# Chapitre 3

# Introduction à l'optimisation

Sommaire



*Optimiser : rendre optimal, donner à quelque chose les meilleures conditions d'utilisation, de fonctionnement ou de rendement au regard de certaines circonstances.*

(Déf. du LAROUSSE).

## 3.1 Qu'est-ce qu'un problème d'optimisation ?

Soit $X$ est un sous-ensemble non vide de $\mathbb{R}^n$. Considérons un problème d'optimisation de la forme :

$$\min f(x) \quad \text{s.c.} \quad x \in X, \tag{3.1}$$

La fonction $f : \mathbb{R}^n \to \mathbb{R}$ est appelée fonction *coût, objectif ou critère*. L'ensemble $X$ est appelé *ensemble ou domaine des contraintes*. Tout point $x \in \mathbb{R}^n$ vérifiant : $x \in X$, est appelé *point admissible* ou *point réalisable* du problème (3.1).

Chercher une solution du problème avec contraintes (3.1) revient à chercher un point de minimum local de $f$ dans l'ensemble des points admissibles, au sens de la définition suivante :



**Définition 3.1: Minimum local/Minimum global**

- $x_0 \in \mathbb{R}^n$ est un point de minimum local de $f$ sur $X \subset \mathbb{R}^n$ si et seulement si

$$x_0 \in X \quad \text{et} \quad \exists \mathcal{V}_{x_0} \text{ un voisinage de } x_0 \text{ tq : } \forall x \in \mathcal{V}_{x_0} \cap X, f(x) \geqslant f(x_0) \tag{3.2}$$

- $x_0 \in \mathbb{R}^n$ est un point de minimum global de $f$ sur $X$ si et seulement si

$$x_0 \in X \quad \text{et} \quad \forall x \in X, f(x) \geqslant f(x_0). \tag{3.3}$$

A noter que tout point de minimum global est aussi local. Les notions de maximum local et global sont définies de façon tout à fait similaire. Ces définitions sont illustrées sur la Figure 3.1

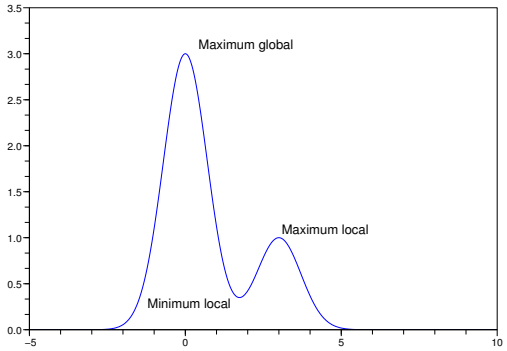



FIGURE 3.1 – Minima et maxima locaux et globaux de la fonction $x \mapsto 3e^{-x^2} + e^{-(x-3)^2}$.

On peut facilement démontrer que les problèmes (avec ou sans contraintes) $\min_x f(x)$ et $\max_x -f(x)$ sont équivalents dans le sens où ils ont même ensemble de solutions et :

$$\min_x f(x) = -\max_x -f(x) \quad \text{ou encore} \quad \max_x f(x) = -\min_x -f(x).$$

Ainsi la recherche d'un maximum pouvant se ramener à la recherche d'un minimum, nous porterons une attention plus particulière à la recherche du minimum.

Les problèmes d'optimisation peuvent être classés en plusieurs grandes familles :

- *Optimisation numérique :* $X \subset \mathbb{R}^n$.
- *Optimisation discrète (ou combinatoire) :* $X$ fini ou dénombrable.
- *Commande optimale :* $X$ est un ensemble de fonctions.
- *Optimisation stochastique :* données aléatoires
  (à ne pas confondre avec les méthodes stochastiques d'optimisation).
- *Optimisation multicritères* : plusieurs fonctions objectifs.

Dans ce cours on mettra l'accent sur l'optimisation numérique : une étape importante en pratique consiste à identifier le type de problème auquel on a affaire afin de savoir quelle famille d'algorithme peut être pertinente.

| Fonction coût | Contraintes | Domaine $X$ | Terminologie |
|---|---|---|---|
| Linéaire<br>$f(x) = c^\top x, c \in \mathbb{R}^n$ | linéaires | Polytope / Polyèdre | Programmation linéaire (P.L.) |
| Linéaire | linéaires | $X \subset \mathbb{Z}$ | P.L. en nombres entiers |
| Quadratique<br>$f(x) = c^\top x + x^\top Q x$<br>$c \in \mathbb{R}^n,\ Q \in \mathbb{S}_n$ | linéaires | Polytope / Polyèdre | Programmation quadratique |
| Convexe | convexes | convexe | Programmation convexe |
| qq | qq | qq | Programmation non linéaire (P.N.L.) |



| | | |
|---|---|---|
| PROGRAMMATION LINÉAIRE | | |
| | | Méthode du simplexe<br>Algorithme des points intérieurs |
| PROGRAMMATION NON LINÉAIRE | | |
| SANS CONTRAINTE | AVEC DÉRIVÉES | Méthodes de type gradient<br>Méthodes de Newton et quasi-Newton<br>Gauss-Newton, Levenberg-Marquardt<br>(pbs de moindres carrés) |
| | SANS DÉRIVÉES | (DFO, NEWUOA, MADS, NOMADS,...)<br>Méthodes heuristiques : Nelder-Mead, surfaces de réponses (réseaux de neurones, krigeage)<br>Méthodes stochastiques : méthodes à 2 phases algos génétiques, algos évolutionnaires, recuit simulé, recherche tabou |
| | NON-DIFF. | Méthodes de sous-gradient<br>Méthodes de faisceaux<br>Méthodes d'échantillonnage de gradient<br>Algorithmes proximaux |
| | $f(x) = g(x) + h(x)$<br>avec $g$ convexe diff.<br>$h$ convexe non diff. | Algorithmes de splitting : descente de gradient proximale (+variante accélérée)<br>Douglas Rachford, ADMM |
| AVEC CONTRAINTES | AVEC DÉRIVÉES | Gradient projeté, algorithme de Uzawa<br>Méthode SQP (Newton)<br>Points intérieurs<br>Méthodes de pénalisation, Lagrangien augmenté |

## 3.2 Résultats d'existence et d'unicité en optimisation

Dans ce paragraphe, nous nous intéresserons aux théorèmes qui permettent d'assurer qu'il existe un minimum à un problème d'optimisation.

Considérons à nouveau le problème (3.1) mais d'un point de vue ensembliste : résoudre

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{sous la contrainte : } x \in X.$$

revient à chercher le minimum de l'image directe $f(X) = \{f(x) : x \in X\}$ de $X$ par $f$.

Il existe principalement deux théorèmes donnant des conditions suffisantes d'existence d'un point de minimum : le premier dans le cas où l'ensemble des contraintes est fermé borné, le second pour un ensemble de contraintes fermé mais non borné.

### 3.2.1 Cas où l'ensemble $X$ des contraintes est borné

**Théorème 3.2: Théorème de Weierstrass**

Soit $X$ un ensemble fermé borné non vide de $\mathbb{R}^n$ et $f : X \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ une application continue sur $X$. Alors $f$ est bornée et atteint ses bornes. Autrement dit, $f$ admet au moins un point $\underline{x} \in X$ de minimum global de $f$ sur $X$ :

$$\forall y \in X,\ f(\underline{x}) \leqslant f(y),$$

et au moins un point $\bar{x} \in X$ de maximum global de $f$ sur $X$ :

$$\forall y \in X,\ f(\bar{x}) \geqslant f(y).$$

**Remarque 3.1.** La condition de continuité de $f$ n'est pas nécessaire, on peut la remplaçer par la condition plus faible de semi-continuité inférieure qui dit essentiellement que $f(\lim x_n) \leqslant \lim f(x_n)$, alors que la continuité impose l'égalité.

