Synthèse sur les méthodes newtoniennes en optimisation numérique non linéaire: écriture d'algorithmes efficaces

Matthieu Guilbert

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Synthèse sur les méthodes newtoniennes en optimisation numérique non linéaire: écriture d'algorithmes efficaces

Matthieu Guilbert

N° 0325

Août 2006

Thème NUM

ISSN 0249-0803

Synthèse sur les méthodes newtoniennes en optimisation numérique non linéaire: écriture d'algorithmes efficaces

Matthieu Guilbert

Thème NUM — Systèmes numériques
Projets Bipop

Rapport technique n° 0325 — Août 2006 — 12 pages

Résumé : Ce document décrit les méthodes newtoniennes en optimisation non-linéaire avec et sans contrainte. Après un bref rappel des conditions d'optimalité, nous présentons les principes généraux de l'algorithmique en optimisation et des méthodes de Newton. Comme les méthodes newtoniennes sont locales, c'est à dire qu'elles sont uniquement valables proche de la solution, il faut utiliser des techniques de globalisation de la recherche de l'optimum, ces techniques ont certains avantages, mais elles peuvent entraîner des problèmes numériques, surtout en optimisation avec contraintes. Pour calculer une direction de descente, les méthodes newtoniennes ont besoin des dérivées secondes de la fonction à minimiser (où du Lagrangien dans le cas d'une minimisation avec contraintes) ce qui peut être coûteux et peut causer des problèmes numériques sérieux; c'est pourquoi nous exposons une méthode alternative de quasi-Newton dans les cas avec et sans contraintes.

Mots-clés : Newton, quasi-Newton, Lagrangien, Pénalisation exacte, Effet Maratos

Unité de recherche INRIA Rhône-Alpes

Synthesis on newtonian methods in non-linear numerical optimization: to efficient algorithms writing

Abstract: This document deals with numerical methods in non-linear optimization with and without constraints. After a brief recapitulation of the optimality conditions, we present the general principles of the algorithmic in optimization and of newtonian methods. Since these methods are only valid near the solution, we must use globalization techniques, such techniques have several advantages but they can severely unsettle the convergence when minimizing a function subject to constraints. Moreover to compute a descent direction, Newton methods need second derivatives of the function to minimize (or of the Lagrangian in the case of minimization with constraints), what may be costly and may cause serious numerical problems; that is why we expose another way of finding a solution using quasi-Newton methods.

Key-words: Newton, quasi-Newton, Lagrangian, Exact penalty function, Maratos effect

Table des matières

1 Introduction

L'objectif de ce document est d'exposer les méthodes Newtoniennes en optimisation non-linéaire de manière très succincte mais en insistant sur les problèmes numériques inhérents à ce genre de méthodes. J'espére que cette synthèse sur les méthodes Newtoniennes permettra au lecteur de se repérer dans les méandres de la littérature sur ce sujet, mais surtout qu'il comprendra mieux le fonctionnement et le comportement de ces algorithmes très complexes. Mais avant tout, je tiens à remercier tout particulièrement Claude Lemaréchal pour ses conseils avisés et ses relectures indispensables.

L'optimisation non-linéaire cherche à résoudre des problèmes du type :

$$\min_{x \in \mathbb{R}^n} F(x)$$

$$\begin{aligned} c_E(x) &= 0 \quad (1) \\ c_I(x) &\leq 0 \quad (2) \end{aligned}$$

avec $F : \mathbb{R}^n \to \mathbb{R}$ la fonction objectif à minimiser, $c_E : \mathbb{R}^n \to \mathbb{R}^{me}$ les contraintes dégalités et $c_I : \mathbb{R}^n \leftrightarrow \mathbb{R}^{mi}$ les contraintes d'inégalités. Ce document expose dans une première partie les méthodes Newtoniennes en optimisation sans contrainte, et ceci afin d'exposer la structure générale d'un algorithme d'optimisation tout en insistant sur les algorithmes de quasi-Newton qui sont couramment utilisés aujourd'hui. Une seconde partie expose les méthodes Newtoniennes en optimisation avec contraintes, tout en insistant sur des effets numériques indésirables liés à l'utilisation d'une fonction de pénalisation non différentiable utilisée pour globaliser la recherche du minimum.

2 Méthodes numériques d'Optimisation sans Contrainte

Un problème d'optimisation sans contrainte s'écrit :

$$\min_{x \in \mathbb{R}^n} F(x).$$

Il existe différentes méthodes pour trouver une solution à ce problème [BGLS03], mais nous allons nous concentrer sur les méthodes de Newton et quasi-Newton après avoir exposé brièvement les conditions d'optimalité (les conditions que doit satisfaire F au minimum). Nous insistons sur les méthodes de Newton car elles sont aujourd'hui les plus efficaces pour trouver le minimum d'une fonction non-linéaire générique.

2.1 Conditions d'Optimalité [HU01]

$$\begin{aligned} \text{Premier ordre :} \quad & \nabla_x F(x^*) = 0 \\ \text{Second ordre :} \quad & \nabla_x^2 F(x^*) \succeq 0 \end{aligned}$$

avec x^* le minimum, $\nabla_x F(x^*)$ le gradient de F au minimum, et $\nabla_x^2 F(x^*)$ la hessienne de F au minimum et $\nabla_x^2 F(x^*) \succeq 0$ désignant la semi-définie positivité de $\nabla_x^2 F(x^*)$ au minimum. Il est facile d'avoir une explication intuitive de ces conditions. En effet si x^* est un minimum local de F alors $dF \geq 0$ (dF est une différentielle de F) pour toute différentielle dx issue de x^* (pour les notions de différentielles, le lecteur peut se référer à [BMP04]), il est alors facile de démontrer que $\nabla_x F(x^*)$ doit être nul, mais aussi que $\nabla_x^2 F(x^*) \succeq 0$. Connaissant les conditions que doit satisfaire F au minimum et partant d'un itéré initial x_0, une méthode de Newton permet de calculer une approximation au second ordre de l'optimum si on est assez proche de x^*.

