Taylor limité nous donne

$$0 \geq g_i(y_k) = g_i(x) + \varphi_k \nabla g_i(x) d_k + \mathcal{O}(\varphi_k),$$

où $\varphi_k = \|y_k - x\|$. Puisque $g_i(y_k) \leq 0$ et $g_i(x) \leq 0$ il est commode de distinguer deux cas :

1. $g_i(x) < 0$: dans ce cas, quelle que soit la suite y_k convergeant vers x, éventuellement on aura que $g_i(y_k) < 0$, et donc aucune analyse additionnelle n'est requise.
2. $g_i(x) = 0$: dans l'expression de $g_i(y_k)$, il ne reste que

$$0 \geq g_i(y_k) = \nabla g_i(x) d_k \sim \mathcal{O}(\varphi_k),$$

et en passant à la limite,

$$\nabla g_i(x) \bar{d} \leq 0.$$

Donc, pour toutes les composantes i de l'application g telles que $g_i(x) = 0$, ce qui s'écrit

$$\nabla g_{I^*}(x) \bar{d} \leq 0,$$

où ∇g_{I^*} représente le Jacobien de l'application g_{I^*}, c'est-à-dire une matrice dont les lignes sont $\nabla g_i(x), i \in I^*$. Donc, toute direction réalisable d en x satisfait à la relation $\nabla g_{I^*}(x) d \leq 0$.

Comme au chapitre précédent, introduisons une hypothèse de qualification des contraintes g. Cette hypothèse stipule que x^* est un point régulier du système $g_{I^*}(x) = 0$, c'est-à-dire, la matrice $\nabla g_{I^*}(x^*)$ est une matrice de plein rang-lignes, $|I^*| \leq n$. Cette hypothèse permet de montrer que toute direction satisfaisant $\nabla g_{I^*}(x) d \leq 0$ est une direction réalisable, complétant ainsi la caractérisation des directions réalisables. Remarquons que cette qualification dépends des fonctions g et non pas de l'ensemble des points réalisables, qui pourrait être exprimé à l'aide d'autres contraintes. Nous approfondirons l'étude des qualifications des contraintes dans un contexte plus général au prochain chapitre.

Théorème 6.1.1 *Soit x^* un point régulier des contraintes $g(x^*) \leq 0$, c'est-à-dire que la matrice $\nabla g_{I^*}(x^*)$ est une matrice de plein rang. Alors, d est une direction réalisable en x^* si et seulement si $\nabla g_{I^*}(x^*) d \leq 0$.*

Exercice 6.1.1 *[Preuve]* Imitez la preuve du théorème 5.1.1 pour démontrer le théorème 6.1.1. □

On voit donc que les directions réalisables coïncident avec les directions réalisables des contraintes linéaires $0 \geq L(y) = g(x) + \nabla g_{I^*}(x) y$. Par conséquent, nous pouvons appliquer les développements du chapitre 4 pour obtenir des conditions d'optimalité.

Condition du premier ordre

Théorème 6.1.2 *Soit x^* un minimum local de la fonction f sous les contraintes $g(x) \leq 0$ tel que $\nabla g_{I^*}(x^*)$ est une matrice de plein rang-ligne ; alors, il existe un vecteur $0 \leq \lambda^* \in \mathbb{R}^m$ tel que $\nabla f(x^*) + {\lambda^*}^t \nabla g(x^*) = 0$ et $\lambda_i^* g_i(x^*) = 0$; ceci est équivalent à la condition ${Z^*}^t \nabla f(x^*) = 0$, où les colonnes de Z^* forment une base du noyau de la matrice $\nabla g_{I^*}(x^*)$.*

6.1.2 Conditions de second ordre

Nous venons de voir que les conditions de premier ordre expriment le fait que les directions réalisables, donc appartenant au cône tangent aux contraintes $g(x) \leq 0$, ne sont pas des directions de descente stricte.

Théorème 6.1.3 *Si x^*, point régulier des contraintes $g(x) \leq 0$, est un minimum local du programme (6.1), alors il existe un vecteur $0 \leq \lambda^* \in I\!R^m$ tel que $\nabla_x L(x^*, \lambda^*) = 0$, $\nabla_\lambda L(x^*, \lambda^*)^t \lambda^* = 0$ et de plus $d^t \nabla^2_{xx} L(x^*, \lambda^*) d \geq 0$ pour tout d tel que $\nabla g_{I^*}(x^*) d = 0$. Par ailleurs, s'il existe un vecteur $0 \leq \lambda^* \in I\!R^m$ tel que $\nabla_x L(x^*, \lambda^*) = 0$, $\nabla_\lambda L(x^*, \lambda^*)^t \lambda^* = 0$, et de plus $d^t \nabla^2_{xx} L(x^*, \lambda^*) d > 0$ pour tout d tel que $\nabla g_{I^*_\lambda}(x^*) d = 0$, alors x^* est un minimum local pour le programme (6.1).*

Remarquons que l'hypothèse de qualification des contraintes n'est requise que pour les conditions nécessaires, c'est-à-dire pour démontrer l'existence de multiplicateurs λ^*.

6.1.3 Une analyse basée sur les pénalités

Il est remarquable que l'on puisse déduire les conditions d'optimalité nécessaires à partir des fonctions de pénalité introduites au chapitre précédent. De plus, la preuve simple qui suit ne suppose pas la condition d'indépendance linéaire des contraintes actives, mais plutôt une condition de régularité beaucoup plus faible, en fait, la condition la plus faible assurant que l'ensemble des multiplicateurs de Lagrange est non-vide et borné.

Considérons le problème général sous la forme suivante :

$$\min_{x \in I\!R^n} \quad f(x) \tag{6.2}$$

$$\text{sujet à} \quad g(x) \leq 0, \tag{6.3}$$

$$h(x) = 0. \tag{6.4}$$

Théorème 6.1.4 *Soit un minimum local x^* pour lequel la condition de régularité suivante est satisfaite :*

$$\left\{ \begin{array}{c} \lambda \nabla g(x^*) + \mu \nabla h(x^*) = 0 \\ \lambda g(x^*) = 0 \\ \lambda \geq 0 \end{array} \right\} \Longrightarrow \left\{ \begin{array}{c} \lambda = 0 \\ \mu = 0 \end{array} \right\}.$$

Alors, il existe des vecteurs $\lambda^* \geq 0$ et μ^* tels que

$$\nabla f(x^*) + \lambda^* \nabla g_{I^*}(x^*) + \mu^* \nabla h(x^*) = 0.$$

Preuve Tout d'abord, on peut récrire les contraintes $g(x) \leq 0$ à l'aide de l'opérateur de partie positive $g_i^+(x) = max(g_i(x), 0)$ et son extension vectorielle comme $g^+(x) = 0$. On peut donc exprimer les contraintes à l'aide de la fonction $\tilde{h}(x) = \binom{g^+(x)}{h(x)} = 0$.