*Démonstration.* Soit $(x_n)_{n\in\mathbb{N}}$ une suite minimisante dans $f(X)$, i.e. d'éléments de $X$ telle que

$$\lim_{n\to+\infty} f(x_n) = \inf f(X).$$

Comme $X$ est fermé borné, il existe une sous-suite extraite $(x_{\sigma(n)})_{n\in\mathbb{N}}$ qui converge vers un $x \in X$. Cette suite extraite vérifie

$$x_{\sigma(n)} \longrightarrow x \quad \text{et} \quad f(x_{\sigma(n)}) \longrightarrow \inf_{y\in X} f(y).$$

Or $f$ est continue, d'où par unicité de la limite, il suit

$$f(x) = \inf_{y\in X} f(y) \text{ avec } x \in X,$$

et $f$ réalise son minimum sur $X$. □

### 3.2.2 Cas où l'ensemble $X$ des contraintes est non borné

Dans cette section nous nous intéressons au cas où l'ensemble des contraintes est non borné. C'est le cas tout particulièrement lorsqu'il n'y a pas de contraintes, c'est-à-dire $X = \mathbb{R}^n$. Le problème supplémentaire par rapport au cas précédent est qu'il faut empêcher l'infimum d'être à l'infini. Une fonction qui illustre ce problème est la fonction $f : x \mapsto 1/x$ qui admet $0$ comme infimum sur $\mathbb{R}^+$ pour $x = +\infty$. Pour empêcher l'infimum d'être à l'infini, on peut utiliser l'hypothèse de "coercivité" de la fonction :



**Définition 3.3**

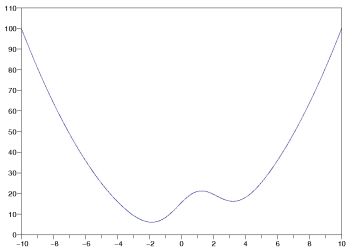
Une application $f : X \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ est dite infinie à l'infini (ou coercive) sur $X$ ssi

$$\forall A \in \mathbb{R}, \exists R > 0 \text{ tel que } \forall x \in X, \text{ si } \|x\| \geqslant R \text{ alors } f(x) \geqslant A. \tag{3.4}$$

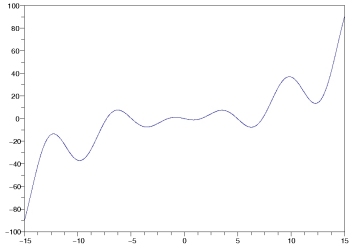
On note : $\lim\limits_{\substack{\|x\|\to+\infty \\ x\in X}} f(x) = +\infty$.

**Exemple 3.2.**

1. $f_1(x) = \|x\|_2$ *est coercive.*
2. $f_2(x) = x_1^2 - x_2^2$ *n'est pas coercive : en effet, la suite de terme général* $x_n = (0, n)$, $n \in \mathbb{N}$, *est telle que :* $\lim\limits_{n\to+\infty} \|x_n\| = \lim\limits_{n\to+\infty} n = +\infty$ *mais :* $\lim\limits_{n\to+\infty} f_2(x_n) = \lim\limits_{n\to+\infty} -n^2 = -\infty$.



*Exemple de fonction coercive ($f_1$)*



*Exemple de fonction non coercive ($f_2$).*

Comme la définition 3.3 n'est pas facile à manier en pratique, on utilise souvent la proposition suivante, qui est une hypothèse un peu plus forte, pour montrer que la fonction est infinie à l'infini.

**Proposition 3.4**

Soit $f : X \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ une application et $g : \mathbb{R} \longrightarrow \mathbb{R}$ vérifiant

$$f(x) \geqslant g(\|x\|) \quad \text{avec} \quad \lim_{t\to+\infty} g(t) = +\infty.$$

Alors, $f$ est infinie à l'infini.

*Démonstration.* Comme $g$ tend vers $+\infty$ en $+\infty$

$$\forall A \in \mathbb{R}, \quad \exists R > 0 \mid \forall t \in \mathbb{R} \quad t \geqslant R \Longrightarrow g(t) \geqslant A.$$

Avec $t = \|x\|$ et comme $g(x) \geqslant f(\|x\|)$, nous obtenons (3.4). □

**Théorème 3.5**

Soient $F$ un fermé non vide de $\mathbb{R}^n$ et $f : F \to \mathbb{R}$ une application continue infinie à l'infini sur $F$. Alors $f$ admet un point de minimum global sur $F$, i.e. il existe $x \in F$ tel que

$$\forall y \in F,\ f(y) \geqslant f(x).$$

### 3.2.3 Cas particulier de contraintes d'égalités et/ou d'inégalités

L'hypothèse $X$ fermé est assez difficile à montrer en pratique sauf dans le cas (fréquent en optimisation) où $X$ est défini par des égalités et des inégalités :

$$X = \{x \in \mathbb{R}^n : h(x) = 0,\ g(x) \leqslant 0\}$$

où $h : \mathbb{R}^n \to \mathbb{R}^p$ et $g : \mathbb{R}^n \to \mathbb{R}^q$. L'écriture "$h(x) = 0$" représente en fait $p$ contraintes d'égalité :

$$h_i(x) = 0, \quad i = 1, \dots, p,$$

et de même "$g(x) \leqslant 0$" représente $q$ contraintes d'inégalité :

$$g_i(x) \leqslant 0, \quad i = 1, \dots, q.$$

Dans le cas où les fonctions contraintes $g$ et $h$ sont continues, on a le résultat suivant :

**Proposition 3.6**

Soient $g : \mathbb{R}^n \to \mathbb{R}^q$ et $h : \mathbb{R}^n \to \mathbb{R}^p$ deux fonctions continues.
- $X = \{x \in \mathbb{R}^n : h(x) = 0,\ g(x) \leqslant 0\}$ est un ensemble fermé de $\mathbb{R}^n$.
- $X = \{x \in \mathbb{R}^n : g(x) < 0\}$ est un ouvert de $\mathbb{R}^n$.

Ainsi , on peut conclure directement que si $f$, $g$ et $h$ sont continues et si :
- soit l'ensemble des contraintes $X = \{x \in \mathbb{R}^n : h(x) = 0,\ g(x) \leqslant 0\}$ est borné,
- soit $f$ est infinie à l'infini,

alors le problème :

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ \text{s.t. :} & g(x) \leqslant 0,\ h(x) = 0. \end{array}$$

admet au moins une solution globale. Autrement dit, $f$ admet au moins un point de minimum global sur $X$.



### 3.2.4 Convexité et optimisation

Les résultats précédents ne nous donnent aucune information quant à l'unicité éventuelle d'un point de minimum global, ni sur le lien entre possibles minima locaux et minima globaux. C'est la notion de *convexité* qui permet de garantir que les minima locaux sont en fait des minima globaux.

Comme nous l'avons vu, la notion de convexité est une notion globale (elle doit être valable en toute paire de points), elle implique donc des propriétés globales sur les problèmes de minimisation. Notamment que tout point de minimum local devient global.

**Théorème 3.7**

Soit $x^\star$ un point de minimum local d'un problème de minimisation.

(i) Si le problème est convexe, alors $x^\star$ est un point de minimum global.

(ii) Si le problème est strictement convexe, alors $x^\star$ est l'unique point de minimum global.

# Chapitre 4

# Optimisation différentiable sans contrainte

Sommaire



Considérons un problème d'optimisation très général de la forme :

$$(P) \qquad \min_{x\in\mathbb{R}^n} f(x), \tag{4.1}$$

où $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ une fonction que l'on suppose différentiable.

## 4.1 Conditions d'optimalité

**Proposition 4.1**

Soit $f : \mathbb{R}^n \to \mathbb{R}$ supposée différentiable. Si $x^\star$ est un point de minimum local de $f$ sur $X$ alors :

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



C'est ce que nous appellerons la condition nécessaire d'optimalité du premier ordre pour l'optimisation sans contrainte.