2.2 Méthodes Numériques de Newton

Le but des méthodes numériques en optimisation est de trouver une suite $\{x_k\}$ qui converge vers x^*. A chaque itération on cherche un pas d_k qui fasse décroître la fonction objectif, le schéma numérique d'un algorithme d'optimisation se présente comme suit :

La plus grande difficulté dans ce schéma est de trouver l'incrément d_k et un bon premier itéré x_0. Le choix du premier itéré est laissé à l'utilisateur qui pourra le choisir en fonction de ses connaissances sur le problème. En ce qui concerne l'incrément d_k, il est important de souligner qu'il doit faire diminuer la fonction objectif. En effet la condition $F(x_k + d_k) < F(x_k)$ est fondamentale en optimisation numérique, tout algorithme d'optimisation

(i)	Initialiser $k \leftarrow 0$, l'itéré initial x_0 et la tolérance d'arrêt ϵ
(ii)	Trouver un incrément d_k telle que $F(x_k + d_k) < F(x_k)$
(iii)	$x_{k+1} \leftarrow x_k + d_k$
(iv)	Si $\|\nabla_x F(x_{k+1})\| > \epsilon$ aller au pas (ii) sinon s'arrêter

TAB. 1 – Schéma numérique d'un algorithme d'optimisation

doit forcer la descente de la fonction objectif. Il existe diverses méthodes pour trouver des directions de descente [BGLS03, p. 2-35], mais nous allons nous concentrer sur les méthodes de Newton qui possèdent aujourd'hui les meilleures propriétés de convergence par rapport aux (trop simples) méthodes de gradient, par exemple.

Partant d'un itéré courant x_k, notre objectif est de trouver un pas d_k qui fasse décroître F voire qui trouve le minimum de F, c'est à dire que $\nabla_x F(x_k + d_k) = 0$. Pour cela, linéarisons les condtions d'optimalité en faisant un développement de Taylor :

$$\nabla_x F(x_k + d_k) = \nabla_x F(x_k) + \nabla_x^2 F(x_k) d_k + o(\|d_k\|). \tag{3}$$

Négligeons le terme $o(\|d_k\|)$, on obtient alors le système linéaire :

$$\nabla_x F(x_k) + \nabla_x^2 F(x_k) d_k = 0 \tag{4}$$

à résoudre pour trouver l'incrément. Si on est assez proche du minimum alors $\nabla_x^2 F(x_k)$ est semi-définie positive et peut donc être inversée, on obtient alors le pas de descente :

$$d_k = -\left(\nabla_x^2 F(x_k)\right)^{-1} \nabla_x F(x_k) \tag{5}$$

Le gros avantage d'une méthode de Newton est bien connu : elle converge très rapidement [BGLS03, p. 51]. Si $\nabla_x^2 F(x_k)$ est continue et inversible à proximité de x^*, la méthode de Newton converge superlinéairement, si en plus F est de classe C^3 elle converge quadratiquement. Pour que la méthode de Newton soit efficace, il faut que x_k soit assez proche de x^*, dans le cas contraire la convergence ne peut pas être assurée, il faut adapter la méthode pour forcer la convergence globale.

Pour cela d_k est dorénavant considérée comme une direction le long de laquelle une recherche linéaire est utilisée afin de diminuer la fonction $q(t) = F(x_k + td_k)$, et dans ce cas on a $x_{k+1} \leftarrow x_k + td_k$. Pour une description détaillée des différentes recherches linéaires, on peut se référer à [BGLS03, 37-50].

Il reste cependant un problème de taille, c'est le calcul de la hessienne exacte $\nabla_x^2 F(x_k)$ qui peut être lourd (en considérant qu'elle est symétrique, il y a $\frac{1}{2}n(n+2)$ éléments à calculer). Un deuxième problème est que loin du minimum, elle n'est pas forcément définie positive. Ce problème peut être résolu en utilisant une hessienne augmentée de termes sur la diagonale afin de la rendre définie positive. On utilise alors $(\nabla_x^2 F(x_k) + \nu I_n)$ (où I_n est la matrice identité de dimension $n \times n$) au lieu de la hessienne, c'est la méthode de Levenberg-Marquardt ([Fle87, p. 48]) pour contrer le mauvais conditionnement de la hessienne (elle est souvent utilisée pour des problèmes de moindre carrés).

Remarque 2.1. *Le système linéaire (4) correspond aux conditions d'optimalité du problème quadratique suivant :*

$$(\mathcal{P}_{QT}) \quad \min_{d_k \in \mathbb{R}^n} F(x_k) + \nabla_x F(x_k)^T d_k + \frac{1}{2} d_k^T \nabla_x^2 F(x_k) d_k \tag{6}$$

qui est appelé "problème quadratique tangent" car il correspond à minimiser une approximation quadratique du problème initial. C'est un point clé dans la définition des Programmes Quadratiques Successifs (PQS ou SQP en anglais) utilisés dans l'optimisation avec contraintes. On peut dès à présent voir les méthodes de Newton comme une succession de problèmes quadratiques tangents à résoudre.