Puisque x^* est un optimum local, alors il existe un voisinage V de x^*, que l'on peut prendre convexe, tel que $f(x) \geq f(x^*) \forall x \in C \cap V$. En remplaçant au besoin la fonction objectif par $\tilde{f}(x) = f(x) + \epsilon\|x - x^*\|^2$, on peut supposer que x^* est en fait l'unique minimum global du problème :

$$\begin{aligned} &\min_{x \in V} \tilde{f}(x) \\ &\text{sujet à } \tilde{h}(x) = 0. \end{aligned} \tag{6.5}$$

Alors, on peut envisager la pénalisation quadratique telle que vue au chapitre précédent,

$$\min_{x \in V} \phi_k(x) = \tilde{f}(x) + \frac{1}{2\rho_k}\|\tilde{h}(x)\|^2.$$

Puisque l'on minimise dans V, le minimum de la fonction ϕ_k est toujours atteint. Or, d'après l'exercice 5.2.1, tout point d'accumulation de la suite $x^k = x(\rho_k)$ est un minimum global pour le programme 6.5, dont l'unique minimum global est x^*.

Pour que la preuve soit complète, justifions la convergence globale des x^k en écrivant :

$$\phi_k(x^k) \le \phi_k(x^{k+1}) \le \phi_{k+1}(x^{k+1}) \le \phi_{k+1}(x^k),$$

d'où l'on tire que $\tilde{f}(x^{k+1}) \ge \tilde{f}(x^k)$, $\phi_k(x^k) \le \inf_{x:\tilde{h}(x)=0} \phi_k(x) = \tilde{f}(x^*)$; si $\rho_k \searrow 0$, la dernière inégalité nous apprend que $\tilde{h}(x^k) \to 0$. Soit un point d'accumulation $\bar{x}$, comme $\tilde{h}(\bar{x}) = 0$, $f(\bar{x}) \ge f(x^*)$, ce qui entraîne que $\bar{x} = x^*$ puisque x^* est l'unique minimum global dans V.

Donc, on a $x^k \to x^*$, et si k est assez grand, $x^k \in int(V)$. Puisque x^k minimise $\phi_k(x)$, on doit avoir, lorsque $x^k \in int(V)$, que $\nabla\phi_k(x^k) = 0$, donc

$$\nabla\tilde{f}(x^k) + \frac{\tilde{h}(x^k)^t}{\rho_k}\nabla\tilde{h}(x^k) = 0.$$

Considérons maintenant la suite $\nu^k = \frac{\tilde{h}(x^k)}{\rho_k}$. Deux cas peuvent se produire : cette suite peut être bornée, ou non-bornée. Si elle est bornée, ses points d'accumulation $\bar{\nu}$ satisfont à la condition recherchée, et on a

$$\nabla\tilde{f}(x^*) + \bar{\nu}\nabla\tilde{h}(x^*) = \nabla\tilde{f}(x^*) + (\bar{\lambda}, \bar{\mu})\begin{pmatrix}\nabla g^+(x^*) \\ \nabla h(x^*)\end{pmatrix} = 0.$$

Si cette suite n'est pas bornée, considérons la suite bornée $\frac{\nu^k}{\|\nu^k\|}$ et un de ses points d'accumulation $\bar{\nu} \neq 0$. On retrouve

$$\frac{\nabla\tilde{f}(x^k)}{\|\nu^k\|} + \frac{\nu^k}{\|\nu^k\|}\nabla\tilde{h}(x^k) = 0;$$

Or, $\|\nu^k\|$ diverge, de telle sorte que l'on a

$$\bar{\nu}\nabla\tilde{h}(x^*) = (\bar{\lambda}, \bar{\mu})\begin{pmatrix}\nabla g^+(x^*) \\ \nabla h(x^*)\end{pmatrix} = 0,$$

ce qui contredit la condition de régularité. □

La condition de régularité s'exprime comme une généralisation naturelle de l'indépendance linéaire des gradients des contraintes actives. En fait, cette condition est équivalente à la condition usuelle de Mangasarian-Fromowitz qui s'exprime comme :

$$(MF) \qquad \exists d : \begin{pmatrix} \nabla h(x^*)d = 0 \\ \nabla g(x^*)d < 0 \end{pmatrix}.$$

L'équivalence des deux conditions est démontrée en considérant le programme linéaire

$$\begin{array}{rl} \min\limits_{d,u \in \boldsymbol{R}^n \times \boldsymbol{R}} & u \\ \text{sujet à} & \nabla g(x)d \leq u, \\ & \nabla h(x)d = 0, \end{array}$$

et son dual,

$$\begin{array}{rl} \max\limits_{y \in \boldsymbol{R}^m} & 0 \\ \text{sujet à} & \nabla g(x)^t y_1 + \nabla h(x) y_2 = 0, \\ & e^t y_1 = -1 \\ & y_1 \geq 0. \end{array}$$

Lorsque le dual est réalisable, la régularité n'est pas satisfaite, et la valeur de u dans le primal est nulle, donc (MF) n'est pas satisfaite. Autrement, la régularité est satisfaite, et le primal n'est pas borné inférieurement, donc (MF) est satisfaite.

Corollaire 6.1.1 *Soit un minimum local x^* pour lequel la condition de régularité est satisfaite. Alors, l'ensemble*

$$\Lambda^* = \{\lambda, \mu : \nabla f(x^*) + \lambda \nabla g_{I^*}(x^*) + \mu \nabla h(x^*) = 0\}$$

est non-vide et borné.

