

Chapitre 4

Les Méthodes Newtoniennes

Chapitre 4

Les Méthodes Newtoniennes

4.1 Définition de l'algorithme

La méthode de Newton est un procédé général de résolution des systèmes d'équations non linéaires de la forme

$$g(x) = 0 \tag{4.1}$$

où $g : \mathbb{R}^n \rightarrow \mathbb{R}^n$ est supposée régulière. Pour le problème (P) déjà posé, on utilise cette méthode avec $g(x) = \nabla f(x)$, donc (4.1) résume la condition nécessaire d'optimalité du problème (P).

Dans ces conditions l'algorithme de Newton génère une suite de vecteurs $\{x_k\}$ de la manière suivante : Supposant connu l'itéré courant x_k , l'équation (4.1) linéarisée en x_k est :

$$g(x_k) + g'(x_k)(x - x_k) = 0 \tag{4.2}$$

si g est le gradient d'une fonction f à minimiser alors, g' est son hessien. On écrit autrement l'équation (4.2) :

$$g(x_k) + H(x_k)(x - x_k) = 0.$$

Lorsque $H(x_k)$ est inversible la solution de l'équation est la nouvelle itéré de l'algorithme de Newton donc

$$x_{k+1} = x_k + d_k \quad (4.3)$$

ou

$$d_k = -H^{-1}(x_k)g(x_k) \quad (4.4)$$

Et on peut écrire l'algorithme de Newton :

Algorithme 4.1.

Etape 0 (Initialisation) : x_0 et $\varepsilon > 0$ donnée, $k = 0$

Etape 1 : Test d'arrêt : calculer $g(x_k)$, si $\|g(x_k)\| < \varepsilon$ alors stop sinon

Etape 2 : Calcul de la direction $d_k = -H^{-1}(x_k)g(x_k)$

Etape 3 : $x_{k+1} = x_k + d_k$

Etape 4 : Remplacer k par $k + 1$ et aller à l'étape 1.

Remarque 4.1.

- A l'étape 2, on calcul la direction d_k en résolvant l'équation (dite de Newton)

$$H(x_k)d_k = -g(x_k) \quad (4.5)$$

- Dans la pratique, si $H(x_k)$ est définie positive, il est préférable d'utiliser la méthode de décomposition de Choleski, on calcule la matrice L triangulaire inférieure telle que $H(x_k) = LL^\top$. La résolution de (4.5) revient à résoudre deux système triangulaires

$$Ly = -g(x_k) \quad (4.1)$$

puis

$$L^\top d_k = y \quad (4.2)$$

mieux encore, on peut décomposer $H(x_k)$ en $LDL^\top = (LD^{1/2}D^{1/2}L^\top)$ avec D , une matrice diagonale.

4.1.1 Etude de la convergence

On va étudier la convergence d'une méthode un peu plus générale, dans laquelle la direction d_k est donnée par

$$d_k = -B_k^{-1}g(x_k) \quad (4.3)$$

B_k étant une matrice inversible.

Théorème 4.1. Soit x^* une solution de (4.1). Supposant que g est de classe C^1 dans un voisinage V_{x^*} de x^* et $H(x^*)$ est inversible. S'il existe $\varepsilon > 0$ et $\delta > 0$ assez petits tels que

$$\|x_0 - x^*\| \leq \varepsilon \text{ et } \|B_k - H(x_k)\| \leq \delta$$

alors l'algorithme 4.1 avec d_k donné par (4.3) génère une suite $\{x_k\}$ convergeant linéairement vers x^* . Si de plus $\{B_k\}$ converge vers $H(x^*)$, alors la convergence est super-linéaire. Enfin, si H est lipschitzien et si $B_k - H(x^*) = o(\|x_k - x^*\|)$, alors la convergence est quadratique.

Démonstration.

