UNIVERSITE SIDI MOHAMED BEN ABDELLAH
FACULTE DES SCIENCES ET TECHNIQUES
Département des Mathématiques

Licence Sciences et Techniques (LST)

MATHEMATIQUES ET APPLICATIONS

MEMOIRE DE FIN D'ETUDES

Pour l'obtention du Diplôme de Licence Sciences et Techniques

Méthodes De Descente En Optimisation Numérique

Présenté par :

- **SOULAMI Hanae**

Encadré par :

- **Pr. BENAICHA Khadija**

Soutenu Le 9 Juin 2016 devant le jury composé de:

- **Pr. BENAICHA Khadija**
- **Pr. AMMOR Ouafae**
- **Pr. RAHMOUNI HASSANI Aziza**

Stage effectué à FST de Fès

Année Universitaire 2015 / 2016

Remerciements :

Je tiens tout d'abord à remercier Mme. BENAICHA Khadija, d'avoir accepter de m' encadrer, pour sa compétence, ses bonnes directives, et pour le temps qu'elle a consacré à la réalisation de ce travail, qu'elle trouve ici l'expression de ma profonde gratitude.

Je tiens également à remercier les membres de l'honorable Jury qui ont assisté à ma soutenance.

Je ne passerai cette occasion sans remercier tous les professeurs qui m'ont enseigné à la FST de FES, et particulièrement les professeurs du département de Mathématiques.

Finalement, je ne peux oublier de remercier profondément les êtres les plus chers, mes parents, pour leur soutien, leurs encouragements et la confiance qu'ils m'accordaient depuis toujours, ainsi que mes amis pour tous les bons moments passés ensembles. ENCORE MERCI !

Table des matières

Introduction

Depuis bien longtemps, l'homme cherche à optimiser : minimiser un coût ou bien maximiser un gain ou un profit, pour se faire il a mis en évidence certaines procédures qui permettent d'optimiser tout problème rencontré.

Commençons tout d'abord par rappeler le concept d'un problème d'optimisation.

Tout problème d'optimisation comporte une étape essentielle : la modélisation mathématique.

Elle consiste en trois étapes :

1. Identification des variables de décisions qui seront désignées dans ce travail par un vecteur $x \in \mathbb{R}^n$: ce sont les paramètres sur lesquels l'utilisateur peut agir pour faire évoluer le système considéré.
2. Définition d'une fonction coût (appelée fonction objectif) permettant d'évaluer l'état du système (ex : rendement, performance,…).
3. Description des contraintes imposées aux variables de décision.

Le problème d'optimisation consiste alors à déterminer les variables de décision conduisant aux meilleures conditions de fonctionnement du système (ce qui revient à minimiser ou maximiser la fonction coût), tout en respectant les contraintes d'utilisation définies à l'étape 3.

Dans le présent travail, nous nous intéressons aux problèmes d'optimisation numérique sans contraintes dont l'ensemble des contraintes est défini sur $\mathbb{R}^n$tout entier, agissants sur des fonctionnelles non-linéaires, continues et différentiables.

Le premier chapitre commence par traiter des notions mathématiques fondamentales à maîtriser avant de s'intéresser à la résolution à proprement parler du problème d'optimisation, ainsi que la description mathématique d'un problème d'optimisation sans contraintes, la notion de solution locale et globale et quelques éléments d'analyse convexe.

Dans les chapitres suivants, nous entrons dans le vif du sujet en nous intéressant à la résolution des problèmes d'optimisation non linéaire sans contraintes, tout en introduisant les méthodes et algorithmes de résolution appropriées.

Partie 1 : Etude théorique

Chapitre 1

I. Généralités et étude théorique sur les problèmes d'optimisation sans contraintes.

1-1 Rappels et notions élémentaires.

Commençons ce paragraphe par quelques rappels et notions élémentaires.

1) Pour tout $n \in \mathbb{N}^*$, $\mathbb{R}^n$ désigne **l'espace euclidien** $\mathbb{R} \times \mathbb{R} \times \dots \mathbb{R}$ (produit n fois).

En général un vecteur $x \in \mathbb{R}^n$ sera noté $x = (x_1, x_2 \dots x_n)^T$ (vecteur colonne).

2) On considère dans cette partie une fonction f définie par :

$$f: \mathbb{R}^n \to \mathbb{R}.$$

On dit que f est continue en $x \in \mathbb{R}^n$ si $f(x^{(k)}) \to f(x)$ pour toute suite $x^{(k)} \subset \mathbb{R}^n$ telle que : $x^{(k)} \to x$.
On dit que f est continue sur $\mathbb{R}^n$ si f est continue en tout point $x \in \mathbb{R}^n$.

3) **Notion de Dérivée totale et de différentiabilité :**

Si f admet une dérivée partielle en un point $x \in \mathbb{R}^n$ qu'on note $f'(x)$, alors cette fonction peut être approchée par une approximation de Taylor.
Si $f'(x)$ existe, nous notons $E(x;h)$ la différence :

$$E(x;h) = \frac{f(x+h) - f(x)}{h} - f'(x), \quad si\ h \neq 0.$$

Et $E(x;h) = 0$. si, $h \to 0$.
De la formule précédente, nous obtenons la formule :

$$f(x+h) = f(x) + f'(x)h + hE(x;h).$$

Qui a également lieu pour $h = 0$.
C'est ce que l'on appelle le développement de Taylor du premier ordre pour approcher $f(x+h) - f(x)$ par $f'(x)h$.
Ce point de vue qui consiste à approcher une fonction différentiable par une fonction linéaire suggère de détendre le concept de differentiabilité à des dimensions supérieures.

Définition 1-1-2:

Nous disons que f est différentiable en x si il existe une transformation linéaire : $T_x: \mathbb{R}^n \to \mathbb{R}$,
Et une fonction $E(x;h)$ telle que :

$$f(x+h) = f(x) + T_x(h) + \|h\|E(x;h), (1.1)$$

Où $\lim_{\|h\|\to 0} E(x;h) = 0$, la transformation linéaire T_x est appelée la dérivée totalede f en x, ou la différentielle de f en x noté aussi $Df(x)$.
L'équation (1 .1) est appelée formule de Taylor au premier ordre pour $f(x+h)$.
Ceci donne une approximation linéaire,$T_h(h)$, de la différence $f(x+h) - f(x)$.
Le prochain théorème montre que, si la dérivée totale existe, alors elle est unique.
Il nous dit également comment calculer la dérivée totale $T_x(h), \quad \forall h \in \mathbb{R}^n$.

Théorème 1-1-1:
Supposons que f soit différentiable en x et de dérivée totale T_x.
Alors, la dérivée $f'(x;y)$ existe pour tout $y \in \mathbb{R}^n$ et nous avons :

$$f'(x;y) = \sum_{k=1}^{n} D_k f(x) y_k = Df(x).y$$

Avec $f'(x;y) = df(x;y) = \langle \nabla f(x), y \rangle$;
Ou $\nabla f(x)$ est le vecteur dont les composantes sont les dérivées partielles premières (quand elles existent) de $f$en x, appelé **gradient** de f au point x.
Nous pouvons, sous ses notations, réécrire la formule de Taylor au premier ordre :

$$f(x+h) = f(x) + \nabla f(x)h + \|h\|E(x;h).$$

Où, $\lim_{\|h\|\to 0} E(x;h) = 0$.

Théorème 1-1-2 : (une condition suffisante de différentiabilité)
Supposons que les dérivées partielles existent et sont continues en un point x.
Alors f est différentiable en x.
Remarque :
Une fonction vérifiant les hypothèses du théorème précédent est dite continument différentiable.

4) **Dérivée directionnelle :**

Définition 1-1-3 :
Soit $f: \mathbb{R}^n \to \mathbb{R}$ une application continue.
Soit $x \in \mathbb{R}^n$ et $d \in \mathbb{R}^n$.
La dérivée directionnelle de f en x dans la direction d est définie par :

$$df(x;d) := \lim_{t\to 0^+} \frac{f(x+td)-f(x)}{t}.$$ Si cette limite existe.

Proposition 1-1-1 :
Si f est différentiable en un point $x \in \mathbb{R}^n$, alors pour tout $d \neq 0, f$ admet une dérivée dans la direction d en x et :

$$df(x;d) = Df(x)(d) = \nabla f(x)^T d.$$

On rappelle que la réciproque est fausse : La dérivée directionnelle de f en x selon tout vecteur d n'implique pas nécessairement la différentiabilité de $f$en x.

5) On dit qu'un point $x^* \in \mathbb{R}^n$ est **un point critique(ou stationnaire)** pour la fonction f si $\nabla f(x^*) = 0$.

1-2 Position du problème.

On considère ici le cas particulier où l'ensemble de contraintes est défini sur l'espace $\mathbb{R}^n$ tout entier.
On va considérer le problème de minimisation suivant:

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

Où $f: \mathbb{R}^n \to \mathbb{R}$ est une fonctionnelle donnée.
C'est un problème de **minimisation sans contraintes.**
On pourra donc écrire ce problème sous forme :

$$\begin{cases} Trouver\ x^* \in \mathbb{R}^n\ tel\ que \\ f(x^*) \leq f(x), \qquad \forall x \in \mathbb{R}^n. \end{cases}$$

On supposera que x^* existe (éventuellement qu'il est unique) et on se propose de trouver une **approximation numérique** de x^*, en construisant une suite d'itérés $\{x^{(k)}\}_{k \in \mathbb{N}} \subset \mathbb{R}^n$ telle que : $x^{(k)} \to x^*$ pour $k \to +\infty$.
Cette suite d'itérés sera construite par les méthodes numériques de minimisation que nous verrons plus loin.
Résoudre un problème d'optimisation sans contraintes revient à chercher des points de minimum local (ou global, c'est encore mieux !), au sens de la définition suivante.

Définition 1-2-1:(Minimum local/Minimum global)
Soit $f: U \subset \mathbb{R}^n \to \mathbb{R}$ une fonctionnelle donnée.

- $x^* \in \mathbb{R}^n$ est un point de minimum local de f sur U si :
  $x^* \in U$ et $\exists r > 0$ $tq\ \forall x \in U \cap B(x^*, r), \qquad f(x^*) \leq f(x)$.

On dit alors que $f(x^*)$ est un minimum local de f sur U.

- $x^* \in \mathbb{R}^n$ est un point de minimum global de f sur U si et seulement si :
  $x^* \in U$ et $\forall x \in U, \quad f(x^*) \leq f(x)$.

On dit alors que $f(x^*)$ est un minimum global de f sur U.
Les notions de maximum local et global sont définies de façon tout à fait similaire.
En fait, on peut facilement démontrer que les problèmes sans contraintes :
$\min_{x \in \mathbb{R}^n} f(x)$ et $\max_{x \in \mathbb{R}^n} -f(x)$
Sont équivalents dans le sens où ils ont même ensemble de solutions
Ainsi la recherche d'un maximum pouvant se ramener à la recherche d'un minimum, nous porterons donc une attention particulière à la recherche d'un minimum.

1-3 Convexité et optimisation.

La convexité joue un rôle extrêmement important en optimisation numérique.
Les problèmes dont les données sont convexes, constituent une classe importante, car ils sont fréquemment rencontrés dans les applications et à la base de nombreuses méthodes développées pour des problèmes plus généraux.

Définition 1-3-1 :(ensemble convexe)

Un ensemble C est dit convexe si, pour tous les points x et y de C, le segment $[x, y]$ est inclus dans C, i.e. $\forall t \in [0,1], tx + (1-t)y$ est un point de C.

Exemple 1-3-1: $\mathbb{R}^n$est un ensemble convexe.

Définition 1-3-2:(fonction convexe)

Une fonction f définie sur un ensemble convexe C est dite convexe si et seulement si :

$$\forall x, y \in C, \quad \forall t \in [0,1], \quad f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

La fonction est dite strictement convexe si et seulement si :

$$\forall x, y \in C, \quad x \neq y, \quad \forall t \in]0,1[, \quad f(tx + (1-t)y) < tf(x) + (1-t)f(y).$$

Lorsqu'une fonction convexe est dérivable, la caractérisation suivante sera utile.

Proposition 1-3-1:

Soit f une fonction différentiable définie sur un convexe C de $\mathbb{R}^n$, alors f est convexe si et seulement si :

$$\forall x, y \in C, \quad \nabla f(x)(y-x) \leq f(y) - f(x).$$

Preuve :

- Soit f convexe. On considère $\theta \in [0,1]$,et $x, y \in C$.

Puisque C est convexe : $x + \theta(y+x) \in C$ et en utilisant la convexité de f, on a :

$$f\big(x + \theta(y-x)\big) \leq (1-\theta)f(x) + \theta f(y).$$

D'où :

$$f(y) - f(x) \geq \frac{f\big(x + \theta(y-x)\big) - f(x)}{\theta} \qquad 1)$$

On conclut alors en faisant tendre θ vers 0 dans 1).

- Réciproquement, on part des deux inégalités :

$$f(y) \geq f\big(y + \theta(x-y)\big) - \theta f\big(y + \theta(x-y)\big)(x-y)$$
$$f(x) \geq f\big(y + \theta(x-y)\big) + (1-\theta)f\big(y + \theta(x-y)\big)(x-y).$$

En combinant ces deux inégalités on obtient :

$f(x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$, ce qui établit la convexité.

Remarque :

On a une caractérisation similaire pour les fonctions strictement convexe : remplacer convexe par strictement convexe, et l'inégalité large par une inégalité stricte dans la proposition précédente.
Toutefois le premier sens de la preuve ne peut être directement adapté.
En effet, si l'on obtient bien 1) avec une inégalité stricte, cette dernière n'est pas nécessairement conservée lorsque θ tend vers 0.