2.3 Méthodes Numériques de Quasi-Newton

Considérons le schéma numérique de Newton de la section précédente, mais plutôt que de calculer la hessienne exacte à chaque itération et de résoudre le système linéaire pour l'inverser, une autre idée serait de trouver directement une approximation de $\left(\nabla_x^2 F(x_k)\right)^{-1}$ par une matrice H_k ; c'est ce que fait le schéma 2. La question dans ce shéma réside dans le pas (iv) : trouver une matrice d'adaptation C_k telle que :

– elle soit plus facile à calculer que $\left(\nabla_x^2 F(x_k)\right)^{-1}$

(i)	Initialiser $k \leftarrow 0$, $H_k \leftarrow I_n$, l'itéré initial x_0, et la tolérance d'arrêt ϵ
(ii)	$d_k \leftarrow -H_k \nabla_x F(x_k)$
(iii)	Faire une recherche linéaire le long de d_k sur $q(t_k) = F(x_k + t_k d_k)$
(iv)	$x_{k+1} \leftarrow x_k + t_k d_k$
(iv)	$H_{k+1} \leftarrow H_k + C_k$, $k \leftarrow k+1$,
(v)	Si $\|\nabla_x F(x_{k+1})\| > \epsilon$ aller au pas (ii) sinon s'arrêter

TAB. 2 – Schéma numérique d'un algorithme de quasi-Newton

- H_{k+1} soit toujours définie positive
- seules les dérivées premières de la fonction F soient nécessaires

Il nous faut trouver une relation entre H_{k+1} et $\nabla_x^2 F(x_k)$, pour cela faisons un développement de Taylor du gradient :

$$\nabla_x F(x_k + \delta_k) = \nabla_x F(x_k) + \nabla_x^2 F(x_k)\delta_k + o(\|\delta_k\|), \tag{7}$$

en posant $\gamma_k = \nabla_x F(x_{k+1}) - \nabla_x F(x_k)$ et $\delta_k = x_{k+1} - x_k$, on peut alors écrire :

$$\gamma_k = \nabla_x^2 F(x_k)\delta_k + o(\|\delta_k\|). \tag{8}$$

Afin que H_{k+1} mime correctement l'inverse de la hessienne dans la direction définie par γ_k, il faut que :

$$H_{k+1}\gamma_k = \delta_k; \tag{9}$$

cette relation est souvent appelée relation fondamentale de quasi-Newton. Toutes les méthodes d'adaptation de H_k devront, quoi qu'il se passe, respecter cette condition. La manière la plus simple pour mettre à jour H_k est de l'augmenter d'une matrice de rang 1 ; on renvoie à [Fle87, p.53] pour des détails sur cette méthode. Les méthodes qui nous intéressent plus particulièrement sont les méthodes de mise à jour de rang 2, c'est à dire qu'on ajoute deux matrices de rang 1 à H_k :

$$H_{k+1} = H_k + auu^T + bvv^T. \tag{10}$$

On peut alors appliquer le principe fondamental (9) sur la formule (10), on obtient :

$$\delta_k = H_k\gamma_k + auu^T\gamma_k + bvv^T\gamma_k, \tag{11}$$

un choix logique est donc $u = \delta_k$ et $v = H_k\gamma_k$, on peut alors identifier a et b en posant $au^T\gamma_k = 1$ et $bv^T\gamma_k = -1$, on obtient alors la formule de mise à jour de Davidon, Fletcher et Powell (DFP) :

$$H_{k+1}^{DFP} = H + \frac{\delta\delta^T}{\delta^T\gamma} - \frac{H\gamma\gamma^T H}{\gamma^T H\gamma}. \tag{12}$$

On remarquera qu'on a omis l'indice k dans le second membre de l'équation (12) par souci de simplification. En faisant d'autres choix lors de l'identification de a, b, u, v, on peut obtenir la mise à jour de Broyden, Fletcher, Goldfarb et Shanno (BFGS) :

$$H_{k+1}^{BFGS} = H + \left(1 + \frac{\gamma^T H\gamma}{\delta^T\gamma}\right)\frac{\delta\delta^T}{\delta^T\gamma} - \left(\frac{\delta\gamma^T H + H\gamma\delta^T}{\delta^T\gamma}\right) \tag{13}$$

Il est intéressant de calculer l'inverse de H_{k+1}^{BFGS}, notons la B_{k+1}^{BFGS} :

$$B_{k+1}^{BFGS} = B + \frac{\gamma\gamma^T}{\gamma^T\delta} - \frac{B\delta\delta^T B}{\delta^T B\delta}. \tag{14}$$

La ressemblance est notable avec la formule DFP (12), il suffit d'échanger γ avec δ et B avec H et on obtient la formule DFP ; ces deux formules sont dites complémentaires ou duales, on renvoie le lecteur à [Fle87, p. 49-68] pour plus de détails à ce sujet. On a donc une matrice qui est mise à jour au cours des itérations, qui est plus simple à calculer que la hessienne exacte, qui mime son comportement dans la direction de descente et qui n'utilise que des dérivées premières. Cependant les formules DFP ou BFGS n'imposent pas que H_{k+1} (ou B_{k+1}) soit définie positive, néanmoins dans [Fle87, p. 54] et [BGLS03, p. 56] on peut trouver le théorème suivant :

Théorème 2.1. *Si $\delta_k\gamma_k > 0$ alors les formules BFGS ou DFP assurent que H_{k+1} est définie positive si H_k l'est.*

Remarque 2.2. *Si dans le schéma numérique (1), on utilise une formule d'adaptation DFP ou BFGS, alors la condition $\delta_k\gamma_k > 0$ est automatiquement satisfaite si on utilise la recherche linéaire dite de Wolfe [BGLS03, section 3.4].*

Un algorithme de quasi-Newton utilisant la formule BFGS et la recherche linéaire de Wolfe converge alors superlinéairement vers x^* ; pour la démonstration on renvoie à [BGLS03, p. 65].