$g \in C^1(V_{x^*}) \Rightarrow H$ est continue sur V_{x^*} et puisque $H(x^*)$ est inversible, donc $H(x_k)$ est inversible pour $x_k \in V_{x^*}$. Si de plus δ est assez petit alors B_k sera aussi inversible et la direction (4.3) est bien définie, donc

$$\begin{aligned} x_{k+1} - x^* &= x_k + d_k - x^* = x_k - x^* - B_k^{-1}g(x_k) = \\ &= B_k^{-1}(B_k(x_k - x^*) - g(x_k)) \end{aligned}$$

En utilisant la formule (1.6) pour la fonction vectorielle g

$$g(x_k) = g(x^* + (x_k - x^*)) = g(x^*) + \int_0^1 H(x^* + t(x_k - x^*))(x_k - x^*) dt$$

Puisque : $g(x^*) = 0$, on a

$$\begin{aligned} x_{k+1} - x^* &= B_k^{-1}(B_k - H(x_k))(x_k - x^*) + \\ &+ B_k^{-1} \int_0^1 (H(x_k) - H(x^* + t(x_k - x^*)))(x_k - x^*) dt \end{aligned}$$

La formule de la moyenne pour $t_0 \in]0, 1[$

$$\|x_{k+1} - x^*\| \leq \|x_k - x^*\| \left\| B_k^{-1} \right\| \cdot [\|B_k - H(x_k)\| + \|H(x_k) - H(x^* + t_0(x_k - x^*))\|] \quad (4.4)$$

si ε, δ est tels que

$$\|B_k - H(x_k)\| \leq \frac{r_1}{\|B_k^{-1}\|} \text{ et } \|H(x_k) - H(x^* + t_0(x_k - x^*))\| \leq \frac{r_2}{\|B_k^{-1}\|}$$

avec $r_1 + r_2 = r \in [0, 1[$, on a

$$\|x_{k+1} - x^*\| \leq r \|x_k - x^*\| \quad (4.5)$$

donc $\{x_k\}$ converge linéairement vers x^* . Si de plus, $B_k \rightarrow H(x^*)$, alors

$$\lim \left\| B_k^{-1} \right\| [\|B_k - H(x_k)\| + \|H(x_k) - H(x^* + t_0(x_k - x^*))\|] = 0$$

de (4.4) on a

$$\lim \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \lim \frac{o(\|x_k - x^*\|)}{\|x_k - x^*\|} = 0 \quad (4.6)$$

ce qui montre que la convergence de $\{x_k\}$ est super-linéaire. Enfin, si on suppose que H est

M-Lipschitzien et que $B_k - H(x^*) = o(\|x_k - x^*\|)$, puisque $t_0 \in]0, 1[$ on obtient

$$\begin{aligned} \|H(x_k) - H(x^* + t_0(x_k - x^*))\| &\leq M \|x_k - (x^* + t_0(x_k - x^*))\| \\ &= M \|(1 - t_0)(x_k - x^*)\| < M \|x_k - x^*\| \end{aligned}$$

donc de (4.4)

$$\begin{aligned} \|x_{k+1} - x^*\| &\leq \|x_k - x^*\| \|B_k^{-1}\| [o(\|x_k - x^*\|) + M \|x_k - x^*\|] \\ \Rightarrow \lim \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} &= M' > 0 \end{aligned}$$

□

Remarque 4.2. Le théorème précédent n'entraîne nullement la convergence globale de l'algorithme de Newton, mais il dit que si x_k est proche de la solution du problème (4.1), alors x_{k+1} est infiniment plus proche. en fait les inconvénients de la méthodes de Newton sont bien connue :

1. L'algorithme n'est pas globalement convergeant.
2. L'algorithme n'est pas définie en un point x où $H(x)$ est singulière.
3. Si $H(x_k)$ n'est pas définie positif, la direction (4.4) n'est pas une direction de descente.
4. Le point stationnaire obtenue peut être un minimum local, un maximum local où un point de selle.
5. Un système linéaire d'ordre n doit être résolut à chaque itération.

4.2 Algorithme de Newton modifié

On note que contrairement aux méthodes du premier ordre (Chapitre 1) l'algorithme (4.3)-(4.4) fournit non seulement une direction, mais aussi une longueur de pas le long de cette direction, en d'autres termes, le pas $t = 1$ est privilégié (pour avoir une convergence super-linéaire vue le théorème 4.1). Ceci est valable seulement pour x_0 proche de la solution, loin de là on n'a pas de convergence globale, pire encore la direction (4.4) n'assure pas la descente, c'est-à-dire

$$f(x_{k+1}) = f(x_k + d_k) \leq f(x_k).$$

Pour cela on peut interpréter (4.4) comme une direction, le long de laquelle on effectue une recherche linéaire pour forcer la descente entraînant la convergence globale.