Proposition 1-3-2 :

Soit f une fonction convexe définie sur un ensemble convexe C. Alors ;

- Tout minimum local de f sur C est un minimum global.
- Si f est strictement convexe, il y'a au plus un minimum global (unicité du minimum global).

Preuve :

- Soit $x_0 \in C$ un minimum local, i.e. il existe $r > 0$ tel que $B(x_0,r) \subset C$ et :

$$\forall x \in B \quad f(x) \geq f(x_0). \qquad 2)$$

Soit $y \in C$. On considère g définie par :
$g(t) = f(ty + (1-t)x_0)$. 3)
Soit $z = \partial B \cap [x,y]$. Il existe $\epsilon \in [0,1]$ tel que : $z = \epsilon y + (1-\epsilon)x_0$. La relation 2) implique que ;
$\forall t \in [0,\epsilon] \quad g(t) \geq f(x_0)$.
En utilisant cette dernière inégalité et le caractère convexe de f nous obtenons :
$\forall t \in [0,\epsilon] \quad f(x_0) \leq g(t) \leq tf(y) + (1-t)f(x_0)$.
D'où :
$t(f(y) - f(x_0)) \geq 0 \quad \forall t \in [0,\epsilon]$, et en prenant $t = \epsilon$ on obtient $f(y) \geq f(x_0)$.

- Soit f strictement convexe. On suppose que f admet deux minimum globaux distincts x, y.
- Soit alors $t \in]0,1[$ et $z = tx + (1-t)y$.

On aboutit à la contradiction suivante :$f(z) < tf(x) + (1-t)f(y) = \min_C f$.
D'où l'unicité du point de minimum global.

1-4 Résultat d'existence, cas d'unicité:

La plupart des théorèmes d'existence de minimum sont des variantes du théorème classique suivant : une fonction continue sur un compact admet un minimum.
Commençons par le théorème suivant :

Théorème 1-4-1 :

Soit f une fonction continue sur un sous-ensemble C fermé de $\mathbb{R}^n$.
On suppose que :

- Ou bien C est borné,
- Ou bien C est non borné et $\lim_{\|x\| \to +\infty} f(x) = +\infty$ (on dit que f est coercive),

Alors f possède un minimum sur C.

Le résultat d'unicité du point de minimum global est prouvé précédemment, sous la condition de convexité stricte de f.

1-5 Condition d'optimalité :

Nous supposons dans tout ce paragraphe que f est un ou deux fois différentiable.
On notera x^* un minimum (local) de f.
Ce qui suit reste valable dans le cas où le minimum x^* se trouve à l'intérieur de l'ensemble des contraintes.
Nous donnons les conditions nécessaires de minimum, puis celles suffisantes.

Théorème 1-5-1:(condition nécessaires.)

Les deux conditions nécessaires sont les suivantes.

- Condition au premier ordre : si f est différentiable en x^*, on a :

$$\nabla f(x^*) = 0,$$

- Condition au second ordre : si f est deux fois différentiable au point x^*, alors la forme quadratique $D^2 f(x^*)$ est positive i.e.

$$\langle D^2 f(x^*)y, y\rangle \geq 0,$$

Où $D^2 f(x^*)$ est la matrice Hessienne, définie par les coefficients : $\frac{\partial^2 f}{\partial x_i \partial x_j}(x^*)$.

Preuve :

i) On considère un développement limité à l'ordre un en x^* :

$f(x^* + h) = f(x^*) + \nabla f(x^*)h + \|h\|E(x^*; h)$.
Soit $t \geq 0$. On choisit $h = -t\nabla f(x^*)$ et on pose $\alpha = \|\nabla f(x^*)\|$.
Ce qui donne :
$f(x^* + h) = f(x^*) + t\alpha^2 + t\alpha E(x^*; h)$.
Supposons que $\alpha > 0$. Puisque $E(x^*; h) \underset{t\to 0}{\longrightarrow} 0$ on vérifie alors qu'il existe $t^* > 0$ tel que :
$\forall t \leq t^*$: $f(x^* + h) < f(x^*)$. Ceci indique que x^* n'est pas un minimum. Ainsi il est nécessaire que $\alpha = 0$.

ii) Supposons qu'il existe $y \in \mathbb{R}^n, y \neq 0$ tel que $\alpha := \langle D^2 f(x^*)y, y\rangle < 0$. On considère un développement limité à l'ordre deux en x^* :

$$f(x^* + ty) = f(x^*)\frac{1}{2}t^2\alpha + t^2\alpha^2\alpha E_2(x^*; ty).$$

Puisque $E_2(x^*; ty) \underset{t\to 0}{\longrightarrow} 0$, on vérifie qu'il existe $t^* > 0$ tel que $\forall t \in [0; t^*]$:
$f(x^* + ty) < f(x^*)$. Ceci indique que x^* n'est pas un minimum.
Ainsi il est donc nécessaire que $D^2 f(x^*)$ soit positive.
Ce qui achève la preuve de ce théorème.
Nous énonçons à présent des conditions nécessaires et suffisantes pour qu'un point x^* soit un minimum, dans le cas où f est suffisamment régulière.

Théorème 1-5-2:(Conditions nécessaires et suffisantes.)

Soit f une fonction de classe C^1 définie sur $\mathbb{R}^n$. On suppose que : $\nabla f(x^*) = 0$ et que f deux fois différentiable en x^*.
Alors, x^* est un minimum (local) de f si et seulement si l'une des deux conditions suivantes est vérifiée :

i) $D^2 f(x^*)$ est définie positive.

ii) $\exists r > 0$ tel que f est deux fois différentiable sur $B(x^*, r)$ et la forme quadratique $D^2 f(x)$ est positive pour tout $x \in B(x^*, r)$.

Preuve :

- la condition nécessaire au premier ordre et au deuxième ordre en x^* est vérifiée dans les deux cas i) et ii).
- On vérifie que i) ou ii) est une condition suffisante.

Pour ii) : on considère la formule de Taylor-Mac-Laurin en x^* à l'ordre deux, pour tout $h \in B(0, r)$ il existe $\lambda = \lambda(h) \in [0,1]$, donnée par :

$$f(x^* + h) = f(x^*) + \frac{1}{2}\langle D^2 f(x^* + \lambda h)h, h\rangle \geq f(x^*),$$

Ce qui montre bien que x^* est un minimum.
Ce qui achève la preuve du théorème.
Dans le cas où f est convexe, la condition suffisante s'exprime beaucoup plus facilement.
En effet, nous avons la proposition suivante :

Proposition 1-5-1 :

Soit f une fonction convexe de classe C^1, définie sur $\mathbb{R}^n$ et x^* un point de $\mathbb{R}^n$. Alors, x^* est un minimum (global) de f si et seulement si :
$\nabla f(x^*) = 0$.

Preuve :
Il suffit de montrer que la condition est suffisante.
Soit y quelconque. D'après la proposition précédente nous avons :
$f(y) - f(x^*) \geq \nabla f(x^*)(y - x^*) = 0$,
Ce qui montre bien que x^* est un minimum.

Chapitre 2

II. Généralités sur les méthodes de descente.

Une grande classe d'algorithmes que nous allons considérer dans ce chapitre pour la résolution des problèmes d'optimisation sans contraintes ont la forme générale suivante :

$$\begin{cases} x^{(0)} \text{ étant donné dans } \mathbb{R}^n. \\ \text{calculer } x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}. \end{cases}$$

Le vecteur $d^{(k)}$ s'appelle direction de descente, $\rho^{(k)}$ le pas de la méthode à la $k - ième$ itération.

Considérons une fonction J définie par :

$J: \mathbb{R}^n \to \mathbb{R}$.

En pratique, on choisi $\rho^{(k)}$ et $d^{(k)}$ de façon à satisfaire l'inégalité suivante :

$J(x^{(k+1)}) \leq J(x^{(k)})$.

De tels algorithmes sont souvent appelés algorithmes de descente.

Essentiellement, la différence entre ces algorithmes réside dans le choix de la direction de descente $d^{(k)}$.

2-1 Direction de descente.

Le gradient joue un rôle essentiel en optimisation.

Dans le cadre des méthodes d'optimisation, il sera également important d'analyser le comportement de la fonction objectif dans certaines directions.

Pour cela, introduisons la notion de dérivée directionnelle évoquée au chapitre 1), qui nous permet d'avoir des informations sur la pente de la fonction dans la direction d, tout comme la dérivée qui donne des informations sur la pente des fonctions à une variable.

En particulier :

- Si $dJ(x;d) > 0$ alors J est croissante dans la direction d.
- Si $dJ(x;d) < 0$ alors J est décroissante dans la direction d.

Dans ce dernier cas, on dira que d est une direction de descente de J.

Définition 2-1-1:(direction de descente d)

Soient $J: \mathbb{R}^n \to \mathbb{R}$ et $x \in \mathbb{R}^n$. Le vecteur $d \in \mathbb{R}^n$ est une direction de descente pour J à partir du point x si $t \to J(x + td)$ est décroissante en $t = 0$.

C'est-à-dire s'il existe $\mu > 0$ tel que :

$\forall t \in]0, \mu], J(x + td) < J(x)$.

Proposition 2-1-1:

Soient $J: \mathbb{R}^n \to \mathbb{R}$ différentiable et $x \in \mathbb{R}^n$ tel que : $\nabla J(x) \neq 0$.

Le vecteur $d \in \mathbb{R}^n$ est une direction de descente pour J à partir du point x si et seulement si la dérivée directionnelle de J en x dans la direction d vérifie :

$dJ(x;d) = \nabla J(x)^T d < 0$.

Autrement dit, la relation (*) nous dit que la décroissance de la fonction objectif J en faisant un pas de taille t dans la direction d, est au moins le pas fois une fraction β de la pente.

Exemple 2-1-1:

Allure de la fonction $f: x \to \frac{1}{2}x_1^2 + 2x_2^2$ au point $x = (1,1)^T$ dans plusieurs directions.

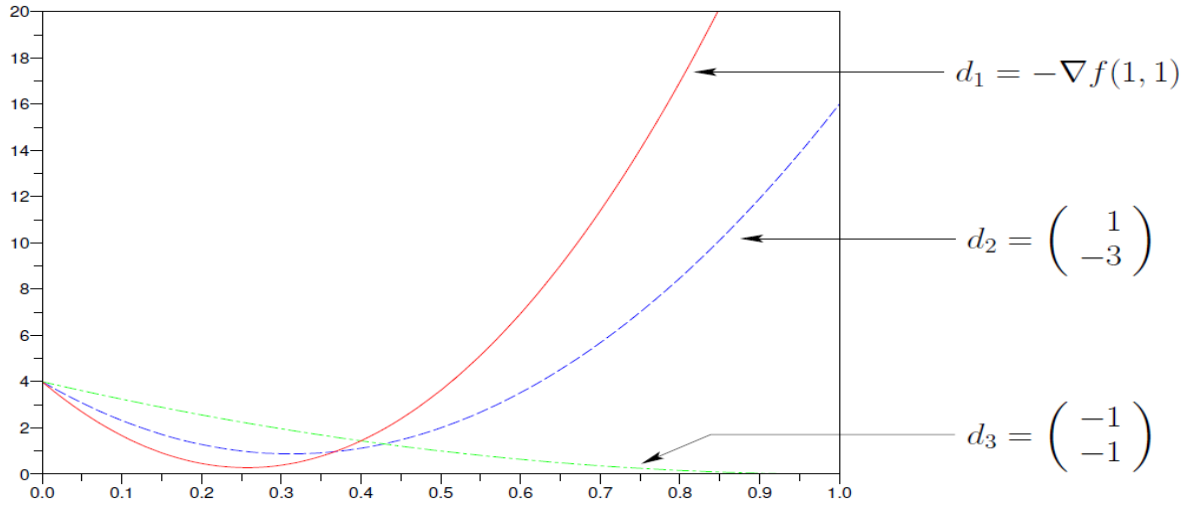


Figure 1. Allure de la fonction $f: x \to \frac{1}{2}x_1^2 + 2x_2^2$ dans plusieurs directions.

Nous avons donc le résultat suivant ;

Théorème 2-1-1:

Soit $J: \mathbb{R}^n \to \mathbb{R}$ une fonctionnelle différentiable.
Soit $x \in \mathbb{R}^n$, alors pour toute direction de norme constante égale à $\|d\| = \|\nabla J(x)\|$, on a :
$(-\nabla J(x))^T \nabla J(x) \leq d^T \nabla J(x)$,
La direction $d^* = -\nabla J(x)$ est appelée direction de plus forte descente.

Preuve :

Soit $d \in \mathbb{R}^n$ une direction quelconque de norme : $\|d\| = \|\nabla J(x)\|$.
On a alors :
D'après l'inégalité de Cauchy-Schwarz et le fait que : $\|d\| = \|\nabla J(x)\|$.
Nous avons :
$(-d)^T \nabla J(x) \leq \|-d\| \|\nabla J(x)\| \leq \|\nabla J(x)\|^2 = \nabla J(x)^T \nabla J(x)$ (**)
On en déduit donc la formule (**).
Sans surprise, si l'opposé du gradient correspond à la plus forte descente, le gradient correspond, lui, à la plus forte montée.

Corollaire 2-1-1:

Le vecteur $\nabla J(x)$ est appelé direction de plus forte pente de J au point $x \in \mathbb{R}^n$.
On remarquera que si x^* est un point minimum local de J, alors il n'existe aucune direction de descente pour J au point x^*.

Cette direction étant choisie nous sommes plus ou moins ramenés à un problème unidimensionnel pour la détermination de $\rho^{(k)}$.
Pour ces raisons, commençons par analyser ce qui se passe dans le cas de dimension un.

2-2 Recherche linéaire : Méthodes de recherche du pas de descente.

