

6 Méthodes à directions de descente

Il est aisé d'en conclure que la valeur de $f(x - \alpha \nabla f(x))$ deviendra inférieure à $f(x)$ si α est suffisamment petit. Si, maintenant, α vient à croître, et si, comme nous l'avons supposé, la fonction f est continue, la valeur de $f(x - \alpha \nabla f(x))$ décroîtra jusqu'à ce qu'elle [...] coïncide avec une valeur minimum, déterminée par l'équation à une inconnue $D_\alpha f(x - \alpha \nabla f(x)) = 0$. Il suffira donc, ou de résoudre cette dernière équation, ou du moins d'attribuer à α une valeur suffisamment petite, pour obtenir une nouvelle valeur de f inférieure à $f(x)$. Si la nouvelle valeur de f n'est pas un minimum, on pourra en déduire, en opérant toujours de la même manière, une troisième valeur plus petite encore ; et, en continuant ainsi, on trouvera successivement des valeurs de f de plus en plus petites, qui convergeront vers une valeur minimum de f .

L.-A. CAUCHY, extrait de [103 ; 1847, page 537], récrit avec les notations du présent ouvrage.

Ce chapitre introduit une classe importante d'algorithmes de résolution des problèmes d'optimisation sans contrainte. Le concept central est celui de direction de descente (section 6.1). On le retrouvera dans des contextes variés, également pour résoudre des problèmes avec contraintes. Tous les algorithmes d'optimisation n'entrent pas dans ce cadre. Une autre classe importante de méthodes se fonde sur la notion de région de confiance qui sera vue au chapitre 8.

Après avoir décrit le fonctionnement d'un algorithme à directions de descente (section 6.1), nous donnons quelques exemples d'algorithmes de ce type (section 6.2). Nous décrivons ensuite les principales règles de recherche linéaire (section 6.3) et étudions la contribution de la recherche linéaire à la convergence des algorithmes à directions de descente (section 6.4). À la section 6.5, nous énonçons des critères qui permettent d'estimer la qualité de la direction de descente proche d'une solution : celui de l'admissibilité asymptotique du pas unité (section 6.5.1) et celui de la convergence superlinéaire (section 6.5.2). L'algorithme du gradient est une bien mauvaise méthode (elle converge trop lentement), mais son analyse sert de référence à l'étude d'autres algorithmes plus complexes ; nous l'aborderons à la section 6.6. Nous concluons ce chapitre en étudiant l'algorithme proximal pour la minimisation de fonctions convexes (section 6.7). Celui-ci, bien que non implémentable en pratique, sera utilisé pour interpréter certains algorithmes de dualité au chapitre 14.

Connaissances supposées. La section 6.7 sur l'algorithme proximal suppose connues les notions de sous-différentiabilité de fonctions convexes, de [point proximal](#) et de [régularisée de Moreau-Yosida](#) (sections 3.5 et 3.6).

6.1 Principes généraux

Considérons le problème d'optimisation sans contrainte

$$\min_{x \in \mathbb{R}^n} f(x), \quad (6.1)$$

où $f : \mathbb{R}^n \rightarrow \mathbb{R}$ est supposée régulière. On supposera donné un produit scalaire $\langle \cdot, \cdot \rangle$ sur \mathbb{R}^n et on notera $\|\cdot\| = \langle \cdot, \cdot \rangle^{1/2}$ la norme associée. On note aussi $\nabla f(x)$ et $\nabla^2 f(x)$ le gradient et le hessien de f en x pour ce produit scalaire.

On s'intéresse ici à une classe d'algorithmes qui sont fondés sur la notion de direction de descente. On dit que d est une *direction de descente* de f en $x \in \mathbb{R}^n$ si

$$f'(x) \cdot d < 0.$$

Par définition du gradient (section 6.6), il revient au même de dire que $\langle \nabla f(x), d \rangle < 0$ ou encore que d fait avec l'opposé du gradient $-\nabla f(x)$ un angle θ , appelé *angle de descente*, qui est strictement plus petit que 90° :

$$\theta := \arccos \frac{\langle -\nabla f(x), d \rangle}{\|\nabla f(x)\| \|d\|} \in \left[0, \frac{\pi}{2}\right]. \quad (6.2)$$

La notion d'angle définie ci-dessus dépend du produit scalaire et n'est pas invariante par rotation des vecteurs ! L'ensemble des directions de descente de f en x ,

$$\{d \in \mathbb{R}^n : \langle \nabla f(x), d \rangle < 0\},$$

forme un demi-espace ouvert de \mathbb{R}^n (illustration à la figure 6.1).

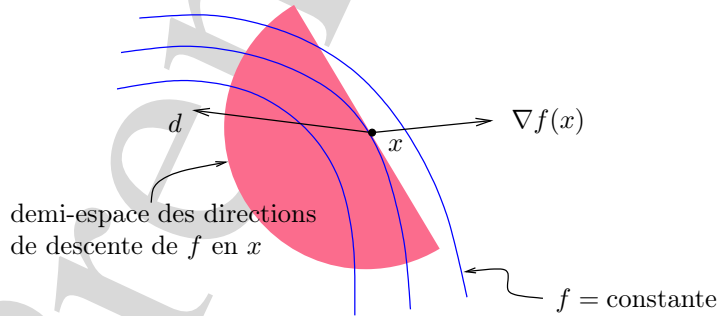
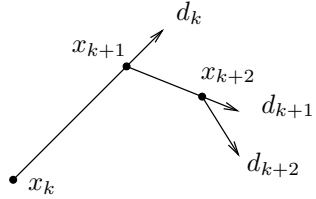


Fig. 6.1. Demi-espace (translaté) des directions de descente d de f en x .

Par définition de la dérivée, on voit que si d est une direction de descente,

$$f(x + \alpha d) < f(x), \text{ pour tout } \alpha > 0 \text{ suffisamment petit}$$

et donc que f décroît strictement dans la direction d . De telles directions sont intéressantes en optimisation car, pour faire décroître f , il suffit de faire un déplacement le long de d . Les méthodes à directions de descente utilisent cette idée pour minimiser une fonction (voir ce qu'en disait Cauchy dans le texte en épigraphe de ce chapitre, lorsque $d = -\nabla f(x)$). Elles construisent la suite des itérés $\{x_k\}_{k \geq 1}$ approchant une solution x_* de (6.1) par la récurrence



$$x_{k+1} = x_k + \alpha_k d_k, \quad \text{pour } k \geq 1, \quad (6.3)$$

où $\alpha_k > 0$ est appelé le *pas* et d_k est une direction de descente de f en x_k . Pour définir une méthode à directions de descente il faut donc spécifier deux choses :

- dire comment la direction d_k est calculée; la manière de procéder donne le nom à l'algorithme;
- dire comment on détermine le pas α_k ; c'est ce que l'on appelle la *recherche linéaire*; la section 6.3 sera consacrée à la description des principales techniques de recherche linéaire.

Décrivons cette classe d'algorithmes de manière précise.

Algorithme 6.1 (méthode à directions de descente — une itération)

On suppose qu'au début de l'itération k , on dispose d'un itéré $x_k \in \mathbb{R}^n$.

1. *Test d'arrêt* : Si $\nabla f(x_k) \simeq 0$, arrêt de l'algorithme.
 2. Choix d'une direction de descente $d_k \in \mathbb{R}^n$.
 3. *Recherche linéaire* : déterminer un pas $\alpha_k > 0$ le long de d_k de manière à « faire décroître f suffisamment ».
 4. $x_{k+1} := x_k + \alpha_k d_k$.
-

Le test de l'étape 1 est discuté dans les paragraphes qui suivent. Le sens à donner à l'expression « faire décroître f suffisamment » à l'étape 3 sera précisé dans la section 6.3.

Dans les problèmes sans contrainte, le test d'arrêt de l'étape 1 porte sur la petitesse du gradient : $\nabla f(x_k) \simeq 0$. C'est en effet ce que suggère la condition nécessaire d'optimalité du premier ordre $\nabla f(x_*) = 0$ (voir (4.13)). Comme x_k n'est jamais exactement égal à x_* , ce test ne pourra marcher que si $\nabla f(x)$ est faible en norme pour x voisin de x_* , ce qui revient pratiquement à supposer que f est de classe C^1 . Par ailleurs, un tel test d'arrêt suggère qu'un algorithme à directions de descente ne peut pas trouver mieux qu'un point stationnaire de f . C'est en effet souvent le cas, mais ce point faible est rarement rédhibitoire en pratique. On peut noter qu'il existe une

version élaborée des méthodes à régions de confiance qui permet de trouver un minimum local, évitant ainsi les points stationnaires qui n'ont pas cette propriété de minimalité. Nous étudierons cette approche au chapitre 8.

On est parfois tenté d'arrêter l'algorithme si le critère f ne décroît presque plus. Ceci n'est pas sans risque et il vaut mieux ne pas utiliser un tel test d'arrêt, car une faible variation du critère peut se produire loin d'une solution. En effet, au premier ordre, $f(x_{k+1}) \simeq f(x_k)$ revient à $\alpha_k \langle \nabla f(x_k), d_k \rangle \simeq 0$, ce qui peut arriver si le pas α_k est petit (c'est en général très suspect) ou si la direction de descente fait avec l'opposé du gradient un angle proche de 90 degrés, une situation qui se rencontre fréquemment (si l'algorithme est bien conçu, cela traduit un mauvais conditionnement du problème).

Même si le test d'arrêt de l'étape 1 est suggéré par la théorie, on peut s'interroger sur sa pertinence, du point de vue suivant : peut-on préciser dans quelle mesure le fait d'avoir un petit gradient implique que l'itéré est proche d'un point stationnaire de f ? Le cas où f est quadratique strictement convexe est instructif :

$$f(x) = \frac{1}{2} x^\top A x - b^\top x, \quad \text{avec } A \succ 0.$$

Minimiser f revient alors à déterminer l'unique solution x_* du système linéaire $Ax = b$. Par ailleurs, le gradient de f (pour le produit scalaire euclidien) est le résidu du système linéaire : $\nabla f(x) = Ax - b$. Or on sait bien que, si le conditionnement de A est élevé, on peut très bien avoir $\|Ax - b\|_2$ petit et une erreur $\|x - x_*\|_2$ importante. Le test d'arrêt portant sur la petitesse du gradient doit donc être interprété avec précaution. Le lemme suivant apportera un nouvel éclairage sur cette question, toujours dans le cas où la fonction est quadratique.