Une telle recherche est possible que si $H(x_k)$ est définie positif (pour avoir $g(x_k)d_k < 0$). Sinon, on peut modifier le hessien, car ce cas est facilement repérable (revenant aux remarques 4.1) pendant la décomposition de $H(x_k)$ (par la méthode de Choleski), on s'aperçoit très vite qu'on doit extraire une racine carrée négative, on peut alors ajouter un terme positive là où il le faut, de façon à obtenir la définie positivité de la matrice hessienne. Il faut mentionner que cette technique est très gourmande en nombre d'opération effectuer, donc elle est peut efficace.

En effet, pour avoir la matrice $L \in \mathcal{M}_n$ on effectue un total de $\frac{2n^3 + 15n^2 + n}{6}$ opérations élémentaires. La résolution des systèmes triangulaires (4.1) et (4.2) demande $2n^2$ opérations élémentaires. Donc, pour inverser la matrice $H(x_k)$ il faudrait faire $\frac{2n^3 + 15n^2 + n}{6} + 2n^2 = \frac{2n^3 + 27n^2 + n}{6}$ opérations élémentaires à chaque itération de l'algorithme (penser au cas où $n = 10^3, \dots, 10^6$).

Algorithme 4.2 (Newton avec décomposition de Choleski).

Etape 0 (Initialisation) : x_0 et $\varepsilon > 0$ donnée, $k = 0$

Etape 1 : Test d'arrêt : calculer $g(x_k)$, si $\|g(x_k)\| < \varepsilon$ alors stop sinon,

Etape 2 : Calcul de la direction d_k :

Effectuer une décomposition de Choleski de la matrice $H(x_k)$: $H(x_k) = LL^T$,

Résoudre les deux systèmes : $Ly = -g(x_k)$ puis $L^T d_k = y$,

Etape 3 : Recherche linéaire, trouver t_k solution du problème $\min_{t>0} f(x_k + td_k)$,

Etape 4 : Si recherche linéaire réussie $x_{k+1} = x_k + t_k d_k$ remplacer k par $k + 1$ et aller en 1.

4.2.1 Algorithme de Newton tronqué

A l'étape 2 de l'algorithme 4.2, et vue les remarques précédentes, on voit bien qu'il est pratiquement très coûteux de calculer exactement la direction d_k , on peut alors résoudre

l'équation de Newton (4.5) de manière approchée. Dans la pratique cette résolution se fait par un processus itératif interne (le plus souvent il s'agit de l'algorithme du gradient conjugué linéaire), ce processus sera stoppé avant convergence d'où le nom d'algorithme de Newton tronqué. Voici un exemple de telles méthodes.

Algorithme 4.3.

Etape 0 (Initialisation) : x_0 et $\varepsilon > 0$ donnée, $k = 0$

Etape 1 : Test d'arrêt : calculer $g(x_k)$, si $\|g(x_k)\| < \varepsilon$ alors stop sinon

Etape 2 : Calcul de la direction d_k : Résoudre d'une façon approchée le système : $H(x_k)d_k = -g(x_k)$

Etape 3 : Recherche linéaire, trouver t_k solution du problème $\min_{t \geq 0} f(x_k + td_k)$

Etape 4 : Si la recherche linéaire réussie, $x_{k+1} = x_k + t_k d_k$, remplacer k par $k + 1$ et aller en 1.

Théorème 4.2 (Convergence globale). Supposons que g Lipschitzien sur la tranche

$$\mathcal{L} = \{x \in \mathbb{R}^n / f(x) \leq f(x_0)\}$$

et soit $\{x_k\}$, $\{B_k\}$ la suite des vecteurs et la suite des matrices générées par l'algorithme 4.2 ou 4.3, utilisant une recherche linéaire de Wolfe. Si

- La suite $\{f(x_k)\}$ est bornée inférieurement,
- La suite de matrices $\{B_k\}$ est symétrique et définie positive,
- Il existe $\mu > 0$ tel que $\|B_k\| \|B_k^{-1}\| < \mu$, $\forall k \in \{0, 1, \dots\}$.

Alors, $\liminf_{k \rightarrow \infty} \|g(x_k)\| = 0$.

Démonstration.

Il suffit de démontrer que $c_k = -\frac{g^t(x_k)d_k}{\|g(x_k)\| \|d_k\|} \geq \bar{\mu} > 0$ et on applique le théorème 3.7.

En effet, si $\lambda_n = \max(\sigma(B_k))$ et puisque B_k est symétrique définie positive $\Rightarrow B_k^{-1}$ symétrique et définie positive et $\lambda_n^{-1} = \min(\sigma(B_k^{-1}))$.