En optimisation mathématique, la **recherche linéaire** est l'une des deux approches classiques permettant de forcer et d'accélérer la convergence des algorithmes de calcul d'un minimum x^* d'une fonction $J: \mathbb{R}^n \to \mathbb{R}$, lorsque le premier itéré est éloigné d'un tel minimum.
L'autre méthode est celle **des régions de confiance**.
Dans cette partie nous nous intéressons à la recherche linéaire du pas de descente d'un point minimum x^*.
Soit $q(\rho)$ la fonction coût que l'on cherche à minimiser.
On pourra prendre par exemple :

$$q(\rho) = J(x^{(k)} + \rho d^{(k)})$$

Afin d'appliquer les idées au cas de la méthode de descente.
La recherche du pas n'est qu'une étape d'un algorithme plus complexe pour minimiser J.
La philosophie générale est alors de plutôt essayer d'avoir une approximation satisfaisante du pas optimal.
Si nous considérons $q(\rho) = J(x + \rho d)$, avec $\rho > 0$, nous avons :

$$q'(p) = \nabla J(x + \rho d).\, d,$$

Par conséquent, puisque d est une direction de descente, $q'(p) = \nabla J(x).d < 0,$ il s'ensuit que si nous prenons ρ un peu à droite de 0, on est sûr de faire décroître q.
Toutefois, il faut faire attention car deux éléments contradictoires sont à prendre en compte.

- Si ρ est trop grand, on risque de ne pas faire décroître la fonction q ou son comportement peut être oscillant,
- Si ρ est trop petit, l'algorithme n'avancera pas assez vite.

Il existe des règles classiques pour parvenir à cette fin.

a) Règle d'Armijo (1966) :

La règle d'Armijo se base sur le choix d'un paramètre $0 < m < 1$ et détermine une valeur approchée de ρ sous la condition :

$$q(\rho) \leq q(0) + m\rho q'(0).$$

Le risque de cette méthode est de favoriser les valeurs trop petites, aussi, elle est rarement utilisée seule.

b) Règle de Goldstein (1967) :

On choisit deux paramètres (m_1, m_2) tels que $0 < m_1 < m_2 < 1$ et on cherche une valeur ρ qui vérifie :

$$\begin{cases} q(\rho) \leq q(0) + m_1 \rho q'(0) \\ q(\rho) \geq q(0) + m_2 \rho q'(0) \end{cases}$$

c) Règle de Wolfe (1969) :

On choisit deux paramètres $m_1$et m_2, avec $0 < m_1 < m_2 < 1$ (par exemple $m_1 = 0.1$ et $m_2 = 0.7$), et on recherche ρ qui vérifie :

$$\begin{cases} q(\rho) \leq q(0) + m_1 \rho q'(0) \\ q(\rho) \geq m_2 q'(0) \end{cases}$$

L'algorithme associé à la règle de Wolfe est alors donné par la table ci-dessous, celui pour la méthode de Goldstein étant similaire

Soit J la fonction à minimiser.

A l'itération k, nous avons $x = x^{(k)}$ et $d = d^{(k)}$, et on veut calculer :

$x^{(k+1)} = x^{(k)} + \rho d^{(k)}$

pour une valeur ρ à déterminer.

Partir de $k = 0$; Choisir un point $x^{(0)}$ de $\mathbb{R}^n$.

Choisir $m1$ et $m2$ tel que : $0 < m1 < m2 < 1$;

En pratique on prendra ($m1 = 10\text{^}(-4)$ et $m2 = 0{,}9999$) ;

Poser $\rho = 1$; $\rho_+ = 0$; $\rho_- = 0$.

Tant que la première condition de Wolfe n'est pas vérifiée :

$q(\rho) \leq q(0) + m1\rho q'(0)$; et $(k \leq kmax)$;

Alors poser : $\rho_+^{(k)} = \rho^{(k)}$; Et $\rho^{(k+1)} = \frac{\rho_- + \rho_+}{2}$;

Poser $k \leftarrow k + 1$; Fin tant que ;

Tant que la deuxième condition de Wolfe n'est pas vérifiée :

$q(\rho) \geq m2q'(0)$; et $(k \leq kmax)$;

Alors poser : $\rho_-^{(k)} = \rho^{(k)}$;

Si $\rho_+ = 0$ on a : $\rho^{(k+1)} = 2\rho^{(k)}$;

Sinon, on a : $\rho^{(k+1)} = \frac{\rho_- + \rho_+}{2}$; Fin si

Poser $k \leftarrow k + 1$; Fin tant que ;

Retourner $\rho^{(k)}$; Fin.

Figure 2. Algorithme de Wolfe.

2-3 Algorithme de descente.

Partant d'un point $x^{(0)}$ arbitrairement choisi, un algorithme de descente va chercher à générer une suite d'itérés $(x^{(k)})_{k\in\mathbb{N}}$ tel que :
$\forall k \in \mathbb{N}, J(x^{(k+1)}) \leq J(x^{(k)})$.
D'après la caractérisation de la descente, il s'agit donc à chaque itération k, de trouver un point $x^{(k+1)}$ dans une direction d vérifiant : $\nabla J(x^{(k)})^{\mathrm{T}} d < 0$.
Le schéma général d'un algorithme de descente est le suivant :

Entrée : $J: \mathbb{R}^n \to \mathbb{R}$ supposé au moins différentiable, $x^{(0)}$ un point initial arbitrairement choisi.

Sortie : une approximation de la solution du problème : $\min_{x\in\mathbb{R}^n} J(x)$.

1) Poser $k = 0$.
2) Tant que « test de convergence » non satisfait.
   a) Trouver une direction de descente $d^{(k)}$ tel que : $\nabla J(x^{(k)})^T d^{(k)} < 0$.
   b) Recherche linéaire : Choisir un pas $\rho^{(k)} > 0$ à faire dans cette direction tel que :
   $$J(x^{(k)} + \rho^{(k)} d^{(k)}) \leq J(x^{(k)}).$$
   c) Mise à jour :
   $$x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}.$$
   Poser $k = k + 1$;
   Fin tant que.
3) Retourner $x^{(k)}$.

Figure 3. Algorithme de descente.

- **Test de convergence / Test d'arrêt.**

Soit x^*un point de minimum local du critère de *J* à optimiser.
Supposons que l'on choisisse comme test d'arrêt dans l'algorithme de descente, le critère idéale : «$x^{(k)} = x^*$ ».

En pratique , un test d'arrêt devra être choisi pour garantir que l'algorithme s'arrête toujours après un nombre fini d'itérations et que le dernier point calculé soit suffisamment proche de x^*.
Soit $\varepsilon > 0$ la précision demandée, on test si $\|\nabla J(x_k)\| \leq \varepsilon$.
Dans ce cas l'algorithme s'arrête et fournit l'itéré courant $x^{(k)}$ comme solution.
En pratique, le test d'optimalité n'est pas toujours satisfait et on devra faire appel à d'autres critères (fondés sur l'expérience numérique) ;

- Stagnation de solution : $\|x^{(k+1)} - x^{(k)}\| < \varepsilon \|x^{(k)}\|$.
- Stagnation de la valeur courante : $\|J(x^{(k+1)}) - J(x^{(k)})\| < \varepsilon \|J(x^{(k)})\|$.

- Nombre d'itérations dépassant un seuil fixé à l'avance : $k < k_{max}$.

Et généralement une combinaison de ces critères :
Critère d'arrêt = Test d'optimalité satisfait
OU (Stagnation de la valeur courante et stagnation de la solution)
OU Nombre d'itérations maximum autorisé dépassé.

2-4 Convergence et vitesse de convergence d'un algorithme.

Etudier la convergence d'un algorithme, c'est étudier la convergence de la suite des itérés $(x^{(k)})_{k\in\mathbb{N}}$ générées par l'algorithme.
Un algorithme de descente selon le modèle précédent, est dit convergent si la suite de ses itérés $(x^{(k)})_{k\in\mathbb{N}}$ converge vers un point limite x^*, solution du problème :

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

De plus, la convergence est dite locale sil elle n'a lieu que pour des points initiaux $x^{(0)}$ dans un voisinage de x^*.
Sinon elle est dite globale.
En pratique, le but d'un algorithme d'optimisation est de trouver un point critique (i.e. un point vérifiant la condition d'optimalité du premier ordre : $\nabla J(x^*) = 0$).
On introduit alors la notion de convergence d'un algorithme d'optimisation.

Définition 2-4-1:
Soit un algorithme itératif qui génère une suite $(x^{(k)})_{k\in\mathbb{N}}$ dans $\mathbb{R}^n$ afin de résoudre le problème :

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

Où, $J: \mathbb{R}^n \to \mathbb{R}$ est une application de classe C^1.

L'algorithme est dit globalement convergent si quel que soit le point initial $x^{(0)} \in \mathbb{R}^n$, $\lim_{k\to+\infty} \|\nabla J(x^{(k)})\| = 0$.
Cette propriété garantit que le critère d'arrêt $\|\nabla J(x^{(k)})\| \leq \varepsilon$ sera satisfait à partir d'un certain rang quelle que soit la précision $\varepsilon > 0$, demandée.
Il est bien entendu très important de garantir la convergence d'un algorithme sous certaines hypothèses, mais la vitesse de convergence et la complexité sont également des facteurs à prendre en compte lors de la conception ou de l'utilisation d'un algorithme ; en effet, on a tout intérêt à ce que la méthode choisie soit à la fois rapide, précise et stable.
Pour cela, on introduit les notions de vitesse (ou taux) de convergence qui mesurent l'évolution de l'erreur commise :$\|x_k - x^*\|$.

Définition 2-4-2 :

Soit $(x^{(k)})_{k\in\mathbb{N}}$ une suite d'itérés générées par un algorithme convergent donné.
On note x^* la limité de la suite et on suppose que : $\forall k \in \mathbb{N}\ , x_k \neq x^*$(sinon l'algorithme convergerait en un nombre fini d'itérations).

La convergence de l'algorithme est dite :

- Linéaire si l'erreur $e^{(k)} = \left\|x^{(k)} - x^*\right\|$ décroit linéairement i.e s'il existe $\tau \in]0;1[$ tel que

$$\lim_{k\to+\infty} \frac{\left\|x^{(k+1)} - x^*\right\|}{\left\|x^{(k)} - x^*\right\|} = \tau.$$

- Super-linéaire si

$$\lim_{k\to+\infty} \frac{\left\|x^{(k+1)} - x^*\right\|}{\left\|x^{(k)} - x^*\right\|} = 0.$$

- D'ordre p s'il existe $\tau \geq 0$ tel que :

$$\lim_{k\to+\infty} \frac{\left\|x^{(k+1)} - x^*\right\|}{\left\|x^{(k)} - x^*\right\|^p} = \tau.$$

En particulier, si p=2, la convergence est dite quadratique (grosso modo à partir d'un certain rang, le nombre de chiffres significatifs exacts double à chaque itérations).

Bien entendu, on a intérêt à ce que la convergence d'un algorithme soit la plus élevée possible afin de converger vers la solution en un minimum d'itérations pour une précision donnée.

Chapitre 3

III. Méthodes et Algorithmes de minimisation.

3-1 Méthode du gradient.

La méthode du gradient fait partie de la classe des méthodes dites de descente.
La suite de vecteurs $x^{(k)}$ sera donc générée de la manière suivante :

$$\begin{cases} x^{(0)} donné\ dans\ \mathbb{R}^n. \\ x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}. \end{cases}$$

Avec $d^{(k)} \in \mathbb{R}^n$ et $\rho^{(k)} > 0$.
De nombreux choix existent pour $d^{(k)}$et $\rho^{(k)}$.
Nous devons en premier lieu choisir la direction de descente.
Rappelons que le développement de Taylor de J au premier ordre au voisinage de $x^{(k+1)}$ est donné par :

$$J\left(x^{(k+1)}\right) = J\left(x^{(k)} + \rho^{(k)} d^{(k)}\right)$$
$$= J\left(x^{(k)}\right) + \rho^{(k)} \nabla J\left(x^{(k)}\right) d^{(k)} + \rho^{(k)} \left\| d^{(k)} \right\| E\left(x^{(k)}; \rho^{(k)} d^{(k)}\right).$$

Où : $\lim_{\rho^{(k)} d^{(k)} \to 0} E\left(x^{(k)}; \rho^{(k)} d^{(k)}\right) = 0.$

Or, puisque l'on désire avoir : $J\left(x^{(k+1)}\right) \leq J\left(x^{(k)}\right)$, une solution évidente consiste à prendre : $d^{(k)} = -\nabla J(x^{(k)})$.
Puisqu' alors

$$J\left(x^{(k+1)}\right) - J\left(x^{(k)}\right) = -\rho^{(k)} \left\| \nabla J(x^{(k)}) \right\|^2 + o\left(\rho^{(k)}\right).$$

Nous voyons que si $\rho^{(k)}$est suffisamment petit, $x^{(k+1)}$ minimisera mieux J que ne le faisait $x^{(k)}$.
La méthode obtenue avec le choix $d^{(k)} = -\nabla J(x^{(k)})$ est appelée méthode du gradient.
Lorsque l'on travaille sur une résolution numérique d'un problème, on se donne en général un critère d'arrêt de la forme :$\left\| x^{(k+1)} - x^{(k)} \right\| < \varepsilon$.
De plus, puisque la convergence n'est pas toujours assurée, une règle de base est de fixer un nombre maximum d'itérations $k^{\max}$. Sinon l'algorithme génère une suite infinie de $x^{(k)}$.
On obtient alors l'algorithme du Gradient suivant :

poser $k = 0$
choisir $\mathbf{x}^{(0)}$
tant que $\left(\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \geq \varepsilon\right)$ et $\left(k \leq k^{\max}\right)$ faire
    calculer $\mathbf{d}^{(k)} = -\nabla J\left(\mathbf{x}^{(k)}\right)$
    calculer $\rho^{(k)}$
    poser $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \rho^{(k)} \mathbf{d}^{(k)}$
fin tant que

Figure 4.Algorithme du Gradient.