2.4 Ce qu'il faut retenir

Les méthodes de Newton utilisent les dérivées premières et secondes de la fonction à minimiser pour calculer un pas de descente d_k à partir de la formule (5) ; d_k est aussi la solution du problème quadratique tangent (6). Cette méthode est valable si on est assez proche du minimum de F. Dans le cas où on est loin, il faut globaliser l'optimisation en utilisant une recherche linéaire le long de d_k. L'avantage d'une méthode de Newton est qu'elle converge superlinéairement (voire quadratiquement), mais il faut calculer la hessienne de la fonction à minimiser ce qui peut être coûteux. Un autre problème est que loin de l'optimum, la hessienne peut ne pas être définie positive et donc ne pas être inversible ou d_k ne pas être une direction de descente. Les méthodes de quasi Newton répondent à ces problèmes en mettant à jour une matrice qui mime la hessienne dans la direction indiquée par les gradients. Cette matrice peut être mise à jour à chaque itération par la formule de BFGS (13), elle est plus facile à calculer que la hessienne exacte et est toujours définie positive sous la condition indiquée dans le théorème 2.1. Cette dernière condition est automatiquement vérifiée si on utilise la recherche linéaire de Wolfe. Un algorithme de quasi-Newton complet peut donc se dérouler ainsi comme dans la table (2) en utilisant une recherche linéaire de Wolfe.

3 Méthodes numériques d'Optimisation avec Contraintes

Un problème d'optimisation avec contraintes s'écrit comme :

$$\min_{x\in\mathbb{R}^n} F(x)$$

$$\begin{aligned} c_E(x) &= 0 && (15)\\ c_I(x) &\leq 0, && (16) \end{aligned}$$

avec $c_E(x) : \mathbb{R}^n \to \mathbb{R}^{m_E}$ et $c_I(x) : \mathbb{R}^n \to \mathbb{R}^{m_I}$, on notera dorénavant $m = m_E + m_I$. Il existe plusieurs méthodes pour trouver une solution à ce problème [Fle87]. Après un exposé bref des conditions d'optimalité d'un tel problème, nous allons exposer les méthodes de type Programmation Quadratique Successive (PQS) basées sur les méthodes de Newton. Enfin, nous nous concentrerons tout particulièrement sur les méthodes quasi-Newton qui sont largement utilisées aujourd'hui.

3.1 Conditions d'Optimalité

Les conditions d'optimalité du premier ordre d'un problème d'optimisation sous contraintes sont :

$$\begin{aligned} \nabla_x F(x^*) + \nabla_x c_E(x^*)^T\lambda_E^* + \nabla_x c_I(x^*)^T\lambda_I^* &= 0 && (17)\\ c_E(x^*) &= 0 && (18)\\ c_I(x^*) &\leq 0 && (19)\\ \lambda_I^* &\geq 0 && (20)\\ c_I(x^*)^T\lambda_I^* &= 0 && (21) \end{aligned}$$

avec λ_E et λ_I des vecteurs appelés multiplicateurs de Lagrange. En posant $\lambda = (\lambda_E, \lambda_I)^T$ et $c(x) = (c_E(x), c_I(x))^T$ et le lagrangien :

$$L(x, \lambda) = F(x) + c(x)\lambda^T \quad (22)$$

alors les conditions (17) et (18) expriment que :

$$\nabla_{(x,\lambda_E)} L(x,\lambda) = 0. \tag{23}$$

La condition de complémentarité (21) peut être vue comme une condition d'activation des contraintes, en effet si $c_I^k(x) < 0$ alors $\lambda_I^k = 0$ et si $c_I^k(x) = 0$ alors $\lambda_I^k \geq 0$, et ceci pour tout $1 \leq k \leq m$. Tout comme pour l'optimisation sans contrainte, on peut trouver une explication intuitive de ces conditions. En effet si x^* est un minimum local alors $dF \geq 0$ dans toutes les directions autorisées par les contraintes d'égalités et d'inégalités actives. La condition d'optimalité du second ordre est que la hessienne du Lagrangien doit être semi-définie positive sur le sous espace tangent (noyau de la jacobienne des contraintes actives), pour plus de détails le lecteur peut se référer à [HU01].

Remarque 3.1. *Pour éviter des problèmes d'optimisation pathologiques, il est commode de supposer que la jacobienne des contraintes actives soit de rang plein, pour plus de détails le lecteur peut se référer à [HU01]. Ces conditions de qualification des contraintes sont nécessaires pour écrire les conditions d'optimalité (17)-(21).*

Pour plus de détails sur les différentes interprétations géométriques ou analytiques et sur la théorie de l'optimisation sous contraintes, on renvoie le lecteur à [HU01, p. 54-58] et au chapitre 9 de [Fle87]. Maintenant qu'on connaît les conditions que doivent vérifier la fonction et les contraintes au minimum, les méthodes de type Newton sont à l'heure actuelle les plus efficaces pour résoudre un problème d'optimisation avec contraintes.

