□□□□□□ □□□□□□□□□□ □□□□□□□□ □□□□□□□□
République Algérienne Démocratique et Populaire
□□□□□□ □□□□ □ □□□□□ □□□□□□□ □□□□□
Minstère de l'Enseignement Supérieur et de la Recherche Scientifique

CENTRE UNIVERSITAIRE DE MILA
INSTITUT DES SCIENCES ET DE LA TECHNOLOGIE

Réf. ***/11***

Mémoire de fin d'étude
Présenté pour l'obtention du diplôme de

Licence Académique

Domaine : **Mathématiques et Informatique**
Filière : **Mathématiques**
Spécialité : **Mathématiques Fondamentales**

Thème

Les méthodes de quasi-Newton pour un problème d'optimisation sans contraintes

Présenté par :
- *Benmrara Meriem*
- *Amraoui Samah*

Dirigé par :
Dr. Boukaroura Khelladi Samia

Année universitaire 2010-2011

Table des matières

Introduction générale

Dans la vie courante, nous sommes fréquemment confrontés à des problèmes d'optimisation plus au moins complexes. Cela peut commencer au moment où l'on tente de ranger son bureau, de placer son mobilier, et aller jusqu'à un processus industriel, par exemple pour la planification des différentes taches. Ces problèmes peuvent être exprimés sous la forme générale d'un "problème d'optimisation". On définit alors une fonction objectif (fonction de coût ou fonction profit), que l'on cherche à optimiser (minimiser ou maximiser) par rapport à tous les paramètres (contraintes) concernés.

Il existe deux grandes catégories de problèmes celle des problèmes d'optimisation sans contraintes et celle des problèmes d'optimisation avec contraintes. Pour chaque catégorie de problèmes il existe un grand nombre de méthodes de résolution.

Dans ce mémoire on s'intéresse aux méthodes pour les problèmes d'optimisation sans contraintes.

Dans le premier chapitre on donne la méthode de Newton comme étant une méthode de déscente, ensuite dans le second chapitre on étudie les méthodes quasi newtoniennes, pour résoudre des problèmes d'optimisation non linéaires sans contraintes, à savoir celles de DFP et BFGS, suivie par des implémentations numériques sur des fonctions tests.

Chapitre 1

Méthode de Newton

1.1 *Méthodes de descente*

Soit le problème d'optimisation sans contraintes :

$$(P)\begin{cases}\min f(x)\\ x\in\mathbb{R}^n\end{cases}, \qquad \text{où} \quad f:\mathbb{R}^n\longrightarrow\mathbb{R}.$$

Une grande classe des méthodes existantes pour résoudre ce type de problème est celle des méthodes de descente, dont l'idee principale est la suivante :

On veut minimiser la fonction f, pour cela on se donne un point de départ x^0, pour construire l'itéré suivant x^1, il faut savoir qu'on veut se rapprocher du minimum x^* de f,on veut donc que

$$f(x^1)\leq f(x^0)$$

On cherche alors x^1 sous la forme :

$$x^1=x^0+\rho_0 d^0, \quad \text{où} \quad \rho_0\in\mathbb{R}_+^* \quad \text{et} \quad d^0\in\mathbb{R}^n, d^0\neq 0.$$

Donc, on cherche d^0 et ρ_0 de telle sorte que :

$$f(x^1)=f(x^0+\rho_0 d^0)\leq f(x^0),$$

ainsi de suite, on construit une suite $(x^k)_{k\geq 0}$ à partir de x^0, qui tend vers x^*.
Cette suite $(x^k)_{k\geq 0}$ est générée par la formule suivante :

$$\begin{cases} x^0\in\mathbb{R}^n \ \text{(donnée)}\\ x^{k+1}=x^k+\rho_k d^k \quad / \quad d^k\in\mathbb{R}^n\backslash\{0\}\text{, et } \rho_k>0\end{cases}$$

où ρ_k et d^k sont choisis de telle sorte que : $f(x^{k+1}) \leq f(x^k)$.

Le vecteur d^k s'appelle la direction de descente de f, et ρ_k est le pas déplecement de la méthode à la $k^{ième}$ itération.

Définition 1.1.1 *(direction de descente)*

Soit $f : \mathbb{R}^n \to \mathbb{R}$ une fonction de classe C^1 et $d \in \mathbb{R}^n \setminus \{0\}$. On dit que d est une direction de déscente de f au point x si :

$$\langle \nabla f(x), d \rangle < 0.$$

Remarque 1.1.2 .

***1**-Le pas de déplacement peut être fixé ou changé à chaque itération.*

***2**-Si la direction d^k est donnée en fonction de la matrice hessienne $H\left(x^k\right)$ on dit que la méthode est d'ordre 2.*

1.1.1 *Choix du pas de déplascement*

Le choix du pas de déplacement ρ_k peut être effectuer de plusieurs manières donnant plusieurs appélations.

a. Méthode à pas fixe :

On prend : $\rho_k = \rho > 0$, $\forall k$, (pour toutes les itérations).

b. Méthode à pas optimale :

ρ_k est choisi tel que :

$$\varphi(\rho_k) = \min \varphi(\rho) = \min(f(x^k - \rho \nabla f(x^k)), \rho > 0.$$

$\varphi(\rho)$ est une fonction à une seule variable ($\varphi : \mathbb{R}_+^* \longrightarrow \mathbb{R}$).

c. Méthode à pas variable :

On peut faire varié le pas ρ_k à chaque itération par exemple $\rho_k = \frac{1}{k}$.

1.2 *Méthode de Newton*

On suppose que la fonction f est deux fois continument différentiable sur $\mathbb{R}^n (f \in C^2(\mathbb{R}^n))$ et que l'on sait calculer toutes ses dérivées secondes .

L'idée consiste à remplacer, au voisinage du point courant x^k, la fonction f par son approximation quadratique $q(x)$.