Remarque :

Même si ces méthodes sont conceptuellement très simples et qu'elles peuvent être programmées directement, elles sont souvent lentes dans la pratique.
Elles convergent mais sous des conditions de convergence souvent complexes.
A titre d'exemple, donnons le résultat suivant.

Théorème 3-1-1:

Soit J une fonction de classe C^1 de $\mathbb{R}^n$ dans $\mathbb{R}$, x^* est un minimum de J Supposons que :

i) J est α-elliptique, c'est-à-dire :

$$(\exists \alpha > 0)\ (\forall (x,y) \in R^n \times R^n)\langle (\nabla J(x) - \nabla J(y)), (x-y)\rangle \geq \alpha \|x-y\|^2.$$

ii) L'application ∇J est Lipchitzienne, c'est-à-dire :

$$(\exists M > 0)(\forall (x,y) \in R^n \times R^n)(\|\nabla J(x) - \nabla J(y)\| \leq M\|x-y\|).$$

S'il existe deux réels a et b tels que $\rho^{(k)}$ satisfasse $0 < a < \rho^{(k)} < b < \frac{2\alpha}{M^2}$, pour tout $k \geq 0$, alors , la méthode du gradient définie par :

$$x^{(k+1)} = x^{(k)} - \rho^{(k)} \nabla J(x^{(k)})$$

Converge pour tout choix de $x^{(0)}$ de façon géométrique, c-à-dire :

$$\exists \beta \in]0;1[\ , \left\|x^{(k)} - x^*\right\| \leq \beta^k \left\|x^{(0)} - x^*\right\|.$$

Preuve :

Nous savons que x^* est l'unique solution de l'équation d'Euler :

$$\nabla J(x^*) = 0.$$

Notre but est de montrer que $x^{(k)} - x^* \to 0$ pour $k \to +\infty$.
Nous avons :

$$x^{(k+1)} - x^{(*)} = x^{(k)} - \rho^{(k)} \nabla J\left(x^{(k)}\right) - x^* = x^{(k)} - x^* - \rho^{(k)}[\nabla J\left(x^{(k)}\right) - \nabla J(x^*)].$$

Utilisons la formule : $\|x-y\|^2 = \|x\|^2 + \|y\|^2 - 2\langle x,y\rangle$ pour tous $x, y \in \mathbb{R}^n$.
Nous avons :

$$\left\|x^{(k+1)} - x^*\right\|^2 = \left\|x^{(k)} - x^*\right\|^2 + (\rho^{(k)})^2 \left\|\nabla J\left(x^{(k)}\right) - \nabla J(x^*)\right\|^2 - 2\rho^{(k)}\langle x^{(k)} - x^*, \nabla J\left(x^{(k)}\right) - \nabla J(x^*)\rangle.$$

En utilisant l'ellipticité de J et le fait que ∇J est lipchitzienne, nous obtenons :

$$\left\|x^{(k+1)} - x^*\right\|^2 \leq \left\|x^{(k)} - x^*\right\|^2 + M^2(\rho^{(k)})^2 \left\|x^{(k)} - x^*\right\|^2 - 2\rho^{(k)}\alpha \left\|x^{(k)} - x^*\right\|^2$$

C'est-à-dire :

$$\left\|x^{(k+1)} - x^*\right\|^2 \leq \left(1 - 2\alpha\rho^{(k)} + M^2(\rho^{(k)})^2\right) \left\|x^{(k)} - x^*\right\|^2 \quad (1)$$

Considérons maintenant la fonction $\varphi: \mathbb{R} \to \mathbb{R}$ donnée par :

$$\varphi(p) = 1 - 2\alpha\rho^{(k)} + M^2\rho^2.$$

Comme $a \leq \rho^{(k)} \leq b$, il est facile de voir que :

$$\varphi\left(\rho^{(k)}\right) \leq max\{\varphi(a), \varphi(b)\} \quad (2)$$

D'autre part, le minimum de φest atteint en $\frac{\alpha}{M^2}$,

ce qui nous donne :

$$\varphi\left(\rho^{(k)}\right) \geq 1 - \frac{\alpha^2}{M^2} \quad (3)$$

En utilisant l'inégalité de Cauchy-Schwarz, nous avons pour tous $x, y \in \mathbb{R}^n$:
$\alpha\|x-y\|^2 \leq (\nabla J(x) - \nabla J(y), x-y) \leq \|\nabla J(x) - \nabla J(y)\|\|x-y\| \leq M\|x-y\|^2$.
Ce qui nous donne, en prenant $x \neq y$:

$$\alpha \leq M. \quad (4)$$

Comme $\varphi(0) = \varphi\left(\frac{2\alpha}{M^2}\right) = 1$, on déduit grâce aux inégalités $(3) et\ (4)$ que :

$$\varphi(y) \in [0,1[, \qquad \forall y \in]0, \frac{2\alpha}{M^2}[.$$

Posons alors $\beta = \sqrt{\max\{\varphi(a), \varphi(b)\}}$ .Comme $a, b \in]0, \frac{2\alpha}{M^2}[$ on a clairement :

$$0 \leq \beta < 1.$$

A l'aide de (2) on obtient :

$$\varphi(\rho^{(k)}) \leq \beta^2$$

Ce qui avec (1) nous donne :

$$\left\|x^{(k+1)} - x^*\right\| \leq \beta\left\|x^{(k)} - x^*\right\|, \quad \forall k \in \mathbb{N}.$$

Par récurrence, nous déduisons $\forall k \in \mathbb{N}^*$:

$$\left\|x^{(k)} - x^*\right\| \leq \beta\left\|x^{(k-1)} - x^*\right\| \leq \beta^2\left\|x^{(k-2)} - x^*\right\| \leq \cdots \leq \beta^k\left\|x^{(0)} - x^*\right\|$$

Ce qui achève la preuve du théorème.

3-1-1 Méthode du gradient à pas fixe.

La méthode du gradient à pas fixe est définie par :

$$\begin{cases} x^{(0)} donné\ dans\ \mathbb{R}^n \\ x^{(k+1)} = x^{(k)} - \rho\nabla J(x^{(k)}) \end{cases}$$

C'est une méthode de descente utilisant un pas fixe $\rho^{(k)} = \rho$ (indépendamment de k) et la stratégie de Cauchy (i.e. $d^{(k)} = -\nabla J\left(x^{(k)}\right)$, pour le choix de la direction de descente.
La convergence de la méthode est assurée sous les hypothèses du théorème précédent.
En particulier pour un choix de pas ρ vérifiant :

$$0 < \rho < \frac{2\alpha}{M^2}.$$

C'est la méthode de gradient la plus simple à utiliser.

Remarque :

- En général il est difficile de connaître α et M (en supposant qu'ils existent).

Alors dans la pratique on prend un pas ρ assez petit pour être sûr d'avoir la convergence. Mais dans ce cas la convergence peut être très lente !

- Dans le cas quadratique, α et M sont connues.

3-1-2 Méthode du gradient à pas optimal.

- **Cas d'une fonctionnelle J quelconque.**

La méthode du gradient à pas optimal consiste à choisir $\rho^{(k)}$ comme étant le minimum de la fonction $q(\rho) = J(x^{(k)} - \rho \nabla J(x^{(k)})$.
Dans ce cas, nous avons le même résultat de convergence que précédemment sous les hypothèses faibles sur J.

Nous avons donc le résultat suivant.

Théorème 3-1-2:

Soit $J: \mathbb{R}^n \to \mathbb{R}$ une fonction α-elliptique et M-lipchitzienne.
Alors la méthode de gradient à pas optimal donnée par :

$$\begin{cases} x^{(0)} \text{ donné dans } \mathbb{R}^n \\ x^{(k+1)} = x^{(k)} - \rho^{(k)} \nabla J(x^{(k)}) \end{cases}$$

Avec $\rho^{(k)}$ définie par :

$$J(x^{(k)} - \rho^{(k)} \nabla J(x^{(k)}) = \min_{\rho \in \mathbb{R}} J\left(x^{(k)} - \rho \nabla J(x^{(k)})\right).$$

Est bien définie et converge vers x^* la solution du problème d'optimisation.

Preuve :

Par hypothèse :
Comme la fonction J est α-elliptique et M-lipchitzienne, donc il existe un unique point de minimum x^* de cette fonction.
Ceci montre que la méthode de gradient à pas optimal est bien définie.
La convergence est donc admise.

- **Cas d'une fonctionnelle J quadratique.**

On suppose ici que la fonction J associée à une matrice A symétrique définie positive (SDP) est quadratique.
Nous devons calculer $\rho^{(k)} \in \mathbb{R}$ qui minimise la fonction $q: \mathbb{R} \to \mathbb{R}$ donnée par :
$q(\rho) = J(x^{(k)} - \rho \nabla J(x^{(k)})$.
Alors $\rho^{(k)}$ satisfait nécessairement $q'(\rho^{(k)}) = 0$.
Un calcul simple nous donne :

$$q'(\rho) = -< \nabla J\left(x^{(k)} - \rho \nabla J(x^{(k)})\right), \nabla J(x^{(k)}) >$$

C'est-à-dire, comme : $\nabla J(x^{(k)}) = Ax^{(k)} - b$.

$$\begin{aligned} q'(\rho) &= -< A\left(x^{(k)} - \rho \nabla J(x^{(k)})\right) - b, Ax^{(k)} - b > \\ &= -\left\|Ax^{(k)} - b\right\|^2 + \rho < A(Ax^{(k)} - b), Ax^{(k)} - b > \end{aligned}$$

On obtient alors,

$$\rho^{(k)} = \frac{\left\|Ax^{(k)} - b\right\|^2}{< A(Ax^{(k)} - b), Ax^{(k)} - b >}$$

Remarquons qu'on a,

$$< A\left(Ax^{(k)} - b\right), Ax^{(k)} >> 0 \quad \left(car A\ est\ SDP\ et\ Ax^{(k)} - b = \nabla J\left(x^{(k)}\right) \neq 0\right).$$

Donc la méthode du gradient à pas optimal dans le cas J quadratique est donnée par :

$$x^{(k+1)} = x^{(k)} - \rho^{(k)}\left(Ax^{(k)} - b\right)$$

Avec $\rho^{(k)}$ est donnée par la formule précédente (valable uniquement pour $Ax^{(k)} - b \neq 0$).

Comme $q'\left(\rho^{(k)}\right) = 0$, on déduit immédiatement que deux directions successives de descente sont orthogonales, c.-à-d. :

$$< \nabla J\left(x^{(k+1)}\right), \nabla J\left(x^{(k)}\right) > = 0.$$

Remarque :

Même pour le gradient à pas optimal qui est en principe la meilleure de ces méthodes d'un point de vue de rapidité de convergence, celle-ci peut être lente car elle est altérée par un mauvais conditionnement de la matrice Hessienne H de J.

Par ailleurs, on peut considérer des critères de convergence sur le gradient de J en $x^{(k)}$: $||\nabla J(x^{(k)})|| < \varepsilon_1$.

3-1-3 Méthode du gradient conjugué :

- **cas quadratique :**

Considérons la forme quadratique suivante :

$$J(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

Avec $A \in M_n(\mathbb{R})$ une matrice symétrique définie positive, $b \in \mathbb{R}^n$.

On considère dans ce paragraphe le problème de minimisation sans contraintes suivant :

$$\min_{x\in\mathbb{R}^n} J(x) = \min_{x\in\mathbb{R}^n} \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

Dans cette méthode, on démarre d'un point

$$x^{(0)} \in \mathbb{R}^n, d^{(0)} = -r^{(0)} = \nabla J\left(x^{(0)}\right) = Ax^{(0)} - b.$$

Les directions $d^{(1)}, \dots, d^{(n-1)}$ sont calculées à chaque itération.

A l'itération k on a :

$$d^{(k)} = -r^{(k)} + \beta^{(k-1)} d^{(k-1)}$$

Où $\beta^{(k-1)}$ est obtenu de sorte que $d^{(k)}$ soit A-conjugué avec les autres vecteurs $d^{(0)}, \dots, d^{(k-1)}$. En d'autres termes on doit avoir :

$$\langle Ad^{(k)}, d^{(i)}\rangle = 0\ , \quad i = 0, \dots, k-1.$$

Dans l'appellation gradient conjugué on trouve les deux mots : gradient et conjugué.

a) le mot gradient est utilisé car $d^{(k)}$ est calculée à partir du gradient au point $x^{(k)}$.

b) Le mot conjugué est aussi justifié, car et comme on le verra plus loin, les directions $d^{(0)}, \dots, d^{(n-1)}$ sont A-conjuguées.

Principe de la méthode :

On démarre d'un point quelconque : $x^{(0)} \in \mathbb{R}^n$.
Le principe de la méthode consiste à partir d'un point $x^{(0)}$ et à minimiser $J(x)$ successivement suivant n directions linéairement indépendantes $d^{(0)}, \dots, d^{(n-1)}$ possédant la propriété d'être mutuellement conjuguées par rapport à la forme quadratique $J(x)$.