3.2 Programmation Quadratique Successive : contraintes d'égalité

Afin de mieux comprendre comment s'écrit un algorithme PQS, nous allons nous limiter aux contraintes d'égalité et ceci sans véritable perte de généralité. En effet, on peut estimer qu'en étant assez proche de l'optimum, l'état d'activation des contraintes d'inégalité ne change plus et on peut les traiter comme des contraintes d'égalité ; de plus on verra que l'activation des contraintes est faite au niveau du PQ (Programmation Quadratique ou QP en anglais) qui ne sera pas développé dans ce document. De manière identique au cas sans contrainte, on fait un développement de Taylor au premier ordre des conditions d'optimalité (23), on obtient alors :

$$\nabla_x L(x_k + dx, \lambda_k + d\lambda) \approx \nabla_x L(x_k,\lambda_k) + \nabla^2_{xx} L(x_k,\lambda_k)dx + \nabla^2_{x\lambda} L(x_k,\lambda_k)d\lambda \tag{24}$$

$$\nabla_\lambda L(x_k + dx, \lambda_k + d\lambda) \approx \nabla_\lambda L(x_k,\lambda_k) + \nabla^2_{\lambda x} L(x_k,\lambda_k)dx + \nabla^2_{\lambda\lambda} L(x_k,\lambda_k)d\lambda. \tag{25}$$

En écrivant que ceci doit être nul, on obtient alors le système linéaire :

$$\nabla_x L(x_k,\lambda_k) + \nabla^2_{xx} L(x_k,\lambda_k)dx + \nabla_x c(x_k)^T d\lambda = 0 \tag{26}$$

$$c(x_k) + \nabla_x c(x_k)dx = 0 \tag{27}$$

Une petite astuce consiste à remarquer que $\nabla_x L(x_k,\lambda_k)$ est linéaire par rapport à λ_k, l'équation (26) devient alors :

$$\nabla_x F(x_k) + \nabla^2_{xx} L(x_k,\lambda_k)dx + \nabla_x c(x_k)^T(\lambda_k + d\lambda) = 0 \tag{28}$$

Comme $\lambda_{k+1} = \lambda_k + d\lambda$, on obtient alors le nouveau système linéaire à résoudre :

$$\begin{pmatrix} \nabla^2_{xx} L(x_k,\lambda_k) & \nabla_x c(x_k)^T \\ \nabla_x c(x_k) & 0 \end{pmatrix} \begin{pmatrix} dx \\ \lambda_{k+1} \end{pmatrix} = - \begin{pmatrix} \nabla_x F(x_k) \\ c(x_k) \end{pmatrix} \tag{29}$$

Remarque 3.2. *La matrice de ce système est inversible si (i) $\nabla_x c(x_k)$ est de rang plein et (ii) $\nabla^2_{xx} L(x_k,\lambda_k)$ est définie positive sur le sous espace tangent (le noyau de $\nabla_x c(x_k)$) (cf remarque 3.1), pour plus de détails le lecteur peut se référer à [BGLS03, p.174].*

Dans le cas où il n'y a que des contraintes d'égalité, le système d'équations (29) n'est autre que les conditions d'optimalité du programme quadratique suivant :

$$\begin{cases} \min\limits_{d_k \in \mathbb{R}^n} \nabla_x F(x_k)^T dx_k + \dfrac{1}{2} dx_k^T \nabla^2_{xx} L(x_k,\lambda_k) dx_k \\ c_E(x_k) + \nabla_x c_E(x_k) dx_k = 0 \end{cases} \tag{30}$$

Sans rentrer dans les détails inutiles dans ce document, on peut étendre ce résultat aux contraintes d'inégalité et écrire le problème quadratique tangent au problème d'optimisation avec contraintes d'égalité et d'inégalité :

$$(\mathcal{P}_{QT}) \quad \begin{cases} \min\limits_{d_k \in \mathbb{R}^n} \nabla_x F(x_k)^T dx_k + \dfrac{1}{2} dx_k^T \nabla^2_{xx} L(x_k, \lambda_k) dx_k \\ c_E(x_k) + \nabla_x c_E(x_k) dx_k = 0 \\ c_I(x_k) + \nabla_x c_I(x_k) dx_k \leq 0 \end{cases} \tag{31}$$

Le problème d'activation des contraintes d'inégalités est donc renvoyé au niveau de la résolution du problème quadratique tangent. On peut alors écrire un PQS à hessienne exacte comme dans le schéma numérique 3.

(i)	Initialiser $k \leftarrow 0$, l'itéré initial (x_0, λ_0) et la tolérance d'arrêt ϵ
(ii)	Calculer $\nabla_x F(x_k)$ et $\nabla^2_{xx} L(x_k, \lambda_k)$
(ii)	Calculer dx_k et λ_{k+1} en résolvant $(\mathcal{P}_{QT})$ (31)
(iii)	$x_{k+1} \leftarrow x_k + d_k$
(iv)	Si $\|\nabla_{(x,\lambda)} L(x_{k+1}, \lambda_{k+1})\| > \epsilon$ aller au pas (ii) sinon s'arrêter

TAB. 3 – Schéma numérique d'un PQS à hessienne exacte

Le pas (ii) est l'étape non triviale de cet algorithme. La résolution d'un Programme Quadratique (PQ ou QP en anglais) est basée sur le système linéaire (29), cependant ce système ne traduit pas le problème lié à l'activation des contraintes. La résolution d'un PQ ne faisant pas l'objet de ce document, on renvoie le lecteur au chapitre 10 du livre [Fle87] pour de plus amples détails sur l'activation des contraintes.

Comme pour les méthodes de Newton sans contrainte, le talon d'Achille du schéma (3) est le calcul de la hessienne exacte qui peut être lourd, et elle peut ne pas être définie positive si on est loin du minimum.