En utilisant le développement limité de Taylor de f jusqu'à l'ordre 2, au voisinage de x^k on obtient :

$$f(x) \simeq q(x) = f(x^k) + \langle x - x^k, \nabla f(x^k) \rangle + \frac{1}{2} \langle H(x^k)(x - x^k), x - x^k \rangle$$

où $\nabla f(x^k)$ est le gradient de f en x^k, et $H(x^k) = H_k$ est la matrice hessienne de f au point x^k

On prend alors comme point x^{k+1} le minimum de $q(x)$ l'orsqu'il existe.

Ceci ne peut être le cas que si H_k est une matrice définie positive, alors $q(x)$ est strictement convexe et a un minimum unique qu'on note : x^{k+1} qui vérifie :

$$\nabla q(x^{k+1}) = 0 \Longleftrightarrow \nabla f(x^k) + H_k(x^{k+1} - x^k) = 0.$$

Ce qui nous conduit au système linéaire :

$$\begin{aligned} \left(H_k(x^{k+1} - x^k) = -\nabla f(x^k)\right) &\iff \left(H_k d^k = -\nabla f(x^k) \diagup d^k = x^{k+1} - x^k\right) \\ &\iff \left(d^k = -\left[H_k\right]^{-1} \nabla f(x^k)\right) \end{aligned} \tag{1.1}$$

où H_k est définie positive, d^k est dite direction de *Newton*.

D'où la formule itérative est la suivante :

$$x^{k+1} = x^k - \left[H_k\right]^{-1} \nabla f(x^k) = x^k + d^k \tag{1.2}$$

Cette formule n'est autre que la méthode de Newton appliquée à la résolution du système d'équations non linéaires

$$\nabla f(x) = 0$$

1.2.1 *Algorithme de Newton*

Soit le problème d'optimisation sans contraintes :

$$(P)\begin{cases} \min f(x) \\ x \in \mathbb{R}^n \end{cases}, \qquad \text{où} \quad f : \mathbb{R}^n \longrightarrow \mathbb{R}.$$

On suppose que la fonction f est deux fois continument différentiable sur $\mathbb{R}^n$ avec $\nabla f(x^k)$ le gradient de f en x^k, et $H(x^k) = H_k$ la matrice hessienne de f au point x^k.

Alors l'algorithme de Newton est donné comme suit :

Algorithm 1.2.1 *(Newton)*

1. Initialisation

Poser $k = 0$, *choisir* x^0 *dans un voissinage de* x^*, *donner* $\varepsilon > 0$ *(précision demandée).*

2. Itération k

Calculer d^k, *solution du systéme* $H_k d^k = -\nabla f(x^k)$.

Poser $x^{k+1} = x^k + d^k$.

3. Critère d'arrêt

Si $\| \nabla f(x^{k+1}) \| < \varepsilon$, *stop.*

Sinon poser $k = k + 1$ *et retourner à* 2.

Remarque 1.2.2 .

On peut choisir d'autre critères d'arrêt tels que :

$$\left\| f(x^{k+1}) - f(x^k) \right\| < \varepsilon \text{ ou bien } \left\| x^{k+1} - x^k \right\| < \varepsilon.$$

1.2.2 *Convergence de la méthode de Newton*

Soit le problème d'optimisation sans contraintes :

$$(P) \begin{cases} \min f(x) \\ x \in \mathbb{R}^n \end{cases}, \qquad \text{où} \quad f : \mathbb{R}^n \longrightarrow \mathbb{R}.$$

La convergence de la méthode de Newton est donnée à travers le théorème suivant :

Théorème 1.2.3 *(convergence)*

Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ *de classe* $C^3(\mathbb{R}^n)$, x^**un minimum local de* f. *On suppose que la matrice hessienne* $H(x^*)$ *définie positive alors :*

1. *Il existe un voisinage* v^* *de* x^**telle que : si* $x^0 \in v^*$ *alors la suite* $(x^k)_{k \in \mathbb{N}}$ *générée à partir de* x^0 *par l'algorithme de Newton converge vers* x^*.

2. *La convergence est au moins quadratique.*

Remarque 1.2.4 .

En général les conditions de convergence d'une méthode sont suffisantes (non pas necessaires), donc une méthode peut converger sans que la fonction vérifie les conditions de convergence.

1.2.3 *Avantages de la méthode de Newton*

L'avantage principal de la méthode de Newton est sa grande rapidité, sa vitesse de convergence est quadratique en plus elle est simple et très facile à implémenter.

Exemple 1.2.5 .

La méthode de Newton appliquée àune fonction quadratique strictement convexe converge en une seule itération.

Preuve.

En effet, soit f une fonction quadratique strictement convexe sous la forme :

$$f(x) = \frac{1}{2}x^T Ax + b^T x + c$$

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

On a (f strictement convexe) $\Longleftrightarrow$ (A définie positive)

Montrons que l'algorithme de Newton converge en une seule itération $\forall x^0$ le point de départ.

On a :

$f(x) = \frac{1}{2}x^T Ax + b^T x + c$, $\forall x \in \mathbb{R}^n$

$\Rightarrow \nabla f(x) = Ax + b$, $\forall x \in \mathbb{R}^n$

$\Rightarrow H(x) = A, \forall x \in \mathbb{R}^n$

1. *Soit $x^0 \in \mathbb{R}^n$ (quelconque), soit $\varepsilon > 0$.*

2. *Calculons d^0 solution de $H_0 d^0 = -\nabla f(x^0)$ / $H_0 = H(x^0)$.*

On a $H_0 = A$ définie positive $\Rightarrow [H_0]^{-1} = A^{-1}$ existe

$\Rightarrow d^0 = -A^{-1}\nabla f(x^0)$ $/\nabla f(x^0) = Ax^0 + b$

$\Rightarrow d^0 = -A^{-1}(Ax^0 + b)$

$\Rightarrow d^0 = -x^0 - A^{-1}b$

On a $x^1 = x^0 + d^0 = x^0 - x^0 - A^{-1}b.$

$\Rightarrow x^1 = -A^{-1}b.$

3. *Critère d'arrêt :*

$\nabla f(x^1) = Ax^1 + b = A(-A^{-1}b) + b = -b + b = 0$

$\Rightarrow \nabla f(x^1) = 0 \Rightarrow \|\nabla f(x^1)\| = 0 < \varepsilon$. *Stop.*

$\Rightarrow x^* = x^1$ *est le minimum de f sur $\mathbb{R}^n$.* ■

1.2.4 *Inconvénients de la méthode de Newton*

Les principaux inconvenients de la méthode de Newton sont :

1. Sa convergence locale, elle dépend du choix du point de départ x^0, elle peut diverger si $x^0 \notin v^*(x^*)$.

2. Le calcul couteux de la direction d^k (soit par résolution du système d'équations, soit par le calcul de la matrice inverse $[H_k]^{-1}$).