Lemme 6.2 (Wilkinson) Soient $A \in \mathbb{R}^{n \times n}$ non nul, $b \in \mathbb{R}^n$ et $x \in \mathbb{R}^n$ non nul. Alors

$$\frac{\|Ax - b\|_2}{\|A\|_2 \|x\|_2} = \min \left\{ \frac{\|\Delta A\|_2}{\|A\|_2} : (A + \Delta A)x = b \right\}.$$

DÉMONSTRATION. Notons $r := Ax - b$. Si ΔA vérifie $(A + \Delta A)x = b$, on a $r = -(\Delta A)x$, dont on déduit $\|r\|_2 \leq \|\Delta A\|_2 \|x\|_2$ ou encore

$$\frac{\|r\|_2}{\|A\|_2 \|x\|_2} \leq \frac{\|\Delta A\|_2}{\|A\|_2}.$$

On obtient l'égalité ci-dessus en prenant $\Delta A = -rx^\top / \|x\|_2^2$, qui vérifie bien $(A + \Delta A)x = b$. \square

Ce lemme nous apprend que si le gradient $\nabla f(x) = Ax - b$ est petit devant $\|A\|_2 \|x\|_2$, alors x est solution d'un système linéaire $(A + \Delta A)x = b$ (ou minimiseur de la fonction quadratique précédente avec A remplacé par $A + \Delta A$), avec ΔA petit devant A . On dit que l'on a interprété la précision de x en termes d'erreur *amont* (*backward error* [284]).

Certains algorithmes convergent lorsque $\alpha_k = 1$ pour tout indice k , donc sans faire de recherche linéaire. Il en est ainsi de l'algorithme proximal de la section 6.7 ou de l'algorithme de Newton étudié au chapitre 9. Mais le plus souvent, ces algorithmes

ne sont définis et ne convergent que si le premier itéré x_1 est suffisamment proche d'une solution (c'est le cas pour l'algorithme de Newton) ou si la fonction possède des propriétés particulières (c'est la convexité du critère qui joue un rôle important dans l'algorithme proximal). Pour ces algorithmes, on peut voir l'introduction du pas α_k calculé par recherche linéaire comme une technique de *globalisation*, c'est-à-dire une technique permettant de forcer la convergence de la suite des itérés même lorsque le premier itéré est loin d'une solution. Nous verrons à la section 6.4 comment la recherche linéaire contribue en effet de manière significative à la convergence des itérés.

L'introduction d'un algorithme se fait d'ailleurs souvent comme suit. On le conçoit à partir de considérations locales (dans l'algorithme de Newton, on linéarise les équations qui déterminent le type de solution recherchée), pour qu'il ait une bonne vitesse de convergence proche d'une solution. On utilise ensuite une technique de globalisation, comme la recherche linéaire (section 6.3) ou les régions de confiance (chapitre 8), pour forcer la convergence. Des conditions pour que la recherche linéaire n'empêche pas de retrouver l'algorithme initial local (avec $\alpha_k = 1$ donc) seront mises en évidence à la section 6.5.1.

6.2 Exemples de méthodes à directions de descente

Oublions un instant la recherche linéaire et concentrons nous sur quelques exemples de directions de descente. On note

$$g_k := \nabla f(x_k)$$

le gradient de f en x_k pour un produit scalaire $\langle \cdot, \cdot \rangle$.

6.2.1 Algorithme du gradient (ou de la plus profonde descente)

Dans cet algorithme, on prend pour direction de recherche

$$d_k = -g_k,$$

appelée *direction du gradient* ou de la *plus profonde descente*. Cette dernière appellation vient du fait que, si g_k est non nul, la direction est parallèle à la solution du problème en $d \in \mathbb{R}^n$ ci-dessous dans lequel on minimise le modèle affine de f (développement au premier ordre) sur une boule de rayon $\Delta > 0$ quelconque :

$$\begin{cases} \min f(x_k) + \langle g_k, d \rangle \\ \|d\| \leq \Delta. \end{cases}$$

Lorsque $\Delta > 0$ et $g_k \neq 0$, la solution de ce problème est en effet $d = -(\Delta/\|g_k\|)g_k$ (exercice 4.11).

La direction du gradient est évidemment une direction de descente si x_k n'est pas un point stationnaire ($g_k \neq 0$) puisque

$$f'(x_k) \cdot (-g_k) = \langle g_k, -g_k \rangle = -\|g_k\|^2 < 0.$$

L'algorithme qui utilise cette direction de descente porte le nom d'*algorithme du gradient* ou d'*algorithme de la plus profonde descente*.

Par son utilisation de directions de plus profonde descente et par sa simplicité de mise en œuvre, l'algorithme du gradient semble séduisant. Cependant, si le produit scalaire utilisé pour calculer le gradient n'est pas bien choisi, cet algorithme convergera très lentement. Si l'on n'a pas d'idées sur ce que doit être le bon produit scalaire, il vaudra donc mieux éviter cette méthode. On notera que, pour minimiser une fonction quadratique strictement convexe de deux variables (ce qui correspond à résoudre un système linéaire de deux équations linéaires à deux inconnues), l'algorithme demande en général un nombre infini d'itérations, alors que la solution est évidente et aisément calculable à la main ou par d'autres algorithmes en un nombre fini d'opérations. En pratique, on observe souvent que $-g_k$ est une bonne direction de descente loin d'une solution mais qu'elle est à éviter dès que l'on entre dans le voisinage d'une solution x_* , là où les termes du second ordre d'un développement de Taylor de f autour de x_* jouent un grand rôle. En fait, comme nous le verrons, le défaut de cet algorithme est d'ignorer la courbure de f , qui est décrite par son hessien.

Malgré ses piètres performances numériques, cet algorithme mérite d'être étudié. Les techniques utilisées pour l'analyser servent en effet souvent de guide dans l'étude d'algorithmes plus complexes. C'est à ce titre que nous en en dirons davantage à la section 6.6.

6.2.2 Algorithme du gradient conjugué

L'*algorithme du gradient conjugué* peut être vu comme une légère modification de l'algorithme du gradient puisque la direction le long de laquelle le pas α_k sera déterminé s'écrit ($k = 1$ est l'indice du premier itéré) :

$$d_k = \begin{cases} -g_1 & \text{si } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{si } k \geq 2. \end{cases}$$

Cette direction est appelée *direction du gradient conjugué*. Le scalaire $\beta_k \in \mathbb{R}$ peut prendre différentes valeurs, ce qui donne à l'algorithme des propriétés différentes.

La forme de cette direction sera justifiée au chapitre 7 dans lequel l'algorithme est étudié en détail. Remarquons déjà que si l'on choisit α_{k-1} de manière à minimiser le critère le long de la direction précédente (c'est-à-dire si α_{k-1} minimise la fonction $\alpha \mapsto f(x_{k-1} + \alpha d_{k-1})$, ce qui implique que $\langle g_k, d_{k-1} \rangle = 0$), la direction d_k est bien de descente en un point non stationnaire ($g_k \neq 0$), puisque

$$f'(x_k) \cdot d_k = \langle g_k, d_k \rangle = -\|g_k\|^2 < 0.$$

Cette manière de déterminer le pas n'est pas acceptable en pratique lorsque le critère est non linéaire, sans propriétés particulières (voir la section 6.3.2). Nous renvoyons le lecteur à la section 7.2.6 pour une discussion sur ce sujet.

6.2.3 Algorithme de Newton

Dans l'*algorithme de Newton* pour l'optimisation sans contrainte, on détermine une direction d_k par la formule suivante :

$$d_k = -\nabla^2 f(x_k)^{-1} g_k.$$

Cette direction est appelée *direction de Newton*. Il faut évidemment que le hessien de f en l'itéré courant soit inversible pour que cette définition ait un sens. Cet algorithme sera étudié en détail au chapitre 9.

Remarquons que si x_* est un minimum vérifiant les conditions d'optimalité du second ordre (CS2), $\nabla^2 f(x_*)$ est définie positive ($\langle \nabla^2 f(x_*)v, v \rangle > 0$, pour tout $v \neq 0$), et donc $\nabla^2 f(x)$ est également définie positive lorsque x est proche de x_* . Dans le voisinage d'une telle solution, d_k est bien définie et est une direction de descente puisque (on suppose aussi que $g_k \neq 0$)

$$f'(x_k) \cdot d_k = -\langle g_k, \nabla^2 f(x_k)^{-1} g_k \rangle = -f''(x_k) \cdot (\nabla^2 f(x_k)^{-1} g_k)^2 < 0.$$

6.2.4 Algorithmes de quasi-Newton

Les *algorithmes de quasi-Newton* s'inspirent de la méthode de Newton pour définir la direction de recherche. Celle-ci s'écrit :

$$d_k = -M_k^{-1} g_k,$$

où M_k est une matrice d'ordre n auto-adjointe (pour le produit scalaire $\langle \cdot, \cdot \rangle$). Cette matrice M_k est générée par des formules de mise à jour qui seront étudiées au chapitre 10. La direction d_k ci-dessus est appelée *direction de quasi-Newton*.

En optimisation, on s'arrangera souvent pour que M_k soit également définie positive ($\langle M_k v, v \rangle > 0$, pour tout $v \neq 0$). Dans ce cas, d_k est une direction de descente de f puisqu'avec $v_k = M_k^{-1} g_k \neq 0$, on a

$$f'(x_k) \cdot d_k = -\langle g_k, M_k^{-1} g_k \rangle = -\langle M_k v_k, v_k \rangle < 0.$$

On observera qu'alors l'angle de descente θ_k défini en (6.2) est contrôlé par le *conditionnement* de la matrice M_k , noté $\kappa(M_k)$. D'après l'exercice 4.12, on a en effet

$$\cos \theta_k = \frac{\langle -g_k, d_k \rangle}{\|g_k\| \|d_k\|} = \frac{\langle g_k, M_k^{-1} g_k \rangle}{\|g_k\| \|M_k^{-1} g_k\|} \geq \frac{\lambda_{\min}(M_k^{-1})}{\lambda_{\max}(M_k^{-1})} = \frac{1}{\kappa(M_k)}. \quad (6.4)$$

6.2.5 Algorithme de Gauss-Newton

On s'intéresse ici à un problème d'optimisation sans contrainte particulier, celui de minimiser la norme ℓ_2 d'une fonction $r : \mathbb{R}^n \rightarrow \mathbb{R}^m$ (en général $m \gg n$), dont les composantes r_i sont appelées les *résidus* :

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|r(x)\|_2^2. \quad (6.5)$$

C'est ce qu'on appelle un *problème de moindres-carrés non linéaire*.

On note $J(x) = r'(x)$ la jacobienne $m \times n$ de r en x . Alors le gradient et le hessien du critère f de (6.5) pour le produit scalaire euclidien s'écrivent

$$\nabla f(x) = J(x)^\top r(x) \quad \text{et} \quad \nabla^2 f(x) = J(x)^\top J(x) + \sum_{i=1}^m r_i(x) \nabla^2 r_i(x).$$

Dans l'algorithme de Gauss-Newton, on détermine d_k comme solution particulière (il peut y en avoir plusieurs) du système linéaire

$$(J(x_k)^\top J(x_k)) d_k = -J(x_k)^\top r(x_k). \quad (6.6)$$

Si $J(x_k)$ est injective, on obtient

$$d_k = - (J(x_k)^\top J(x_k))^{-1} J(x_k)^\top r(x_k).$$

Cette direction est appelée *direction de Gauss-Newton*. Comparée à la direction de Newton sur (6.5), elle n'utilise qu'une partie du hessien de f , de manière à éviter le calcul des dérivées secondes des résidus, qui sont souvent coûteuses à évaluer. Cet algorithme sera étudié en détail au chapitre 17.