6.2 Construction de plusieurs familles d'algorithmes de pénalité

Contentons-nous pour l'instant de produire un algorithme qui calcule des points stationnaires pour le programme (6.1), c'est-à-dire des points qui satisfont aux conditions d'optimalité de premier ordre : il existe un vecteur $\lambda^* \in \mathbb{R}^m$ tel que

$$\begin{aligned} \nabla f(x^*) + \lambda^{*t} \nabla g(x^*) &= 0 \\ g(x^*) &\leq 0 \\ \lambda_i^* g_i(x^*) &= 0. \end{aligned} \tag{6.6}$$

Comme au chapitre précédent, nous allons perturber les conditions d'optimalité pour en déduire des algorithmes de pénalité, des trajectoires, et analyser le comportement asymptotique de ces algorithmes. On peut envisager de perturber soit l'inéquation $g(x) \leq 0$, soit la condition de complémentarité $\lambda_i g_i(x) = 0$. De plus, l'inégalité $g(x) \leq 0$ se prête à deux formes de perturbations conduisant à des algorithmes distincts.

6.2.1 Perturbation des conditions d'optimalité I : $g(x) \leq 0$

De la condition de complémentarité, nous savons que lorsque $g_i(x) < 0$, $\lambda_i = 0$. Perturbons donc l'inégalité $g(x) \leq 0$ par l'égalité équivalente $\max(g(x), 0) = 0$ et perturbons cette dernière en remplaçant le 0 de droite par $\rho\lambda$, où ρ est un paramètre que nous laisserons tendre vers zéro. Le but de cette manœuvre est de calculer des solutions $x(\rho)$ dépendant du paramètre ρ en espérant pouvoir montrer que $\lim_{\rho \searrow 0} x(\rho) = x^*$. Cet manipulation permet d'écrire, en notant $g^+(x) \stackrel{\text{def}}{=} \max(g(x), 0)$

$$\nabla f(x) + \lambda^t \nabla g(x) = 0$$
$$g^+(x) = \rho\lambda,$$

d'où l'on tire que $\lambda = \frac{g^+(x)}{\rho}$, ce qui permet de réduire ces deux équations à une seule :

$$P(x, \rho) \stackrel{\text{def}}{=} \nabla f(x) + \frac{g^+(x)}{\rho}^t \nabla g(x) = 0,$$

qui sont précisément les conditions d'optimalité du problème

$$\min_{x \in \boldsymbol{R}^n} p(x, \rho) \stackrel{\text{def}}{=} f(x) + \frac{\|g^+(x)\|^2}{2\rho}.$$

Ce dernier problème est nommé *pénalisation quadratique* du programme (6.1).

Un inconvénient majeur de ces fonctions de pénalité vient des termes g_i^+, qui ne sont pas différentiables aux points où la fonction $g_i = 0$. Puisque ces termes apparaîssent au carré, la fonction $p(x, \rho)$ est différentiable, mais pas de classe $\mathcal{C}^2$. Par ailleurs, au voisinage de minima satisfaisant aux conditions suffisantes d'optimalité et à la condition de stricte complémentarité, la convergence se fait de l'*extérieur* des contraintes actives, d'où le nom de *méthodes de pénalité extérieures* sous lequel ces algorithmes sont connus. En effet, si la complémentarité stricte est satisfaite au point limite, puisque $\frac{g^+(x)}{\rho}$ s'approche de λ^*, si $g_i(x)$ s'approche de 0, $\frac{g_i(x)}{\rho}$ s'approche d'une quantité positive, d'où l'interprétation que x s'approche de l'extérieur. Dans ce cas, la discontinuité des dérivées secondes n'a pas d'importance puisque les itérations demeurent dans un ensemble (l'extérieur des contraintes actives et l'intérieur des autres) où les dérivées secondes sont continues.

6.2.2 Perturbation des conditions d'optimalité II : $\lambda g(x) = 0$

Une autre possibilité consiste à perturber le terme de complémentarité. La perturbation la plus simple que l'on puisse imaginer est $\lambda g(x) = -\rho$, où encore une fois $\rho \searrow 0$. Supposons maintenant que les solutions des conditions d'optimalité ainsi perturbées satisfont $g(x(\rho)) < 0$ ainsi que $\lambda(\rho) > 0$, et donc l'inégalité $g(x) \leq 0$ est toujours satisfaite et $\lambda = \frac{-\rho}{g(x)}$ peut être substitué dans la première relation, ce qui donne

$$B(x, \rho) = \nabla f(x) - \sum \frac{\rho}{g_i(x)} \nabla g_i(x) = 0,$$

et correspond aux conditions d'optimalité de

$$\min b(x, \rho) = f(x) - \rho \sum \log(-g_i(x)).$$

Par la définition de la fonction *log*, les minima locaux de la fonction $b(x, \rho)$ seront tous des points *intérieurs* aux contraintes, et les valeur implicites $\lambda_i = -\frac{\rho}{g_i(x)}$ seront strictement positives, donc dans l'*intérieur* de l'ensemble $\lambda \geq 0$. Ces algorithmes sont connus sous le nom de *méthodes intérieures.*

6.2.3 Perturbation des conditions d'optimalité III : méthodes semi-intérieures

Une autre famille perturbe les contraintes, mais en utilisant un vecteur $\lambda > 0$ intérieur. On parle donc de perturbation conduisant à une pénalisation extérieure pour les variables x et intérieure pour les variables λ. Il s'agit d'écrire $g(x) = \rho \log \lambda$. On peut alors substituer $\lambda_i = e^{\frac{g_i(x)}{\rho}}$ et obtenir

$$P_E(x, \rho) = \nabla f(x) + \sum e^{\frac{g_i(x)}{\rho}} \nabla g_i(x)$$

et

$$p_e(x, \rho) = f(x) + \rho \sum e^{\frac{g_i(x)}{\rho}}.$$

La stratégie symétrique, de considérer une perturbation intérieure pour les variables x et extérieure pour les variables λ conduit aux formulations suivantes, déduites de la perturbation $\lambda = -\rho \log(-g(x))$:

$$P_L(x, \rho) = \nabla f(x) - \rho \sum \log(-g_i(x)) \nabla g_i(x)$$

et

$$p_l(x, \rho) = f(x) + \rho \sum g_i(x)(\log(-g_i(x)) - 1).$$

6.3 Propriétés générales des algorithmes de pénalisation

Nous étudions d'abord quelques propriétés générales des algorithmes de pénalité.

Les algorithmes extérieurs possèdent des propriétés similaires aux algorithmes de pénalité quadratique présentés au chapitre précédent. Par ailleurs, les deux autres types de pénalités possèdent également des propriétés semblables.