3. Á la fin de l'algorithme on obtient un point critique, il faudra ensuite vérifier que c'est un minimum.

Pour remédier aux difficultés de la méthode de Newton il faut introduire un certain nombre de modifications.

Tout d'abord, comme l'approximation de $f(x)$ par $q(x)$ n'est valable que dans un voisinage de x^k, on peut agir sur le pas de déplacement, en utilisant une formule itérative du type :

$$x^{k+1} = x^k - \rho_k \left[H_k\right]^{-1} \nabla f(x^k)$$

où : le pas de déplacement ρ_k est un scalaire choisi, par exemple, de façon à ce que la norme $\| x^{k+1} - x^k \|$ ne soit pas trop grande.

On peut aussi le choisir de façon à ce que x^{k+1} minimise $\varphi(\rho) = f(x^k + \rho d^k)$ dans la direction $d^k = -\left[H_k\right]^{-1} \nabla f(x^k)$.

Une autre façon de déterminer le pas de déplacement est donnée par le lemme suivant :

Proposition 1.2.6

Essayer la valeur $\lambda_k = 1$. Si $f(x^k + d^k) \geq f(x^k)$, alors ρ^* minimisant

$$\varphi(\rho) = f(x^k + \rho d^k)$$

est certainement compris entre 0 et 1.

Preuve.

En effet,

$\varphi(\rho) = f(x^k + \rho d^k) = f(x^k) + \rho \left(d^k\right)^T \nabla f(x^k) + \frac{\rho^2}{2} \left(d^k\right)^T H_k d^k$

$\varphi(1) = f(x^k + d^k) = f(x^k) + \left(d^k\right)^T \nabla f(x^k) + \frac{1}{2} \left(d^k\right)^T H_k d^k$

$\varphi(0) = f(x^k)$

On a $f(x^k + d^k) = f(x^k) + \left(d^k\right)^T \nabla f(x^k) + \frac{1}{2} \left(d^k\right)^T H_k d^k \geq f(x^k) f(x^k + d^k)$

$\Rightarrow \left(d^k\right)^T \nabla f(x^k) + \frac{1}{2} \left(d^k\right)^T H_k d^k \geq 0$

$\varphi'(\rho) = \left(d^k\right)^T \nabla f(x^k) + \rho \left(d^k\right)^T H_k d^k$

$\Rightarrow \varphi'(0) = \left(d^k\right)^T \nabla f(x^k) < 0$, *car* d^k *est une direction de déscente.*

et $\varphi'(1) = \left(d^k\right)^T \nabla f(x^k) + \left(d^k\right)^T H_k d^k > 0$,

car $\left(d^k\right)^T \nabla f(x^k) + \left(d^k\right)^T H_k d^k > \left(d^k\right)^T \nabla f(x^k) + \frac{1}{2} \left(d^k\right)^T H_k d^k \geq 0$

Donc $\varphi'(0) < 0$ *et* $\varphi'(1) > 0$

$\Rightarrow \exists \rho^* \in [0, 1]$ *tel que* $\varphi'(\rho^*) = 0$. ■

On peut alors utiliser une méthode de dichotomie jusqu'à trouver un ρ tel que :

$$f(x^k + \rho_k d^k) < f(x^k).$$

Une autre difficulté peut apparaître l'orsque le hessien H_k n'est pas défini positif. Dans ce cas, en effet, la direction de déplacement : $-\left[H_k\right]^{-1} \nabla f(x^k)$ peut ne pas être une

direction de descente, et la convergence globale de la méthode n'est pas assurée. Lorsque ceci se produit, certains auteurs ont suggéré de perturber légèrement le hessien H_k de façon à obtenir une matrice S_k définie positive. On aboutit ainsi à la formule itérative : $x^{k+1} = x^k - \rho_k \left[S_k\right]^{-1} \nabla f(x^k)$

où le scalaire ρ_k est déterminé par l'une des techniques décrites ci-dessus.

Remarquer ici que le fait que S_k soit définie positive assure que la direction de déplacement $d^k = -\left[S_k\right]^{-1} \nabla f(x^k)$ est une direction de descente.

En effet :

$$\nabla f^T(x^k) d^k = -\nabla f^T(x^k) \left[S_k\right]^{-1} \nabla f(x^k) < 0.$$

Pour construire S_k à partir de H_k on pourra, par exemple, utiliser une perturbation du type : $S_k = \mu_k I + H_k$ où $\mu_k > 0$ est un scalaire choisi minimal avec la contrainte que toutes les valeurs propres de S_k soient supérieures ou égales à une constante $\delta > 0$ donnée. La convergence globale de cette procédure est facile à démontrer (pour μ_k trés grand, on retrouve la méthode du gradient).

Pour remédier à ces inconvénients, plusieurs variantes ont été developpés, dite méthode de quasi-Newton telle que la méthode de Levenberg-Marquadt, la méthode DFP et la méthode BFGS qu'on va voir dans le chapitre 2.

Chapitre 2

Les méthodes quasi newtoniennes

2.1 *Principe général*

Le principe de ces méthodes consiste essentiellement en une généralisation de la formule itérative de Newton :

$$x^{k+1} = x^k - \rho_k \left[H_k\right]^{-1} \nabla f(x^k).$$

On a vu que la limitation importante de la méthode de Newton consistait dans la restriction H_k définie positive.

Une extension naturelle consiste à remplacer $[H_k]^{-1}$ par une matrice M_k définie positive donnant la direction de déplacement à partir du gradient $\nabla f(x^k)$.

D'où une formule itérative du type :

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

où ρ_k est choisi de façon à minimiser $\varphi(\rho) = f(x^k + \rho d^k)$ dans la direction $d^k = -M_k \nabla f(x^k)$, (ou au moins de telle sorte que $\varphi(\rho_k) < \varphi(0)$).