3.3 Méthodes de Quasi-Newton avec Contraintes

Le développement des méthodes de quasi-Newton avec contraintes est basé sur le même principe que celui des problèmes sans contrainte exposé dans la section 2.3, il faut cependant remplacer la fonction $F(x)$ par le lagrangien $L(x, \lambda)$. Ainsi, en posant :

$$\gamma_k^l = \nabla_x L(x_{k+1}, \lambda_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}) \quad et \quad \delta_k = x_{k+1} - x_k, \tag{32}$$

on remarquera qu'on évalue le gradient du lagrangien en λ_{k+1} pour calculer γ_k^l, ceci se justifie par le fait qu'on connait ce λ_{k+1} et qu'on cherche à mimer la hessienne du lagrangien par rapport à x et pas par rapport à λ. De manière analogue au cas sans contrainte, on peut réécrire l'équation fondamentale de quasi-Newton (9), mais aussi la formule de BFGS pour la hessienne du lagrangien :

$$B_{k+1}^{Lagr} = B + \frac{\gamma\gamma^T}{\gamma^T\delta} - \frac{B\delta\delta^T B}{\delta^T B \delta}. \tag{33}$$

Pour globaliser l'algorithme, on peut utiliser une recherche linéaire ou une recherche par région de confiance. On fait le choix ici de décrire uniquement les méthodes avec recherche linéaire. Cette recherche doit être faite sur une fonction qui prenne en compte à la fois le critère à minimiser et les contraintes, de telles fonctions sont appelées fonctions de pénalisation.

3.3.1 Fonction de pénalisation non différentiable

Il existe une multitude de fonctions de pénalisation, cependant on peut classer ces fonctions en deux familles : les fonctions de pénalisation inexactes et exactes. Nous allons nous concentrer sur les fonctions de pénalisation exacte et plus précisément sur la pénalisation L_1 car elle est exacte sans faire appel à un schéma itératif. A l'inverse, d'autres pénalisations comme les fonctions barrières ou le lagrangien augmenté sont aussi couramment utilisées ; pour plus de détails on renvoie le lecteur à [Fle87, p.277-304]. La fonction de pénalisation L_1 est définie par :

$$\Phi(x) = F(x) + \sum_{i=1}^{m_E} \sigma_i |c_i(x)| + \sum_{i=m_E+1}^{m} \sigma_i |\max(c_i(x), 0)| \tag{34}$$

Cette méthode de pénalisation distingue les contraintes d'égalités et d'inégalités, en effet seules les contraintes d'inégalités actives influent sur la valeur de la fonction de pénalisation et cette activation est faite à l'aide

de la fonction max. Pour que la pénalisation (34) soit exacte, il faut que les σ_i soit assez grands pour que la croissance des c_i compense la décroissance de la fonction coût F. Ainsi plutôt que de fixer les σ_i très grands, ce qui pourrait entrainer des problèmes numériques (les contraintes seraient alors prépondérantes par rapport à la fonction coût), il serait plus judicieux de l'ajuster au cours de l'optimisation pour que sa valeur soit toujours juste assez grande. Pour cela, la proposition 15.1 de [BGLS03] propose une relation entre σ et λ_{k+1} (λ_{k+1} étant calculé en résolvant le problème quadratique tangent (31)) : si $\sigma_i \geq \|\lambda_{k+1}\|_\infty$ alors la pénalisation (34) est exacte. On peut alors établir une règle d'ajustement de σ_k comme dans la table (4).

si	$\sigma_{k-1} \geq 1.1(\|\lambda_{k+1}\|_\infty + \bar{\sigma})$,
alors	$\sigma_k = (\sigma_{k-1} + \|\lambda_{k+1}\|_\infty + \bar{\sigma})/2$;
sinon	si $\sigma_{k-1} \geq \|\lambda_{k+1}\|_\infty + \bar{\sigma}$,
	alors $\sigma_k = \sigma_{k-1}$.
	sinon $\sigma_k = \max(1.5\sigma_{k-1}, \|\lambda_{k+1}\|_\infty + \bar{\sigma})$.

TAB. 4 – Règle d'ajustement de σ

Ici $\bar{\sigma} > 0$ est une constante fixée au préalable, elle permet que σ_{k+1} soit toujours nettement plus grand que $\|\lambda_{k+1}\|_\infty$, c'est un peu une barrière de sécurité pour que σ_{k+1} soit toujours assez grand. Cette règle d'ajustement a été proposée historiquement par Mayne et Polak et on renvoie le lecteur à [BGLS03, p. 239-242] pour plus de détails sur cette règle d'ajustement de σ.

A cause des valeurs absolues et de la fonction max(.,.) dans la pénalisation (34), il est clair que cette dernière fonction n'est pas différentiable partout, et entre autre au minimum. Ceci risque de mettre en péril l'utilisation d'une recherche linéaire de type Armijo, Wolfe ou même Goldstein et Price qui ont au moins besoin de la dérivée directionnelle de la fonction de pénalisation dans la direction dx_k. Le lemme 14.3 de [BGLS03] indique que si $F(.)$, $c_E(.)$ et $c_I(.)$ admettent une dérivée directionnelle en x_k et dans la direction dx_k alors $\Phi(.)$ admet aussi une dérivée directionnelle le long de dx_k et elle vaut :

$$\Phi'(x_k, dx_k) = \nabla_x F(x_k) dx_k + \|S_k P_{c(x_k)} \nabla_x c(x_k) dx_k\|_1 \tag{35}$$

avec $S_k = diag(\sigma_{i,k})$, et l'opérateur $P_v u$ identique à l'opérateur utilisé dans [BGLS03, p. 225] et défini par :

$$(P_v u)_i = \begin{cases} u_i & si \quad 1 \leq i \leq m_E \\ \max(u_i, 0) & si \quad m_E \leq i \leq m \quad et \quad v_i = 0 \\ 0 & si \quad m_E \leq i \leq m \quad et \quad v_i < 0. \\ u_i & si \quad m_E \leq i \leq m \quad et \quad v_i > 0. \end{cases} \tag{36}$$

Cet opérateur permet une écriture synthétique de la dérivée directionnelle (35). On peut donc utiliser cette fonction de pénalisation pour faire une recherche linéaire et globaliser la recherche du minimum.