Pour $k = 0$;
Calculer $d^{(0)} = -r^{(0)} = b - Ax^{(0)}$,

$$\rho^{(0)} = -\frac{\langle r^{(0)}, d^{(0)} \rangle}{\langle Ad^{(0)}, d^{(0)} \rangle}$$

Supposons qu'à l'itération k on ait : $x^{(k)}$ et $d^{(k)}$.
Ceci nous permettra de calculer :

$$r^{(k)} = Ax^{(k)} - b$$
$$\rho^{(k)} = -\frac{\langle r^{(k)}, d^{(k)} \rangle}{\langle Ad^{(k)}, d^{(k)} \rangle},$$
$$x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)},$$
$$r^{(k+1)} = Ax^{(k+1)} - b,$$
$$d^{(k+1)} = -r^{(k+1)} + \beta^{(k)} d^{(k)}.$$

$\beta^{(k)}$ est choisi de sorte que :

$$\langle Ad^{(k+1)}, d^{(k)} \rangle = 0,$$

Puisque $d^{(k+1)} = -r^{(k+1)} + \beta^{(k)} d^{(k)}$, ce qui nous donne :

$$\langle A\left(-r^{(k+1)} + \beta^{(k)} d^{(k)}\right), d^{(k)} \rangle = 0,$$

Ou encore :

$$\langle A\left(\beta^{(k)} d^{(k)}\right), d^{(k)} \rangle = \langle Ar^{(k+1)}, d^{(k)} \rangle,$$

Et finalement :

$$\beta^{(k)} = \frac{\langle Ar^{(k+1)}, d^{(k)} \rangle}{\langle Ad^{(k)}, d^{(k)} \rangle}.$$

En résumé on aura l'algorithme suivant :

1. Choisir $x^{(0)} \in \mathbb{R}^n$.
2. Calculer $r^{(0)} = Ax^{(0)} - b$. Si $r^{(0)} = 0$ stop. Sinon poser : $d^{(0)} = -r^{(0)}$.Poser $k = 0$.
   Tant que $\|x^{(k+1)} - x^{(k)}\| \geq \varepsilon$ et $k \leq kmax$. faire
3. Calculer :
$$\rho^{(k)} = -\frac{\langle r^{(k)}, d^{(k)} \rangle}{\langle Ad^{(k)}, d^{(k)} \rangle}.$$
4. Calculer :
$$x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}$$
5. Calculer :
   $r^{(k+1)} = Ax^{(k+1)} - b$. si $r^{(k+1)} = 0$ Stop,
6. Calculer :
$$\beta^{(k)} = \frac{\langle Ar^{(k+1)}, d^{(k)} \rangle}{\langle Ad^{(k)}, d^{(k)} \rangle}$$
7. Calculer :
$$d^{(k+1)} = -r^{(k+1)} + \beta^{(k)} d^{(k)}.$$
8. Poser $k = k + 1$ et retourner à 3.
   Fin tant que.
   Fin.

Figure 5.Algorithme du gradient conjugué d'une fonction quadratique.

Théorème 3-1-3: (convergence de la méthode du gradient conjugué quadratique). Partant d'un point initial $x^{(0)} \in \mathbb{R}^n$ quelconque, l'algorithme de gradient conjugué converge vers la solution optimale unique x^*du problème dans n itérations, c'est-à-dire qu'on a :
$x^{(n)} = x^*$et $Ax^{(n)} = Ax^* = b$.

- **Cas non-quadratique avec la variance de Fletcher - Reeves :**

La méthode de Fletcher et Reeves (1964) est une extension directe de la méthode précédente au cas des fonctions quelconques.
Cette méthode est très intéressante, d'une part parce qu'elle nécessite le stockage de très peu d'information (essentiellement trois vecteurs de dimension n) ; d'autre part, par sa vitesse de convergence très supérieure à celle des algorithmes de gradient classique.

Méthode de Fletcher-Reeves

Partir de $k = 0$.

1. Choisir $x^{(0)} \in \mathbb{R}^n$ quelconque.
   Choisir $\varepsilon > 0$.
   Choisir $\varepsilon_1 > 0$.
2. Pour $k = 0$, calculer $r^{(0)} = \nabla J(x^{(0)})$.
   Poser $d^{(0)} = -r^{(0)}$.
3. Tant que ($\|x^{(k+1)} - x^{(k)}\| \geq \varepsilon$ et $(k \leq kmax)$ faire ;
   Si ($\|r^{(k)}\| < \varepsilon_1$) alors arrêt : $x^* = x^{(k)}$. Sinon
4. Calculer :

$$\beta^{(k)} = \frac{\|\nabla J(x^{(k)})\|^2}{\|\nabla J(x^{(k-1)})\|^2};$$

5. Calculer $d^{(k+1)} = -r^{(k+1)} + \beta^{(k)} d^{(k)}$;
6. Calculer $\rho^{(k)}$approchant le minimum $\rho \to J(x^{(k)} + \rho d^{(k)})$ en utilisant une recherche linéaire du pas de descente.
7. Calculer $x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}$.
8. Poser $k = k + 1$;
   Fin tant que ;
   Retourner à l'étape 4.
   Fin.

Figure 6.Algorithme du gradient conjugué pour une fonction quelconque.

Remarque :
On voit que l'on garde en mémoire à chaque étape, que les gradients en $x^{(k)}$ et $x^{(k+1)}$ et la direction de déplacement courante $d^{(k)}$.
Il est important de remarquer que la convergence globale de la méthode Fletcher-Reeves n'est assurée que si l'on procède à une réinitialisation périodique.
Par exemple, toutes les n itérations, on repartira du dernier point obtenu avec, comme direction de déplacement, le gradient en ce point.

La convergence globale de cette procédure découle alors de la convergence des méthodes de gradients à pas optimal.

3-2 Méthode de Newton.

La méthode de Newton n'est pas a proprement parlé une méthode d'optimisation.
C'est une méthode de recherche de zéros d'une fonction $F: \mathbb{R}^n \to \mathbb{R}^n$ selon : $F(x) = 0$.
L'idée de cette méthode consiste dans notre cas de résoudre le système d'équations : $\nabla J(x) = 0$ avec $\nabla J: \mathbb{R}^n \to \mathbb{R}^n$.
Condition nécessaire de premier ordre pour la détection d'extrema d'une fonction.
$\nabla J(x) = 0$ est donnée par un système $n \times n$ d'équations non linéaires.
On suppose que J est deux fois continument différentiable (i.e. $J \in C^2$), que l'on sait calculer toutes ses dérivées secondes, et que la matrice $D^2J(x^{(k)})$ est inversible.
La méthode s'écrit formellement :

$$x^{(k+1)} = x^{(k)} - [D^2J(x^{(k)})]^{-1}\nabla J(x^{(k)}).$$

Où ; $d^{(k)} = -\left(D^2J(x^{(k)})\right)^{-1}\nabla J(x^{(k)})$ est appelée **direction de Newton.**
Dans le cas où la matrice $D^2J(x^{(k)})$ est définie positive à chaque itération, la méthode de Newton est une méthode de descente à pas fixe égal à 1.

Cherchons à résoudre, pour $f: \mathbb{R} \to \mathbb{R}$, une équation $f(x) = 0$.
Nous supposons que f est de classe C^1.
L'algorithme de Newton est donné par la table ci-dessous.

poser $k = 0$
choisir $x^{(0)}$ dans un voisinage de x^*
choisir $\varepsilon > 0$
tant que $\left(|x^{(k+1)} - x^{(k)}| \geq \varepsilon\right)$ et $\left(k \leq k^{\max}\right)$ faire
    poser $x^{(k+1)} = x^{(k)} - \dfrac{f(x^{(k)})}{f'(x^{(k)})}$
    poser $k = k + 1$
fin tant que

Figure 7. Algorithme de Newton Unidimensionnelle.

Une interprétation géométrique simple de cette méthode est donnée à partir de la tangente au point $x^{(k)}$.
La convergence de la méthode doit être précisée ainsi que la définition de la suite $(f'((x^{(k)}))_k$.
Généralisons maintenant cette méthode au cas d'une fonction J donné de $\mathbb{R}^n$dans $\mathbb{R}$.
Soit $F: \mathbb{R}^n \to \mathbb{R}^n$ une fonction de classe C^1.

On suppose que l'équation $F(x) = 0$ possède au moins une solution notée x^* et que la matrice $DF(x^{(k)})$ est une matrice inversible.
La continuité de DF permet en fait d'assurer l'inversibilité de $DF(x^{(k)})$ pour tout point $x^{(k)}$ se trouvant dans un voisinage de x^* et permet de définir l'itéré $x^{(k+1)}$.
L'extension de la seconde étape est réalisée par la résolution du système linéaire :

$$[DF(x^{(k)})]\delta^{(k)} = F(x^{(k)}).$$

Puis on pose $x^{(k+1)} = x^{(k)} - \delta^{(k)}$.

Remarque :

Lorsque l'on utilise la méthode de Newton pour résoudre: $\nabla J(x^*) = 0$, on prend bien sûr $F = \nabla J$.
La méthode donnera alors les points critiques de J, la propriété de minimum étant à vérifier a posteriori.
Dans ce cas, DF est la matrice Hessienne D^2J .

En résumé, l'algorithme de Newton peut s'écrire sous la forme présentée dans la table ci-dessous.

poser $k = 0$
choisir $\mathrm{x}^{(0)}$ dans un voisinage de x^*
choisir $\varepsilon > 0$
tant que $\left(\left\|\mathrm{x}^{(k+1)} - \mathrm{x}^{(k)}\right\| \geq \varepsilon\right)$ et $\left(k \leq k^{\max}\right)$ faire
    resoudre le systeme lineaire $\left[D\mathrm{F}(\mathrm{x}^{(k)})\right]\delta^{(k)} = \mathrm{F}(\mathrm{x}^{(k)})$
    poser $\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} - \delta^{(k)}$
    poser $k = k + 1$
fin tant que

Figure 8. Algorithme de Newton multidimensionnelle.

En ce qui concerne la convergence de ce dernier algorithme, nous avons le résultat suivant ;

Théorème 3-2:(convergence de la méthode de Newton)

Soit $F: \mathbb{R}^n \to \mathbb{R}^n$ une fonction de classe C^1 et x^* un zéro de F.
On suppose que ce zéro est isolé et que $DF(x^*)$ est inversible (DF désigne la dérivée première de F).
Alors, il existe une boule $B(x^*;r)$ telle que, pour tout point $x^{(0)} \in B(x^*;r)$, la suite $(x^{(k)})_k$ générée par la méthode de Newton est entièrement contenue dans B et converge vers x^*, seul zéro de Fdans B.
De plus, la convergence est géométrique : il existe $\beta \in]0;1[$ tel que :

$$\forall k \geq 0\ , \left\|x^{(k)} - x^*\right\| \leq \beta^k \left\|x^{(0)} - x^*\right\|.$$

Par conséquent, si nous choisissons $x^{(0)}$ « suffisamment près » de x^*, la méthode de Newton converge.

Remarque :

Un des gros problèmes de cette méthode est que le choix de $x^{(0)}$ joue un grand rôle sur la convergence ou non de la méthode de Newton.
La méthode est très sensible à l'initialisation.
Il peut aussi arriver que la méthode converge vers un extremum qui n'est pas celui recherché.
Une approche possible consiste à faire tourner quelques itérations d'une méthode de gradient pour approcher x^* et de considérer l'itéré résultant comme le point de départ de la méthode de Newton.
L'avantage de la méthode de Newton est sa grande rapidité de convergence : la convergence de la méthode de Newton est quadratique.
Un des inconvénients de la méthode de Newton réside également dans le fait que nous avons à évaluer et à ''inverser '' (en fait résoudre un système plein associé) la matrice $DF(x^{(k)})$ à chaque itération.
Certaines méthodes proposent d'utiliser non pas $DF(x^{(k)})$ comme matrice mais plutôt une approximation de celle-ci (plus précisément, dans les méthodes de quasi-Newton décrites ci-dessous, on construit une suite de matrices $S^{(k)}$ qui approchent $[DF(x^{(k)})]^{-1}$).

3-2-1 Méthodes de quasi-Newton (ou méthodes à métrique variable).

Principe général :

L'idée ici est d'imiter l'algorithme de Newton tout en évitant de calculer D^2J et «son inverse ».
Plus précisément, cela signifie que, à l'itération k, nous allons chercher à construire une approximation $S^{(k)}$ symétrique définie positive de $[D^2J(x^{(k)})]^{-1}$ et $\rho^{(k)}$ un paramètre positif par un algorithme de minimisation unidimensionnel le long de la direction $d^{(k)} = -S^{(k)}\nabla J(x^{(k)})$ tel que l'itération classique :

$$x^{(k+1)} = x^{(k)} - [D^2J(x^{(k)})]^{-1}\nabla J(x^{(k)}),$$

Soit remplacée par une itération plus simple et beaucoup moins couteuse

$$x^{(k+1)} = x^{(k)} - \rho^{(k)}S^{(k)}\nabla J(x^{(k)}).$$

1) Méthode de Davidson-Fletcher-Powell (DFP) :

Le but est de calculer «une bonne approximation » $S^{(k)}$de $[D^2J(x^{(k)})]^{-1}$, c'est-à-dire telle que la différence soit petite pour une norme matricielle.
Cette méthode permet notamment, pour une fonctionnelle quadratique, de construire l'inverse du Hessien.
Elle engendre également des directions conjuguées.
Cette méthode utilise la formule de correction (de rang 2) suivante :

$$S^{(k+1)} = S^{(k)} + \frac{\delta^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}} - \frac{S^{(k)}\gamma^{(k)}\gamma^{(k)^T}S^{(k)}}{\gamma^{(k)^T}S^{(k)}\gamma^{(k)}} \quad (*)$$

Où le point $x^{(k+1)}$ est obtenu à partir de $x^{(k)}$ par déplacement dans la direction : $d^{(k)} = -S^{(k)}\nabla J\left(x^{(k)}\right)$, Et où :
$\delta^{(k)} = x^{(k+1)} - x^{(k)}$, $\gamma^{(k)} = \nabla J\left(x^{(k+1)}\right) - \nabla J\left(x^{(k)}\right)$.
Le résultat suivant montre que, sous certaines conditions, la formule (*) conserve la définie-positivité des matrices $S^{(k)}$.

Théorème 3-2-1 :

On suppose que la matrice $S^{(k)}$ est définie positive.
Alors si la condition $\delta^{(k)^T}\gamma^{(k)} > 0$ est vérifiée, la matrice $S^{(k+1)}$ donnée par (*) 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 $d^{(k)} = -S^{(k)}\nabla J\left(x^{(k)}\right)$.
Cette propriété de conservation de la définie positivité est essentielle, car elle garantit, en particulier, que les directions $d^{(k)}$, successivement engendrées par l'algorithme, sont des directions de descente.