La matrice H_k est évidement modifiée à chaque itération, de telle manière que, pour une fonction quadratique de la forme :

$$q(x) = \frac{1}{2} x^T A x + b^T x + c, \text{ avec } A \text{ définie positive,}$$

M_k converge vers l'inverse A^{-1} du hessien de q.

Enfin de convergence, on retrouvera donc la méthode de Newton.

L'orsqu'on applique la méthode à une fonction f quelconque, M_k peut alors être considérée, à chaque instant, comme une approximation (définie positive) de l'inverse du

hessien de f.

Il existe évidemment beaucoup de variantes possibles dans le choix de la formule de mise à jour de la matrice M_k. Généralement on impose la relation :

$$M_k \left[\nabla f(x^k) - \nabla f(x^{k-1})\right] = x^k - x^{k-1},$$

car $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, $\nabla f(x^k) = Ax^k + b$, et $\nabla f(x^{k-1}) = Ax^{k-1} + b$
$\Rightarrow \nabla f(x^k) - \nabla f(x^{k-1}) = A(x^k - x^{k-1})$
$\Rightarrow A^{-1}(\nabla f(x^k) - \nabla f(x^{k-1})) = x^k - x^{k-1}$
$\Rightarrow M_k(\nabla f(x^k) - \nabla f(x^{k-1})) = x^k - x^{k-1}$

La formule de corretion qui permet d'obtenir la matrice M_{k+1}, à partir de la matrice M_k utilise les nouvelles informations obtenues lors de l'étape k de l'algorithme, c'est-à-dire essentiellement le gradient $\nabla f(x^{k+1})$ au point x^{k+1} (généralement obtenu par recherche unidimensionnelle dans la direction $d^k = -M_k \nabla f(x^k)$ à partir de x^k).

Différentes formules de correction du type :

$$M_{k+1} = M_k + \triangle_k,$$

ont été proposées. Suivant que la matrice $\triangle_k$ est de rang 1 ou de rang 2 on parlera de correction de rang 1 ou de rang 2.

Une des premières formules de correction utilisées pour construire une approximation de l'inverse du hessien consiste à choisir une matrice (de rang 1) de la forme :

$$\triangle_k = \alpha_k u_k u_k^T$$

où u_k et α_k sont respectivement un vecteur et un scalaire choisis de telle sorte que l'on ait :

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

On remarque que si la matrice M_0 est symétrique, la corretion ci-dessus préserve la symétrie des matrices M_k.

En posant :

$$\begin{aligned} \delta_k &= x^{k+1} - x^k. \\ \gamma_k &= \nabla f(x^{k+1}) - \nabla f(x^k) \end{aligned}$$

Montrons comment on peut déterminer α_k et u_k tels que :

$$M_{k+1}\gamma_k = \delta_k.$$

Soit encore :

$$[M_k + \alpha_k(u_k u_k^T)]\gamma_k = \ \delta_k.$$

En prenant le produit scalaire des deux membres avec γ_k on obtient :

$$\gamma_k^T M_k \gamma_k + \alpha_k(\gamma_k^T u_k)(u_k^T \gamma_k) = \ \gamma_k^T \delta_k.$$

Soit :

$$\alpha_k(u_k^T \gamma_k)^2 = \gamma_k^T(\delta_k - M_k\gamma_k).$$

En utilisant l'identité :

$$\alpha_k(u_k u_k^T) = \frac{(\alpha_k u_k u_k^T \gamma_k)(\alpha_k u_k u_k^T \gamma_k)^T}{\alpha_k.(u_k^T \gamma_k)^2}$$

et en remplaçant

$$\alpha_k u_k u_k^T \gamma_k \text{ par } \delta_k - M_k\gamma_k$$
$$\alpha_k(u_k^T \gamma_k)^2 \text{ par } \gamma_k^T(\delta_k - M_k\gamma_k)$$

On obtient la formule de correction (de rang 1) suivante :

$$M_{k+1} - M_k = \alpha_k(u_k u_k^T) = \frac{(\delta_k - M_k\gamma_k)(\delta_k - M_k\gamma_k)^T}{\gamma_k^T(\delta_k - M_k\gamma_k)} \tag{2.1}$$

Proposition 2.1.1 .
Si M_k est définie positive et si $\gamma_k^T(\delta_k - M_k\gamma_k) > 0$, alors M_{k+1} est définie positive.

Preuve.
On a

$$\begin{aligned} x^T M_{k+1} x &= x^T M_k x + x^T(\alpha u u^T)x \\ &= x^T M_k x + x^T \frac{(\delta_k - M_k\gamma_k)(\delta_k - M_k\gamma_k)^T}{\gamma_k^T(\delta_k - M_k\gamma_k)} x \\ &= x^T M_k x + \frac{1}{\gamma_k^T(\delta_k - M_k\gamma_k)}(x^T(\delta_k - M_k\gamma_k))^2 > 0. \end{aligned}$$

Donc M_{k+1} est définie positive. ■

Remarque 2.1.2 .

La condition $\gamma_k^T(\delta_k - M_k\gamma_k) > 0$ n'est pas nécessairement remplie.

Dans ce cas, il est conseillé de ne pas mettre à jour la matrice M_k, et de conserver la même matrice d'une itération à l'autre.

Exemple 2.1.3 .

on a

$$M_k = I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \delta_k = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \gamma_k = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

donnent

$$M_{k+1} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

qui n'est pas définie positive puisque ses valeurs propres sont $\{-1, 1\}$.

La validité de cette formule provient du résultat suivant :

Théorème 2.1.4 .

Soit f une fonction quadratique, A son hessien (supposé défini positif).