6.3.1 Propriété de convergence globale

Tout comme au chapitre précédent, observons que les points d'accumulation d'une suite de minima *globaux* des fonctions pénalisées constituent des minima globaux du problème 6.1.

Explicitons un tel résultat abstrait pour une suite d'ϵ_k-minima des fonctions pénalisées. Considérons une pénalisation extérieure (quadratique ou exponentielle) et une suite de points x_k tels que

$$p(x_k, \rho_k) \leq \min_{x \in \boldsymbol{R}^n} p(x, \rho_k) + \epsilon_k,$$

où la suite $\epsilon_k \searrow 0$.

Théorème 6.3.1 *Supposons que les fonctions f et g_i sont continues, avec f bornée inférieurement. Soit $x_k \in I\!R^n$ un ϵ_k-minimum de $p(x, \rho_k)$, où $\epsilon_k \geq 0$ et $\rho_k > 0$ convergent toutes deux vers zéro. Alors, tout point d'accumulation de la suite $\{x_k\}$ est un minimum global du problème 6.1. La suite $\{x_k\}$ admets au moins un point d'accumulation dès que f est inf-compact, ou de façon plus générale, lorsqu'il existe $\eta > 0$ tel que*

$$\lim_{\substack{\|x\| \to \infty \\ c_i(x) \leq \eta}} f(x) = +\infty \tag{6.7}$$

Preuve Soit $\bar{x}$ un point d'accumulation de la suite $\{x_k\}$. Quitte à extraire une sous-suite, supposons que la suite est convergente, $\lim_{k\to\infty} x_k = \bar{x}$. L'$\epsilon_k$-optimalité nous donne

$$p(x_k, \rho_k) \leq p(x, \rho_k) + \epsilon_k \qquad \forall\ x \in I\!R^n$$

et donc

$$p(x_k, \rho_k) \leq \inf_{g_i(x) \leq 0} f(x) + \rho_k m + \epsilon_k, \tag{6.8}$$

où m borne la pénalisation dans le domaine réalisable ; on peut remplacer m par 0 pour la pénalisation quadratique. On en déduit que x^* doit être réalisable car autrement, le côté gauche de la précédente relation diverge alors que le côté droit converge vers $\inf_{g_i(x)\leq 0} f(x)$. De plus, puisque $f(x_k) \leq p(x_k, \rho_k)$ nous déduisons en passant à la limite que

$$f(\bar{x}) \leq \inf_{g_i(x) \leq 0} f(x)$$

ce qui démontre l'optimalité de $\bar{x}$.

Pour justifier l'existence de points d'accumulation, supposons que f est bornée inférieurement et que la relation (6.7) est satisfaite. Alors, la suite $\{x_k\}$ doit être bornée car de la relation (6.8), on tire que $\rho_k \sum \exp(g_i(x_k)/\rho_k)$ est borné et donc $g_i(x_k) \leq \eta$ pour tout $i = 1, \ldots, m$ et tout k suffisamment grand. Également de la relation (6.8), on déduit que $f(x_k)$ est bornée supérieurement, de sorte que la conclusion est obtenue à partir de la relation (6.7). □

Ce résultat montre la convergence de minima globaux des sous-problèmes pénalisés vers un minimum global du problème 6.1. Comme pour l'algorithme de pénalité quadratique appliqué à des problèmes avec contraintes d'égalité, le mieux que l'on puisse montrer au sujet d'une suite de minima locaux est le fait que les points d'accumulation satisfont aux conditions nécessaires d'optimisation.

Théorème 6.3.2 *Supposons qu'un point d'accumulation $\bar{x}$ de la suite $\{x(\rho_k)\}$ soit un point régulier des contraintes actives. Supposons de plus que les points $x(\rho_k)$ sont des minima locaux des sous-problèmes pénalisés. Alors, $\bar{x}$ satisfait aux conditions nécessaires pour un minimum local du problème (6.1).*

Exercice 6.3.1 *[Convergence globale]* Démontrez le théorème 6.3.2 pour la pénalité exponentielle et pour la pénalité logarithmique □

Exercice 6.3.2 *[Convergence faible]* On peut construire un exemple d'une seule variable à l'aide d'une fonction f telle que $f'(x^*) = f''(x^*) = 0$, x^* est un *maximum* global de f et une contrainte telle que $g(x^*) < 0$ pour laquelle x^* est un minimum local de $b(x, \rho)$ et ce pour tout $\rho > 0$.

a) Fournissez des exemples de fonctions f et g, illustrant ainsi qu'une suite de minima locaux peut converger vers un maximum global.

b) Comment se comporte la pénalité exponentielle sur votre exemple ?

c) Pourquoi la pénalité quadratique ne présente pas la même faiblesse sur votre exemple ?

d) Construisez un exemple pour lequel une suite de minima locaux de la pénalité quadratique converge vers un maximum global.

□

Remarquons que les exemples de convergence vers des maxima globaux ne contredisent pas le théorème 6.3.2 : un maximum global peut très bien satisfaire aux conditions nécessaire d'optimalité pour un minimum!

6.3.2 Propriété d'identification des contraintes actives

Considérons une sous-suite convergente $\{x_k\}$: $\lim_{k\to\infty} x_k = \bar{x}$. Nommons $\bar{I}$ l'ensemble des indices pour lesquels $g_i(\bar{x}) = 0$. Alors, $g_i(x_k)$ converge vers zéro pour les $i \in \bar{I}$ et vers une certaine constante négative pour les autres indices $i \notin \bar{I}$. Si l'on peut évaluer la vitesse à laquelle $g_{\bar{I}}(x_k)$ s'approche de zéro, il devient possible d'identifier les contraintes actives. Nous montrerons plus loin que sous certaines conditions, $\|x_k - \bar{x}\| \sim \mathcal{O}(\rho_k)$ tout comme c'était le cas pour la pénalité quadratique du chapitre précédent. Alors, asymptotiquement, les contraintes de $\bar{I}$ sont celles qui vérifient $g_i(x_k) \geq -(\rho_k^{\nu}), 1 < \nu < 1$.

D'autre part, sous l'hypothèse que $\lambda_{\bar{I}} > 0$, la pénalité quadratique approche le domaine réalisable de l'extérieur, c'est-à-dire que pour des valeurs assez grandes de k, $g_i(x_k) > 0$, $i \in \bar{I}$ alors que bien entendu, $g_i(x_k) < 0$, $i \notin \bar{I}$.