Théorème 3-2-2 :(cas d'une fonction quadratique)

Appliqué à une fonction J quadratique (de Hessien D^2J défini positif), alors la méthode DFP engendre des directions $\delta^{(0)}, \ldots . \delta^{(k)}$, vérifiant, pour tout k :

$$\delta^{(i)^T} D^2 J \delta^{(j)} = 0, \quad 0 \le i \le j \le k$$
$$\delta^{(k+1)} D^2 J \delta^{(i)} = \delta^{(i)}, \quad 0 \le i \le k.$$

Ceci implique que la méthode converge en n itérations, Et $S^{(k)} = [\mathrm{D}^2 J]^{-1}$.

Remarque :

La propriété $\delta^{(k)^T}\gamma^{(k)} > 0$ est vérifiée également par des méthodes de recherche linéaire ne nécessitant pas l'optimum exact dans la direction.
Par exemple si l'on utilise une recherche linéaire« économique » avec la règle de Wolfe, on obtient un point $x^{(k+1)}$ tel que :

$$d^{(k)^T}\nabla J\left(x^{(k+1)}\right) \ge m3 d^{(k)^T}\nabla J\left(x^{(k)}\right) \quad (avec\ m3 < 1)$$

D'où : $d^{(k)^T}\nabla J\left(x^{(k+1)}\right) < -d^{(k)^T}\nabla J\left(x^{(k)}\right)$.
Et par la suite, $\delta^{(k)^T}\gamma^{(k)} > 0$.
Si $S^{(0)} = Id$ (la matrice identité), on a la méthode du gradient conjugué.

La méthode utilisant la correction (*) est donnée par l'algorithme ci-dessous.

choisir $S^{(0)}$, matrice symetrique definie positive
poser $k = 0$
choisir $\mathbf{x}^{(0)} \in \mathbb{R}^n$
choisir $\varepsilon > 0$
tant que $\left(\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \geq \varepsilon\right)$ et $\left(k \leq k^{\max}\right)$ faire
calculer $\rho^{(k)}$ par une methode de recherche lineaire sur $\mathbf{d}^{(k)} = -S^{(k)}\nabla J(\mathbf{x}^{(k)})$
poser $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \rho^{(k)}\mathbf{d}^{(k)}$
poser $\delta^{(k+1)} = \rho^{(k)}\mathbf{d}^{(k)}$
calculer $\gamma^{(k)} = \nabla J(\mathbf{x}^{(k+1)}) - \nabla J(\mathbf{x}^{(k)})$
calculer $S^{(k+1)} = S^{(k)} + \dfrac{\delta^{(k)}\delta^{(k)T}}{\delta^{(k)T}\gamma^{(k)}} - \dfrac{\delta^{(k)}\gamma^{(k)}\gamma^{(k)T}\delta^{(k)}}{\gamma^{(k)T}\delta^{(k)}\gamma^{(k)}}$
fin tant que

Figure 9. Algorithme de Davidson-Fletcher-Powell (DFP)

2) Méthode de Broyden-Fletcher-Goldfarb-Shanno (BFGS) :

Cette dernière méthode est considéré comme très robuste et moins sensible aux erreurs dans les recherches linéaires.
Elle a été développé indépendamment 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 (*) rappelée par la formule suivante:

$$S^{(k+1)} = S^{(k)} + \frac{\delta^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}} - \frac{S^{(k)}\gamma^{(k)}\gamma^{(k)^T}S^{(k)}}{\gamma^{(k)^T}S^{(k)}\gamma^{(k)}}$$

On sait que la matrice $S^{(k+1)}$ obtenue par (*) vérifie la relation :
$S^{(k+1)}\gamma^{(k)} = \delta^{(k)}$ (**)
Comme l'a remarqué Fletcher (1970), si l'on intervertit les rôles de $\delta^{(k)}$ et de $\gamma^{(k)}$ dans (*) et que l'on considère la suite des matrices définies par :

$$\begin{cases} G^{(0)} \text{ symétrique définie positive quelconque.} \\ G^{(k+1)} = G^{(k)} + \dfrac{\delta^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}} - \dfrac{G^{(k)}\delta^{(k)}\delta^{(k)^T}G^{(k)}}{\gamma^{(k)^T}S^{(k)}\gamma^{(k)}} \end{cases} \quad (***)$$

Les matrices obtenues vérifient la relation « inverse » de (**)
$G^{(k+1)}\delta^{(k)} = \gamma^{(k)}$
On voit alors que la formule (***) permet de construire une approximation du Hessien lui-même (et non pas de son inverse).
Si l'on veut, à partir de (***), obtenir une formule de correction pour approximer l'inverse du Hessien, il suffit donc de prendre l'inverse des deux membres de (***).
Le calcul donne :

$$[G^{(k+1)}]^{-1} = [G^{(k)}]^{-1} + \left[1 + \frac{\gamma^{(k)^T}[G^{(k)}]^{-1}\gamma^{(k)}}{\delta^{(k)^T}\gamma^{(k)}}\right]\frac{\delta^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}} - \frac{\delta^{(k)}\gamma^{(k)^T}[G^{(k)}]^{-1} + [G^{(k)}]^{-1}\gamma^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}}$$

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

$$S^{(k+1)} = S^{(k)} + \left[1 + \frac{\gamma^{(k)^T}S^{(k)}\gamma^{(k)}}{\delta^{(k)^T}\gamma^{(k)}}\right]\frac{\delta^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}} - \frac{\delta^{(k)}\gamma^{(k)^T}S^{(k)} + S^{(k)}\gamma^{(k)}\delta^{(k)^T}}{\delta^{(k)^T}\gamma^{(k)}}$$

L'algorithme de la méthode est représenté en résumé dans la table ci-dessous.

choisir $S^{(0)}$, matrice symetrique definie positive
poser $k = 0$
choisir $\mathbf{x}^{(0)} \in \mathbb{R}^n$
choisir $\varepsilon > 0$
tant que $\left(\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \geq \varepsilon\right)$ et $\left(k \leq k^{\max}\right)$ faire
    calculer $\rho^{(k)}$ par une methode de recherche lineaire sur $\mathbf{d}^{(k)} = -S^{(k)}\nabla J(\mathbf{x}^{(k)})$
    poser $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \rho^{(k)}\mathbf{d}^{(k)}$
    poser $\delta^{(k+1)} = \rho^{(k)}\mathbf{d}^{(k)}$
    calculer $\gamma^{(k)} = \nabla J(\mathbf{x}^{(k+1)}) - \nabla J(\mathbf{x}^{(k)})$
    calculer $S^{(k+1)} = S^{(k)} + (1 + \frac{\gamma^{(k)T}\delta^{(k)}\gamma^{(k)}}{\delta^{(k)T}\gamma^{(k)}})\frac{\delta^{(k)}\delta^{(k)T}}{\delta^{(k)T}\gamma^{(k)}} - \frac{\delta^{(k)}\gamma^{(k)T}\delta^{(k)} + \delta^{(k)}\gamma^{(k)}\delta^{(k)T}}{\delta^{(k)T}\gamma^{(k)}}$
fin tant que

Figure 10. Algorithme de Broyden-Fletcher-Goldfarb-Shanno.

3-2-2 Méthode de Gauss-Newton.

Terminons ce chapitre par le cas particulier des problèmes de moindres carrés linéaires et non linéaires.
Un problème de moindres carrés est un problème d'optimisation de la forme :

$$\min_{x \in \mathbb{R}^n} r(x) \text{ Avec } r(x) = \frac{1}{2}\|F(x)\|_2^2 \quad 1)$$

Où F désigne une application de $\mathbb{R}^n$ dans $\mathbb{R}^m$, en supposant $m \geq n$.
Remarquons que si le système $F(x) = 0$ a des solutions, alors ce sont également les solutions du problème de minimisation.
De tels problèmes se rencontrent fréquemment en pratique dans le cadre de l'identification de paramètres.
Les variables d'optimisation x_i sont les paramètres du modèle physique non linéaire proposé.

On effectue m mesures et on cherche les x_i qui permettent d'ajuster au mieux le modèle aux mesures.
Nous allons voir dans ce paragraphe que la structure particulière de ces problèmes les rend plus simples à résoudre qu'un problème d'optimisation sans contraintes général.
Remarquons d'abord que l'on a des expressions explicites des dérivées de r en fonction de F, de sa Jacobienne J_F et des Hessiens $H_{F_i}(x)$:

$$\nabla r(x) = J_F(x)^T F(x) = \sum_{i=1}^{m} F_i(x) \nabla F_i(x)$$

$$H_r(x) = J_F(x)^T J_F(x) + \sum_{i=1}^{m} F_i(x) H_{F_i}(x)$$

Une caractéristique des algorithmes de moindres carrés est qu'à condition de connaître la Jacobienne de F en tout point, on peut calculer une partie du Hessien H_r sans calcul supplémentaire.
De plus ce terme $J_F(x)^T J_F(x)$ est souvent prépondérant (ou dominant) par rapport au second terme $\sum_{i=1}^{m} F_i(x) H_{F_i}(x)$ pour deux raisons : tout d'abord à cause de la presque linéarité du modèle au voisinage de la solution (i.e. $H_{F_i}(x)$ petit), ou bien à cause du résidu F_j petit (ou négligeable) au voisinage de la solution.
La plupart des algorithmes de moindres carrés non linéaires existants exploitent cette structure particulière du Hessien de r.

1) Problèmes de moindres carrés linéaires :

On se place dans le cas où la fonction F est linéaire i.e. :

$$F(x) = Ax - b, \text{ Avec } A \in M_{n,p}(\mathbb{R}),$$

Et on résout le problème de moindres carrés linéaire suivant :

$$\min_{x \in \mathbb{R}^n} r(x) \text{ Avec } r(x) = \frac{1}{2}\|Ax - b\|_2^2 \quad 2)$$

On calcule sans difficulté la Jacobienne de F en tout point :
On pose $J_F = A$,
Puis :

$$\nabla r(x) = A^T(Ax - b) \text{ Et}$$
$$H_r(x) = J_F(x)^T J_F(x) = A^T A.$$

La Hessienne de r est donc positive en tout point, le problème de moindres carrés linéaire est donc convexe.
Cherchons les points critiques du problème 2) i.e. vérifiant : $\nabla r(x) = 0$, d'où le système suivant à résoudre :
$A^T A x = A^T b$
Appelé système d'équations normales du problème de moindres carrés linéaire 2).
Ainsi, x^* est solution du problème 2) si et seulement si x^* vérifie les équations normales.
De plus si A est de rang plein, alors la fonction r est strictement convexe et x^* est l'unique solution de 2).
Appliquons maintenant la méthode de Newton au problème 2).
La direction de Newton donnée par :

$$d^{(k)} = x^{(k+1)} - x^{(k)}$$

Est solution du système :

$$A^T A d^{(k)} = -A^T\left(Ax^{(k)} - b\right),$$

D'où :

$$A^T A x^{(k+1)} = A^T b,$$

Et ceci quel que soit $x^{(k)}$.
On reconnaît ici le système d'équations normales du problème 2).
Ceci signifie donc que la méthode de Gauss-Newton identifie la solution en une seule itération lorsque la fonction F est linéaire.

2) Méthode et algorithme de Gauss-Newton :

Revenons au problème de moindres carrés non linéaire général 1)

$$\min_{x\in\mathbb{R}^n} \{r(x) := \frac{1}{2}\|F(x)\|_2^2\},$$

Il existe deux façons de présenter l'algorithme de Gauss-Newton : l'une consiste à appliquer la méthode de Newton au problème 1) en négligeant le terme de second ordre $\sum_{i=1}^{m} F_i(x) H_{F_i}(x)$ dans la Hessienne de r, pour les raisons évoquées précédemment.
Nous nous intéressons ici à une autre : l'idée est de remplacer à chaque itération le problème de moindres carrés non linéaire par un problème approché de moindres carrés linéaire.
Soit $x^{(k)} \in \mathbb{R}^n$ le point courant.
Remplaçons au voisinage de $x^{(k)}$ le problème 1) par :

$$\min_{y\in\mathbb{R}^n} \{\check{r}(y) = \frac{1}{2}\left\|F(x^{(k)}) + J_F(x^{(k)})(y - x^{(k)})\right\|_2^2,\} \quad 3)$$

Où l'on a remplacé F par une approximation du premier ordre en $x^{(k)}$.
Le problème 3) est un problème de moindres carrés linéaire dont les solutions $x^{(k+1)}$ vérifient les équations normales associées, à savoir :

$$J_F(x^{(k)})^T J_F(x^{(k)})(x^{(k+1)} - x^{(k)}) = -J_F(x^{(k)})^T F(x^{(k)}).$$

La direction $d^{(k)} = x^{(k+1)} - x^{(k)}$ ainsi définie, est appelée la direction de Gauss-Newton.

En résumé, l'algorithme de la méthode est décrit dans la table ci-dessous :

Partir de $k = 0$;

Choisir $x^{(0)} \in \mathbb{R}^n$;

Choisir $\varepsilon > 0$;

Poser $d = -grad(x)$;

Tant que $(\|x^{(k+1)} - x^{(k)}\| \geq \varepsilon)$ et $(k \leq kmax)$;

Calcul d'une direction de descente $d^{(k)}$ solution de :

$$J_F(x^{(k)})^T J_F(x^{(k)}) d^{(k)} = -J_F(x^{(k)})^T F(x^{(k)});$$

$$d^{(k)} = -(J_F(x^{(k)})^T J_F(x^{(k)}))^{-1} J_F(x^{(k)})^T F(x^{(k)});$$

Calculer $\rho^{(k)}$ en utilisant une recherche linéaire du pas de descente.