Considérons un processus itératif qui, partant d'un point x^0, engendre successivement les points :

$x^2 = x^1 + \delta_1$,..., $x^{n+1} = x^n + \delta_n$ par déplacement le long de n directions indépendantes δ_1, δ_2, ..., δ_n successivement. Alors la suite des matrices M_k obtenues par :

$$\begin{cases} M_0 \text{ matrice symétrique quelconque} \\ M_{k+1} = M_k + \frac{(\delta_k - M_k\gamma_k)(\delta_k - M_k\gamma_k)^T}{\gamma_k^T(\delta_k - M_k\gamma_k)} \end{cases}$$

converge, en au plus n étapes, vers A^{-1}, l'inverse du hessien de f.

avec $\gamma_k = \nabla f(x^{k+1}) - \nabla f(x^k) = A(x^{k+1} - x^k) = A\delta_k$.

Preuve.

Si le hessien de f est constant, égal à A, on a $\forall\ i,\ \ A\delta_i = \gamma_i$.

On a vu que M_{k+1} est construite de telle sorte que $M_{k+1}\gamma_k = \delta_k$.

Démontrons que l'on a aussi $M_{k+1}\gamma_i = \delta_i$ *pour* $i = 1, 2, ..., k-1$.

Raisonnons par récurrence en supposant que cette propriété est vraie pour M_k, autrement dit que l'on a :

$M_k\gamma_i = \delta_i$ *pour* $i = 1, 2, ..., k-1$.

Soit $i \leq k-1$ *quelconque.*

On a

$$M_{k+1}\gamma_i = M_k\gamma_i + \frac{(\delta_k - M_k\gamma_k)(\delta_k^T\gamma_{i-}\gamma_k^T M_k\gamma_i)}{\gamma_k^T(\delta_k - M_k\gamma_k)}$$

Par l'hypothèse de récurrence $M_k\gamma_i = \delta_i$.

Donc $\delta_k^T\gamma_i = \delta_k^T A\delta_i = \delta_k^T\gamma_i$ *et par suite* $M_{k+1}\gamma_i = M_k\gamma_i = \delta_i$.

On en déduit que $\forall i = 1, 2, ..., k$, *on a* $M_{k+1}\gamma_i = \delta_i$.

En particulier, on a au bout de n étapes $M_{n+1}\gamma_i = \delta_i$ ($\forall i = 1, 2, ..., n$).

En tenant compte du fait que $\gamma_i = A\delta_i$, *ceci s'écrit* $M_{n+1}A\delta_i = \delta_i \quad (\forall i = 1, 2, ..., n)$. *Comme les* δ_i *constituent* n *directions linéairement indépendantes, il vient* $M_{n+1}A = I$ *ou encore* $M_{n+1} = A^{-1}$. *D'où le résultat.* ■

On remarque que la formule de correction (2.1) présente l'avantage que le point x^{k+1} n'a pas besoin d'être choisi comme le minimum de la fonction f dans la direction δ_k à partir du point x^k. C'est pourquoi de nombreux auteurs ont envisagé l'utilisation de cette formule pour construire des algorithmes ne nécessitant pas de minimisation unidimensionnelle.

En revanche la formule (2.1) présente l'inconvénient que, même si la fonction f est quadratique, et que M_0 et le hessien de f sont définis positifs, les matrices M_k obtenues ne sont pas nécessairement définies positives. D'autre part, la formule (2.1) peut ne pas avoir de sens, par exemple si le terme : $\gamma_k^T(\delta_k - M_k\gamma_k)$ est nul ou simplement de trés faible valeur.

L'orsque ces circonstances se produisent, il faut alors prévoir des procédures spéciales pour la mise à jour des matrices M_k (on peut par exemple poser $M_{k+1} = M_k$) mais alors la convergence de M_k vers l'inverse du héssien, n'est plus assurée.

Les méthodes quasi newtoniennes qui vont être décrites maintenant, et qui sont basées sur des formules de correction de rang 2, ne présentent pas ces inconvénients. Par contre, elles nécessitent l'utilisation d'une procédure d'optimisation unidimensionnelle exacte (algorithme de DFP) ou approchée (algorithme BFGS).

2.2 *Méthode de Davidon-Fletcher-Powell (DFP)*

Cet algorithme utilise la formule de correction (de rang 2) suivante :

$$M_{k+1} = M_k + \frac{\delta_k\delta_k^T}{\delta_k^T\gamma_k} - \frac{M_k\gamma_k\gamma_k^T M_k}{\gamma_k^T M_k\gamma_k} \tag{2.2}$$

où le point x^{k+1} est obtenu à partir de x^k par déplacement dans la direction

$$d^k = -M_k\nabla f(x^k),$$

et où

$$\delta_k = x^{k+1} - x^k,$$

$$\text{et } \gamma_k = \nabla f(x^{k+1}) - \nabla f(x^k).$$

Théorème 2.2.1 .

On suppose que la matrice M_k est définie positive. Alors si la condition $\delta_k^T.\gamma_k > 0$ est vérifiée, la matrice M_{k+1} donnée par la formule (2.3) est définie positive.

Cette condition est satisfaite, en particulier, si le point x^{k+1} est obtenu à partir de x^k par minimisation unidimensionnelle dans la direction $d^k = -M_k \nabla f(x^k)$.

Preuve.

Soit x un vecteur quelconque $x \neq 0$. Il faut montrer que $x^T M_{k+1} x > 0$.

On a

$$x^T M_{k+1} x = x^T M_k x + \frac{(x^T \delta_k)^2}{\delta_k^T \gamma_k} - \frac{(x^T M_k \gamma_k)^2}{\gamma_k^T M_k \gamma_k}$$

En posant : $u = (M_k)^{\frac{1}{2}} x$ et $v = (M_k)^{\frac{1}{2}} \gamma_k$ ceci s'écrit

$$x^T M_{k+1} x = \frac{(u^T u)(v^T v) - (u^T v)^2}{(v^T v)} + \frac{(\delta_k^T x)^2}{\delta_k^T \gamma_k}$$

Le premier terme du second membre est positif ou nul (d'après l'inégalité de Cauchy-Schwartz) et le seconde terme est positif ou nul si $\delta_k^T.\gamma_k > 0$.

D'autre part, les deux termes ne peuvent s'annuler simultanément. En effet, si le premier terme s'annule on a $u = \lambda v$ ($\lambda \neq 0$ scalaire), d'où $x = \lambda.\gamma_k$.