3.3.2 Recherche linéaire et correction de Powell

Dans les méthodes de quasi-Newton sans contrainte, on avait justifié l'emploi d'une recherche linéaire de Wolfe car elle permettait d'assurer la condition $\gamma_k \delta_k > 0$ qui implique que B_{k+1} est définie positive si B_k l'est. Dans le cas avec contraintes, on ne fait pas la recherche linéaire sur le lagrangien mais sur la fonction de pénalisation (34), ainsi même si on utilise une recherche linéaire de Wolfe on ne pourra jamais assurer la condition $\gamma_k^l \delta_k > 0$. C'est pourquoi la recherche linéaire de Armijo est souvent préférée à la recherche linéaire de Wolfe car elle est plus simple à implanter, les qualités de la recherche de Wolfe sont surtout utiles en optimisation sans contrainte. Si pour un pas unitaire la fonction de pénalisation ne diminue pas, le pas est divisé par deux par exemple (mais ce n'est pas le plus judicieux, on préférera utiliser la méthode du "cubic fitting" exposée dans [BGLS03, p. 40-41]). La recherche linéaire ne permet donc pas d'assurer la positivité de la matrice de BFGS.

Pour compenser ce problème, Powell a proposé de modifier γ_k^l (le γ de l'équation (33)) jusqu'à ce que le scalaire $(\gamma_k^l)^T \delta_k$ soit positif. Dans un premier temps, il faut calculer un pas t_k le long de la direction dx_k calculée par le PQ afin de diminuer une fonction de mérite (la fonction de pénalisation (34), par exemple) ce qui donne le nouvel itéré $x_{k+1} \leftarrow x_k + t_k dx_k$. γ_k^l et δ_k sont définies par (32), et le but est de calculer un γ_k^P comme une combinaison convexe de γ_k^l et $B_k^{Lagr} \delta_k$:

$$\gamma_k^P \leftarrow \theta \gamma_k^l + (1 - \theta) B_k^{Lagr} \delta_k \tag{37}$$

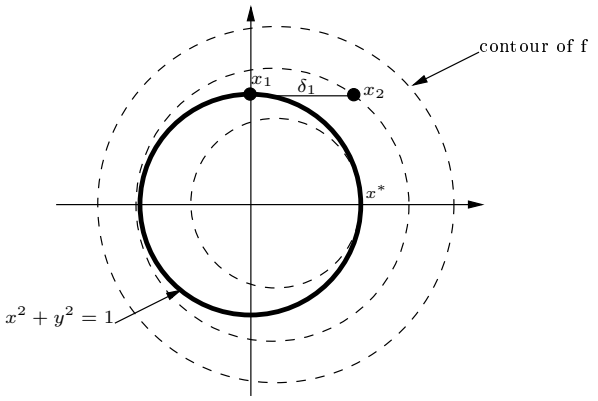

FIG. 1 – Illustration de l'effet Maratos

On choisit donc $\gamma_k^P = B_k\delta_k$ pour $\theta = 0$ comme issue de secours, on a alors $B_{k+1}^{Lagr} = B_k^{Lagr}$ et on est assuré que B_{k+1}^{Lagr} est définie positive. Cependant le but est de modifier au minimum γ_k^l afin de préserver le maximum d'informations fournie par les données du problème, il faut donc choisir θ le plus grand possible dans $[0, 1]$ tel qu'on ait :

$$(\gamma_k^P)^T\delta_k \geq 0.2\delta_k^T B_k^{Lagr}\delta_k$$

Comme B_k^{Lagr} est définie positive, cette inégalité est satisfaite pour $\theta = 0$, Powell a donc proposé :

$$\theta = \begin{cases} 1 & si\ (\gamma_k^P)^T\delta_k \geq 0.2\delta_k^T B_k^{Lagr}\delta_k, \\ 0.8\frac{\delta_k^T B_k\delta_k}{\delta_k^T B_k\delta_k - (\gamma_k^l)^T\delta_k} & sinon. \end{cases} \tag{38}$$

Cette technique est appelée la correction de Powell. Pour plus de détails sur cette correction, on peut se référer à [BGLS03, p.269-270] et [Fle87, p. 310]. Cette correction de Powell corrige un problème intrinsèquement lié à l'utilisation d'une fonction de pénalisation dans des méthodes de quasi-Newton. Il existe encore un autre problème qui est lui lié à la non différentiabilité de la fonction de pénalisation : l'effet Maratos.

3.3.3 Du global au local : l'effet Maratos

Comme on l'a vu précédemment, une recherche linéaire permet de globaliser l'optimisation, c'est à dire qu'elle permet de converger vers un minimum même si l'itéré initial est loin de celui-ci. Plus on se rapproche du minimum, plus l'algorithme avec recherche linéaire doit tendre vers l'algorithme local exposé dans la section 3.2, c'est à dire que le résultat de la recherche linéaire doit tendre vers $t_k = 1$. Cette propriété peut être appelée "admissibilité asymptotique du pas unitaire". Cependant il est possible que même très proche du minimum le pas unitaire soit rejeté car $\Phi(x_k + dx_k) > \Phi(x_k)$ (on rappelle ici que $\Phi(.)$ est la fonction de pénalisation (34)) ; ce phénomène est connu sous le nom d'effet Maratos et est illustré par l'exemple 3.1.

Exemple 3.1. *On peut illustrer ce phénomène en utilisant l'exemple souvent utilisé [BGLS03, p.254-255] et illustré par la Figure 1 :*

$$\min_{x,y}(2(x^2 + y^2 - 1) - x_1), \text{ soumis à } x^2 + y^2 - 1 = 0$$

La solution optimale de ce problème est $s^* = (1, 0)^T$. La figure 1 illustre un pas de SQP partant de $s_1 = (0, 1)^T$, la résolution du problème quadratique tangent donne une direction $\delta = (1, 0)^T$ pour atteindre la position $s_2 = (1, 1)^T$. Dans cet exemple, le pas δ sera automatiquement rejeté (un pas $t_k = 0$) par la recherche linéaire vu que la fonction de pénalisation L1 augmente dans la direction fournie par la résolution du problème quadratique tangent.