Rappelons (sans démonstration) les conditions nécessaires du premier et du second ordre pour des problèmes d'optimisation différentiable sans contrainte :

- CONDITIONS NÉCESSAIRES D'OPTIMALITÉ LOCALE

  Si $x^\star \in \mathbb{R}^n$ réalise un minimum local (resp. maximum local) de $f$, alors :

  $\nabla f(x^\star) = 0$ (CN d'optimalité du $1^{er}$ ordre)

  $H[f](x^\star)$ est semidéfinie positive (resp. $H[f](x^\star)$ est semidéfinie négative) (CN d'optimalité du $2^{nd}$ ordre)

- CONDITION SUFFISANTE D'OPTIMALITÉ LOCALE

  Soit $X$ un ouvert de $\mathbb{R}^n$ et $x^\star \in X$. Si :

  $\nabla f(x^\star) = 0$ et $H[f](x^\star)$ symétrique, définie positive (resp. définie négative)

  Alors $x^\star$ est un point de minimum local (resp. maximum local) de $f$ sur $X$.

- CONDITION SUFFISANTE D'OPTIMALITÉ GLOBALE

  Supposons $\nabla f(x^\star) = 0$.

  (i) Si $f$ est convexe, alors $x^\star$ est un point de minimum global de $f$.

  (ii) Si $f$ est strictement convexe, alors $x^\star$ est l'unique point de minimum global de $f$.

Les conditions nécessaires d'optimalité du premier et du second ordre expriment le fait qu'il n'est pas possible de "descendre" à partir d'un point de minimum (local ou global). Cette observation va servir de point de départ à l'élaboration des méthodes dites de *descente* étudiées dans ce chapitre.

## 4.2 Principe général des méthodes de descente

Partant d'un point $x_0$ arbitrairement choisi, un algorithme de descente va chercher à générer une suite d'itérés $(x_k)_{k\in\mathbb{N}}$ définie par :

$$x_{k+1} = x_k + s_k d_k$$

et telle que :

$$\forall k \in \mathbb{N}, \quad f(x_{k+1}) \leqslant f(x_k).$$

Un tel algorithme est ainsi déterminé par deux éléments : le choix de la direction $d_k$ appelée direction de descente, et le choix de la taille du pas $s_k$ à faire dans la direction $d_k$. Cette étape est appelée *recherche linéaire*.

### Définition d'une direction de descente

Un vecteur $d \in \mathbb{R}^n$ est une direction de descente pour $f$ à partir d'un point $x \in \mathbb{R}^n$ si $t \mapsto f(x+td)$ est décroissante en $t=0$, c'est-à-dire s'il existe $\eta > 0$ tel que :

$$\forall t \in ]0, \eta], f(x+td) < f(x). \tag{4.2}$$

Il est donc important d'analyser le comportement de la fonction $f$ dans certaines directions. Lorsqu'elle existe, la dérivée directionnelle donne des informations sur la pente de la fonction dans la direction $d$.

**Proposition 4.2**

Le vecteur $d \in \mathbb{R}^n$ est une direction de descente de $f$ au point $x \in \mathbb{R}^n$ si $f'(x;d) < 0$.

Dans le cas où $f$ est différentiable, on obtient une définition plus pratique d'une direction de descente :

**Proposition 4.3**

Supposons $f : \mathbb{R}^n \to \mathbb{R}$ différentiable. Un vecteur $d \in \mathbb{R}^n$ est une direction de descente de $f$ au point $x$ ssi :

$$f'(x;d) = \nabla f(x)^\top d < 0. \tag{4.3}$$

De plus pour tout $\beta < 1$, il existe $\bar{\eta} > 0$ tel que :

$$\forall t \in ]0, \bar{\eta}], f(x+td) < f(x) + t\beta \nabla f(x)^\top d < f(x). \tag{4.4}$$

Cette dernière inégalité garantit une décroissance minimum de la fonction $f$ dans la direction $d$. Le schéma général d'un algorithme de descente est alors le suivant :



---

ALGORITHME DE DESCENTE MODÈLE.

*Données:* $f : \mathbb{R}^n \to \mathbb{R}$ différentiable, $x_0$ point initial arbitraire.
*Sortie:* une approximation de la solution du problème : $\min_{x \in \mathbb{R}^n} f(x)$.

1. $k := 0$
2. Tant que "test d'arrêt" non satisfait,
   (a) Trouver une direction de descente $d_k$ telle que : $\nabla f(x_k)^\top d_k < 0$.
   (b) *Recherche linéaire :* Choisir un pas $s_k > 0$ à faire dans cette direction et tel que :
   $$f(x_k + s_k d_k) < f(x_k).$$
   (c) Mise à jour : $x_{k+1} = x_k + s_k d_k$ ; $k := k+1$ ;
3. Retourner $x_k$.

---

### Choix de la direction de descente

Une fois la théorie bien maîtrisée, calculer une direction de descente est relativement simple. Dans le cas différentiable, il existe deux grandes stratégies de choix de direction de descente :

- la stratégie de Cauchy : $d_k = -\nabla f(x_k)$, conduisant aux *algorithmes de gradient* décrits au paragraphe 4.4.
- la stratégie de Newton : $d = -H[f](x_k)^{-1} \nabla f(x_k)$, conduisant aux *algorithmes Newtoniens* décrits au paragraphe 4.5.

Remarquons que si $x_k$ est un point stationnaire ($\nabla f(x_k) = 0$) non optimal alors toutes ces directions sont nulles et aucun de ces algorithmes ne pourra progresser. Ce problème peut être résolu en utilisant des approches de type région de confiance qui ne seront pas étudiées dans le cadre de ce cours.

### Critère d'arrêt

Soit $x^\star$ un minimum local du critère $f$ à optimiser. Supposons que l'on choisisse comme test d'arrêt dans l'algorithme de descente modèle, le critère idéal : "$x_k = x^\star$". Dans un monde idéal (i.e. en supposant tous les calculs exacts et la capacité de calcul illimitée), soit l'algorithme s'arrête après un nombre fini d'itérations, soit il construit (théoriquement) une suite infinie $x_0, x_1, \ldots, x_k, \ldots$ de points de $\mathbb{R}^n$ qui converge vers $x^\star$.

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 $x^\star$.

Soit $\varepsilon > 0$ la précision demandée. Plusieurs critères sont à notre disposition : tout d'abord (et c'est le plus naturel), un critère d'optimalité basé sur les conditions nécessaires

d'optimalité du premier ordre présentées dans le chapitre **??** : en optimisation différentiable sans contrainte, on testera si

$$\|\nabla f(x_k)\| < \varepsilon, \tag{4.5}$$

auquel 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 du numérique) :

- Stagnation de la solution : $\|x_{k+1} - x_k\| < \varepsilon(1 + \|x_k\|)$.
- Stagnation de la valeur courante : $\|f(x_{k+1}) - f(x_k)\| < \varepsilon(1 + |f(x_k)|)$.
- Nombre d'itérations dépassant un seuil fixé à l'avance : $k < \mathrm{IterMax}$.

et généralement une combinaison de ces critères :

Critère d'arrêt = Test d'optimalité satisfait
OU (Stagnation de la valeur courante & Stagnation de la solution)
OU Nombre d'itérations maximum autorisé dépassé.

**Remarque 4.1.** En pratique, on préférera travailler avec les erreurs relatives plutôt qu'avec les erreurs absolues, trop dépendantes de l'échelle.

## 4.3 La recherche linéaire

Supposons pour l'instant résolu le problème du choix de la direction de descente et intéressons nous uniquement au calcul du pas : c'est la phase de recherche linéaire.

Soit $x \in \mathbb{R}^n$ un point de $\mathbb{R}^n$ non critique et $d$ une direction de descente de $f$ en $x$. Nous cherchons à calculer un pas $s > 0$ de sorte que :

$$f(x + sd) < f(x).$$

Le choix de ce pas répond généralement à deux objectifs souvent contradictoires : trouver le meilleur pas possible et effectuer le moins de calculs possibles. Ces deux objectifs ont donné naissance à deux grandes familles : les algorithmes à pas fixe et ceux à pas optimal.