Mais alors : $\delta_k^T.x = \lambda.\delta_k^T.\gamma_k \neq 0$ et le second terme n'est pas nul.

Vérifions, en particulier, que $\delta_k^T.\gamma_k > 0$ si le point x^{k+1} est obtenu à partir de x^k par minimisation unidimensionnelle dans la direction $d^k = -M_k.\nabla f(x^k)$, on a

$$\begin{aligned} &\delta_k = x^{k+1} - x^k = \theta d^k = -\theta M_k.\nabla f(x^k) \qquad (\text{où } \theta > 0) \\ &\left(d^k\right)^T.\nabla f(x^{k+1}) = 0 \Rightarrow \delta_k^T.\nabla f(x^{k+1}) = 0. \end{aligned}$$

Dans ces conditions on a :

$$\begin{aligned} \delta_k^T.\gamma_k &= \delta_k^T(\nabla f(x^{k+1}) - \nabla f(x^k) \\ &= -\delta_k^T \nabla f(x^k) \\ &= \theta \nabla f^T(x^k) M_k \nabla f(x^k) > 0 \end{aligned}$$

■

Cette propriété de conservation de la définie positivité est essentielle, car elle garantit, en particulier, que les directions d^k, successivements engendrées par l'algorithme, sont des directions de descente.

La méthode utilisant la formule de correction (2.2) est alors donnée par l'algorithme suivant :

2.2.1 *Algorithme de Davidon-Fletcher-Powell*

Algorithm 2.2.2 *(DFP)*

1. Initialisation

Pose $k = 0$, *choisir* x^0 *point de départ. Choisir* M_0 *définie positive quelconque (par exemple la matrice unité) ;*

2. Itération k

Déterminer la direction de déplacement

$d^k = -M_k.\nabla f(x^k)$.

Déterminer le pas $\lambda_k = \underset{\lambda \geq 0}{argmin} f(x^k + \lambda d^k)$

$x^{k+1} = x^k + \lambda_k d^k$

Poser $\delta_k = x^{k+1} - x^k$.

Calculer $\gamma_k = \nabla f(x^{k+1}) - \nabla f(x^k)$ *puis*

$$M_{k+1} = M_k + \frac{\delta_k.\delta_k^T}{\delta_k^T.\gamma_k} - \frac{M_k.\gamma_k.\gamma_k^T.M_k}{\gamma_k^T.M_k.\gamma_k}$$

3. Critère d'arrêt

si $\left\|x^{k+1} - x^k\right\| < \varepsilon$, *stop. Sinon poser* $k = k + 1$ *et retourner à* 2.

La validité de cet algorithme provient du résultat suivant :

Théorème 2.2.3 .

Appliqué à une fonction f *quadratique (hessien* A *défini positif), l'algorithme de Davidon-Fletcher-Powell engendre des directions* δ_0,δ_1,...,δ_k *vérifiant,*

pour tout k $\begin{cases} \delta_i^T A \delta_j = 0 \qquad (0 \leq i < j \leq k).....(*) \\ M_{k+1} A \delta_i = \delta_i \quad (0 \leq i \leq k).....(**) \end{cases}$

Preuve.

En utilisant (2.2), *on a*

$$M_{k+1} A \delta_k = M_{k+1} \gamma_k = M_k \gamma_k + \frac{\delta_k \delta_k^T \gamma_k}{\delta_k^T \gamma_k} - \frac{M_k \gamma_k \gamma_k^T M_k \gamma_k}{\gamma_k^T M_k \gamma_k} = \delta_k, \quad \forall k$$

Les relations $(*)$ *et* $(**)$ *sont donc vraies pour* $k = 0$.

En les supposant vraies pour $k-1$, démontrons qu'elles sont vraies pour k.

On peut écrire pour $0 \leq i < k$ $\nabla f(x^k) - \nabla f(x^i) = \gamma_i + \gamma_{i+1} + ... + \gamma_{k-1}$.

Puisque le hessien de f est constant et égal à A, on a $\forall i : A\delta_i = \gamma_i$.

Par suite :

$\nabla f(x^k) - \nabla f(x^i) = A(\delta_i + \delta_{i+1} + ... + \delta_{k-1})$

Comme x^i est optimum de f dans la direction δ_{i-1} on a $\delta_{i-1}^T \nabla f(x^i) = 0$.

Par conséquent pour $i = 1,2,...,k-1$ $\delta_{i-1}^T \nabla f(x^k) = \delta_{i-1}^T A(\delta_i + \delta_{i+1} + ... + \delta_{k-1}) = 0$ *(par l'hypothèse de récurrence).*

En utilisant $(**)$ *et l'hypothèse de réccurence, on a* $\delta_{i-1} = M_k A \delta_{i-1}$.

D'où $\forall i$ $(1 \leq i \leq k)$ $\delta_{i-1}^T A M_k \nabla f(x^k) = 0$.

Comme $\delta_k = x^{k+1} - x^k = -\theta M_k \nabla f(x^k)$ *(avec* $\theta > 0$*) ceci entraîne* $\delta_{i-1}^T A \delta_k = 0$, $\forall i = 1, 2,..., k$ *et par suite la relation* $(*)$ *est vérifiée pour k.*

Démontrons maintenant que $M_{k+1} A \delta_i = \delta_i$, $\forall i = 1, 2,..., k-1$ *(la relation est vraie pour $i = k$ comme on l'a vérifié plus haut).*

On a

$$M_{k+1} A \delta_i = M_k A \delta_i + \frac{\delta_k \delta_k A \delta_i}{\delta_k^T \gamma_k} - \frac{M_k \gamma_k \gamma_k^T M_k A \delta_i}{\gamma_k^T M_k \gamma_k}.$$

Le second terme du second membre est nul car : $\delta_k^T A \delta_i = 0 \quad$ *pour* $i \leq k-1$.

Le troisième terme est également nul, si l'on remarque que, par l'hypothèse de récurrence $M_k A \delta_i = \delta_i$ *et que* $\gamma_k^T = \delta_k^T A$,

donc

$$\gamma_k^T M_k A \delta_i = \gamma_k^T \delta_i = \delta_k^T A \delta_i = 0, \quad \text{pour } i \leq k-1.$$

Par conséquent on a bien

$$M_{k+1} A \delta_i = M_k A \delta_i = \delta_i \quad (0 \leq i \leq k)$$

■

Remarque 2.2.4 .