Cet effet s'explique par le fait que la décroissance de $F(.)$ ne compense plus la croissance de $\|c(.)\|_1$ et est parfaitement illustré dans [BGLS03, Exemple 15.6]. Ce phénomène est une conséquence directe de la séparation franche entre le calcul de la direction à l'aide du problème quadratique tangent (31) et le calcul du pas à l'aide de la fonction de pénalisation (34). Le problème vient de la fonction de pénalisation, ou du moins de la manière dont on l'utilise.

Il existe une correction du second ordre de dx_k qui permet de compenser ce phénomène, on calcule alors dx_k^m :

$$dx_k^m \leftarrow dx_k - (\nabla_x c_A(x_k))^- c_A(x_k + dx_k), \tag{39}$$

avec $c_A(.)$ les contraintes d'égalités et les contraintes d'inégalités actives, $(\nabla_x c_A(x_k))^-$ la pseudo-inverse à droite de la jacobienne des contraintes correspondantes. On rappelle que :

$$(\nabla_x c_A(x_k))^- = (\nabla_x c_A(x_k) \nabla_x c_A(x_k)^T)^{-1} \nabla_x c_A(x_k)^T.$$

Pour plus de détails sur les différentes méthodes de compensation de l'effet Maratos, le lecteur peut se référer à [Fle87, p. 393-395] et [BGLS03, p.254-260].

3.4 Ce qu'il faut retenir

(i)	Choisir un itéré initial (x_1, λ_1), et la tolérance d'arrêt ϵ calculer $F(x_1)$, $c(x_1)$, $\nabla_x F(x_1)$ et $\nabla_x c(x_1)$ fixer une constante $\omega \in]0, \frac{1}{2}[$ (modificateur de pente dans les conditions d'Armijo), et $\bar{\sigma} > 0$ (seuil du paramètre de pénalité), $\beta \in]0, \frac{1}{2}[$ (paramètre de sécurité dans la recherche linéaire d'Armijo), et $M_1 \leftarrow I_n$
(ii)	Calculs de δ_k et γ_k^P en utilisant la correction de Powell (37),
(iii)	Mise à jour de M_k en utilisant la formule de BFGS (13),
(iv)	Calculs de (dx_k, λ_{k+1}), solution du problème quadratique tangent (31),
(v)	Mise à jour des paramètres de pénalité σ_k en utilisant la règle 4,
(vi)	Calcul de dx_k^m en utilisant la correction du second ordre (39),
(vii)	Calcul du pas t_k, solution de la recherche linéaire d'Armijo sur la fonction de pénalisation (34) dans la direction dx_k^m,
(iix)	$x_{k+1} \leftarrow x_k + t_k dx_k$, $\lambda_{k+1} \leftarrow \lambda_k$,
(ix)	$k \leftarrow k + 1$, si les équations (17) à (21) sont vérifiées à ϵ près on arrête, sinon aller au pas (ii).

TAB. 5 – Schéma numérique d'un PQS de quasi-Newton

On a commencé par écrire les conditions d'optimalité pour un problème de minimisation avec contraintes d'égalités et d'inégalités. Tout comme pour les problèmes d'optimisation sans contrainte, il a été possible d'écrire une méthode de Newton, ce qui nous a amené à écrire un premier algorithme de minimisation à hessienne exacte de la table 3. Tout comme pour les problèmes sans contrainte, le calcul de la hessienne exacte est souvent un travail lourd et peut mener à des problèmes numériques lorsqu'on est loin du minimum, c'est pourquoi on utilise couramment des méthodes de quasi-Newton où la hessienne du lagrangien est estimée au cours des itérations à l'aide de la formule de BFGS (33). Afin de globaliser cette optimisation, on utilise une recherche linéaire sur une fonction de pénalisation dont le minimum x^* est, sous certaines conditions, le même que celui du problème original. Faire une recherche linéaire sur une fonction autre que la fonction coût implique que la définie positivité de B_{k+1} n'est plus assurée par la recherche linéaire, ceci a amené Powell à définir une correction pour assurer cette définie positivité. Dès lors, il n'y a plus de réel avantage à utiliser une recherche linéaire de Wolfe, une simple recherche linéaire basée sur les conditions d'Armijo suffit. Pour que l'algorithme fonctionne bien, le résultat de cette recherche linéaire doit asymptotiquement tendre vers un pas unitaire. Dans certains cas où la décroissance de $F(.)$ ne compense plus la croissance de $\|c(x_k)\|_1$, ce pas unitaire est rejeté même très proche du minimum ; ceci est appelé l'effet Maratos et peut être compensé par une modification de la direction de descente. Un PQS fonctionnel ressemble à l'algorithme 5.

Références

[BGLS03] Joseph Frédéric Bonnans, Jean-Charles Gilbert, Claude Lemaréchal, and Claudia Sagastizabal. *Numerical Optimization : Theoritical and Practical Aspects.* Springer, Collection, 2003.

[BMP04] Vincent Beck, Jérôme Malick, and Gabriel Peyré. *Objectif Agrégation.* H-K, 2004.

[Fle87] Roger Fletcher. *Practical Methods of Optimization, Second Edition.* John Wiley and Sons, 1987.

[HU01] Jean-Baptiste Hiriart-Urruty. *Optimisation.* Que sais-je ?, 2001.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-0803