6.3.3 Propriété numériques

Nous avons soulevé au chapitre précédent le problème du mauvais conditionnement des sous-problèmes pénalisés lorsque ρ_k s'approche de zéro. Bien évidemment, ce problème afflige tout autant les algorithmes de pénalité introduits dans ce chapitre.

6.4 Propriétés asymptotiques des algorithmes

Afin d'analyser les algorithmes introduits ci-haut, nous allons utiliser des hypothèses qui semblent difficile à contourner. Puisque nous nous intéressons aux propriétés asymptotiques, nous supposons évidemment que la suite $\{x_k\}$ converge vers un certain x^*.

H1 x^* satisfait aux conditions suffisante d'optimalité.

H2 $\nabla g_{I^*}(x^*)$ est une matrice de plein rang.

H3 $\lambda_{I^*} > 0$.

À l'aide de ces hypothèses, nous pouvons imiter pas à pas l'analyse que nous avons faite au chapitre précédent.

Constatons que les contraintes actives, celles de I^*, jouent un peu le même rôle que les contraintes d'égalité du chapitre précédent. En effet, ces contraintes sont satisfaites avec égalité alors que les autres contraintes n'ont que peu d'influence localement. En fait, pour les pénalités extérieures, les contraintes inactives disparaîssent littéralement car $(g_i(x))^+ = 0$ dans un voisinage de x^*. Par conséquent, l'analyse asymptotique du chapitre précédent demeure valable car les hypothèses H1 et H2 correspondent aux hypothèses du chapitre précédent alors que l'hypothèse H3 assure que, localement, $(g_i(x))^+ = g_i(x) > 0$ pour les $i \in I^*$.

Nous examinons dans la suite les deux autres pénalités (logarithmique et exponentielle). Pour ces dernières, l'influence des contraintes inactives devient négligeable mais nécessite tout de même d'adapter les développements.

6.4.1 Élimination du mauvais conditionnement

Les techniques permettant de contourner le problème du mauvais conditionnement peuvent s'adapter aux contraintes d'inégalité à condition de connaître l'ensemble I^*. Supposons que nous connaissions I^*, et voyons comment transformer les équations. Dans une section subséquente, nous discuterons du problème d'estimer I^*. Nous effectuons les développements pour la pénalité logarithmique, laissant en exercice les détails pour la pénalité exponentielle.

Méthode primale-duale bien conditionnée

Revenons aux équations

$$\begin{aligned} \nabla f(x) + \lambda^t \nabla g(x) &= 0 \\ g(x)\lambda &= \rho. \end{aligned}$$

Séparons les contraintes en I^* et J

$$\begin{aligned} \nabla f(x) + {\lambda_{I^*}}^t \nabla g_{I^*}(x) + {\lambda_J}^t \nabla g_J(x) &= 0 \\ g_i(x) &= \frac{\rho}{\lambda_i} \quad i \in I^* \\ \lambda_j &= \frac{\rho}{g_j(x)} \quad j \notin I^*. \end{aligned}$$

Le Jacobien de ce dernier système est le suivant :

$$\begin{pmatrix} \nabla^2 f(x) + \sum \lambda_i \nabla^2 g_i(x) & \nabla g_{I^*}(x) & \nabla g_J(x) \\ \nabla g_{I^*}^t(x) & diag(\frac{\rho}{\lambda_i^2}), i \in I^* & 0 \\ diag(\frac{\rho}{g_j^2(x)}), j \notin I^* & 0 & I \end{pmatrix}.$$

En $\rho = 0$, on retrouve

$$\left(\begin{array}{cc|c} \nabla^2 f(x^*) + \sum \lambda_i \nabla^2 g_i(x^*) & \nabla g_{I^*}(x^*) & \nabla g_J(x^*) \\ \nabla g_{I^*}^t(x^*) & 0 & 0 \\ \hline 0 & 0 & I \end{array}\right).$$

Cette dernière matrice est inversible car son premier bloc est inversible (lemme 5.3.1).

Méthode primale bien conditionnée

Reprenons l'équation du hessien de la pénalisation

$$H(x,\rho) = \nabla^2 f(x) + \sum \frac{\rho}{g_i(x)} \nabla^2 g(x) + \sum \frac{\rho}{g_i^2(x)} \nabla g_i(x) \nabla g_i(x)^t,$$

et observons que le mauvais conditionnement n'a lieu que dans les sous-espaces image de la matrice ∇g_{I^*}. Puisque cette matrice est supposée de plein rang, il existe une matrice Q_x orthogonale telle que

$$Q_x \nabla g_{I^*}(x)^t = \begin{pmatrix} U \\ 0 \end{pmatrix},$$

avec U une matrice triangulaire supérieure, de sorte que

$$Q_x \nabla g(x)^t = \begin{pmatrix} U & R \\ 0 & S \end{pmatrix}.$$

Par conséquent,

$$Q_x H(x,\rho) Q_x^t$$

se réécrit

$$Q_x \left(\nabla^2 f(x) + \sum \frac{\rho}{g_i(x)} \nabla^2 g(x) \right) Q_x^t + \begin{pmatrix} U & R \\ 0 & S \end{pmatrix} diag \left(\frac{\rho}{g_i^2(x)} \right) \begin{pmatrix} U^t & 0 \\ R^t & S^t \end{pmatrix}.$$

En dénotant par $G = Q_x \left(\nabla^2 f(x) + \sum \frac{\rho}{g_i(x)} \nabla^2 g(x) \right) Q_x^t$, et par $\Delta = diag(\frac{\rho}{g_i^2(x)})$, on retrouve

$$G + \begin{pmatrix} U\Delta_1 U^t + R\Delta_2 R^t & R\Delta_2 S^t \\ S\Delta_2 R^t & S\Delta_2 S^t \end{pmatrix}.$$

Remarquons maintenant que $\rho\Delta_1$ s'approche de $\Lambda_{I^*}^2 = diag(\lambda_i^2), i \in I^*$ alors que Δ_2 converge vers zéro. Le mauvais conditionnement est ainsi enlevé en multipliant le premier groupe d'équations par ρ :

$$\begin{pmatrix} \rho G_{11} & \rho G_{12} \\ G_{21} & G_{22} \end{pmatrix} + \begin{pmatrix} U\Lambda_{I^*}^2 U^t + \rho R\Delta_2 R^t & \rho R\Delta_2 S^t \\ S\bar{\Delta}_2 R^t & S\bar{\Delta}_2 S^t \end{pmatrix}$$

qui converge vers la matrice inversible

$$\begin{pmatrix} U^* \Lambda_{I^*}^2 U^{*t} & 0 \\ G_{21}^* & G_{22}^* \end{pmatrix}.$$

Lemme 6.4.1 *Si x^* est un point régulier des contraintes, minimum local du programme (6.1) satisfaisant aux conditions suffisantes de second ordre, alors la matrice*

$$M = \begin{pmatrix} U^* \Lambda_{I^*}^2 U^{*t} & 0 \\ G_{21}^* & G_{22}^* \end{pmatrix}$$

est inversible.