La direction de Gauss-Newton d_k est de descente lorsque x_k n'est pas stationnaire ($J(x_k)^\top r(x_k) \neq 0$), puisque

$$\begin{aligned} f'(x_k) \cdot d_k &= \nabla f(x_k)^\top d_k = -r(x_k)^\top J(x_k) d_k \\ &= -d_k^\top (J(x_k)^\top J(x_k)) d_k = -\|J(x_k) d_k\|_2^2 < 0. \end{aligned}$$

La stricte négativité vient du fait que $J(x_k) d_k = 0$ impliquerait par (6.6) que $J(x_k)^\top r(x_k) = 0$, ce que l'on a supposé ne pas avoir lieu.

6.3 La recherche linéaire

6.3.1 Vue d'ensemble

Dans cette section, nous allons décrire les différentes manières de déterminer un pas $\alpha_k > 0$ le long d'une direction de descente d_k . C'est ce que l'on appelle *faire de la recherche linéaire*. Il s'agit de réaliser deux objectifs.

Le premier objectif est de *faire décroître f suffisamment*. Cela se traduit le plus souvent par la réalisation d'une inégalité de la forme

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \text{« un terme négatif »}. \quad (6.7)$$

Le terme négatif, disons ν_k , joue un rôle-clé dans la convergence de l'algorithme utilisant cette recherche linéaire. L'argument est le suivant. Si $f(x_k)$ est minorée (il existe une constante C telle que $f(x_k) \geq C$ pour tout k), alors ce terme négatif tend nécessairement vers zéro : $\nu_k \rightarrow 0$. C'est souvent à partir de la convergence vers zéro de cette suite que l'on parvient à montrer que le gradient lui-même doit tendre vers zéro. Le terme négatif devra prendre une forme bien particulière si l'on veut pouvoir en tirer de l'information. En particulier, il ne suffit pas d'imposer $f(x_k + \alpha_k d_k) < f(x_k)$.

Le second objectif de la recherche linéaire est d'*empêcher le pas $\alpha_k > 0$ d'être trop petit*, trop proche de zéro. Le premier objectif n'est en effet pas suffisant car l'inégalité (6.7) est en général satisfaite par des pas $\alpha_k > 0$ arbitrairement petit. Or

ceci peut entraîner une « fausse convergence », c'est-à-dire la convergence des itérés vers un point non stationnaire, comme le montre l'observation suivante. Si l'on prend

$$0 < \alpha_k \leq \frac{\epsilon}{2^k \|d_k\|},$$

la suite $\{x_k\}$ générée par (6.3) est de Cauchy, puisque pour $1 \leq l < k$ on a

$$\|x_k - x_l\| = \left\| \sum_{i=l}^{k-1} \alpha_i d_i \right\| \leq \sum_{i=l}^{k-1} \frac{\epsilon}{2^i} \rightarrow 0, \quad \text{lorsque } l \rightarrow \infty.$$

Donc $\{x_k\}$ converge, disons vers un point \bar{x} . En prenant $l = 1$ et $k \rightarrow \infty$ dans l'estimation ci-dessus, on voit que $\bar{x} \in \bar{B}(x_1, \epsilon)$ et donc \bar{x} ne saurait être solution s'il n'y a pas de solution dans $B(x_1, \epsilon)$. On a donc arbitrairement forcé la convergence de $\{x_k\}$ en prenant des pas très petits.

Pour simplifier les notations, on définit la restriction de f à la droite $\{x_k + \alpha d_k : \alpha \in \mathbb{R}\}$ comme la fonction

$$h_k : \alpha \mapsto h_k(\alpha) = f(x_k + \alpha d_k). \quad (6.8)$$

6.3.2 Recherches linéaires « exactes »

Comme on cherche à minimiser f , il semble naturel de chercher à minimiser le critère le long de d_k et donc de déterminer le pas α_k comme solution du problème

$$\min_{\alpha \geq 0} h_k(\alpha).$$

C'est ce que l'on appelle la *règle de Cauchy* et le pas déterminé par cette règle est appelé *pas de Cauchy* ou pas *optimal* (voir figure 6.2). Dans certains cas, on préférera le plus petit point stationnaire de h_k qui fait décroître cette fonction :

$$\alpha_k = \inf\{\alpha \geq 0 : h'_k(\alpha) = 0, h_k(\alpha) < h_k(0)\}.$$

On parle alors de *règle de Curry* et le pas déterminé par cette règle est appelé *pas de Curry* (voir figure 6.2). De manière un peu imprécise, ces deux règles sont parfois qualifiées de *recherche linéaire exacte* alors que les règles présentées plus loin sont qualifiées de *recherche linéaire inexacte*.

Ces deux règles ne sont utilisées que dans des cas particuliers, par exemple lorsque h_k est quadratique. En effet, pour une fonction non linéaire arbitraire,

- il peut ne pas exister de pas de Cauchy ou de Curry,
- la détermination de ces pas demande en général beaucoup de temps de calcul et ne peut de toute façon pas être faite avec une précision infinie,
- l'efficacité supplémentaire éventuellement apportée à un algorithme par une recherche linéaire exacte ne permet pas, en général, de compenser le temps perdu à déterminer un tel pas,
- les résultats de convergence autorisent d'autres types de règles, moins gourmandes en temps de calcul.

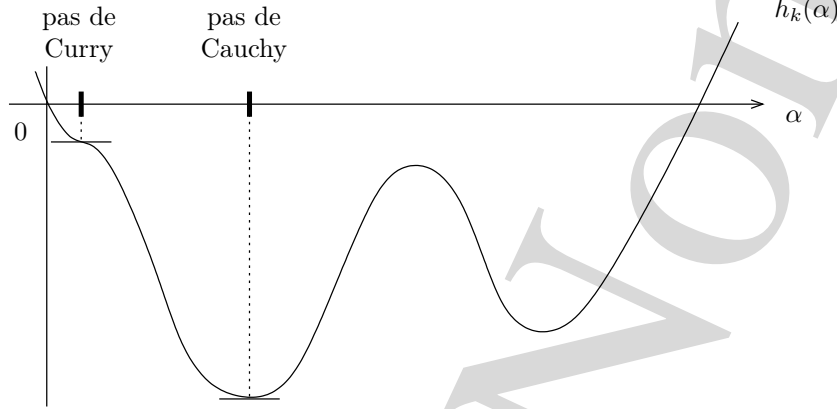


Fig. 6.2. Règles de Cauchy et de Curry.

Au lieu de demander que α_k minimise h_k , on préfère imposer des conditions moins restrictives, plus facilement vérifiées, qui permettent toutefois de contribuer à la convergence des algorithmes. En particulier, il n'y aura plus un unique pas (ou quelques pas) vérifiant ces conditions mais tout un intervalle de pas (ou plusieurs intervalles), ce qui rendra d'ailleurs leur recherche plus aisée. C'est ce que l'on fait avec les règles d'Armijo, de Goldstein et de Wolfe décrites ci-dessous.

6.3.3 Règles d'Armijo et de Goldstein

Une condition naturelle est de demander que f décroisse autant qu'une portion $\omega_1 \in]0, 1[$ de ce que ferait le modèle linéaire de f en x_k . Cela conduit à l'inégalité suivante, parfois appelée *condition d'Armijo* ou *condition de décroissance linéaire*:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \omega_1 \alpha_k \langle g_k, d_k \rangle. \quad (6.9)$$

Elle est de la forme (6.7), car ω_1 devra être choisi dans $]0, 1[$. On voit bien à la figure 6.3 ce que signifie cette condition. Il faut qu'en α_k , la fonction h_k prenne une valeur plus petite que celle prise par la fonction affine $\alpha \mapsto f(x_k) + \omega_1 \alpha \langle g_k, d_k \rangle$. En pratique, la constante ω_1 est prise très petite, de manière à satisfaire (6.9) le plus facilement possible. Typiquement, $\omega_1 = 10^{-4}$. Notons que cette constante ne doit pas être adaptée aux données du problème et donc que l'on ne se trouve pas devant un choix de valeur délicat. On montrera toutefois que, dans certains algorithmes, il est important de prendre $\omega_1 < \frac{1}{2}$ pour que le pas unité ($\alpha_k = 1$) soit accepté lorsque x_k est proche d'une solution (voir par exemple la proposition 6.13).

Il est clair d'après la figure 6.3 que l'inégalité (6.9) est toujours vérifiée si $\alpha_k > 0$ est suffisamment petit. Cela se démontre aussi facilement. En effet dans le cas contraire, on aurait une suite de pas strictement positifs $\{\alpha_{k,i}\}_{i \geq 1}$ convergeant vers 0 lorsque $i \rightarrow \infty$ et tels que (6.9) n'ait pas lieu pour $\alpha_k = \alpha_{k,i}$. En retranchant $f(x_k)$ dans les deux membres, en divisant par $\alpha_{k,i}$ et en passant à la limite quand $i \rightarrow \infty$, on

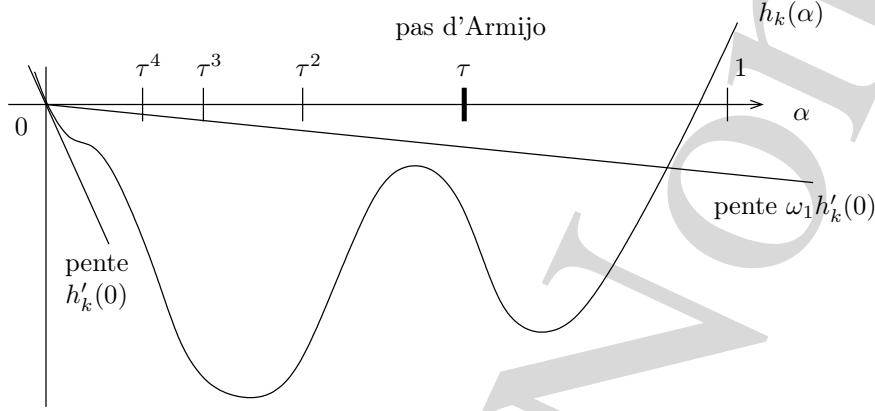


Fig. 6.3. Règle d'Armijo (version simplifiée).

trouverait $\langle g_k, d_k \rangle \geq \omega_1 \langle g_k, d_k \rangle$, ce qui contredirait le fait que d_k est une direction de descente ($\omega_1 < 1$).