---

RECHERCHE LINÉAIRE : PAS FIXE. $s_k = s_{k-1}$

---

RECHERCHE LINÉAIRE : PAS OPTIMAL. $s_k$ solution du problème $\min_{s>0} f(x_k + sd_k)$

---

Illustrées par les méthodes de descente de gradient, aucune de ces deux stratégies ne s'est révélée réellement convaincante : si la première peut être "risquée" du point de vue de la convergence, la seconde est souvent loin d'être triviale à mettre en oeuvre (sauf dans le cas quadratique) et généralement inutilement coûteuse : en effet, à quoi bon calculer très précisément un pas optimal dans une direction qui n'est peut-être pas la bonne ?



(comme c'est par exemple le cas pour la méthode de plus profonde descente). Les recherches linéaires modernes reposent sur l'idée qu'un pas de descente acceptable est un pas qui fait "suffisamment" décroître la fonction objectif. Reste alors à définir les pas qui sont acceptables et ceux qui ne le sont pas.

On notera dans la suite $\varphi : s \in \mathbb{R} \mapsto f(x + sd)$ la fonction dite de *mérite* associée à $f$. La fonction $f$ étant supposée au moins différentiable, $\varphi$ est dérivable sur $\mathbb{R}$, de dérivée : $\varphi'(s) = \nabla f(x + sd)^\top d$ et :

$$\varphi'(0) = \nabla f(x)^\top d < 0. \tag{4.6}$$

**D'autres recherches linéaires élémentaires.** Commençons par décrire quelques améliorations des recherches linéaires à pas fixe et à pas optimal. Le problème majeur des algorithmes à pas fixe est la convergence, en particulier si le pas est trop grand. Il est possible d'assurer la convergence par des stratégies de rebroussement ("backtracking" en anglais) de la façon suivante :

---

RECHERCHE LINÉAIRE AVEC REBROUSSEMENT.

$s_k = s_{k-1}$
Tant que $f(x_k + s_k d_k) \geqslant f(x_k)$ :
  $s_k = s_k/2$

---

On peut également réduire le coût de la recherche linéaire à pas optimal en réduisant l'ensemble sur lequel la recherche linéaire est faite :

---

RECHERCHE LINÉAIRE PARTIELLE.

*Données:* $s_{k-1}$, $T = [1\ a_1\ a_2\ \ldots\ a_k]$ un tableau de réels $> 0$ contenant $1$.
*Sortie:* $s_k$.
$S = s_{k-1} * T = [s_{k-1}\ \ s_{k-1} * a_1\ \ s_{k-1} * a_2\ \ \ldots\ \ s_{k-1} * a_k]$.
$s_k = \arg\min_{s \in S} f(x_k + sd_k)$.

---

## 4.4 Algorithmes de type gradient

Supposons la fonction $f : \mathbb{R}^n \to \mathbb{R}$ différentiable. Un algorithme de type gradient utilise comme direction de descente l'opposée de la direction du gradient de $f$ au point courant :

$$x_0 \in \mathbb{R}^n \text{ quelconque}, \quad x_{k+1} = x_k - s_k \nabla f(x_k) \tag{4.7}$$

où le pas $s_k > 0$ est déterminé par l'une des stratégies de recherche linéaire présentées au paragraphe précédent. On rappelle que $d_k = -\nabla f(x_k)$ est bien une direction de descente de $f$ en $x_k$, et que c'est la direction de "plus forte pente" de $f$ en $x_k$.

**Exercice 4.2.** *Démontrer que $d_k = -\nabla f(x_k)$ est bien une direction de descente de $f$ en $x_k$, et que c'est la direction de "plus forte pente" de $f$ en $x_k$.*

Solution : puisque :

$$\langle \nabla f(x_k), d_k \rangle = -\|\nabla f(x_k)\|^2,$$

et que c'est la direction de plus forte pente de $f$ en $x_k$ au sens où, pour toute direction $d$ de norme constante égale à $\|d\| = \|\nabla f(x)\|$, on a :

$$d_k^\top \nabla f(x_k) = (-\nabla f(x_k))^\top \nabla f(x_k) \leqslant d^\top \nabla f(x_k), \tag{4.8}$$

Dans ce paragraphe nous nous intéressons aux conditions suffisantes de convergence de ce type d'algorithmes. On supposera pour cela que la fonction objectif $f$ est bornée inférieurement, de classe $C^1$ à gradient Lipschitz de constante $L > 0$ sur $\mathbb{R}^n$.

### 4.4.1 CS pour qu'un algorithme de gradient soit une méthode de descente

On cherche une une condition suffisante sur le pas $s_k$ pour que l'algorithme (4.7) soit une méthode de descente, i.e. :

$$\forall k \in \mathbb{N},\ f(x_{k+1}) < f(x_k).$$

**Proposition 4.4**

Soit $f$ une fonction bornée inférieurement, de classe $C^1$ à gradient Lipschitz de constante $L > 0$. Si $s_k < \frac{2}{L},\ \forall k$, alors l'algorithme :

$$x_0 \in \mathbb{R}^n \text{ quelconque}, \quad x_{k+1} = x_k - s_k \nabla f(x_k)$$

est bien un algorithme de descente i.e. : $\forall k \in \mathbb{N},\ f(x_{k+1}) < f(x_k)$.

*Démonstration.* Soit $x_k$ l'itéré courant. Appliquons le Lemme 2.24 aux points $x_k$ et $x_{k+1}$ :

$$\begin{aligned} f(x_{k+1}) &\leqslant f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{L}{2}\|x_{k+1} - x_k\|^2 \\ &\leqslant f(x_k) - s_k \|\nabla f(x_k)\|^2 + \frac{L}{2} s_k^2 \|\nabla f(x_k)\|^2 \\ &\leqslant f(x_k) + s_k \left(\frac{L}{2} s_k - 1\right) \|\nabla f(x_k)\|^2. \end{aligned} \tag{4.9}$$

d'où :

$$f(x_{k+1}) - f(x_k) \leqslant s_k \left(\frac{L}{2} s_k - 1\right) \|\nabla f(x_k)\|^2.$$



Nous avons donc un algorithme de descente (i.e. : $f(x_{k+1}) < f(x_k)$) dès que le pas $s_k$ vérifie :

$$s_k < \frac{2}{L}.$$

Remarquons que le pas assurant la meilleure décroissance de $f$ par ce modèle est donné : $s^\star = \frac{1}{L}$ et qu'on a alors :

$$f(x_{k+1}) \leqslant f(x_k) - \frac{1}{2L}\|\nabla f(x_k)\|^2.$$

□

### 4.4.2 Résultats de convergence globale

Supposons maintenant que le pas $s_k$ vérifie bien : $\forall k \in \mathbb{N},\ s_k < \frac{2}{L}$. On souhaite démontrer la convergence globale de l'algorithme (4.7) i.e. montrer que :

$$\lim_{k \to +\infty} \|\nabla f(x_k)\| = 0.$$

**Théorème 4.5: Convergence du critère**

Soit $f$ une fonction bornée inférieurement, de classe $C^1$ à gradient Lipschitz de constante $L > 0$. Alors :

1. Si $\forall k,\ s_k := s < \frac{2}{L}$, l'algorithme de descente de gradient à pas constant converge globalement et :

   $$0 < s\left(1 - \frac{L}{2}s\right)\|\nabla f(x_k)\|^2 \leqslant f(x_k) - f(x_{k+1}).$$

   De plus, le pas constant assurant la meilleure décroissance possible est : $s_k = \frac{1}{L}$.
2. L'algorithme de gradient à pas optimal converge globalement et :

   $$\frac{1}{2L}\|\nabla f(x_k)\|^2 \leqslant f(x_k) - f(x_{k+1}).$$

*Démonstration.*

1. Soit $k \in \mathbb{N}$. Commençons par écrire que

   $$f(x_{k+1}) - f(x_k) = f(x_k - s\nabla f(x_k)) - f(x_k) = \Big[f(x_k - t\nabla f(x_k))\Big]_0^s$$