Exercice 6.4.1 *[Inversibilité de la matrice M]* Démontrez le lemme 6.4.1. □

On peut maintenant utiliser ces transformations pour calculer la direction de Newton ou encore montrer la différentiabilité des trajectoires en $\rho = 0$.

6.4.2 Ordre (superlinéaire) de convergence locale

Comme au chapitre précédent, nous étudions la convergence locale d'un algorithme pour lequel les sous-problèmes ne sont résolus qu'approximativement. On a toujours la différentiabilité des trajectoires approchées en autant que l'algorithme converge vers un point régulier des contraintes actives qui satisfait aux conditions suffisantes d'optimalité et à la condition de stricte complémentarité. Ces conditions sont *suffisantes* pour démontrer la différentiabilité des trajectoires. Elles ne sont toutefois pas nécessaires, comme nous le verrons dans un chapitre ultérieur, où nous obtiendrons ce résultat sans hypothèse additionnelle pour la programmation linéaire.

Théorème 6.4.1 *Soit x^* un point régulier des contraintes $g_{I^*}(x) = 0$ satisfaisant aux conditions suffisantes de second ordre pour le programme (6.1) et à la condition de stricte complémentarité $\lambda_{I^*} > 0$. Si les fonctions f et g sont de classe $\mathcal{C}^p(I\!R^n)$, alors, il existe des trajectoires différentiables $x(\rho, \Phi)$ et $\lambda(\rho, \Phi)$ de classe $\mathcal{C}^{p-1}(I\!R^n)$ telles que*

1. *$x(0,0) = x^*$ et $\lambda(0,0) = \lambda^*$;*
2. *si ρ et $\|\Phi\|$ sont assez petits, $x(\rho, 0)$ satisfait aux conditions suffisantes de second ordre pour le problème pénalisé $\min f(x) - \rho \sum_{i=1}^m \log(-g_i(x))$, où $x(\rho, \Phi)$, $\lambda(\rho, \Phi)$ sont solutions du système d'équations*

$$\begin{aligned} \nabla f(x) + \lambda^t \nabla g(x) &= \Phi \\ g_i(x)\lambda_i + \rho &= 0 \quad , i = 1, 2, \dots, m. \end{aligned}$$

De plus, les bornes suivantes sont satisfaites :

(a) $\|x(\rho, \Phi) - x^*\| \sim \mathcal{O}(\max(\rho, \|\Phi\|))$;

(b) $\|\lambda(\rho, \Phi) - \lambda^*\| \sim \mathcal{O}(\max(\rho, \|\Phi\|))$;

(c) $\|g(x(\rho, \Phi))\| \sim \mathcal{O}(\rho)$.

Exercice 6.4.2 *[Preuve du théorème 6.4.1]* Démontrez le théorème 6.4.1. □

Nous conservons la propriété importante que la direction de Newton est $\sim \mathcal{O}(\rho_k^2)$.

Lemme 6.4.2 *Sous les mêmes hypothèses que le théorème 6.4.1, si f et g sont des fonctions de classe $\mathcal{C}^2(I\!R^n)$, alors $\|P(\tilde{x}_{k+1}, \rho_{k+1})\| \sim \mathcal{O}(\frac{\rho_k^2}{\rho_{k+1}})$ et la direction de Newton d_N calculée au point $\tilde{x}_{k+1}$ satisfait à la borne $\|d_N\| \sim \mathcal{O}(\rho_k^2)$.*

Exercice 6.4.3 *[Preuve du lemme 6.4.2]* Démontrez le lemme 6.4.2. □

L'ordre de convergence est un peu moins bon que pour l'algorithme de pénalité quadratique. Ceci est dû au fait que le terme d'ordre deux du développement de Taylor de $P(x, \rho)$ fait apparaître des $g_i^2(x)$ au dénominateur alors que la quantité $1/\rho_{k+1}$ n'était pas modifiée par les différentiations successives dans le cas de la pénalité quadratique.

Lemme 6.4.3 *Sous les mêmes hypothèses que le théorème 6.4.1, si f et g sont des fonctions de classe $\mathcal{C}^3(I\!R^n)$, soit d_N la direction de Newton calculée à $\tilde{x}_{k+1}$, et $\bar{x}_{k+1} = \tilde{x}_{k+1} + d_N$. Alors,*

$$\|P(\bar{x}_{k+1}, \rho_{k+1})\| \sim \mathcal{O}\left(\frac{\rho_k^4}{\rho_{k+1}^2}\right)$$

Exercice 6.4.4 *[Preuve du lemme 6.4.3]* Démontrez le théorème 6.4.3. □

6.4.3 Algorithmes globalement et superlinéairement convergents

L'algorithme 5.1 peut être utilisé conjointement à la pénalité logarithmique ou encore la pénalité exponentielle.

Comme tout algorithme de pénalité, lorsqu'au moins une composante de λ^* est non-nulle, donc lorsqu'il y a au moins une contrainte active, le point extrapolé est retenu localement.