D'autre part, on a vu qu'il était dangereux d'accepter des pas trop petits, cela pouvait conduire à une fausse convergence. Il faut donc un mécanisme supplémentaire qui empêche le pas d'être trop petit. On utilise souvent la technique de rebroussement due à Armijo [21 ; 1966] ou celle de Goldstein [239 ; 1965].

Dans sa version la plus simple, la *technique de rebroussement* (*backtracking* en anglais) consiste à prendre $\alpha_k = \tau^{i_k}$, où $\tau \in]0, 1[$ est une constante et i_k est le plus petit entier naturel ($i_k \in \mathbb{N}$) tel que l'on ait (6.9) (voir figure 6.3). C'est le fait de prendre pour α_k le plus grand réel dans $\{1, \tau, \tau^2, \dots\}$ permettant de vérifier (6.9) qui garantit que ce pas ne sera pas trop petit. On voit bien pourquoi cette technique porte le nom de rebroussement : on essaie d'abord $\alpha_k = 1$ et si ce pas n'est pas acceptable, on rebrousse chemin en essayant des pas plus petits τ, τ^2 , etc.

Cette version simplifiée de la technique de rebroussement est souvent améliorée dans les codes soignés. D'abord, s'il est opportun d'essayer le pas unité en premier lieu dans les algorithmes fondés sur la méthode de Newton, ce n'est pas toujours le cas pour d'autres algorithmes. Ensuite les pas intermédiaires τ^i sont imposés a priori, sans que l'on puisse tenir compte des valeurs de f calculées aux points $x_k + \tau^i d_k$. Ces valeurs peuvent en effet servir à estimer un pas α_k vérifiant (6.9). On a plus de liberté en utilisant l'algorithme ci-dessous.

Algorithme 6.3 (règle d'Armijo)

1. Choisir $\alpha_k^1 > 0$ et $\tau \in]0, \frac{1}{2}[$; $i = 1$;
 2. Tant que (6.9) n'est pas vérifiée avec $\alpha_k = \alpha_k^i$:
 - 2.1. Choisir $\alpha_k^{i+1} \in [\tau \alpha_k^i, (1-\tau) \alpha_k^i]$;
 - 2.2. Accroître i de 1;
 3. $\alpha_k = \alpha_k^i$.
-

Typiquement, on prend $\tau \in [10^{-2}, 10^{-1}]$ et le pas α_k^{i+1} est déterminé à l'étape 2.1 par interpolation (voir la section 6.3.5). Le pas déterminé par cette règle de recherche linéaire est appelé *pas d'Armijo*. Cette règle est très souvent utilisée dans l'algorithme de Newton.

La technique de rebroussement a la faiblesse de choisir le pas α_k plus petit que le pas-candidat α_k^1 , qui peut être arbitraire et peut s'avérer trop petit dans certains cas. La *règle de Goldstein* remédie à cet inconvénient. Dans celle-ci, le pas $\alpha_k > 0$ est déterminé de manière à vérifier (6.9) et

$$f(x_k + \alpha_k d_k) \geq f(x_k) + \omega'_1 \alpha_k \langle g_k, d_k \rangle, \quad (6.10)$$

où ω'_1 est une constante choisie dans l'intervalle $]\omega_1, 1[$ (par exemple $\omega'_1 = 0.99$). C'est cette inégalité qui empêche le pas d'être trop petit. Le pas déterminé par cette règle est appelé *pas de Goldstein*.

On déduit facilement du lemme ci-dessous, que l'on peut trouver un pas vérifiant (6.9) et (6.10) dès que h_k est continue et bornée inférieurement (le pas $\bar{\alpha}_k$ dont on montre l'existence convient).

Lemme 6.4 Si $h_k : \mathbb{R}_+ \rightarrow \mathbb{R}$ définie par (6.8) est continue et bornée inférieurement, si d_k est une direction de descente de f en x_k et si $\omega_1 \in]0, 1[$, alors il existe un pas $\bar{\alpha}_k > 0$ tel que $h_k(\bar{\alpha}_k) = f(x_k) + \omega_1 \bar{\alpha}_k \langle g_k, d_k \rangle$.

DÉMONSTRATION. Soit $A = \{\alpha > 0 : (6.9) \text{ est vérifiée pour tout } \alpha_k \in [0, \alpha]\}$. Comme $\omega_1 < 1$ et $h'_k(0) < 0$, A est non vide. D'autre part, $\bar{\alpha}_k = \sup A$ est fini, car h_k est bornée inférieurement et $\omega_1 > 0$. Par continuité de h_k , le résultat est vérifié avec ce pas $\bar{\alpha}_k$. \square

La détermination d'un pas d'Armijo par rebroussement ne présente pas de difficulté particulière : l'algorithme 6.3 est explicite. Par contre, il n'est pas aisé de voir comment on peut trouver un pas de Goldstein en un nombre fini d'étapes. L'exercice 6.3 aborde cette question.

6.3.4 Règle de Wolfe

Dans la *règle de Wolfe*, le pas α_k est déterminé de manière à satisfaire les deux inégalités suivantes, appelées *conditions de Wolfe* (voir figure 6.4) :

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \omega_1 \alpha_k \langle g_k, d_k \rangle \quad (6.11)$$

$$\langle \nabla f(x_k + \alpha_k d_k), d_k \rangle \geq \omega_2 \langle g_k, d_k \rangle, \quad (6.12)$$

où les constantes ω_1 et ω_2 sont choisies telles que

$$0 < \omega_1 < \omega_2 < 1.$$

Typiquement, on prend $\omega_1 = 10^{-4}$ et $\omega_2 = 0.99$. La première inégalité n'est autre que la condition de décroissance linéaire (6.9), tandis que le rôle de (6.12) est d'empêcher le pas d'être trop petit. Cette dernière inégalité s'écrit en effet aussi $h'_k(\alpha_k) \geq \omega_2 h'_k(0)$,

qui n'est pas vérifiée pour $\alpha_k = 0$ (car $\omega_2 < 1$ et $h'_k(0) < 0$) et, par la continuité supposée de h'_k , n'est pas non plus vérifiée pour de petits pas $\alpha_k > 0$. Comme nous le verrons au chapitre 10, cette règle de recherche linéaire est bien adaptée aux algorithmes de quasi-Newton. Le pas déterminé par cette règle est appelé *pas de Wolfe*.

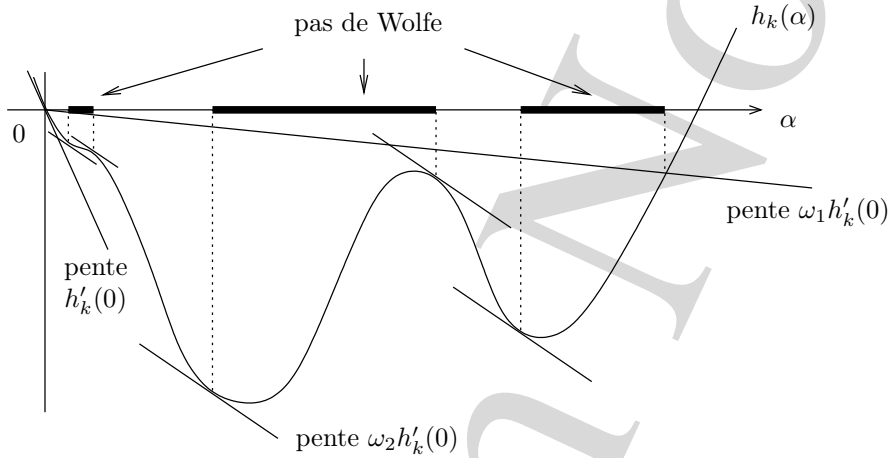


Fig. 6.4. Règle de Wolfe.

Proposition 6.5 Si d_k est une direction de descente de f en x_k , si $h_k : \mathbb{R}_+ \rightarrow \mathbb{R}$ définie par (6.8) est dérivable et bornée inférieurement et si $0 < \omega_1 < \omega_2 < 1$, alors il existe un pas $\alpha_k > 0$ vérifiant les conditions de Wolfe (6.11)–(6.12).

DÉMONSTRATION. Le pas $\alpha_k = \bar{\alpha}_k$ construit dans la démonstration du lemme 6.4 convient. Il vérifie en effet (6.11). D'autre part, par définition de $\bar{\alpha}_k$ (voir la démonstration du lemme 6.4)

$$h_k(\alpha_{k,i}) > f(x_k) + \omega_1 \alpha_{k,i} \langle g_k, d_k \rangle,$$

pour une suite de pas $\alpha_{k,i} \downarrow \bar{\alpha}_k$ quand $i \rightarrow \infty$. En retranchant $h_k(\bar{\alpha}_k) = f(x_k) + \omega_1 \bar{\alpha}_k \langle g_k, d_k \rangle$ dans les deux membres de cette inégalité, en divisant par $(\alpha_{k,i} - \bar{\alpha}_k)$ et en passant à la limite quand $i \rightarrow \infty$, on obtient $h'_k(\bar{\alpha}_k) \geq \omega_1 \langle g_k, d_k \rangle \geq \omega_2 \langle g_k, d_k \rangle$ (car $\omega_1 \leq \omega_2$ et $\langle g_k, d_k \rangle < 0$). Donc (6.12) est vérifiée avec $\alpha_k = \bar{\alpha}_k$. \square

La démonstration précédente n'est pas constructive. En pratique, on utilise des algorithmes spécifiques pour trouver un pas de Wolfe. En voici un, particulièrement simple, dont on peut montrer qu'il trouve un pas de Wolfe en un nombre fini d'étapes (proposition 6.7). Il génère une suite d'intervalles $[\underline{\alpha}, \bar{\alpha}]$ emboîtés, dans lesquels il trouve finalement un pas α_k vérifiant les deux conditions de Wolfe (6.11) et (6.12). Les constantes τ_i et τ_e sont utilisées respectivement dans les phases d'interpolation

(on cherche un nouveau pas dans l'intervalle, étapes 3 et 4.3.4) et d'extrapolation (on cherche un nouveau pas dans un intervalle non borné à droite, étape 4.3.3).

Algorithme 6.6 (Fletcher-Lemaréchal)

1. Soient $\underline{\alpha} := 0$, $\bar{\alpha} := +\infty$, $\tau_i \in]0, \frac{1}{2}[$ et $\tau_e > 1$;
On se donne un premier pas $\alpha > 0$;
2. Répéter :
 - 2.1. Si (6.11) n'est pas vérifiée avec $\alpha_k = \alpha$;
 - 2.2. Alors $\bar{\alpha} = \alpha$ et on choisit un nouveau pas α dans l'intervalle

$$[(1-\tau_i)\underline{\alpha} + \tau_i\bar{\alpha}, \tau_i\underline{\alpha} + (1-\tau_i)\bar{\alpha}]; \quad (6.13)$$

- 2.3. Sinon
 - 2.3.1. Si (6.12) est vérifiée avec $\alpha_k = \alpha$,
 - 2.3.2. Alors on sort avec $\alpha_k = \alpha$;
 - 2.3.3. Sinon
 - 2.3.3.1. $\underline{\alpha} = \alpha$;
 - 2.3.3.2. Si $\bar{\alpha} = +\infty$;
 - 2.3.3.3. Alors choisir un nouveau pas $\alpha \in [\tau_e\underline{\alpha}, \infty[$;
 - 2.3.3.4. Sinon choisir un nouveau pas α dans l'intervalle (6.13).
-

Lorsque l'on sort à l'étape 2.3.2, le pas α_k vérifie clairement (6.11) et (6.12).