   En intégrant, on obtient que

   $$\Big[f(x_k - t\nabla f(x_k))\Big]_0^s = -\int_0^s \langle \nabla f(x_k), \nabla f(x_k - t\nabla f(x_k))\rangle dt$$

En ajoutant le vecteur nul $\nabla f(x_k) - \nabla f(x_k)$ dans le second membre du produit scalaire, on obtient que (attention au signe du terme intégral)

$$f(x_{k+1}) - f(x_k) = -s\|\nabla f(x_k)\|_2^2 + \int_0^s \langle \nabla f(x_k), \nabla f(x_k) - \nabla f(x_k - t\nabla f(x_k))\rangle dt.$$

Majorons le produit scalaire à l'aide de l'inégalité de Cauchy-Schwarz, puis de l'hypothèse de régularité sur $f$,

$$f(x_{k+1}) - f(x_k) \leqslant -s\|\nabla f(x_k)\|_2^2 + \int_0^s tL\|\nabla f(x_k)\|_2^2 dt = -\left(s - \frac{s^2}{2}L\right)\|\nabla f(x_k)\|_2^2.$$

Par hypothèse sur le pas de temps, on a $\left(s - \frac{s^2}{2}L\right) > 0$. On prouve ainsi que la suite des $f(x_k)$ est décroissante. La convergence suit car cette suite est minorée par $\inf f$.

2. Pour le pas optimal, la décroissance étant la meilleure possible, elle est meilleure que si on avait choisi $s_k = \frac{1}{L}$, d'où :

$$\frac{1}{2L}\|\nabla f(x_k)\|^2 \leqslant f(x_k) - f(x_{k+1}).$$

□

**Théorème 4.6: Convergence du critère d'optimalité**

Soit $f$ une fonction bornée inférieurement, de classe $C^1$ à gradient Lipschitz de constante $L > 0$.
Si $\forall k,\ s_k := s < \frac{2}{L}$, l'algorithme de descente de gradient à pas constant assure

$$\lim_{k\to\infty} \|\nabla f(x_k)\|_2 = 0$$

*Démonstration.* Dans le théorème précédent on a établi que pour tout $k \in \mathbb{N}$

$$f(x_{k+1}) - f(x_k) \leqslant -\left(s - \frac{s^2}{2}L\right)\|\nabla f(x_k)\|_2^2.$$

Soit $K \in \mathbb{N}$. Sommons ces inégalités pour $k$ entre 0 et $K$, certains termes se téléscopent :

$$0 \leqslant \left(s - \frac{s^2}{2}L\right)\sum_{k=0}^{K}\|\nabla f(x_k)\|_2^2 \leqslant f(x_0) - f(x_K + 1) \leqslant f(x_0) - \inf f.$$

Puisque le membre de droite dans cette dernière inégalité est constant et fini, on peut faire tendre $K$ vers $+\infty$ et en déduire que la série de terme général $\|\nabla f(x_k)\|_2^2$ converge absolument. Par conséquent le terme général de cette série converge vers zéro. □



### 4.4.3 Convergence dans le cas convexe

**Résultats théoriques** En rajoutant des hypothèses plus fortes, comme la convexité, on peut alors montrer que l'algorithme du gradient à pas constant non seulement converge, mais converge bien vers le minimum. On est de plus capable d'établir une borne sur la vitesse de convergence du critère.

**Théorème 4.7: Convergence du critère vers le minimum**

Supposons que $f$ est bornée inférieurement, convexe, et que $\nabla f$ est $L$-Lipschitz. Considérons l'algorithme de descente de gradient à pas fixe suivant :

$$x_{k+1} \longleftarrow x_k - \frac{1}{L}\nabla f(x_k).$$

Alors,

$$f(x_k) - f^\star \leqslant L\frac{\|x_0 - x^\star\|_2^2}{2k}.$$

*Démonstration.* Par le Lemme 2.24, on a pour tout $x$ et $z$ que

$$f(x) \leqslant f(z) + \langle \nabla f(z), (x - z)\rangle + \frac{L}{2}\|x - z\|_2^2.$$

En particulier, on a

$$f(x) \leqslant g_k(x) := f(x_{k-1}) + \nabla f(x_{k-1})^T(x - x_{k-1}) + \frac{L}{2}\|x - x_{k-1}\|_2^2$$

Notons que $g_k$ est minimale pour $x = x_k$. On peut alors réécrire $g_k(x) = g_k(x_k) + \frac{L}{2}\|x - x_k\|_2^2$, et en particulier pour $x = x^\star$ on a $g_k(x^\star) = g_k(x_k) + \frac{L}{2}\|x^\star - x_k\|_2^2$ . Ainsi,

$$\begin{aligned} f(x_k) &\leqslant g_k(x_k) = g_k(x^\star) - \frac{L}{2}\|x^\star - x_k\|_2^2 \\ &\leqslant f(x_{k-1}) + \nabla f(x_{k-1})^T(x^\star - x_{k-1}) + \frac{L}{2}\|x^\star - x_{k-1}\|_2^2 - \frac{L}{2}\|x^\star - x_k\|_2^2 \\ &\leqslant f^\star + \frac{L}{2}\|x^\star - x_{k-1}\|_2^2 - \frac{L}{2}\|x^\star - x_k\|_2^2 \end{aligned}$$

En sommant pour $k$ de 1 à $K$, on a une somme télescopique

$$K\left(f(x_K) - f^\star\right) \leqslant \sum_{k=1}^{K} f(x_k) - f^\star \leqslant \frac{L}{2}\|x^\star - x_0\|_2^2 - \frac{L}{2}\|x^\star - x_K\|_2^2,$$

ce qui mène au résultat attendu. □

**Vitesse de convergence** C'est une convergence dite sous-linéaire.

**Taux de convergence**

— Taux sous-linéaire : décrit en terme de puissance du nombre d'itération, par exemple

$$r_k \leqslant c/\sqrt{k}$$

— Taux linéaire : décrit en terme exponentiel du nombre d'itération

$$r_k \leqslant c(1-q)^k$$

— Taux quadratique : décrit en terme d'une double exponentielle du nombre d'itération

$$r_k \leqslant cr_k^2$$

**Remarque 4.3.** Les techniques de recherche linéaires de type pas optimal (ou Wolfe) ne peuvent pas améliorer la situation. En effet, les descentes de gradient sans mémoire, qui utilisent uniquement le gradient à l'itération courante, ont au mieux un taux de convergence en $O(1/k)$.

**Remarque 4.4.** Une manière de traduire la proposition 4.7 est la suivante : la suite des itérées $(x_k)_{k\in\mathbb{N}}$ est une suite minimisante de $f$. Ainsi, si $f$ est de plus $\mu$-fortement convexe, alors on en déduit que cette suite converge vers l'unique minimiseur de $f$, avec

$$\forall k \in \mathbb{N}, \qquad 0 \leqslant \|x_k - x^\star\|_2^2 \leqslant \frac{2L\|x_0 - x^\star\|_2^2}{\mu k} = \frac{2\kappa\|x_0 - x^\star\|_2^2}{k}$$

avec $\kappa$ le conditionnement de la fonction $f$. On peut déjà observer que ce taux est d'autant plus intéressant que le conditionnement est bon (qu'il n'explose pas). On verra plus loin qu'en réalité il est possible d'obtenir un taux bien plus intéressant pour les fonctions à gradient Lipschitz et fortement convexes.

**Des méthodes de descente accélérées** On peut montrer que pour toutes les méthodes de premier ordre (ne faisant intervenir que l'information du gradient de la fonction), il existe des fonctions pathologiques telles que $f(x_k) - f^* \geqslant O(1/k^2)$. On peut se demander s'il existe des méthodes optimales qui convergent en $O(1/k^2)$. La réponse est oui, elle a été donnée en 1987 par Y. Nesterov.