Lemme 6.4.4 *Sous les hypothèses du théorème 6.4.1, si $\lambda^* \neq 0$, asymptotiquement, le point extrapolé sera toujours utilisé.*

Exercice 6.4.5 *[Preuve du lemme 6.4.4]* Démontrez le théorème 6.4.4. □

Lemme 6.4.5 *Sous les mêmes hypothèses que le théorème 6.4.1 et pour des valeurs assez petites de ρ_k, il existe deux constantes positives γ_0 and γ_1 telles que*

$$d_N^t P(\tilde{x}) \leq -\gamma_0 \rho_k \|P(\tilde{x})\| \|d_N\|$$
$$\|d_N\| \leq \gamma_1 \|P(\tilde{x})\|.$$

Lemme 6.4.6 *Sous les mêmes hypothèses que le théorème 6.4.1 et pour des valeurs assez petites de ρ_k, pour une constante $0 < \tau_0 < \frac{1}{2}$, la relation suivante est satisfaite :*

$$p(\tilde{x}_{k+1} + d_N, \rho_{k+1}) - p(\tilde{x}_{k+1}, \rho_{k+1}) \leq \tau_0 d_N^t P(\tilde{x}_{k+1}, \rho_{k+1}).$$

Les lemmes techniques précédents fournissent les outils pour démontrer un théorème de convergence globale localement superlinéaire. Cette fois-ci, il faut restreindre la suite $\rho_{k+1} = \rho_k^\alpha$ à un ordre limite $\alpha < 4/3$.

Théorème 6.4.2 *L'algorithme 5.1 utilisant la pénalité logarithmique ou exponentielle est bien défini. Si un point d'accumulation de l'algorithme est un point régulier des contraintes actives, alors il satisfait aux conditions nécessaires d'optimalité. De plus, s'il s'agit d'un point satisfaisant aux conditions suffisantes d'optimalité où la condition de stricte complémentarité est satisfaite, alors pour des valeurs de ρ_k assez petites, on a*

1. *Une seule itération de Newton (modifiée avec une recherche linéaire d'Armijo) depuis $\hat{x}_{k+1}$ fournit directement $x(\rho_{k+1}, \xi_{k+1})$ avec $\|\xi_k\| \leq \epsilon_k$*
2. *de plus, si au moins une composante de λ^* est non-nulle (i.e. s'il y a au moins une contrainte active), la suite $x(\rho_{k+1}, \xi_{k+1})$ converge superlinéairement avec un ordre α.*

Exercice 6.4.6 *[Pénalité exponentielle]* Effectuez l'analyse de convergence locale pour la pénalité p_e. □

Exercice 6.4.7 *[]* Identifiez un obstacle important dans l'analyse de la pénalité p_l. □

6.5 Pénalités et Lagrangiens augmentés

6.5.1 Pénalités extérieures

En s'inspirant du Lagrangien augmenté, si l'on a un estimé $\bar{\lambda} \geq 0$ de λ^*, on peut utiliser la perturbation $(g(x) + \rho\bar{\lambda})^+ = \rho\lambda$, ce qui conduit au problème sans contrainte

$$\min_{x \in \boldsymbol{R}^n} p_a(x, \bar{\lambda}, \rho) \stackrel{\text{def}}{=} f(x) + \sum \begin{cases} \bar{\lambda}_i g_i(x) + \frac{\|g(x)\|^2}{2\rho} & \text{si } (g_i(x) + \rho\bar{\lambda}_i) \geq 0 \\ -\rho\bar{\lambda}_i^2 & \text{si } (g_i(x) + \rho\bar{\lambda}_i) \leq 0 \end{cases}.$$

6.5.2 Pénalités intérieures

Il semble difficile d'adapter une stratégie de mise-à-jour des variables duales au sein des méthodes intérieures. En effet, un Lagrangien augmenté de la barrière logarithmique conduit à la mise-à-jour

$$\lambda^+ \leftarrow \lambda + \frac{\rho}{g(x)},$$

ce qui entraîne que $g(x)$ converge vers zéro lentement.

6.5.3 Pénalités primales-extérieures

La pénalité exponentielle se prête bien à une forme multiplicative de mise-à-jour. Considérons la pénalisation suivante, où $\bar{\lambda}$ constitue un estimé de λ^*

$$p_{ea}(x, \bar{\lambda}, \rho) = f(x) + \rho \sum \bar{\lambda}_i e^{\frac{g_i(x)}{\rho}},$$

avec son gradient

$$P_{EA}(x, \bar{\lambda}, \rho) = \nabla f(x) + \sum \bar{\lambda}_i e^{\frac{g_i(x)}{\rho}} \nabla g_i(x).$$

Ceci correspond à la perturbation

$$g_i(x) = \rho \log\left(\frac{\lambda_i}{\bar{\lambda}_i}\right).$$

Puisque $\bar{\lambda}$ est destiné à approcher λ^*, $e^{\frac{g_i(x)}{\rho}}$ s'approche de l'unité, ce qui entraîne que $g_{I^*}(x)$ converge vers zéro plus vite que ρ. Par ailleurs, λ_J s'approche très vite de zéro car $g_J(x)$ s'approche d'une quantité négative et que la fonction $e^{\frac{-1}{\rho}}$ s'approche de zéro plus vite que tout polynôme en ρ. De plus, pour les contraintes actives,

$$\log\left(\frac{\lambda_i}{\bar{\lambda}_i}\right) = \log(\lambda_i) - \log(\bar{\lambda}_i) \sim \mathcal{O}(\|\lambda_{I^*} - \bar{\lambda}_{I^*}\|).$$

On peut donc envisager utiliser le changement de variables $\mu_i = \rho \log(\frac{\lambda_i^*}{\bar{\lambda}_i}), \quad i \in I^*$ et $\mu_j = \bar{\lambda}_j, \quad j \notin I^*$ pour établir la différentiabilité de fonctions $x(\rho, \mu, \Phi)$ et $\lambda(\rho, \mu, \Phi)$ où $\lambda(\rho, \mu, \Phi) = \bar{\lambda}_i e^{\frac{g_i(x)}{\rho}}$, et en déduire l'équivalent du théorème 5.4.2 :

Théorème 6.5.1 *Soit x^* un point régulier des contraintes $g_{I^*}(x) = 0$ satisfaisant aux conditions suffisantes de second ordre pour le programme (6.1) et à la condition de stricte complémentarité $\lambda_{I^*} > 0$. Si les fonctions f et g sont de classe $\mathcal{C}^p(\mathbb{R}^n)$, alors, il existe des trajectoires différentiables $x(\rho, \bar{\lambda}, \Phi)$ et $\lambda(\rho, \bar{\lambda}, \Phi)$ telles que*