Proposition 6.7 *Si d_k est une direction de descente de f en x_k , si $h_k : \mathbb{R}_+ \rightarrow \mathbb{R}$ définie par (6.8) est dérivable et bornée inférieurement et si $0 < \omega_1 < \omega_2 < 1$, alors l'algorithme de Fletcher-Lemaréchal trouve un pas $\alpha_k > 0$ vérifiant les conditions de Wolfe (6.11)–(6.12) en un nombre fini d'étapes.*

DÉMONSTRATION. La finitude de l'algorithme se démontre par l'absurde, c.-à-d., en supposant que la boucle *Répéter* (instruction 2) est parcourue indéfiniment ou encore que l'on n'exécute jamais l'instruction 2.3.2.

Observons d'abord que l'instruction 2.3.3.3 n'est exécutée qu'un nombre fini de fois. En effet, dans le cas contraire, on aurait $\underline{\alpha} \rightarrow +\infty$ et

$$h_k(\underline{\alpha}) \leq h_k(0) + \omega_1 \underline{\alpha} h'_k(0). \quad (6.14)$$

Mais alors h_k ne serait pas bornée inférieurement sur \mathbb{R}_+ (car $\omega_1 h'_k(0) < 0$ par hypothèse), ce qui est contraire aux hypothèses. Par conséquent, après un nombre fini d'étapes, $\bar{\alpha} < +\infty$. Les nouveaux pas-candidats α sont alors toujours choisis dans l'intervalle (6.13). Comme à chaque tour de boucle, on a soit $\underline{\alpha} = \alpha$ soit $\bar{\alpha} = \alpha$, la longueur de l'intervalle $[\underline{\alpha}, \bar{\alpha}]$ tend vers zéro ($0 < \tau_i < 1/2$) et il existe un $\hat{\alpha} \geq 0$ tel que $\underline{\alpha} \rightarrow \hat{\alpha}$ et $\bar{\alpha} \rightarrow \hat{\alpha}$. Nous allons montrer que les propriétés de h_k en $\hat{\alpha}$ sont contradictoires.

Par continuité de h_k , en passant à la limite dans (6.14), on trouve $h_k(\hat{\alpha}) \leq h_k(0) + \omega_1 \hat{\alpha} h'_k(0)$. D'autre part, par les instructions 2.1 et 2.2, (6.11) n'est pas vérifiée en $\alpha = \bar{\alpha}$:

$$h_k(\bar{\alpha}) > h_k(0) + \omega_1 \bar{\alpha} h'_k(0). \quad (6.15)$$

A la limite, on trouve $h_k(\hat{\alpha}) \geq h_k(0) + \omega_1 \hat{\alpha} h'_k(0)$. Avec l'inégalité précédente vérifiée par $\hat{\alpha}$, on obtient

$$h_k(\hat{\alpha}) = h_k(0) + \omega_1 \hat{\alpha} h'_k(0), \quad (6.16)$$

En $\alpha = \underline{\alpha}$, (6.12) n'est pas vérifiée: $h'_k(\underline{\alpha}) < \omega_2 h'_k(0)$. A la limite, on trouve

$$h'_k(\hat{\alpha}) \leq \omega_2 h'_k(0), \quad (6.17)$$

En retranchant (6.16) de (6.15), en divisant le résultat par $\bar{\alpha} - \hat{\alpha} > 0$ et en passant à la limite, on trouve

$$h'_k(\hat{\alpha}) \geq \omega_1 h'_k(0). \quad (6.18)$$

Les inégalités (6.17), (6.18), $h'_k(0) < 0$ et $\omega_1 < \omega_2$ sont contradictoires. \square

Pour certains algorithmes (par exemple le gradient conjugué non linéaire, voir la section 7.2.6), il est parfois nécessaire d'avoir une condition plus restrictive que (6.12). Dans la *règle de Wolfe forte*, on cherche un pas $\alpha_k > 0$ tel que l'on ait :

$$\begin{aligned} f(x_k + \alpha d_k) &\leq f(x_k) + \omega_1 \alpha_k \langle g_k, d_k \rangle \\ |\langle \nabla f(x_k + \alpha d_k), d_k \rangle| &\leq \omega_2 |\langle g_k, d_k \rangle|. \end{aligned}$$

6.3.5 Mise en œuvre

Choix du premier pas

Certaines directions de descente ont un «pas naturel», un pas qui a de grandes chances d'être accepté par les différentes conditions définissant les règles de recherche linéaire vues aux sections 6.3.3 et 6.3.4. Il en est ainsi des directions newtoniennes (sections 6.2.3 et 6.2.4) pour lesquelles le pas naturel est 1. C'est ce que nous verrons à la section 6.5.1. Ce pas sera alors essayé en premier lieu et, puisqu'il est souvent accepté, permettra de faire de la recherche linéaire sans trop d'évaluations de fonction.

D'autres directions n'ont pas de pas naturel évident. Il en est ainsi des directions du gradient conjugué non linéaire et de Gauss-Newton (sections 6.2.2 et 6.2.5). Pour de telles directions de descente d_k , on détermine parfois le premier pas à partir d'un modèle quadratique

$$\alpha \mapsto \varphi_k(\alpha) = a_{0,k} + a_{1,k} \alpha + \frac{a_{2,k}}{2} \alpha^2,$$

interpolant $\alpha \mapsto f(x_k + \alpha d_k)$ à partir de la donnée de $f_k := f(x_k)$, de $p_k := f'(x_k) \cdot d_k$ et de la *décroissance attendue* du critère à l'itération k , notée Δ_k . On préfère cette dernière quantité à la dérivée directionnelle seconde $f''(x_k) \cdot d_k^2$, de manière à éviter le calcul généralement coûteux de cette dernière et à assurer la stricte convexité du modèle quadratique ($f''(x_k) \cdot d_k^2$ peut en effet ne pas être strictement positif). Les conditions naturelles d'interpolation $\varphi_k(0) = f_k$ et $\varphi'_k(0) = p_k$ permettent de donner une valeur aux deux premiers coefficients :

$$a_{0,k} = f_k \quad \text{et} \quad a_{1,k} = p_k < 0.$$

La courbure $a_{2,k}$ est déterminée en imposant que la décroissance maximale de φ_k , qui vaut $\varphi_k(0) - \inf \varphi_k = p_k^2/(2a_{2,k})$, soit égale à Δ_k :

$$a_{2,k} = \frac{p_k^2}{2\Delta_k} > 0.$$

On prend alors comme premier pas à essayer dans la recherche linéaire, celui qui donne la décroissance maximale du modèle quadratique φ_k de f ainsi obtenu, à savoir

$$\alpha_k := \frac{-2\Delta_k}{f'(x_k) \cdot d_k}. \quad (6.19)$$

Ce pas est appelé le *pas de Fletcher*. Il est aussi parfois utilisé à la première itération des algorithmes quasi-newtoniens (chapitre 10), car le pas unité le long de la première direction $-\nabla f(x_1)$ n'est pas approprié.

Le pas de Fletcher reporte la détermination du premier pas sur celle de la décroissance attendue Δ_k du critère et on peut légitimement se demander si l'on a progressé. C'est le cas si l'on a une estimation f_{\min} de la valeur minimale de f (par exemple, on peut prendre $f_{\min} \simeq 0$ dans les problèmes de moindres-carrés non linéaires à résidu optimal presque nul). Dans ce cas, il est raisonnable de déterminer α_k par (6.19) avec

$$\Delta_k = \gamma \left(f(x_k) - f_{\min} \right),$$

où γ est de l'ordre de 10^{-2} ou 10^{-1} . À défaut d'information venant du problème à résoudre, certains développeurs choisissent parfois $\Delta_k = f(x_{k-1}) - f(x_k)$, qui est la décroissance de f réalisée à l'itération précédente (on suppose que $k \geq 2$). Cette valeur n'est pas recommandée pour les algorithmes à la convergence superlinéaire, car alors $f(x_k) - f(x_{k+1})$ est asymptotiquement beaucoup plus petit que $f(x_{k-1}) - f(x_k)$.

Interpolation et extrapolation ▲

Contrôle du nombre d'essais de pas ▲

Il n'est pas judicieux de se donner a priori un nombre maximal d'essais de pas, car la recherche d'un pas satisfaisant le long d'une direction de descente arbitraire peut requérir de nombreux essais à certaine itération particulièrement difficile. Heureusement, c'est rarement le cas pour les directions de descente classiques. Toutefois, la situation peut se présenter; par exemple à la première itération des algorithmes quasi-newtoniens (chapitre 10) parce que l'algorithme n'a pas eu le temps de mettre à l'échelle la direction de recherche ou encore proche de la convergence lorsque les erreurs d'arrondi prévalent.

Lorsque la recherche linéaire détermine le pas par encadrement comme dans l'algorithme de Fletcher-Lemaréchal, il semble préférable de s'arrêter lorsque l'intervalle où l'on recherche le pas devient trop petit. Cependant, ce n'est pas la longueur minimale de l'intervalle de recherche en α qui importe, mais le déplacement en x correspondant; cela permet en effet de ne pas faire dépendre la longueur minimale de l'intervalle de la grandeur de la direction de descente. Ce point de vue revient en fait

à se donner une *résolution en x* , c'est-à-dire une grosseur aux « pixels » discrétisant l'espace des paramètres x : on considérera alors que l'on ne peut pas distinguer deux points appartenant à un même pixel. En pratique, cette résolution sera donnée par l'utilisateur du code, qui devrait bien connaître l'ordre de grandeur des *variations* des paramètres à optimiser x , au moyen d'un vecteur

$$\delta \in \mathbb{R}_{++}^n,$$

dont les composantes sont petites et strictement positives. Un *pixel* est alors un parallépipède rectangle dont les côtés ont pour longueur les $\delta_i > 0$. On décide que la recherche linéaire ne peut pas distinguer deux points x et $y \in \mathbb{R}^n$ tels que, pour tout i , $|x_i - y_i| \leq \delta_i$. La longueur minimale de l'intervalle de recherche du pas dans la direction d sera alors déterminé par

$$\min_{1 \leq i \leq n} \frac{\delta_i}{|d_i|}.$$

La donnée des δ_i permet aussi de faire une mise à l'échelle du problème, de telle sorte que les nouvelles variables $\tilde{x}_i := t x_i / \delta_i$ aient des *variations* nominales identiques. Dans la transformation précédente, la constante $t > 0$ peut être choisie de manière à avoir la plupart des \tilde{x}_i de l'ordre de l'unité.