L'idée - similaire aux méthodes de gradient conjugué linéaire - est de définir $x_{k+1}$ comme un élément de $\mathrm{vect}(\nabla f(x_0), \dots, \nabla f(x_k))$ plutôt que de n'utiliser que le gradient à l'itération courante. L'intuition derrière ce choix est que le gradient d'une fonction convexe apporte une information globale sur la topologie de la fonction et ne doit pas être considéré simplement comme une direction de descente locale. Les premiers schémas optimaux sont dus à Boris Polyak qui a proposé une classe de méthodes appelée



"Heavy ball" (balle lourde). Un problème important des descentes de gradient est qu'elles ont tendance à produire des directions de descentes orthogonales. L'intuition des méthodes heavy ball est de laisser tomber une boule sur le graphe de la fonction et de la soumettre à son poids en considérant en plus les forces de friction. Ceci peut être fait en considérant l'EDO suivante :

$$\ddot{x} + \alpha\dot{x} + \beta\nabla f(x) = 0.$$

En discrétisant cette EDO, et en autorisant les paramètres $\alpha$ et $\beta$ à varier au cours du temps, on obtient

$$x_{k+1} = x_k - \alpha_k\nabla f(x_k) + \beta_k(x_k - x_{k-1}).$$

On obtient ainsi une classe de méthodes d'optimisation qui doivent permettre de réduire les phénomènes oscillatoires. Notons que la direction de descente est cette fois de la forme $-\alpha_k\nabla f(x_k) + \beta_k(x_k - x_{k-1})$ et qu' en développant le terme $(x_k - x_{k-1})$, on va retrouver des éléments de $\mathrm{vect}(\nabla f(x_0), \dots, \nabla f(x_k))$ tout entier.

**Théorème 4.8: Descente de gradient accélérée**

ACCÉLÉRATION DE NESTEROV.

*Données:* $t_1 = 1$, $x_0 = y_1$, $L$ constante de Lipschitz de $\nabla f$.
*Sortie:* $x_K$.

1. $x_k = y_k - 1/L\nabla f(x_k)$
2. $t_{k+1} = \left(1 + \sqrt{1 + 4t_k^2}\right)/2$
3. $y_{k+1} = x_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1})$

Cet algorithme assure que

$$f(x_k) - f(x^\star) \leqslant \frac{L\|x_0 - x^\star\|_2^2}{k^2}.$$

C'est un taux de convergence optimal.

**Tests numériques : pas fixe/pas optimal** L'objet de ce paragraphe est de mettre en évidence la performance des méthodes de recherche linéaire modernes par rapport aux stratégies pas fixe et pas optimal. Pour cela, nous allons mettre en œuvre les algorithmes de gradient à pas fixe, à pas optimal et à pas de Wolfe sur l'exemple (quadratique) suivant :

$$\min_{(x,y)\in\mathbb{R}^2} f(x,y) = \frac{1}{2}x^2 + \frac{7}{2}y^2,$$

Sans trop d'effort, on vérifie que la fonction $f$ est deux fois différentiable sur $\mathbb{R}^2$, strictement convexe et qu'elle admet un unique point de minimum (global) en $(0,0)$. De plus, le gradient $\nabla f$ est bien Lipschitzien de constante $L = 5\sqrt{2}$ (le vérifier !).

D'un point de vue numérique, soit $X_k = (x_k, y_k) \in \mathbb{R}^2$ l'itéré courant supposé non stationnaire (i.e. : $\nabla f(x_k, y_k) \neq 0$). La direction de recherche est donnée :

$$d_k = -\nabla f(X_k) = \begin{pmatrix} -x_k \\ -7y_k \end{pmatrix}.$$

Les algorithmes de gradient à pas fixe et à pas optimal sont définis de la façon suivante :

- **Descente de gradient à pas fixe.** On fixe $s_k = s$ et à chaque itération de l'algorithme on a :

  $$x_{k+1} = x_k - s\nabla f(x_k).$$

  Rappelons que, d'après la proposition 4.4, l'algorithme de gradient à pas fixe est un algorithme de descente si le pas $s$ est choisi inférieur à $\frac{2}{L} \simeq 0.2828$. Le Tableau 4.1 illustre le comportement de cet algorithme pour différentes valeurs du pas $s$.

| pas | 0.325 | 0.25 | 0.125 | 0.05 | 0.01 |
|---|---|---|---|---|---|
| Nb d'itérations | DV | 49 | 101 | 263 | 1340 |

TABLE 4.1 – Nombres d'itérations de l'algorithme de gradient à pas fixe pour approcher le point de minimum global de $f$ à $10^{-5}$ près, en fonction du pas - Point initial : $x_0 = (7, 1.5)$.

- **Descente de gradient à pas optimal.** A chaque itération, on calcule le pas optimal $s_k$ solution de :

  $$\min_{s>0} f(X_k + sd_k) = \min_{s>0} \frac{1}{2}x_k^2(1-s)^2 + \frac{7}{2}y_k^2(1-7s)^2.$$

  dont la solution est :

  $$s_k = \frac{x_k^2 + 7^2y_k^2}{x_k^2 + 7^3y_k^2}.$$

On retrouve sur la Figure 4.1 le comportement caractéristique des méthodes de gradient à pas fixe ou optimal, à savoir :

- La non-garantie de convergence pour l'algorithme de gradient à pas fixe.
- la lenteur de la méthode de plus profonde descente, caractérisée par le comportement en zigzag des itérés lorsque l'on se rapproche de la solution (deux directions de descente successives sont orthogonales).

En revanche, l'intérêt d'une autre recherche linéaire dite de Wolfe apparaît clairement sur la Figure 4.2 : elle permet de forcer la convergence et d'accélérer la vitesse de convergence des itérés.



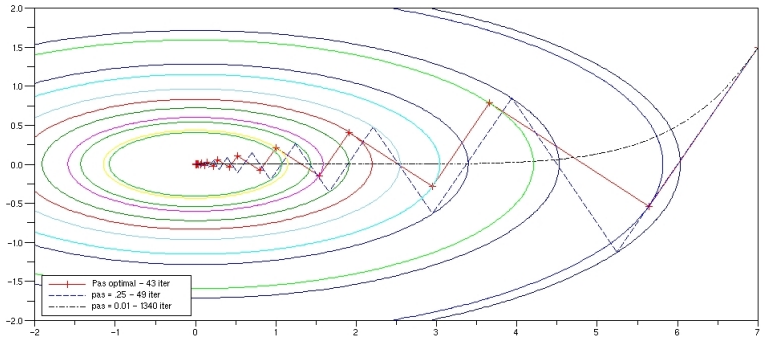



FIGURE 4.1 – Itérations des algos de gradient pas fixe et optimal, générées à partir du point $(7, 1.5)$.

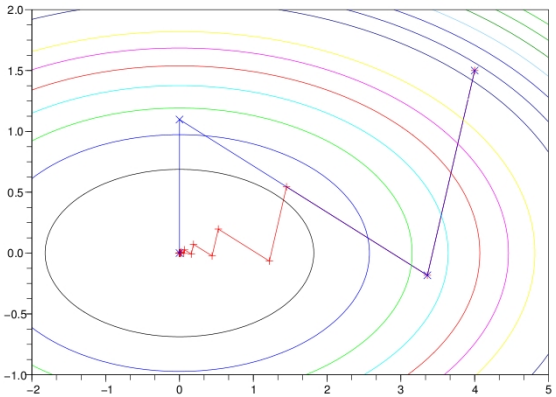


FIGURE 4.2 – 23 it. de l'algorithme de plus profonde descente (en rouge) VS 3 it. l'algorithme de gradient avec recherche linéaire de Wolfe (en bleu) à partir de $(x_0, y_0) = (4, 1.5)$.