Poser : $x^{(k+1)} = x^{(k)} + \rho^{(k)} d^{(k)}$;

Poser $k = k + 1$;

Fin tant que ;

Fin

Figure 11. Algorithme de Gauss-Newton.

Les avantages de cette méthode par rapport à la méthode de Newton sont les suivants :

a. L'approximation faite ne nécessite pas le calcul des dérivées secondes.
b. Dans de nombreuses applications, cette approximation est de bonne qualité.
c. Si $J_F(x^{(k)})$ est de rang plein et si $\nabla r(x^{(k)})$ est non nul, alors la direction de Gauss-Newton est une direction de descente de r en $x^{(k)}$ et donc adaptée pour l'utilisation d'un algorithme de recherche linéaire.

La vitesse de convergence de l'algorithme de Gauss-Newton dépend de la qualité de l'approximation faite du problème initial.
Dans le cas où le résidu est effectivement petit au voisinage de la solution, la convergence est rapide.
Si la Hessienne de F est nulle à la solution, alors la convergence est quadratique.
Sous des hypothèses supplémentaires, on peut démontrer la convergence globale des itérés vers un point stationnaire de r.
Un désavantage de la méthode de Gauss-Newton est que l'on n'a pas de résultat de convergence lorsque la Jacobienne de F en $x^{(k)}$ n'est pas de rang plein.

Partie 2 : Etude Numérique

Chapitre 4

IV. Programmation Matlab

4-1 Application des méthodes du gradient sur des fonctionnelles quelconques.

Fonction de Rosenbrock :
On souhaite dans cette partie de résoudre le problème de minimisation d'une fonctionnelle quelconque.
On définit sur $\mathbb{R}^2$ la fonctionnelle classique de Rosenbrock deux fois continument différentiable, donnée par la formule suivante :

$$f(x,y) = (x-1)^2 + 10(x^2-y)^2$$

1. **Etude théorique :**

a) La fonction de Rosenbrock admet un seul point critique qui est atteint en : $X^* = (1;1)$.

b) On a $f(x,y) = (x-1)^2 + 10 * (x^2-y)^2$;
Donc $f(X^*) = f(1,1) = 0$.
Comme $f(x,y) \geq 0$;
On aura : $f(1,1) \leq f(x,y)$.
De plus f est strictement convexe.

Ce qui nous permet de conclure que f admet un unique minimum global, qu'elle atteint en $X^* = (1;1)$.

c) Le gradient de f au point $X^* = (1,1)$ est donné par :
$\nabla f(1,1) = [0;0]$.

d) On appelle H la matrice Hessienne de f en $X^* = (1,1)$.
La matrice Hessienne en ce point est donc donnée par :
$H[f](1,1) = [82-40; 20-40]$.

e) Le développement limité à l'ordre 2 de la fonction f au point $X^* = (1,1)$ est donné par :

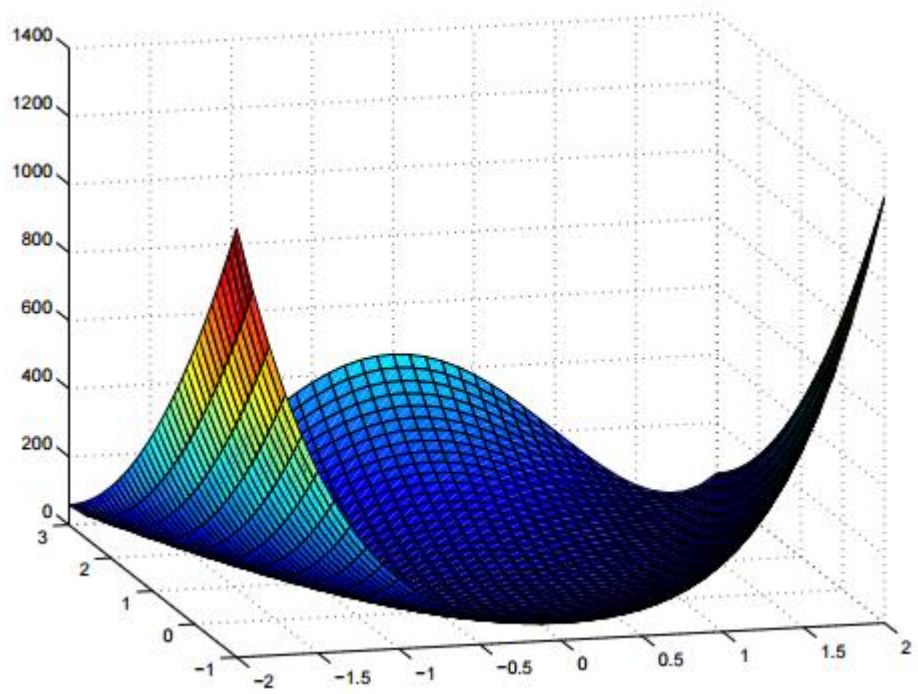
$$f(x,y) = f(1,1) + \frac{\partial f}{\partial x}(1,1)(x-1) + \frac{\partial f}{\partial y}(1,1)(y-1) + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(1,1)(x-1)^2 + \frac{1}{2}\frac{\partial^2 f}{\partial y^2}(1,1)(y-1)^2 + \frac{\partial^2 f}{\partial x \partial y}(1,1)(x-1)(y-1) + \|(x,y)\|^2 \varepsilon(x;y).$$

Ce qui nous donne :
$f(x,y) = 41(x-1)^2 + 10(y-1)^2 - 40(x-1)(y-1) + \|(x,y)\|^2 \varepsilon(x;y)$.
Avec : $\lim_{(x,y)\to 0} \|(x,y)\|^2 \varepsilon(x;y) = 0$.

2. Etude numérique :

a) La représentation de la fonction de Rosenbrock est donnée par le graphe ci-dessous :

```
function z=Rosenbrock(x)
z=(x(1)-1)^2+10*((x(1))^2-x(2))^2;
end
```



b) On souhaite comparer quelques méthodes bien connues de minimisation.

La méthode du gradient à pas fixe, à pas prédéterminé, à pas optimal, puis le gradient conjugué.
Pour comparer ces méthodes, on choisit de démarrer les algorithmes que l'on codera au point de coordonnés x0= (0;1).

➢ Méthode du gradient à pas fixe :

- Pour tester cette méthode, on pourra choisir par exemple un pas initial $\rho = 0{,}01$.

```
function [ x ] = PasFixe( x,e )
tic
d=-grad(x);
k=0;
s=x;
t=0.01;
```

```
while  norm(d,inf)>e & k<1000
  d=-grad(x);
   xopt=x+t*d/norm(d,inf);
   x=xopt;
   s=[s x];
   k=k+1;
end
toc
a=s(1,:);
b=s(2,:);   plot(s(1,:),s(2,:),'-ro')
 end
```

Avec le sous-programme de calcul du gradient de la fonction de Rosenbrock :

```
function [z]=grad(x)
z=[40*(x(1))^(3)+2*x(1)-40*x(1)*x(2)-2;20*x(2)-20*(x(1))^2];
end
```

L'algorithme converge vers un point $= [0\ ;1]$, qui est trop loin du point optimal théorique en 0.000332 seconds $(x^* \neq x)$, donc la convergence n'est pas assurée dans ce cas.

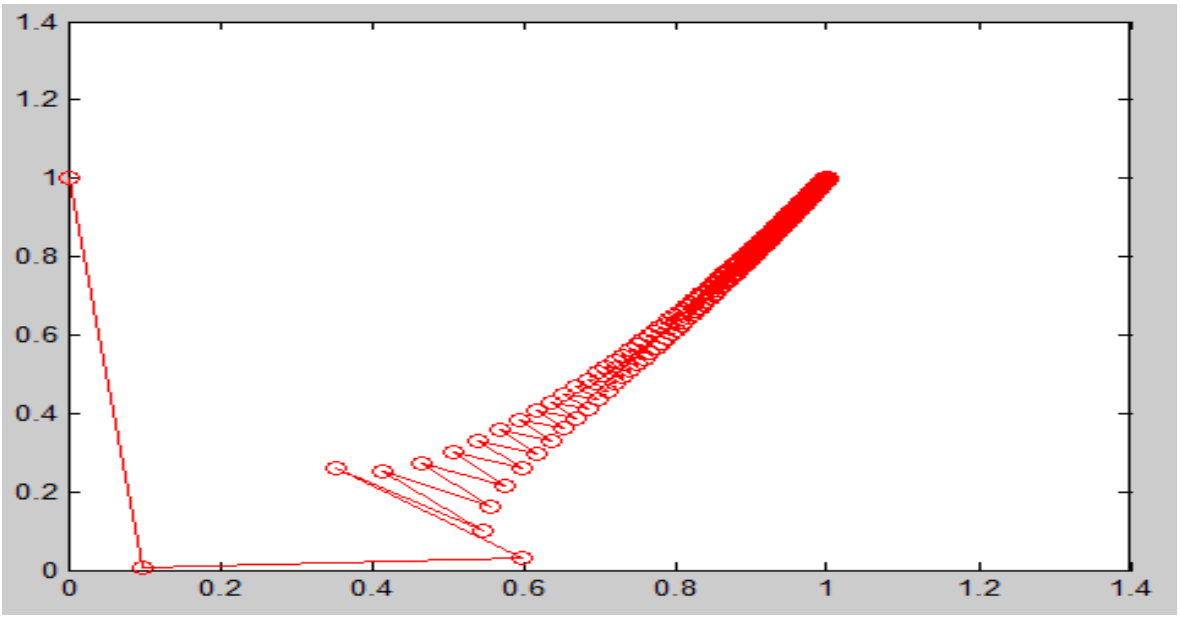
- **Méthode du gradient à pas prédéterminé :**

On utilise le même principe que celui du gradient à pas fixe, sauf que dans ce cas on initialise le pas $\rho = 0.01$, et à chaque itération on calcule de nouveau le pas de descente selon la formule suivante : $\rho = \frac{1}{k}$;

Le programme du gradient à pas prédéterminé est le suivant :

```
function [ x ] = Gradient_pre( x,e )
tic
d=-grad(x);
k=0;
s=x;
while norm(d,inf)>e & k<=1000
   k=k+1;
   t=1/k;
   xopt=x+t*d/norm(d,inf);
   x=xopt;
   d=-grad(x);
   s=[s x];
end
toc
a=s(1,:);
b=s(2,:);
 plot(s(1,:),s(2,:),'-ro')
end
```

la courbe obtenue est la suivante :



On remarque sur la figure obtenue le zig-zag des itères de l'algorithme dû à la variation du pas de descente à chaque itération.
L'algorithme converge vers le point optimal $x = [1.004 ; 0.996]$ qui est très proche du point optimal détecté théoriquement $x^* = [1 ; 1]$,en 0.0089414 seconds.

➢ Méthode du gradient à pas optimal :

- Pour tester cette méthode nous allons choisir la méthode de Wolfe pour déterminer le pas de descente ρ.

Pour cela on se donne le sous-programme de Wolfe suivant :

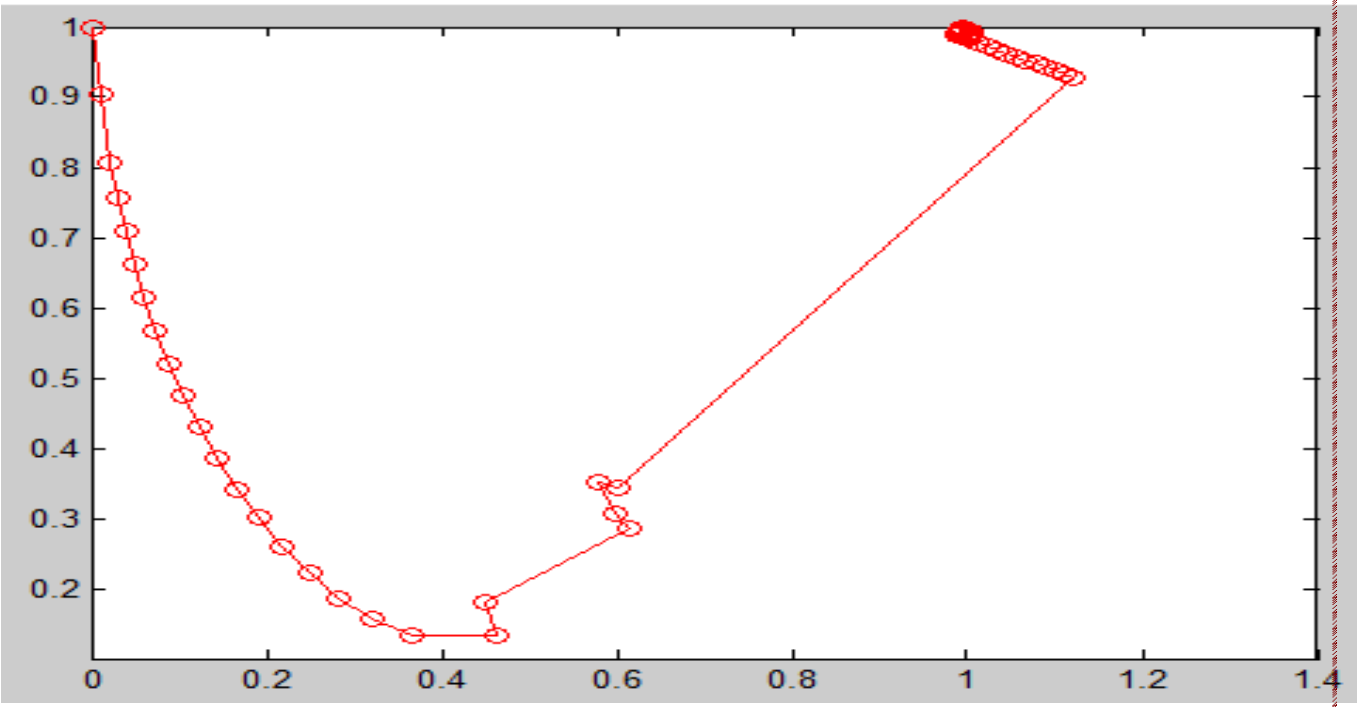
```
function [ alfa ] = Wolfe( x,d )
ti=0;
ts=0;
t=1;
m1= 10^(-4);
m2= 0.9999;
k=0;
while (Rosenbrock(x+t*d) >=Rosenbrock(x)+m1*t*d'*grad(x)) & (k<1000)
   ts=t;
   t=0.5*(ts+ti);
   k=k+1;
end
i=0;
while (d'*grad(x+t*d) <=m2*d'*grad(x))&(k<1000)
   ti=t;
   if (ts==0)
      t=ti+0.001; else
t=0.5*(ts+ti);
 end
   k=k+1 ; end
alfa=t;
```

```
if (alfa<0.0000001)
   alfa=0.0001;
 end
```