6.4 Convergence des méthodes à directions de descente

6.4.1 Condition de Zoutendijk

Dans cette section, on étudie la *contribution* de la recherche linéaire à la convergence des algorithmes à directions de descente. Ce n'est qu'une contribution, parce que la recherche linéaire ne peut à elle seule assurer la convergence des itérés. On comprend bien que le choix de la direction de descente joue aussi un rôle. Cela se traduit par une condition, dite de Zoutendijk, dont on peut tirer quelques informations qualitatives intéressantes.

On dit qu'une règle de recherche linéaire satisfait la *condition de Zoutendijk* s'il existe une constante $C > 0$ telle que pour tout indice $k \geq 1$ on ait

$$f(x_{k+1}) \leq f(x_k) - C \|g_k\|^2 \cos^2 \theta_k, \quad (6.20)$$

où θ_k est l'angle de descente, celui que fait d_k avec $-g_k$ et que nous avons défini en (6.2) par

$$\cos \theta_k = \frac{\langle -g_k, d_k \rangle}{\|g_k\| \|d_k\|}.$$

La proposition suivante et les commentaires qui suivent montrent comment on se sert de la condition de Zoutendijk.

Proposition 6.8 (utilité de la condition de Zoutendijk) *Si la suite $\{x_k\}$ générée par un algorithme d'optimisation vérifie la condition de Zoutendijk (6.20) et si la suite $\{f(x_k)\}$ est minorée, alors*

$$\sum_{k \geq 1} \|g_k\|^2 \cos^2 \theta_k < \infty. \quad (6.21)$$

DÉMONSTRATION. En sommant les inégalités (6.20), on a

$$\sum_{k=1}^l \|g_k\|^2 \cos^2 \theta_k \leq \frac{1}{C} (f(x_1) - f(x_{l+1})).$$

La série est donc convergente puisqu'il existe une constante C' telle que pour tout k , $f(x_k) \geq C'$. \square

Considérons la méthode du gradient. Dans celle-ci, $\cos \theta_k = 1$ et donc sous les hypothèses de la proposition précédente, $g_k \rightarrow 0$. On obtient donc directement la « convergence » de la méthode du gradient (on ne peut montrer la convergence des itérés eux-mêmes que dans de rares cas). Plus généralement, si d_k fait avec $-g_k$ un angle θ_k qui ne se rapproche pas de $\frac{\pi}{2}$ (on veut dire par là que le $\cos \theta_k$ reste uniformément positif), l'algorithme est convergent ($g_k \rightarrow 0$). On est dans ce dernier cas, lorsque $d_k = -M_k^{-1}g_k$ avec une matrice M_k définie positive de conditionnement borné (utiliser la minoration (6.4)).

La démarche que l'on suit pour montrer la convergence d'un algorithme avec recherche linéaire peut à présent être schématisée. Si $\{f(x_k)\}$ est minorée (ce qui est une hypothèse très raisonnable lorsqu'on cherche à minimiser une fonction), le « terme négatif » dans (6.7) doit tendre vers zéro (c'est l'argument utilisé dans la démonstration de la proposition 6.8). Grâce à la recherche linéaire, on parvient à majorer ce terme par une constante positive fois $(-\|g_k\|^2 \cos^2 \theta_k)$ (c'est ce que montreront les propositions 6.9, 6.10 et 6.11, ci-dessous). Dès lors $\|g_k\| \cos \theta_k \rightarrow 0$. Si le cosinus de θ_k reste uniformément positif, $g_k \rightarrow 0$ et l'algorithme « converge ». A posteriori, ceci explique aussi pourquoi on a besoin de trouver un pas tel que l'on ait un peu plus que l'inégalité $f(x_{k+1}) < f(x_k)$: c'est la convergence vers zéro du terme négatif faisant décroître $f(x_k)$ qui contribue à la convergence. S'il n'y a pas de terme négatif, on ne peut en général rien dire sur la convergence de l'algorithme.

Il faut se garder de déduire de cette observation, que l'on dispose à présent d'un moyen efficace pour forcer la convergence d'un algorithme à directions de descente, à savoir celui qui empêcherait les directions d_k de faire avec l'opposé du gradient un angle trop proche de $\frac{\pi}{2}$. Ce serait une bien mauvaise idée. En effet, les directions générées par un algorithme d'optimisation sont en général obtenues à partir de considérations cherchant à ce que l'on ait une convergence rapide vers la solution. Pour obtenir cette propriété, l'algorithme doit parfois construire des directions avec un $\cos \theta_k$ proche de zéro. On sent donc bien qu'il est délicat de donner un seuil à partir duquel la direction d_k devrait être « redressée ». Suivre cette approche pourrait brider

un algorithme efficace. Il faut aussi nuancer ce dernier propos, car les méthodes à régions de confiance (qui seront étudiées au chapitre 8) ont un mécanisme qui redresse la direction de déplacement vers l'opposé du gradient dans les situations critiques, mais cette opération est la conséquence d'un autre principe algorithmique qui a tout son sens.

Les trois propositions suivantes précisent les circonstances dans lesquelles la condition de Zoutendijk (6.20) est vérifiée avec les règles de Cauchy, de Curry, d'Armijo et de Wolfe.

Proposition 6.9 (condition de Zoutendijk pour la règle de Curry) *Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction $C^{1,1}$ dans un voisinage de $\mathcal{N}_1 := \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$. On considère un algorithme à directions de descente d_k dans lequel le pas α_k est déterminé de manière à avoir*

$$f(x_k + \alpha_k d_k) \leq f(x_k + \hat{\alpha}_k d_k),$$

où $\hat{\alpha}_k$ est le pas de Curry (que l'on suppose donc exister). Alors, il existe une constante $C > 0$ telle que, pour tout $k \geq 1$, la condition de Zoutendijk (6.20) est vérifiée.

DÉMONSTRATION. Par hypothèse, pour tout $\alpha \in [0, \hat{\alpha}_k]$, on a

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k + \hat{\alpha}_k d_k) \\ &\leq f(x_k + \alpha d_k) \\ &= f(x_k) + \alpha \langle g_k, d_k \rangle + \int_0^1 (f'(x_k + t\alpha d_k) - f'(x_k)) \cdot (\alpha d_k) dt \\ &\leq f(x_k) + \alpha \langle g_k, d_k \rangle + \frac{\alpha^2 L}{2} \|d_k\|^2. \end{aligned}$$

où L est la constante de Lipschitz de f' . Le membre de droite atteint un minimum lorsque α prend la valeur

$$\tilde{\alpha}_k = \frac{-\langle g_k, d_k \rangle}{L \|d_k\|^2}.$$

D'autre part, $\tilde{\alpha}_k \leq \hat{\alpha}_k$, car par la condition de Lipschitz, on a

$$0 = f'(x_k + \hat{\alpha}_k d_k) \cdot d_k \leq \langle g_k, d_k \rangle + \hat{\alpha}_k L \|d_k\|^2.$$

On peut donc prendre $\alpha = \tilde{\alpha}_k$ dans l'inégalité ci-dessus. Ceci donne

$$f(x_{k+1}) \leq f(x_k) - \frac{\langle g_k, d_k \rangle^2}{2L \|d_k\|^2} = f(x_k) - \frac{1}{2L} \|g_k\|^2 \cos^2 \theta_k,$$

qui est l'inégalité recherchée. \square

Comme la valeur prise par f au pas de Cauchy est inférieure à celle prise au pas de Curry, la conclusion de cette proposition reste vraie pour la règle de Cauchy. On pourrait également étendre les résultats ci-dessous en supposant qu'en $x_k + \alpha_k d_k$, f

prend une valeur inférieure à celle prise avec le pas d'Armijo ou de Wolfe. Un tel raisonnement nous sera utile dans la démonstration du lemme 6.12.

Proposition 6.10 (condition de Zoutendijk pour la règle d'Armijo) *Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction $C^{1,1}$ dans un voisinage de $\mathcal{N}_1 := \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$. On considère un algorithme à directions de descente d_k , qui génère une suite $\{x_k\}$ en utilisant la recherche linéaire d'Armijo avec α_k^1 uniformément positif. Alors, il existe une constante $C > 0$ telle que, pour tout $k \geq 1$, l'une des conditions*

$$f(x_{k+1}) \leq f(x_k) - C|\langle g_k, d_k \rangle| \quad (6.22)$$

ou (6.20) est vérifiée.

DÉMONSTRATION. Si le pas $\alpha_k = \alpha_k^1$ est accepté, on a (6.22), car α_k^1 est uniformément positif. Dans le cas contraire, (6.9) n'est pas vérifiée avec un pas $\alpha'_k \leq \frac{\alpha_k}{\tau}$, c'est-à-dire

$$f(x_k + \alpha'_k d_k) > f(x_k) + \omega_1 \alpha'_k \langle g_k, d_k \rangle.$$

Comme f est $C^{1,1}$, on a pour tout $\alpha > 0$:

$$\begin{aligned} f(x_k + \alpha d_k) &= f(x_k) + \alpha \langle g_k, d_k \rangle + \int_0^1 [f'(x_k + t\alpha d_k) - f'(x_k)] \cdot (\alpha d_k) dt \\ &\leq f(x_k) + \alpha \langle g_k, d_k \rangle + C\alpha^2 \|d_k\|^2, \end{aligned}$$

où $C > 0$ est une constante. Avec l'inégalité précédente, et le fait que $\omega_1 < 1$, on obtient

$$|\langle g_k, d_k \rangle| = \|g_k\| \|d_k\| \cos \theta_k \leq \frac{C}{(1-\omega_1)} \alpha'_k \|d_k\|^2,$$

ce qui permet de minorer $\alpha'_k \|d_k\|$ et donc aussi $\alpha_k \|d_k\|$ par une constante fois $\|g_k\| \cos \theta_k$. Cette minoration et l'expression suivante de (6.9)

$$f(x_k + \alpha_k d_k) \leq f(x_k) - \omega_1 \alpha_k \|g_k\| \|d_k\| \cos \theta_k,$$

conduit à (6.20). □

Proposition 6.11 (condition de Zoutendijk pour la règle de Wolfe) *Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction différentiable sur un voisinage de $\mathcal{N}_1 := \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$ et $C^{1,1}$ sur \mathcal{N}_1 . On considère un algorithme à directions de descente d_k , qui génère une suite $\{x_k\}$ en utilisant la recherche linéaire de Wolfe (6.11)-(6.12). Alors, il existe une constante $C > 0$ telle que, pour tout $k \geq 1$, la condition de Zoutendijk (6.20) est vérifiée.*

DÉMONSTRATION. D'après (6.12), on a

$$(1-\omega_2) |\langle g_k, d_k \rangle| \leq \langle g_{k+1} - g_k, d_k \rangle$$

et du fait que f est $C^{1,1}$

$$(1-\omega_2)\|g_k\| \cos \theta_k \leq L\alpha_k\|d_k\|.$$

On utilise alors (6.11) pour obtenir

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) - \omega_1\alpha_k\|g_k\|\|d_k\| \cos \theta_k \\ &\leq f(x_k) - \frac{\omega_1(1-\omega_2)}{L}\|g_k\|^2 \cos^2 \theta_k. \end{aligned}$$

On en déduit (6.20). \square

6.4.2 Suites minimisantes spéciales

Grâce à la condition de Zoutendijk, en appliquant une des règles de recherche linéaire décrites à la section 6.3 à l'algorithme du gradient, on peut obtenir des suites minimisantes ayant la propriété supplémentaire d'avoir un gradient qui tend vers zéro.