### 4.4.4 Convergence pour $f$ à gradient Lipschitz et fortement convexe

**Théorème 4.9**

Supposons que $f$ est $\mu$-fortement convexe, et que $\nabla f$ est $L$-Lipschitz. Considérons l'algorithme de descente de gradient à pas fixe suivant :

$$x_{k+1} \longleftarrow x_k - \frac{1}{L}\nabla f(x_k).$$

Alors,

$$\|x^\star - x_k\|_2^2 \leqslant \left(\frac{L-\mu}{L+\mu}\right)^k \|x^\star - x_0\|_2^2,$$

et

$$f(x_k) - f^\star \leqslant \left(\frac{L-\mu}{L+\mu}\right)^k \frac{L\|x_0 - x^\star\|_2^2}{2} \leqslant \left(1 - \frac{\mu}{L}\right)^k \frac{L\|x_0 - x^\star\|_2^2}{2}.$$

**Remarque 4.5.** C'est ce que l'on appelle une vitesse de convergence linéaire.

**Remarque 4.6.** La forte convexité transforme le taux de convergence en taux linéaire.

*Démonstration.* On repart de la preuve du Théorème 4.7 :

$$\begin{aligned} f(x_k) &\leqslant f(x_{k-1}) + \nabla f(x_{k-1})^T(x^\star - x_{k-1}) + \frac{L}{2}\|x^\star - x_{k-1}\|_2^2 - \frac{L}{2}\|x^\star - x_k\|_2^2 \\ &\leqslant f^\star + \frac{L-\mu}{2}\|x^\star - x_{k-1}\|_2^2 - \frac{L}{2}\|x^\star - x_k\|_2^2. \end{aligned}$$

On a aussi $f(x_k) \geqslant f^\star + \frac{\mu}{2}\|x_k - x^\star\|_2$ ce qui implique

$$\frac{L}{2}\|x^\star - x_k\|_2^2 \leqslant f^\star - f(x_k) + \frac{L-\mu}{2}\|x^\star - x_{k-1}\|_2^2 \leqslant -\frac{\mu}{2}\|x_k - x^\star\|_2^2 + \frac{L-\mu}{2}\|x^\star - x_{k-1}\|_2^2$$

Et donc,

$$\begin{aligned} \|x^\star - x_k\|_2^2 &\leqslant \frac{L-\mu}{L+\mu}\|x^\star - x_{k-1}\|_2^2 \\ &\leqslant \left(1 - \frac{\mu}{L}\right)\|x^\star - x_{k-1}\|_2^2. \end{aligned}$$

Finalement,

$$\begin{aligned} f(x_k) - f^\star &\leqslant \frac{L}{2}\|x_k - x^\star\|_2^2 \\ &\leqslant \left(1 - \frac{\mu}{L}\right)^k \frac{L}{2}\|x^\star - x_0\|_2^2. \end{aligned}$$

□



Encore une fois, la convergence est sous-optimale ici. On peut construire une version accélérée pour atteindre des taux optimaux dans le cas $f$ à gradient Lipschitz et fortement convexe (l'amélioration n'intervenant que dans la constante).

**Théorème 4.10: Descente de gradient accélérée dans le cas fortement convexe**

ACCÉLÉRATION - CAS FORTEMENT CONVEXE.

*Données:* $x_0 = y_0$, $L$ constante de Lipschitz de $\nabla f$ et $\mu$ paramètre de forte convexité.

*Sortie:* $x_K$.

1. $x_{k+1} = y_k - 1/L\nabla f(y_k)$
2. $y_{k+1} = x_{k+1} + \left(\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}\right)(x_{k+1} - x_k)$

Cet algorithme assure que

$$f(x_k) - f(x^\star) \leqslant \left(\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}\right)^k \frac{L\|x_0 - x^\star\|_2^2}{2} \leqslant \left(1 - \sqrt{\frac{\mu}{L}}\right)^k \frac{L\|x_0 - x^\star\|_2^2}{2}.$$

## 4.5 Méthodes de type Newton

### 4.5.1 Principe

L'algorithme de Newton en optimisation est une application directe de l'algorithme de Newton pour la résolution d'équations du type : $F(x) = 0$. En optimisation sans contrainte, l'algorithme de Newton cherche les solutions de l'équation :

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

autrement dit, les points critiques de la fonction $f$ à minimiser. En supposant $f$ de classe $C^2$ et la matrice hessienne $H[f](x_k)$ inversible, une itération de l'algorithme de Newton s'écrit :

$$x_{k+1} = x_k - H[f](x_k)^{-1}\nabla f(x_k), \tag{4.10}$$

où $d_k = -H[f](x_k)^{-1}\nabla f(x_k)$ est appelée **direction de Newton**. La direction $d_k$ est également l'unique solution du problème :

$$\arg\min_{d\in\mathbb{R}^n} f(x_k) + \langle \nabla f(x_k), d\rangle + \frac{1}{2}\langle H_f(x_k)d, d\rangle.$$

Autrement dit, $d_k$ est le point de minimum global de l'approximation de second ordre de $f$ au voisinage du point courant $x_k$.

A condition que la matrice $H[f](x_k)$ soit définie positive à chaque itération, la méthode de Newton est bien une méthode de descente à pas fixe égal à 1. Les propriétés remarquables de cet algorithme sont :

- ☺ sa convergence quadratique (le nombre de décimales exactes est multiplié par 2 à chaque itération).
- ☹ les difficultés et le coût de calcul de la hessienne $H[f](x_k)$ : l'expression analytique des dérivées secondes est rarement disponible dans les applications.
- ☹ le coût de résolution du système linéaire $H[f](x_k)(x_{k+1} - x_k) = \nabla f(x_k)$.
- ☹ l'absence de convergence si le premier itéré est trop loin de la solution, ou si la hessienne est singulière.
- ☹ Pas de distinction entre minima, maxima et points stationnaires.

La question que l'on se pose dans cette partie est donc : comment forcer la convergence globale de l'algorithme de Newton ? L'idée des méthodes de type Newton consiste à reprendre l'algorithme de Newton en remplaçant les itérations par :

$$x_{k+1} = x_k - s_k H_k^{-1} \nabla f(x_k),$$

où

- la matrice $H_k$ est une approximation de la hessienne $H[f](x_k)$.
- $s_k > 0$ est le pas calculé par une recherche linéaire bien choisie.

Plusieurs questions se posent alors : comment déterminer une matrice $H_k$ qui soit une "bonne" approximation de la hessienne à l'itération $k$ sans utiliser les informations de second ordre et garantir que $-H_k^{-1}\nabla f(x_k)$ soit bien une direction de descente de $f$ en $x_k$, sachant que la direction de Newton, si elle existe, n'en est pas nécessairement une ? Comment conserver les bonnes propriétés de l'algorithme de Newton ?

### 4.5.2 Méthode de Newton avec recherche linéaire

Considérons une itération de type Newton très générale :

$$x_{k+1} = x_k - s_k H_k^{-1} \nabla f(x_k),$$

On veut imposer à $H_k$ d'être définie positive afin de garantir que $-H_k^{-1}\nabla f(x_k)$ est bien une direction de descente de $f$ en $x_k$.

1. Si la matrice $H[f](x_k)$ est définie positive, alors :

$$H_k = H[f](x_k).$$

   Si le pas de Newton ($s_k = 1$) est acceptable au sens des conditions de Wolfe, alors l'algorithme effectue une itération de Newton ; sinon il exécute une recherche linéaire de Wolfe initialisée par un pas égal à 1.



2. Sinon la technique la plus utilisée consiste à générer une matrice $E$ telle que :

$$H_k = H[f](x_k) + E$$

   soit définie positive. Remarquons que ceci est toujours possible si l'on choisit $E$ de la forme $\alpha I$.

**Remarque 4.7.** Effectuer une itération de Newton dès que c'est possible, permet de bénéficier de la rapidité de convervence de cet algorithme.