1. *$x(0, \lambda^*, 0) = x^*$ et $\lambda(0, \lambda^*, 0) = \lambda^*$;*
2. *Si ρ, $\|\Phi\|$ et $\|\bar{\lambda} - \lambda^*\|$ sont assez petits, alors $x(\rho, \bar{\lambda}, 0)$ satisfait aux conditions suffisantes de second ordre pour le problème pénalisé $f(x) + \sum \bar{\lambda}_i e^{\frac{g_i(x)}{\rho}}$, et où $x(\rho, \bar{\lambda}, \Phi)$ et $\lambda(\rho, \bar{\lambda}, \Phi)$*

sont solutions du système d'équations

$$\nabla f(x) + \lambda \nabla g(x) = \Phi$$
$$g_i(x) = \rho \log \left(\frac{\lambda_i}{\bar{\lambda}_i}\right) \qquad i \in I^*;$$
$$\lambda_i = \bar{\lambda}_i e^{\frac{g_i(x)}{\rho}} \qquad i \notin I^*;$$

de plus, les bornes suivantes sont satisfaites :

(a) $\|x(\rho, \bar{\lambda}, \Phi) - x^*\| \sim \mathcal{O}(\max(\rho\|\bar{\lambda} - \lambda^*\|, \|\Phi\|))$ *;*

(b) $\|\lambda(\rho, \bar{\lambda}, \Phi) - \lambda^*\| \sim \mathcal{O}(\max(\rho\|\bar{\lambda} - \lambda^*\|, \|\Phi\|))$.

Exercice 6.5.1 *[]* Démontrez le théorème 6.5.1. □

6.6 Extensions et références

L'analyse d'algorithmes de pénalité (méthodes de points intérieurs) a reçu un regain d'attention depuis les travaux de Karmarkar[20]. Dussault[11] a analysé une famille abstraite d'algorithmes de pénalités, dont la pénalité logarithmique est un cas particulier ; il a introduit la technique basée sur la connaissance des contraintes actives pour contourner le problème du mauvais conditionnement. L'analyse de la pénalité exponentielle est basée sur [5].

6.7 Résumé

Nous avons déduit d'observations élémentaires des conditions d'optimalité pour les problèmes différentiables : $\nabla f(x^*)$ doit être orthogonal au plan tangent aux contraintes actives en x^*, ce qui, en autant que x^* soit un point régulier des contraintes actives, est équivalent à ce que $\nabla f(x^*)$ soit orthogonal au noyau de la matrice $\nabla g(x^*)$, dont les colonnes de la matrice Z^* forment une base. Ainsi, $Z^{*t}\nabla f(x^*) = 0$. De plus, le hessien réduit du Lagrangien doit être (semi-)défini positif.

Nous avons déduit l'algorithme de pénalité par le biais d'une perturbation des équations de Lagrange, et avons présenté ses propriétés de convergence globale et de convergence locale superlinéaire. De plus, nous avons présenté deux techniques pour contourner le problème du mauvais conditionnement (qui constitue l'objection habituelle face aux algorithmes de pénalité). Il en résulte un algorithme robuste et efficace.

Bibliographie

[1] E. Beale. A derivation of conjugate gradients. In F. Lootsma, editor, *Numerical Methods for Nonlinear Optimization*, pages 39–42, Academic Press, London and New York, 1972.

[2] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods.* Academic press, New-York, 1982.

[3] C. G. Broyden and N. Attia. A smooth sequential penalty function method for solving nonlinear programming problems. In P. Thoft-Christenses, editor, *System Modelling and Optimization*, pages 237–245, Springler-Verlag, Berlin, Germany, 1983.

[4] C. G. Broyden and N. Attia. Penalty functions, newton's method, and quadratic programming. *Journal of Optimization Theory and Applications*, 58, 1988.

[5] R. Cominetti and J.-P. Dussault. A stable exponential-penalty algorithm with superlinear convergence. À paraître dans J.O.T.A.

[6] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the Annals of the Mathematics Society*, 49 :1–23, 1943.

[7] G. B. Dantzig. *Linear programming and extensions.* Princeton university press, Princeton, N.J., 1963.

[8] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact newton methods. *S.I.A.M. Journal on Numerical Analysis*, 19(2) :400–408, 1982.

[9] R. S. Dembo and T. Steihaug. Truncated-newton algorithms for large-scale unconstrained optimization. *Mathematical Programming*, 26 :190–230, 1983.

[10] J. E. Dennis Jr. and R. B. Schnabel. *Numerical methods for nonlinear equations and unconstrained optimization.* Prentice Hall, 1983.

[11] J.-P. Dussault. Numerical stability and efficiency of penalty algorithms. À paraître dans S.I.A.M. Journal on Numerical Analysis.

[12] A. V. Fiacco and G. P. McCormick. Nonlinear programming : Sequential unconstrained minimization techniques. *S.I.A.M.*, 1990.

[13] R. Fletcher and C. Reeves. Function minimization by conjugate gradients. *The computer journal*, pages 149–154, 1964.

[14] P. E. Gill, W. Murray, and S. Picken. The implementation of two modified newton algorithms for unconstrained optimization. NPL Report NAC24, 1972.

[15] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, 1981.

[16] N. I. M. Gould. On the accurate determination of search directions for simple differentiable penalty functions. *I.M.A. Journal on Numerical Analysis*, 6 :357–372, 1986.

[17] N. I. M. Gould. On the convergence of a sequential penalty function method for constrained minimization. *S.I.A.M. Journal on Numerical Analysis*, 26 :107–108, 1989.

[18] M. R. Hestenes. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4 :303–320, 1969.

[19] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. N.B.S.*, 49 :409–436, 1952.

[20] N. K. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4 :373—395, 1984.

[21] W. Murray. Second derivative methods. In W. Murray, editor, *Numerical methods for unconstrained optimization*, Academic press, London, 1972.

[22] C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization.* Prentice Hall, 1982.

[23] E. Polak. *Computational methods in optimization : a unified approach.* Academic press, New-York, 1971.

[24] M. J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*, Academic Press, London, 1969.

[25] R. T. Rockafellar. The multiplier method of Hestenes and Powell applied to convex programming. *Journal of Optimization Theory and Applications*, 12 :555–562, 1973.

[26] R. T. Rockafellar. Augmented lagrange multiplier functions and duality in nonconvex programming. *S.I.A.M. Journal on Control and Optimization*, 12 :268–285, 1974.