Grâce à la formule 1.5 et au théorème 1.3, on a

$$\begin{aligned} c_k &= -\frac{g(x_k)^\top d_k}{\|g(x_k)\| \|d_k\|} = \frac{g(x_k)^\top B_k^{-1} g(x_k)}{\|g(x_k)\| \|B_k^{-1} g(x_k)\|} \\ &\geq \frac{\lambda_n^{-1} \|d_k\|^2}{\|g(x_k)\| \|B_k^{-1} g(x_k)\|} \geq \frac{1}{\lambda_n \|B_k^{-1}\|} = \frac{1}{\|B_k\| \|B_k^{-1}\|} \\ &\geq \bar{\mu} \end{aligned}$$

□

Exemple 4.1. On applique les algorithmes 4.2 et 4.3 avec une recherche linéaire de Wolfe sur le problème test de l'exemple 3.1, les résultats sont illustrés dans les tableaux 4.1 et 4.2

k	t_k	x_k	$f(x_k)$	$\ g(x_k)\ $
0	—	(−1.200, 1.000)	24.201	232.867
1	1.000	(−1.170, 1.380)	4.731	4.639
2	0.117	(−0.947, 0.844)	4.067	26.020
3	1.000	(−0.778, 0.576)	3.243	13.716
4	1.000	(−0.513, 0.193)	2.779	22.312
5	1.000	(−0.412, 0.160)	2.005	4.9507
6	0.490	(−0.185, −0.022)	1.729	13.165
7	1.000	(−0.089, −0.001)	1.193	3.101
8	0.490	(0.099, −0.030)	0.973	8.064
9	1.000	(0.198, 0.029)	0.651	2.137
10	1.000	(0.814, 0.642)	7.565×10^{-02}	7.439
15	1.000	(0.850, 0.722)	2.239×10^{-02}	0.314
16	0.700	(0.933, 0.863)	9.590×10^{-03}	2.912
17	1.000	(0.960, 0.922)	1.606×10^{-03}	0.260
18	1.000	(0.994, 0.988)	1.634×10^{-04}	0.511
19	1.000	(0.999, 0.998)	9.929×10^{-07}	6.173×10^{-03}
20	1.000	(0.999, 0.999)	1.030×10^{-10}	4.210×10^{-04}
21	1.000	(1.000, 1.000)	4.459×10^{-19}	4.157×10^{-09}

Tableau 4.1 – Algorithmes Newton-Choleski

k	t_k	x_k	$f(x_k)$	$\ g(x_k)\ $
0	—	(−1.200, 1.000)	24.200	232.867
1	1.000	(−1.175, 1.380)	4.731	4.639
2	0.117	(−0.947, 0.844)	4.067	26.020
3	1.000	(−0.778, 0.576)	3.243	13.716
4	1.000	(−0.513, 0.193)	2.779	22.312
5	1.000	(−0.412, 0.160)	2.005	4.950
6	0.490	(−0.185, −0.022)	1.729	13.165
7	1.000	(−0.089, −0.001)	1.195	3.101
8	0.490	(0.099, −0.030)	0.973	8.064
9	1.000	(0.198, 0.029)	0.651	2.137
10	0.700	(0.387, 0.111)	0.523	9.039
11	1.000	(0.850, 0.722)	2.239×10^{-02}	0.314
16	0.700	(0.933, 0.863)	9.59×10^{-03}	2.912
17	1.000	(0.960, 0.922)	1.606×10^{-03}	0.260
18	1.000	(0.994, 0.988)	1.634×10^{-04}	0.511
19	1.000	(0.999, 0.998)	9.931×10^{-07}	6.174×10^{-03}
20	1.000	(0.999, 0.999)	1.031×10^{-10}	4.211×10^{-04}
21	1.000	(1.000, 1.000)	4.462×10^{-19}	4.159×10^{-09}

Tableau 4.2 – Algorithme Newton tronqué

Exercices

Exercice 4.1. *Utiliser la méthode de Newton pour trouver un point minimisant de*

$$f(x, y) = 5x^4 + 6y^4 - 6x^2 + 2xy + 5y^2 + 15x - 7y + 13$$

Partez de $(x_0, y_0) = (1, 1)$.

Exercice 4.2. Utiliser la méthode de Newton modifiée (avec une recherche unidimensionnelle) pour la résolution de

$$\min_{(x,y) \in \mathbb{R}^n} 2x^2 + y^2 - 2xy + 2x^3 + x^4$$

Effectuer seulement les deux premières itérations et partez de $(2, -1)$.