Le résultat précédent montre que, dans le cas quadratique, les directions δ_0, δ_1,..., δ_k engendrées par l'algorithme sont mutuellement conjugées par rapport à la matrice A de la forme quadratique. Dans ce cas, l'algorithme converge donc en au plus n itérations.

Enfin, pour $k = n-1$, *la relation* $(**)$ *nous donne*

$$M_n A \delta_i = \delta_i \qquad (i = 0, 1, ..., n-1)$$

et, comme les δ_i sont linéairement indépendants (puisque mutuellement conjugués par rapport à A) on en déduit

$$M_n A = I \quad donc \quad M_n = A^{-1}.$$

La formule de correction (2.2) *permet donc de construire, en au plus n étapes dans le cas quadratique, l'inverse du hessien de f.*

2.2.2 *Convergence de l'algorithme DFP*

Comme pour la méthode de Fletcher et Reeves, la convergence globale de la méthode n'est garantie que si l'algorithme est périodiquement réinitialisé : par exemple toutes les n itérations, on prend comme point de départ le dernier point obtenu et on réinitialise la matrice M (par exemple en faisant $M_0 = I$, matrice unité).

Comme la première itération conduit à un déplacement dans la direction du gradient, la convergence globale de l'algorithme d'écoule de la convergence globale de la méthode du gradient. L'orsque l'algorithme n'est pas réinitialisé périodiquement, l'expérience indique que la méthode peut ne pas converger vers un optimum local.

La convergence asymptotique est généralement plus rapide qu'avec la méthode de Fletcher et Reeves. Par contre on observe que le nombre d'information à stocker est beaucoup plus élevé (une matrice $n \times n$), et que le nombre de calculs nécessaires à chaque itération est plus élevé.

Une autre caractéristique de l'algorithme DFP est que ses propriétés de convergence sont assez sensibles aux imprécisions dans les sous-problèmes de minimisation unidimensionnelle. Chaque itération nécessite donc un nombre d'évaluations de la fonction assez élevé pour obtenir la précision requise, et ceci peut effacer en partie les avantages due à une convergence asymptotique quadratique.

2.3 *Méthode de Broyden, Fletcher, Goldfarb et Shanno (BFGS)*

L'algorithme que nous allons décrire maintenant,et qui se rattache encore à la famille des méthodes quasi newtonienne, évite cet inconvénient, tout en conservant les avantages essentiels de l'algorithme DFP.

Cet algorithme développé indépendament par Broyden (1970), Fletcher(1970), Goldfarb(1970) et Shanno(1969) utilise pour construire une approximation de l'inverse du hessien, une formule de correction de rang 2 directement dérivée de la formule (2.2) rappelée

ci-dessous :

$$M_{k+1} = M_k + \frac{\delta_k \delta_k^T}{\delta_k^T \gamma_k} - \frac{M_k \gamma_k \gamma_k^T M_k}{\gamma_k^T M_k \gamma_k}.$$

On sait que la matrice M_{k+1} obtenue par (2.2) vérifie la relation

$$M_{k+1}\gamma_k = \delta_k \tag{2.3}$$

Comme l'a remarqué Fletcher (1970), si l'on intervertit les rôles de δ_k et de γ_k dans (2.2) et que l'on considère la suite des matrices définies par :

$$\begin{cases} G_0 \text{ symétrique définie positive quelconque} \\ G_{k+1} = G_k + \frac{\gamma_k \gamma_k^T}{\gamma_k^T \delta_k} - \frac{M_k \delta_k \delta_k^T M_k}{\delta_k^T M_k \delta_k} \end{cases} \tag{2.4}$$

Les matrices obtenues vérifient la relation <<inverse>> de (2.3) :

$$G_{k+1}\delta_k = \gamma_k \tag{2.5}$$

On voit alors que la formule (2.4) permet de construire une approximation du hessien lui-même (et non pas de son inverse).

Si l'on veut, à partir de (2.4), obtenir une formule de correction pour approximer l'inverse du hessien, il suffit donc de prendre l'inverse des deux membres de (2.4).

Le calcul donne :

$$[G_{k+1}]^{-1} = [G_k]^{-1} + \left[1 + \frac{\gamma_k^T [G_k]^{-1} \gamma_k}{\delta_k^T \gamma_k}\right] \frac{\delta_k . \delta_k^T}{\delta_k^T . \gamma_k} - \frac{\delta_k \gamma_k^T [G_k]^{-1} + [G_k]^{-1} \gamma_k \delta_k^T}{\delta_k^T \gamma_k} \tag{2.6}$$

On voit que (2.5) permet d'exprimer directement $[G_{k+1}]^{-1}$ en fonction de $[G_k]^{-1}$, d'où la formule de correction :

$$M_{k+1} = M_k + \left[1 + \frac{\gamma_k^T M_k \gamma_k}{\delta_k^T \gamma_k}\right] \frac{\delta_k . \delta_k^T}{\delta_k^T \gamma_k} - \frac{\delta_k \gamma_k^T M_k + M_k \gamma_k \delta_k^T}{\delta_k^T \gamma_k} \tag{2.7}$$

L'algorithme BFGS se déduit alors directement de l'algorithme DFP en remplaçant la formule (2.2) par la formule (2.7).

2.3.1 *Algorithme de Broyden, Fletcher, Goldfarb et Shanno*

Algorithm 2.3.1 *(BFGS)*

1. Initialisation

Pose $k = 0$, *choisir* x^0 *point de départ. Choisir* M_0 *définie positive quelconque (par exemple la matrice unité) ;*

2. Itération k

Déterminer la direction de déplacement

$d^k = -M_k.\nabla f(x^k)$.

Déterminer le pas $\lambda_k = \underset{\lambda \geq 0}{argmin} f(x^k + \lambda d^k)$

$x^{k+1} = x^k + \lambda_k d^k$

Poser $\delta_k = x^{k+1} - x^k$.