Pour terminer appliquons les algorithmes de Newton avec et sans recherche linéaire aux problèmes suivants :

$$(P_1) \qquad \min_{(x,y)\in\mathbb{R}^2} f(x,y) = 100(y-x^2)^2 + (1-x)^2 \qquad \text{fonction de Rosenbrock.}$$

$$(P_2) \qquad \min_{(x,y)\in\mathbb{R}^2} g(x,y) = \frac{1}{2}x^2 + x\cos y.$$

Le problème $(P_1)$ admet un unique point critique au point $(1,1)$ qui est un point de minimum global de $f$, tandis que le problème $(P_2)$ admet une infinité de points critiques :

$$((-1)^{k+1}, k\pi),\ k\in\mathbb{Z} \quad \text{points de minimum local de } g$$

$$(0, \frac{\pi}{2} + k\pi),\ k\in\mathbb{Z} \quad \text{points selle de } g$$

Les résultats sont présentés sur les figures 4.3 et 4.4. Comme attendu, la supériorité de la méthode de Newton est évidente par rapport aux méthodes de gradient, que ce soit avec recherche linéaire ou non (voir figure 4.3). La figure 4.4 met également en évidence la globalisation de la méthode de Newton grâce à la recherche linéaire qui force la convergence vers un point de minimum local alors que la méthode de Newton seule se retrouve piégée dans un point selle de la fonction.

**Remarque 4.8.** D'un point de vue algorithmique, on calcule en général une factorisation de Cholesky $L_k L_k^\top$ de $H[f](x_k) + \alpha I$ où le paramètre $\alpha > 0$ est augmenté jusqu'à ce que la factorisation soit rendue possible : on initialise $\alpha$ de la façon suivante :

$$\alpha_0 = 0 \qquad \text{si } H[f](x_k) \text{ est définie positive,} \qquad = \frac{1}{2}\|H[f](x_k)\|_F^2 \qquad \text{sinon.}$$

On résout ensuite :

$$L_k z_k = \nabla f(x_k) \qquad \text{et} \qquad L_k^\top d_k = -z_k.$$

L'avantage est que la hessienne au point courant devient définie positive, et on retrouve l'algorithme de Newton et la convergence quadratique de cette méthode.

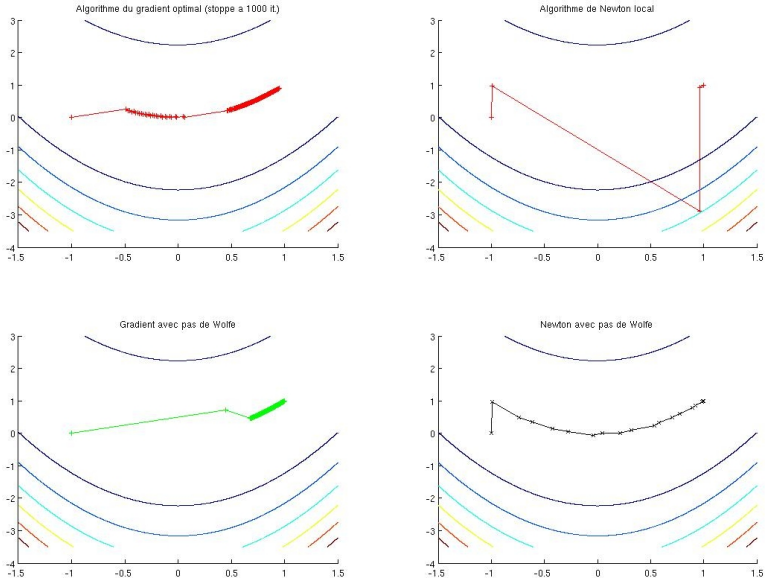



FIGURE 4.3 – Itérations des algorithmes de gradient optimal (stoppé à 10000 itérations !), de Newton (5 itérations), de gradient avec pas de Wolfe (8080 itérations) et de Newton avec pas de Wolfe (19 itérations) pour la minimisation de la fonction de Rosenbrock.

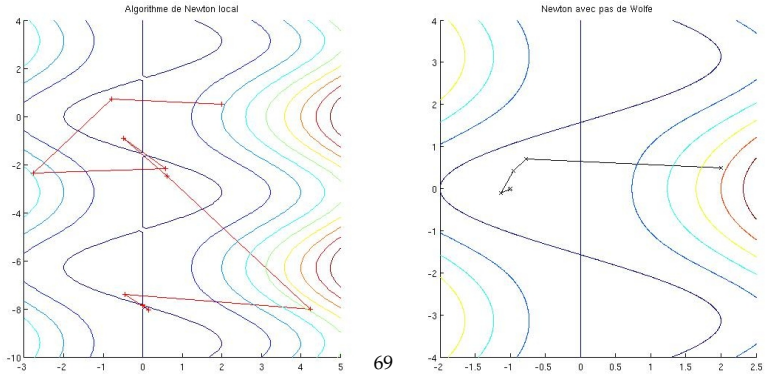



FIGURE 4.4 – Itérations des algorithmes de Newton (10 itérations) et de Newton avec recherche linéaire de Wolfe (6 itérations) pour la minimisation de la fonction $g : (x, y) \mapsto \frac{1}{2}x^2 + x\cos y$.



### 4.5.3 Convergence globale de l'algorithme de Newton avec recherche linéaire

Supposons $f$ de classe $C^1$ à gradient Lipschitz et bornée inférieurement. Le choix de la direction de descente dans cet algorithme est le suivant :

$$d_k = -H_k^{-1}\nabla f(x_k),$$

où $H_k$ est une matrice symétrique définie positive. Couplée avec une recherche linéaire de Wolfe, nous sommes exactement dans le cadre d'application du théorème de Zoutendijk qui nous donne le résultat suivant :

**Proposition 4.11: CS de convergence globale**

Si le conditionnement de la matrice $H_k$ reste borné au cours des itérations, i.e. si :

$$\exists M > 0,\ \forall k \in \mathbb{N},\ \kappa_2(H_k) = \|H_k\|_2\|H_k^{-1}\|_2 \leqslant M,$$

alors l'algorithme de Newton avec pas de Wolfe converge globalement.

*Démonstration.* qui nous donne la convergence de la série : $\sum \cos(\theta_k)^2\|\nabla f(x_k)\|^2$ où :

$$\cos(\theta_k) = \frac{\langle \nabla f(x_k), H_k^{-1}\nabla f(x_k)\rangle}{\|\nabla f(x_k)\|\|H_k^{-1}\nabla f(x_k)\|}.$$

Remarquons que la matrice $H_k$ étant construite de façon à être définie positive, on a :

$$\begin{aligned}\langle \nabla f(x_k), H_k^{-1}\nabla f(x_k)\rangle &\geqslant \lambda_{min}(H_k^{-1})\|\nabla f(x_k)\|^2 \\ &\geqslant \frac{1}{\|H_k\|_2}\|\nabla f(x_k)\|^2\end{aligned}$$

D'où :

$$\cos(\theta_k) \geqslant \frac{1}{\|H_k\|_2\|H_k^{-1}\|_2} = 1/\kappa_2(H_k).$$

Une condition suffisante de convergence globale de l'algorithme de Newton avec recherche linéaire est donc que le conditionnement de la matrice $H_k$ reste borné au cours des itérations. □

On énonce sans démontrer des résultats de vitesses de convergence pour la méthode de Newton.

**Théorème 4.12: Convergence - algo de Newton**

Soit $f$ une fonction convexe, $C^2$. On suppose que la Hessienne est $M$-Lipschitz, càd

$$\|H_f(x) - H_f(y)\|_F \leqslant M\|x - y\|, \qquad \forall x, y \in S.$$

On suppose également que localement la Hessienne est bornée inférieurement, càd que

$$H_f(x^\star) \geq l\mathrm{Id} \qquad \text{avec} \quad l > 0.$$

On suppose que le premier itéré n'est pas loin de la solution $x^\star$ :

$$\|x_0 - x^\star\| < \bar{r} := \frac{2l}{3M}.$$

Alors, l'algorithme de Newton assure que $\|x_k - x^\star\| < \bar{r}$ pour tout $k$ et il converge à vitesse quadratique :

$$\|x_{k+1} - x^\star\| \leqslant \frac{M\|x_k - x^\star\|^2}{2(l - M\|x_k - x^\star\|)}.$$