Exercice 4.3. Utiliser la méthode de Newton pour minimiser

$$f(x) = x - \cos x.$$

Le point initial est $x_1 = 0.5$ et le test d'arrêt est $|x_{k+1} - x_k| < \epsilon$, avec $\epsilon = 10^{-5}$.

Exercice 4.4. Soit la fonction quadratique définie par

$$f(x, y) = \frac{1}{2}(x - y)x + \frac{1}{2}(2y - x)y - x - y + 16$$

1. Calculer le gradient g et le hessien H de f

(a) Déduire de 1) la forme matricielle de f , i.e.

$$f(u) = \frac{1}{2}u^\top A u + b^\top u + c. \text{ } A \in \mathcal{M}_{2 \times 2}, b \in \mathbb{R}^2 \text{ etc } c \in \mathbb{R}.$$

2. Soit le problème de minimisation sans contrainte

$$\min_{x \in \mathbb{R}^2} f(x). \quad (4.7)$$

(a) Trouver (x^*, y^*) , l'unique point stationnaire de f .

(b) Montrer que (x^*, y^*) est un minimum global strict de f .

3. Soit l'algorithme

$$(x_{k+1}, y_{k+1})^\top = (x_k, y_k)^\top + d_k. \quad (4.8)$$

avec $d_k = -[H(x_k, y_k)]^{-1} g(x_k, y_k)$

- (a) Montrez que pour tout $(x_k, y_k) \in \mathbb{R}^2$, le vecteur d_k est une direction de descente.
- (b) Montrer que la méthode (4.8) trouve la solution exacte de (4.7) en une seule itération $\forall (x_1, y_1) \in \mathbb{R}^2$.
- (c) Donner une explication au résultat précédent à partir des conditions d'optimalité.

Exercice 4.5. Montrer que si f est une fonction quadratique de la forme

$$f(x) = \frac{1}{2} x^\top Q x + b^\top x.$$

Où Q est une matrice symétrique, définie positive. Alors la méthode de Newton trouve le minimum de f après une seule itération. Est-ce-que la méthode de Newton modifiée :

$$\begin{aligned} x_{k+1} &= x_k + t_k d_k \\ d_k &= -Q^{-1} g(x_k) \\ t_k &= \arg \min \{f(x_k + t d_k), t > 0\}, \end{aligned}$$

possède la même propriété ? Justifier votre réponse.

Bibliographie

- [1] L. Armijo. Minimization of functions having Lipschitz continuous first derivatives. Pacific Journal of Mathematics 16 (1966) 1-3
- [2] M. S. Bazara. H.D. Sherali, C.M Shetty. Nonlinear programming. J.W&S (1993)
- [3] J. F. Bonnans, J. C. Gilbert, C. Lemaréchal and C. A. Sagastizábal. Numerical optimization : theoretical and practical aspects. Springer Science & Business Media (2006).
- [4] C. G. Broyden. Quasi-Newton method and their application to function minimization, Mathematics of Computation, 21,(1967).
- [5] A. Bukley, A.Lenir. QN-like variable storage conjugate gradients. Math. Prog. 27,(1983)155-175.
- [6] R. Byrd and J. Nocedal. A tool for the analysis of quasi-Newton methods with application to unconstrained minimization, SIAM Journal on Numerical Analysis, 26 (1989) 727-739.
- [7] A. Cauchy. Méthodes générales pour la résolution des systèmes d'équations simultanées. Compte rendus Acad. Sc. Paris, Tome 25 (1847) 536-538.
- [8] F. Chatelin. Eigenvalues of Matrices, J.Wiley & S (1993).
- [9] W. C. Davidon. Variable metric methods for minimization, AEC Research Development, Report ANL-5990 (1959).