- Le programme du gradient à pas optimal est le suivant :

```
function [ x ]=PasOptimal(x,epsilon)
tic
k=0;
e=1;
d=-grad(x);
m1=10^(-4);
m2=0.9999;
s=x;
while k<=1000
   t=Wolfe(x,d);
   xopt=x+t*d/norm(d);
     x=xopt;
      d=-grad(x);
   if norm(d,inf)>epsilon
       e=k;
       break
   end
   s=[s x];
   k=k+1;
end
if e==k
    disp('nombre d''itérations')
    disp(k)
 else
    disp('convergence non atteinte');
 end
toc
c=s(1,:);
v=s(2,:);
 plot(s(1,:),s(2,:),'-ro')
end
```

La courbe obtenue est la suivante :



L'algorithme converge vers le point optimal $x = [1.0056 ; 0.9938]$ qui est très proche du point optimal détecté théoriquement $x^* = [1 ; 1]$, en k=123 itérations en 0.243980 seconds, on remarque aussi le zig-zag des itérés lorsqu'on s'approche de la solution optimale.

➢ Méthode du gradient conjugué :

Le programme de cette méthode est donné par :

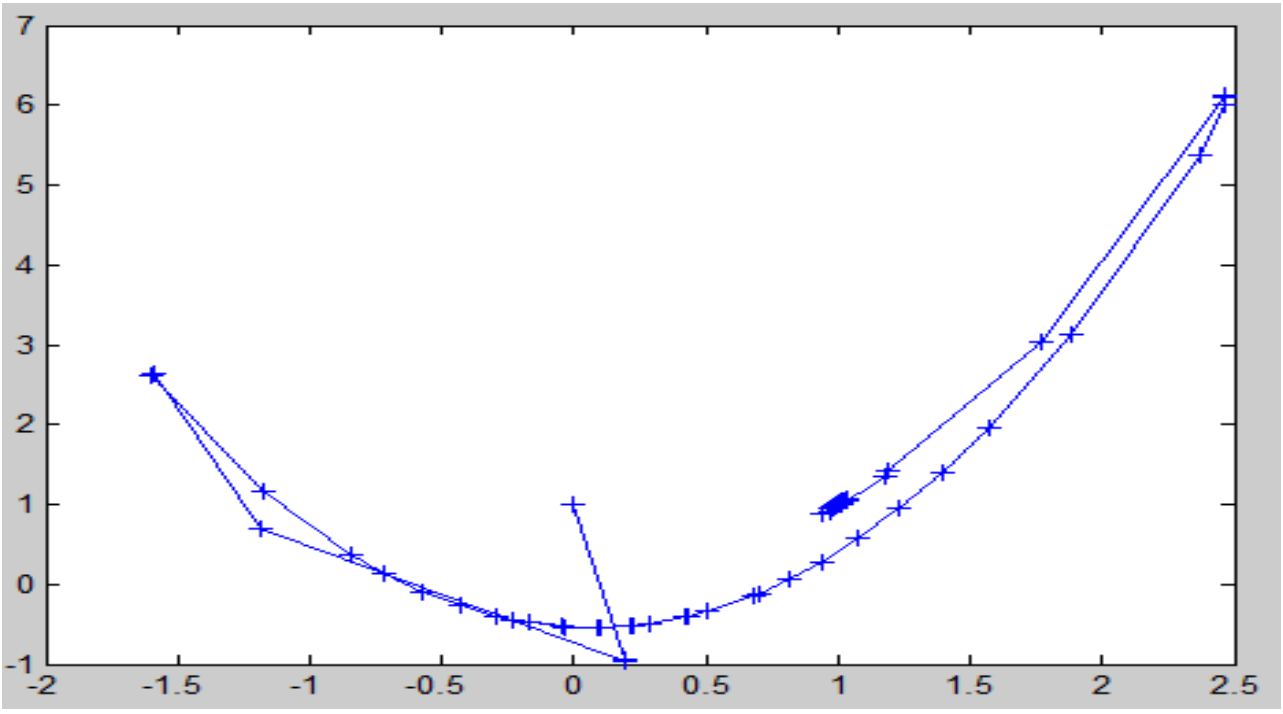
```
function [x]=GradConj_quelconque(x,Nmax,epsilon)
tic
k=0;
e=1;
m1=10^(-4);
m2=0.9999;
r=grad(x);
d=-r;
c1=r'*r;
s=x;
 while(k<=Nmax)
    r=grad(x);
    c2=r'*r;
    beta=c2/c1;
    d=-r+beta*d;
    alpha=Wolfe(x,d);
    xopt=x+alpha*d;
      if norm(r,inf)>epsilon
       e=k;
       break
```

```
    end
    c1=c2;
    k=k+1;

    s=[s x];
     x=xopt;
end
if e==k
    disp('nombre d''itérations')
    disp(k)
else
    disp('convergence non atteinte');
end
    toc
    c=s(1,:);
    v=s(2,:);
plot(s(1,:),s(2,:),'-b+')
end
```

La méthode converge en k=100 itérations en 0.113415 seconds vers le point optimal x=[1.000 ;1.000], le zig-zag des itérés est très clair sur la figue ci-dessous, lorsqu'on se rapproche de la solution optimale.

On obtient la courbe suivante :



Comparaison des méthodes :

On remarque que les méthodes de gradient à pas fixe, à pas prédéterminé et à pas optimal possèdent des comportements caractéristiques différents à savoir :

- La non-garantie de convergence pour l'algorithme de gradient à pas fixe, lorsqu'on part d'un point initial assez loin du point optimal théorique.

- La garantie de convergence des méthodes du gradient à pas prédéterminé, à pas optimal, et celle du gradient conjugué caractérisées par le comportement en zig-zag des itérés lorsqu'on se rapproche de la solution.

Ceci s'explique par le fait que deux directions de descente successives calculées par l'algorithme du gradient à pas optimal sont orthogonales.
On remarque aussi la vitesse de convergence de la méthode du gradient conjugué qui exige le principe des directions conjugués.
En revanche, l'intérêt de la recherche linéaire de Wolfe apparait très clairement, en forçant et accélérant la vitesse de convergence des itérés de la méthode du gradient à pas optimal, par contre elle converge en un nombre d'itération supérieur à celui du gradient conjugué.

- On peut donc conclure que la meilleure de ces méthodes en point de vue de rapidité de convergence est la méthode du gradient conjugué.

4-2 Application des méthodes de descente sur des fonctionnelles cubiques.

On considère la fonctionnelle cubique définie par :

$$J(x) = \langle a, x\rangle + \frac{1}{2}\|Bx\|^2 + \frac{1}{3}\|Cx\|^3$$

Avec $a \in \mathbb{R}^n$, B est une matrice carrée et C l'est aussi.
On souhaite résoudre le problème d'optimisation sans contraintes suivant :

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

On a les expressions du vecteur gradient G et de la matrice Hessienne H données par :

$$G = \nabla J(x) = a + B^T Bx + \|Cx\|Cx \quad (\in \mathbb{R}^n)$$

$$H = H_J(x) = B^T B + \|Cx\|C^T C + \frac{1}{\|Cx\|}C^T Cxx^T C^T C \quad (\in \mathbb{R}^{n\times n})$$

On se propose de comparer quelques méthodes de descente pour le cas quadratique. En utilisant la fonction matlab **fminsearch** pour la détermination du pas de descente, et avec un point de départ x0=[1,1,1,1,1] .

➢ Méthode du gradient à pas optimal :

Le programme de la méthode est donné par :

```
function [x]=pas_optimal(n,a,B,C)
tic
k=0;
epsilon=10^(-6);
```

```
x=ones(n,1);
xx=zeros(n,1);
s=x;
 while norm(xx-x)>epsilon & k<=1000
    xx=x;
    G=a+B'*B*x+(norm(C*x))*C'*C*x;
    f=@(x) a'*(x-t*G)+0.5*(norm(B*(x-t*G))^2)+1/3*(norm(C*(x-t*G))^3);
    t=fminsearch(f,0);
    xopt=x-t*G;
    x=xopt;
    k=k+1;
    s=[s x];
 end
toc
 c=s(1,:);
 plot(s(1,:),'-b+')
end
```

- **Méthode de Newton :**

```
function [x]=newton_quadratique(n,a,B,C)
tic
k=0;
epsilon=10^(-6);
x=ones(n,1);
xx=zeros(n,1);
s=x;
 while norm(xx-x)>epsilon & k<=1000
    xx=x;
    G=a+B'*B*x+(norm(C*x))*C'*C*x;
    H=B'*B+(norm(C*x))*C'*C+(1/(norm(C*x)))*C'*C*x*x'*C'*C;
    T=inv(H);
    xopt=x-T*G;
    x=xopt;
    k=k+1;
    s=[s x];
 end
toc
 c=s(1,:);
 plot(s(1,:),'-b+')
end
```

- **Méthode de Gauss-Newton :**

On considère le problème d'optimisation de moindres carrés suivant :

$$\min_{x\in\mathbb{R}^n} r(x) = \min_{x\in\mathbb{R}^n} \frac{1}{2}\|J(x)\|^2$$

Le programme associé à la méthode est donné par :

```
function [x]=GaussNewton(n,a,B,C)
tic
k=0;
epsilon=10^(-6);
```

```
x=ones(n,1);
xx=zeros(n,1);
s=x;
 while norm(xx-x)>epsilon & k<=1000
    xx=x;
    G=a+B'*B*x+(norm(C*x))*C'*C*x;
    f=@(x) a'*(x-t*G)+0.5*(norm(B*(x-t*G))^2)+1/3*(norm(C*(x-t*G))^3);
    g=G'*G;
    T=inv(g);
    p=-G'*f;
    t=fminsearch(f,0);
    xopt=x+t*T*p;
    x=xopt;
    k=k+1;
    s=[s x];
 end
toc
 c=s(1,:);
 plot(s(1,:),'-b+')
end
```

Comparaison des méthodes :

- On remarque que ces trois méthodes convergent vers le point optimal x=[1 ;1 ;1 ;1 ;1] , ainsi que leur grande rapidité de convergence.

La méthode du gradient à pas optimal converge en : 0.00034 seconds, en dimension finie, en n itérations.
La méthode de Newton converge en : 0.000116 seconds, en une seule itération.
La méthode de Gauss-Newton converge en : 0.000263 seconds en une seule itération.
On peut donc conclure que la meilleure de ces méthodes en termes de rapidité de convergence est la méthode de Newton qui converge en une seule itération, ensuite celle de Gauss-Newton qui évite le calcul du Hessien et de son inverse qui est une opération très coûteuse, et il s'ensuit la méthode du gradient à pas optimal qui converge en un nombre d'itérations plus élevé et qui se révèle peu efficace dans ce cas.

V. Conclusion :

Ces méthodes de descente utilisées pour la résolution des problèmes de minimisation sans contraintes, que nous avons traité dans ce travail sont d'une grande utilité, d'une part par leur grande rapidité de convergence vers le point de minimum global, qui varie d'une méthode à l'autre, et d'autre part par leur grande efficacité pour la minimisation des fonctions objectifs.
Nous avons également remarqué lors de la réalisation de ce travail, qu'il existe une grande variété de ces méthodes dites de descente, et que chaque méthode traitée possède une caractéristique qui la différencie des autres.

Ces méthodes sont jugées de même par rapport à leur efficacité pour des fonctionnelles quadratiques.
La méthode de Newton converge rapidement mais elle est coûteuse, car on a besoin de calculer la matrice Hessienne et son inverse à chaque itération.
Celle de Gauss-Newton est moins rapide mais ne nécessite pas le calcul de l'inverse du Hessien.

Pour les autres méthodes que nous avons traité, on peut considérer l'ordre croissant suivant :

1) La méthode du gradient à pas fixe est de convergence lente, et pas toujours garantie.
2) Les méthodes du gradient à pas optimal et prédéterminé sont de convergence moyenne et leur convergence est garantie.
3) La meilleure de ces méthodes est celle du gradient conjugué, qui est de convergence rapide et garantie.

Bibliographie :

[1] Jean Céa, Optimisation : Théorie et algorithmes, Dunod, (1971).

[2] Michel Minoux, Programmation mathématique : Théorie et algorithmes, Ed Tec & Doc, Lavoisier (2008).

[3] Xavier ANTOINE, Pierre DREYFUSS et Yannick PRIVAT, Introduction à l'optimisation : aspects théoriques, numériques et algorithmes, ENSMN-ENSEM 2A (2006-2007).

[4] Ionel Sorin CIUPERCA, Cours optimisation : Cours en Master M1 SITN.

[5] Aude RONDEPIERRE & Sébastien TORDEUX, Introduction à l'optimisation numérique (2009-2010).

[6] Benzine RACHID, Optimisation sans contraintes Tome 2 : Méthode du gradient conjugué (16 Avril 2016).