Lemme 6.12 (suite minimisante spéciale) Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction bornée inférieurement, dérivable dans le voisinage d'un de ses ensembles de sous-niveau \mathcal{N} et de dérivée lipschitzienne sur \mathcal{N} . Alors, on peut trouver une suite minimisante $\{x_k\} \subset \mathcal{N}$ de f telle que $f'(x_k) \rightarrow 0$.

DÉMONSTRATION. Soit $\{x'_k\}$ une suite minimisante de f incluse dans \mathcal{N} . On construit $\{x_k\} \subset \mathcal{N}$ comme suit. On prend $x_1 = x'_1$. Soit alors x_k donné ($k \geq 1$). Si $\nabla f(x_k) = 0$, on prend $x_{k+1} = x'_{k+1}$. Sinon on calcule d'abord $x''_{k+1} = x_k - \alpha_k \nabla f(x_k)$, où le pas $\alpha_k > 0$ est déterminé par recherche linéaire de Wolfe (par exemple); et on prend pour x_{k+1} le point x'_{k+1} ou x''_{k+1} donnant la plus petite valeur de f . Montrons qu'une sous-suite de la suite $\{x_k\}$ ainsi construite convient.

Observons que, comme $f(x_k) \leq f(x'_k)$ pour tout k , la suite $\{x_k\}$ est aussi minimisante. Par ailleurs, si, pour une sous-suite d'indices k , on a $\nabla f(x_k) = 0$, le résultat est démontré en sélectionnant cette sous-suite. Sinon, pour des indices assez grands, on a $f(x_{k+1}) \leq f(x''_{k+1}) \leq f(x_k) - \omega_1\alpha_k\|\nabla f(x_k)\|^2$ et $\langle \nabla f(x''_{k+1}), -\nabla f(x_k) \rangle \geq \omega_2 \langle \nabla f(x_k), -\nabla f(x_k) \rangle$. En raisonnant comme dans la démonstration de la proposition 6.11, on montre que $\alpha_k \geq (1-\omega_2)/L > 0$, où L est la constante de Lipschitz de f' . Dès lors $\nabla f(x_k) \rightarrow 0$. \square

6.5 Propriétés asymptotiques

6.5.1 Admissibilité asymptotique du pas unité

On s'intéresse à présent à la mise en évidence d'hypothèses assurant que le pas unité $\alpha_k = 1$ est accepté asymptotiquement par les différentes règles de recherche linéaire présentées dans la section 6.3. Par exemple, lorsqu'on considère la règle de

décroissance suffisante (6.9), on cherche à savoir quand est-ce que l'on peut garantir que l'on a

$$f(x_k + d_k) \leq f(x_k) + \omega_1 \langle \nabla f(x_k), d_k \rangle,$$

pour k suffisamment grand et lorsque x_k converge vers un minimum x_* de f . On cherche des conditions sur le type de minimum x_* , sur d_k et sur les constantes intervenant dans l'inégalité à vérifier (ici ω_1). Cette propriété est très souhaitable, car elle permet de dire que le pas unité a quelques chances d'être accepté et que c'est donc le premier pas à essayer dans la recherche linéaire. En pratique, essayer en premier lieu le pas unité permet d'éviter de faire trop d'évaluations de fonction pendant la recherche linéaire. Nous allons montrer qu'il en est ainsi dans le voisinage d'un minimum fort (c.-à-d., vérifiant les conditions d'optimalité du second ordre de la proposition 4.9).

Sans perte de généralité, on peut supposer que la direction de descente est obtenue comme solution du système linéaire

$$M_k d_k = -\nabla f(x_k), \quad (6.23)$$

où M_k est une matrice d'ordre n auto-adjointe inversible. La propriété d'acceptation asymptotique du pas unité est d'ailleurs typique des algorithmes de Newton (chapitre 9), pour lesquels $M_k = \nabla^2 f(x_k)$, et de quasi-Newton (chapitre 10). Une partie des conditions d'acceptation asymptotique du pas unité porte sur la matrice M_k qui doit satisfaire l'estimation (6.24) ci-dessous. Celle-ci est du type $t_k \geq o(\|u_k\|)$, ce qui veut dire qu'il doit exister une suite de réels $\{s_k\}$ tels que $t_k \geq s_k$ et $s_k/\|u_k\| \rightarrow 0$ lorsque $k \rightarrow \infty$. Dès lors, (6.24) est équivalente à $\langle M_k d_k, d_k \rangle \geq \langle \nabla^2 f(x_*) d_k, d_k \rangle + o(\|d_k\|^2)$, qui est vérifiée si les matrices M_k sont assez « grosses ». Cela n'a rien d'étonnant puisqu'alors les directions d_k sont petites (d'après (6.23)) et le pas unité le long de ces directions a en effet plus de chance d'être accepté. D'autre part, (6.24) est clairement vérifiée pour $M_k = \nabla^2 f(x_k)$, donc dans l'algorithme de Newton.

Proposition 6.13 (admissibilité asymptotique du pas unité par l'inégalité de décroissance suffisante) *Soit f une fonction de classe C^2 dans le voisinage d'un point x_* , minimum local fort du problème (6.1). On note $\nabla f(x)$ et $\nabla^2 f(x)$ le gradient et le hessien de f pour un même produit scalaire. Soit $\{x_k\}$ une suite générée par la récurrence $x_{k+1} = x_k + d_k$, où d_k est solution de (6.23). On suppose que $\{x_k\}$ converge vers x_* et que les matrices M_k sont auto-adjointes inversibles et vérifient la condition*

$$\langle (M_k - \nabla^2 f(x_*)) d_k, d_k \rangle \geq o(\|d_k\|^2), \quad \text{lorsque } k \rightarrow \infty. \quad (6.24)$$

Alors, pour k suffisamment grand, on a

$$f(x_k + d_k) \leq f(x_k) + \omega_1 \langle \nabla f(x_k), d_k \rangle,$$

pourvu que $\omega_1 \in]0, \frac{1}{2}[$.

DÉMONSTRATION. On commence par développer $f(x_k + d_k)$ autour de x_k au second ordre :

$$f(x_k + d_k) = f(x_k) + \langle \nabla f(x_k), d_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k) d_k, d_k \rangle + o(\|d_k\|^2).$$

Alors, en utilisant (6.23) et (6.24), on trouve

$$\begin{aligned} & f(x_k + d_k) - f(x_k) - \omega_1 \langle \nabla f(x_k), d_k \rangle \\ &= (1 - \omega_1) \langle \nabla f(x_k), d_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k) d_k, d_k \rangle + o(\|d_k\|^2) \\ &= \left(\frac{1}{2} - \omega_1 \right) \langle \nabla f(x_k), d_k \rangle - \frac{1}{2} \langle (M_k - \nabla^2 f(x_k)) d_k, d_k \rangle + o(\|d_k\|^2) \\ &\leq \left(\frac{1}{2} - \omega_1 \right) \langle \nabla f(x_k), d_k \rangle + o(\|d_k\|^2). \end{aligned}$$

Le résultat sera démontré si l'on montre qu'il existe une constante $C > 0$ telle que $\langle \nabla f(x_k), d_k \rangle = -\langle M_k d_k, d_k \rangle \leq -C\|d_k\|^2$ pour k grand (alors le dernier membre ci-dessus est bien négatif pour k grand). Or ceci découle de (6.24) qui entraîne

$$\langle M_k d_k, d_k \rangle \geq \langle \nabla^2 f(x_*) d_k, d_k \rangle + o(\|d_k\|^2) \geq C\|d_k\|^2,$$

puisque les conditions du deuxième ordre sont supposées être vérifiées en x_* . \square

6.5.2 Conditions de convergence superlinéaire ▲

6.6 Algorithme du gradient ▲

On ne le répétera sans doute jamais assez : *l'algorithme du gradient est une très mauvaise méthode* ! Il est toujours préférable d'utiliser l'algorithme de BFGS à mémoire limitée (section 10.2.4), qui a un champ d'application à peu près identique, oracle (= simulateur) et espace mémoire semblables, et n'est guère plus difficile à implémenter. Il en existe d'ailleurs de bonnes réalisations en distribution libre; voir la section *Logiciels* du chapitre 10.

Le lemme suivant est utilisé pour étudier la vitesse de convergence de la méthode du gradient.

Lemme 6.14 (inégalité de Kantorovitch [313]) Soient M une matrice d'ordre n symétrique définie positive et $v \in \mathbb{R}^n$ de norme 1. Alors

$$(v^\top M v)(v^\top M^{-1} v) \leq \frac{(\lambda_{\max} + \lambda_{\min})^2}{4\lambda_{\max}\lambda_{\min}},$$

où λ_{\max} et λ_{\min} sont les valeurs propres maximale et minimale de M .

DÉMONSTRATION. \square

Proposition 6.15 *Soit f une fonction quadratique strictement convexe de hessien H (défini positif). Utilisée pour minimiser cette fonction, la méthode du gradient à pas optimal génère une suite $\{x_k\}$ convergeant q -linéairement vers l'unique minimum x_* de f . Plus précisément,*

$$\frac{\|x_{k+1} - x_*\|_H}{\|x_k - x_*\|_H} \leq \frac{\kappa - 1}{\kappa + 1}, \quad \text{pour } k \geq 1,$$

où $\|\cdot\|_H = (\cdot^\top H \cdot)^{1/2}$ est la norme associée à H et κ est le conditionnement de H .