- [10] J. E. Denis, Jr., J.J. Moré. A characterisation of superlinear convergence and its application to quasi-Newton methods, *Mathematics of computation*, 28 (1974) 549-560.
- [11] J. E. Dennis, H. Wolkowicz. Sizing and Least Change Secant Methods, CORR Report 93-11 (1993).
- [12] R. Fletcher. A new approach to variable metric algorithms, *Computer Journal*, 13,(1970)317-322.
- [13] R. Fletcher, M.J.D. Powell. A Rapidly convergent descent method for minimisation, *Computer Journal* 6,(1963)163-168.
- [14] R. Fletcher. An overview of Unconstrained Optimization in Algorithms for Continuous Optimization, *The State of the Art*, E. Spedicato, ed., Kluwer academic Publishers, Boston, (1994)109-143.
- [15] A. A. Goldstein, J.F. Price. An effective algorithm for minimization. *Num. Math.* 10,(1967)184-189.
- [16] D. Goldfarb. A family of metric variable methods derived by variation means, *Mathematics Of computation* 24,(1970)23-26.
- [17] A. Griewank, Ph.L. Toint. Local convergence analysis of partitioned quasi-Newton updates, *Numeric Mathematics* 39,(1982)429-448.
- [18] T.G. Kolda. Limited-Memory matrix methods with application, *Applied Mathematics program*, university of Maryland (1998).
- [19] A. S.Lewis. Convex analysis on the hermitian matrices, Technical report CORR 97-09, University of Waterloo (1994).
- [20] D. H. Li, M. Fukushima. A global convergence of BFGS methods for nonconvex unconstrained optimization problems, *SIAM Journal on Numerical Analysis*.7, (1999).
- [21] D. H. Li, M. Fukushima. A modified BFGS method and its global convergence in non-convex minimization, Technical Report 98003, Département of Applied Mathematics and Physics, Kyoto University, (1998).

- [22] D. G. Luenberger. Linear and Nonlinear Programming, 2nd edition, Addison-Wesley (1984).
- [23] P. E. Gill, W. Murray. Conjugate gradient method for large-scale nonlinear optimization, Technical report SOL 79-15, Dept. of Operation Res. Stanford University, (1979).
- [24] J. J. Moré, B.S. Garbow, K.E. Hillstrome. Testing unconstrained optimization software, ACM Trans. Math. Software, 7, (1981)17-41.
- [25] L. . Conjugate gradient methods less dependent on conjugacy. SIAM Review 28,(1986)501-511.
- [26] Y. E. Nesterov. On the a approach to the construction of optimal methods of minimization of smooth convex function, *Economica I Matem.* 24, (1988)504-517.
- [27] J. Nocedal. Adapting quasi-Newton matrices with limited storage, *Mathematics of computation* 35, (1980)773-782.
- [28] J. Nocedal. Theory of algorithms for unconstrained optimization, *acta Numerica*, (1999)199-242.
- [29] J. Nocedal, J. W. Stephen. Numerical optimization. Springer (1999).
- [30] D. P. O'Leary. A discrete Newton algorithm for minimizing a function of many variable, *Math. Prog.* 23, (1982)20-30.
- [31] S. S. Oren, E. Spedicato, Optimal condition of self-scaling variable metric algorithms, *Math. Prg.*, 10, (1976)70-90.
- [32] J. M. Perry. A class of conjugate gradient with two-step variable-metric memory, Discussion paper 269, C.M.S.E.M.S. Northwestern University (Evanston, IL, 1977)
- [33] J. D. Pearson. Variable metric methods of minimization. *Computer Journal*, Vol 12, (1971)21-36.
- [34] M. J. D. Powell. On the convergence of the variable metric algorithm, *Journal of the Institute of Mathematics and its Applications*, (1971)21-36.

- [35] M. J. D. Powell. Some properties of the variable metric algorithm for minimization without exact line searches, in Nonlinear Programming, SIAM-AMS Proceeding, Vol. IX, R.W. Cottle, and C.E. Lemke, eds., SIAM, (1976)35-72.
- [36] L. Schwartz. Analyse, topologie générale et analyse fonctionnelle, Hermann (1970).
- [37] M. Sibony, J.C. Mardon. Analyse Numérique. Systèmes linéaires et non linéaires, HERMAN (1982).
- [38] D. F. Schanno. Conditionning of quasi-Newton methods for function minimization, Mathematics of computation, 24,(1970)641-656.
- [39] D. F. Schanno. On the convergence of new conjugate gradient algorithm, SIAM Journal on numerical analysis, 15,(1978a)1247-1257.
- [40] T. Steihaug. The conjugate gradient method and trust regions in large-scale optimization, SIAM J. Num. Anal. 20, (1983)626-637.
- [41] P. Wolfe. Convergence for ascent methods. SIAM Rev. 11,226-235, 1968.
- [42] W. I. Zangwill. Nonlinear programming : a unified approach, Prentice Hall, Englewood Cliffs, RI, 1969.
- [43] G. Zoutendijk. Non linear programming, computation programming. In J. Abadie (ed), integer and non linear programming. North Holland (1970)37-86.