Calculer $\gamma_k = \nabla f(x^{k+1}) - \nabla f(x^k)$ *puis*

$$M_{k+1} = M_k + \left[1 + \frac{\gamma_k^T M_k \gamma_k}{\delta_k^T \gamma_k}\right] \frac{\delta_k . \delta_k^T}{\delta_k^T \gamma_k} - \frac{\delta_k \gamma_k^T M_k + M_k \gamma_k \delta_k^T}{\delta_k^T \gamma_k}$$

3. Critère d'arrêt

si $\left\|x^{k+1} - x^k\right\| < \varepsilon$, *stop. Sinon poser* $k = k + 1$ *et retourner à* 2.

Remarque 2.3.2 .

La formule (2.7) *a des propriétés trés analogues à celles de la formule* (2.2).

En particulier :

• *Si* $\delta_k^T \gamma_k > 0$ *la définie positivité des matrices* M_k *est préservée.*

• *Appliquée à une fonction quadratique (hessien A défini positif), cette formule permet d'obtenir, en au plus n itération, l'inverse* A^{-1} *du hessien de* f. *De plus les directions* δ_i *successivement engendrées par l'algorithme BFGS sont mutuellement conjuguées par rapport à* A^{-1}.

2.3.2 *Convergence de l'algorithme BFGS*

L'orsque l'algorithme est appliqué à une fonction non linéaire quelconque (non quadratique) il faut, comme pour l'algorithme DFP, procéder à des réinitialisations périodiques pour assurer la convergence globale.

La supériorité de l'algorithme BFGS sur l'algorithme DFP semble reconnue par la plupart des auteurs. La raison essentielle est que BFGS est beaucoup moins sensible que DFP aux imprécisions dans la procédure de recherche unidimensinnelle.

2.4 *Impémentation numérique*

On a mis en oeuvre les méthodes quasi-newtoniennes (DFP et BFGS) définies précédement, et comme en fonction test on a pris la fonction de Rosenbrock, définie par :

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

Le gradient de cette fonction est donné par :

$$\nabla f(x) = \begin{pmatrix} -400x_1(x_2 - x_1^2) - 2(1 - x_1) \\ 200(x_2 - x_1^2) \end{pmatrix}$$

On a pris une précision $\varepsilon = 10^{-4}$ et $M_0 = I$.

Notons que le minimum de cette fonction est atteint au point $x^* = (1, 1)^T$.

Dans ces tests, on change seulement le point initial.

Les résultats obtenus sont présentés dans le tableau si dessous.

	DFP		BFGS	
Point initial	Nombre d'itérations	temps(c.s)	Nombre d'itérations	Temps(c.s)
$(-1, 1)^T$	48	17	13	6
$(-0.2, 0.2)^T$	146	55	36	11
$(0.5, 0.5)^T$	24	11	19	11
$(-2, -2)^T$	pas de convergence		274	99
$(0, 20)^T$	pas de convergence		42	16

Tableau.1

On signale que le temps est calculé en centième de seconde.

D'après les tests q'on a effectués, on peut confirmer la supériorité et la stabilité de l'algorithme BFGS sur celui de DFP.

On a aussi remarqué que l'algorithme BFGS- qui approxime l'inverse du hessien- converge mieux (plus rapidement) si on prend comme matrice initiale une matrice définie positive différente de la matrice unité.

Cette remarque est confirmée par les tests q'on a effectué et qui sont présentés dans les tableaux si dessous.

On prend comme fonction test la fonction de Rosenbrock et comme précision $\varepsilon = 10^{-4}$. Signalons que le temps est calculé en centième de seconde.

Dans ces tests, on prend-à chaque point initial fixe et on change la matrice initiale.

Dans ce premier test, on prend $x_0 = (0.5, 0.5)^T$

La formule BFGS		
Matrice initiale $M^{(0)}$	Nombre d'itérations	Temps de calcul(c.s)
$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	19	11
$\begin{pmatrix} 5 & 2 \\ 2 & 6 \end{pmatrix}$	17	6
$\begin{pmatrix} 4 & 2 \\ 2 & 5 \end{pmatrix}$	16	6

Tableau.2

Dans le second test, on prend $x_0 = (0, 0)^T$.

La formule BFGS		
Matrice initiale $M^{(0)}$	Nombre d'itérations	Temps de calcul(c.s)
$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	27	11
$\begin{pmatrix} 4 & 2 \\ 2 & 5 \end{pmatrix}$	14	6
$\begin{pmatrix} 3 & 2 \\ 2 & 4 \end{pmatrix}$	12	6

Tableau.3

Dans le troisième test, on prend $x_0 = (-2, -2)^T$.

La formule BFGS		
Matrice initiale $M^{(0)}$	Nombre d'itérations	Temps de calcul(c.s)
$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	274	99
$\begin{pmatrix} 3 & 2 \\ 2 & 4 \end{pmatrix}$	91	27
$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$	58	16

Tableau.4

Conclusion générale

En conclusion, on peut dire que la méthode de Newton et ses variantes constituent une des classes les plus performantes pour la résolution des problèmes d'optimisation sans contraintes (ou avec contraites). Elles sont connues plus particulièrement par leur convergence locale quadratique.

Les méthodes quasi-newtoniennes ont prouvé leur performance sur celle de la méthode de Newton, surtout sur le plan pratique, comme le montre les résultats obtenus.

En fin, une comparaison entre les deux méthodes quasi-newtoniennes DFP et BFGS montre que l'algorithme BFGS est plus stable et plus robuste que celui de DFP.

Bibliographie

[1] **P. G. Ciarlet**, *Introduction à l'analyse numérique matricielle et à l'optimisation,* Masson, Paris (1988)

[2] **S. Khelladi Boukaroura**, *Etude du caractère newtonien dans l'algorithme de Karmarkar*, Mémoire de Magistère, Université Ferhat Abbas Sétif (1997).

[3] **M. Minoux**, *Programmation mathématique, théorie et algorithme,*Tome 1, Dunod, Paris (1984).

[4] **A. Rondepierre, S. Tordeux**, *Introduction à l'optimisation numérique,* INSA Toulouse (2010).