DÉMONSTRATION. □

6.7 Algorithme proximal

6.7.1 Définition

Soit \mathbb{E} un espace euclidien (produit scalaire $\langle \cdot, \cdot \rangle$ et norme associée $\|\cdot\|$). À un opérateur auto-adjoint défini positif M , on associe le produit scalaire

$$\langle \cdot, \cdot \rangle_M : (u, v) \in \mathbb{E} \times \mathbb{E} \mapsto \langle u, v \rangle_M := \langle Mu, v \rangle$$

et la norme

$$\|\cdot\|_M : u \in \mathbb{E} \mapsto \|u\|_M := \langle u, u \rangle_M^{1/2}.$$

On s'intéresse dans cette section à la minimisation d'une fonction $f \in \text{Conv}(\mathbb{E})$ (c.-à-d., propre, convexe et fermée), qui n'est pas nécessairement différentiable. Soit $\{M_k\}$ une suite d'opérateurs auto-adjoints définis positifs, qui seront utilisés comme *préconditionneurs* de l'algorithme proximal présenté ci-après, et peuvent être générés au fur et à mesure que l'algorithme progresse. Rappelons que le **point proximal** d'un point $x \in \mathbb{E}$, associé à la fonction f et à l'opérateur M_k , est le point noté et défini par

$$P_k(x) := \arg \min_{y \in \mathbb{E}} \left(f(y) + \frac{1}{2} \|y - x\|_{R_k^{-1}}^2 \right). \quad (6.25)$$

Cette notion a été introduite à la section 3.6.1. Nous ferons évidemment souvent référence aux résultats de cette section. Par la condition nécessaire et suffisante d'optimalité du problème dans (6.25), on a

$$x_p = P_k(x) \iff \exists g_p \in \partial f(x_p) : x_p = x - R_k g_p.$$

où le sous-différentiel $\partial f(x)$ est calculé pour le produit scalaire originel $\langle \cdot, \cdot \rangle$.

L'*algorithme proximal* génère une suite $\{x_k\}$ par la formule

$$x_{k+1} := P_k(x_k) = x_k - R_k g_{k+1}, \quad \text{où } g_{k+1} \in \partial f(x_{k+1}). \quad (6.26)$$

On peut interpréter cet algorithme de diverses manières.

- 1) *C'est une méthode de sous-gradient implicite pour minimiser f .*

Le sous-gradient g_{k+1} est en effet évalué en x_{k+1} (qui est inconnu), plutôt qu'en x_k (dans l'algorithme du gradient de la section 6.6). Donc l'équation (6.26) est non linéaire en x_{k+1} , si bien que le nouvel itéré x_{k+1} s'obtiendra en général par un procédé itératif, par exemple celui qui consiste à résoudre itérativement le problème d'optimisation dans (6.25).

- 2) *C'est un algorithme à direction de descente sur f avec pas unité.*

La direction $-R_k g_{k+1}$ est en effet une direction de descente de f en x_k parce que $R_k g_{k+1}$ est une direction de montée en x_{k+1} et que f est convexe. On a en effet

$$f'(x_{k+1}; R_k g_{k+1}) \geq \langle g_{k+1}, R_k g_{k+1} \rangle \geq 0, \quad (6.27)$$

où la première inégalité vient du fait que $g_{k+1} \in \partial f(x_{k+1})$ (point (i) de la proposition 3.53). Puis, par le point (iv) de la proposition 3.17 et (6.26) :

$$f'(x_k; x_{k+1} - x_k) \leq -f'(x_{k+1}; x_k - x_{k+1}) = -f'(x_{k+1}; R_k g_{k+1}) \leq 0.$$

On observe aussi que l'algorithme s'impose systématiquement un pas unité le long de la direction $-R_k g_{k+1}$. On constate aussi que ce pas fait décroître f à chaque itération, puisqu'en prenant $y = x_k$ dans (6.25), on obtient

$$f(x_{k+1}) \leq f(x_{k+1}) + \frac{1}{2} \|x_{k+1} - x_k\|_{R_k^{-1}}^2 \leq f(x_k). \quad (6.28)$$

- 3) *C'est l'algorithme du gradient, pour le produit scalaire $\langle \cdot, \cdot \rangle_{R_k^{-1}}$, avec pas unité, pour minimiser la régularisée de Moreau-Yosida de f , à savoir la fonction*

$$\tilde{f} : x \in \mathbb{E} \mapsto \tilde{f}(x) := \inf_{y \in \mathbb{E}} \left(f(y) + \frac{1}{2} \|y - x\|_{R_k^{-1}}^2 \right).$$

En effet, d'après la proposition 3.75, le gradient de \tilde{f} pour le produit scalaire $\langle \cdot, \cdot \rangle$ s'écrit

$$\nabla \tilde{f}(x_k) = R_k^{-1}(x_k - x_{k+1}) = g_{k+1}.$$

Dès lors $R_k g_{k+1}$ est le gradient de \tilde{f} en x_{k+1} pour le produit scalaire $\langle \cdot, \cdot \rangle_{R_k^{-1}}$.

Notons que les itérés générés font aussi décroître \tilde{f} de façon monotone. En effet, (6.28) se réécrit $f(x_{k+1}) \leq \tilde{f}(x_k) \leq f(x_k)$, si bien que l'on a

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

Voilà un algorithme bien étrange, puisque pour minimiser f , il faut qu'à chaque itération, l'algorithme proximal minimise la fonction

$$f^k : y \in \mathbb{E} \mapsto f^k(y) := f(y) + \frac{1}{2} \|y - x_k\|_{R_k^{-1}}^2,$$

qui semble bien être aussi compliquée que f . Ce point de vue doit être relativisé au vu des remarques suivantes.

- 1) La fonction f^k à minimiser (à chaque itération, ne l'oublions pas) peut être plus attrayante que f , du fait de sa *forte convexité*. Pour certains algorithmes, cette propriété est une aubaine, permettant d'accélérer leur convergence et de mieux la contrôler. Cette fonction a aussi un minimum unique, ce qui n'est pas nécessairement le cas de f .
- 2) Comme nous le verrons au chapitre 14, certains algorithmes de dualité peuvent être *interprétés* comme des algorithmes proximaux. Ces algorithmes de dualité s'appliquent en fait à des fonctions f dont l'évaluation résulte de la minimisation d'une fonction (le lagrangien) et il n'est pas plus coûteux d'évaluer f que de minimiser f^k (pourvu que $R_k \succ 0$), qui revient à minimiser une autre fonction (le lagrangien augmenté). Ces algorithmes de dualité ont donc tout leur sens. Leur interprétation en termes d'algorithme proximal permet alors d'en obtenir des propriétés difficiles à mettre en évidence autrement.
- 3) Observons que si f est *séparable*, c'est-à-dire si elle s'écrit comme suit

$$f(x) = \sum_{j=1}^N f_j(x_{D_j}),$$

où les D_j sont de *petites* parties disjointes de l'ensemble des indices $\llbracket 1, n \rrbracket$, il en est de même de f^k (pourvu que R_k soit diagonale par blocs). C'est une propriété intéressante lorsqu'on cherche à résoudre de grands problèmes par des techniques de décomposition.

- 4) L'algorithme proximal a un *effet stabilisant*. Lorsque f a plusieurs minimiseurs, l'algorithme génère une suite convergeant vers l'un d'entre eux. L'algorithme proximal est parfois utilisé pour stabiliser des algorithmes qui ne convergeraient pas sans modification lorsque le problème considéré devient singulier.

6.7.2 Convergence

Les propositions 6.16 et 6.17 ci-dessous explorent quelques propriétés de convergence de l'algorithme proximal, dans des situations de plus en plus restrictives.

Un des rôles de l'opérateur R_k dans l'itération de l'algorithme proximal peut se comprendre en examinant l'influence de son ordre de grandeur dans le problème (6.25) dont x_{k+1} est la solution. Si R_k est *très petit*, la pénalité introduite par le terme quadratique est *très grande*, si bien que le déplacement $x_{k+1} - x_k$ peut devenir *très petit* au point d'empêcher le progrès vers une solution (de ce point de vue, l'opérateur $R_k = +\infty I$ conviendrait parfaitement, puisqu'il permettrait de résoudre le problème de minimisation en une seule itération). Comme nous le verrons dans la démonstration de la proposition 6.16 ci-dessous, la condition suivante

$$\sum_{k \geq 0} \lambda_{\min}(R_k) = +\infty \quad (6.29)$$

est suffisante.

L'algorithme proximal à métrique variable R_k est difficile à analyser du fait de la modification de R_k à chaque itération, qui empêche d'utiliser les relations de monotonie issues de la formule (3.60), permettant de comparer deux itérations successives

ou les itérés de l'itération courante à une solution du problème (un argument classique du cas où $R_k = r_k I$). Le résultat de convergence de la proposition 6.16 repose alors sur la monotonie suivante, observée au point 2 de la page 265 : l'algorithme proximal fait décroître la fonction f à chaque itération.

On note $\tilde{f}(x_k)$ la valeur optimale dans (6.25), qui est la valeur de la *régularisée de Moreau-Yosida* en x_k . Comme x_k minimise f si, et seulement si, $\tilde{f}(x_k) = f(x_k)$, il est naturel de s'intéresser à l'écart entre ces deux valeurs, que l'on note

$$\begin{aligned} \delta_k &:= f(x_k) - \tilde{f}(x_k) \\ &= f(x_k) - f(x_{k+1}) - \frac{1}{2} \|x_{k+1} - x_k\|_{R_k^{-1}}^2 \quad [\text{définition de } \tilde{f}(x_k)] \\ &= f(x_k) - f(x_{k+1}) - \frac{1}{2} \langle R_k g_{k+1}, g_{k+1} \rangle \quad [(6.26)]. \\ &\geq 0 \quad [(6.28)]. \end{aligned} \tag{6.30}$$

On peut obtenir mieux que cette inégalité en utilisant l'inégalité de convexité $f(x_{k+1}) + f'(x_{k+1}; x_k - x_{k+1}) \leq f(x_k)$ et (6.27), à savoir

$$f(x_{k+1}) + \langle R_k g_{k+1}, g_{k+1} \rangle \leq f(x_k). \tag{6.31}$$

Proposition 6.16 (algorithme proximal, R_k général) Soient $\{x_k\}$ la suite générée par l'algorithme proximal et $\{g_k\}$ la suite des sous-gradients qui interviennent dans (6.26).

1) La suite $\{f(x_k)\}$ décroît vers une valeur $f_* \in \mathbb{R} \cup \{-\infty\}$.

2) Si $f_* > -\infty$, alors $\sum_k \delta_k < +\infty$.

On suppose désormais que (6.29) a lieu.

3) Si $f_* > -\infty$, alors zéro est point d'adhérence de la suite $\{g_k\}$.

4) Si $\{x_k\}$ est bornée, alors les points d'adhérence de $\{x_k\}$ sont des minima de f (en particulier, f a un minimum).

DÉMONSTRATION. 1) Le fait que $\{f(x_k)\}$ décroisse a été observé en (6.28) ou (6.31). Comme les $f(x_k) \in \mathbb{R}$, $f(x_k)$ décroît vers une limite f_* dans $\mathbb{R} \cup \{-\infty\}$.

2) En sommant les égalités (6.30), on obtient

$$\sum_{k \geq 0} \delta_k + \frac{1}{2} \sum_{k \geq 0} \langle R_k g_{k+1}, g_{k+1} \rangle = f(x_0) - f_* < \infty.$$

Comme la seconde série est positive, on en déduit le résultat.

3) Grâce à l'identité précédente, on a aussi

$$\frac{1}{2} \sum_{k \geq 0} \lambda_{\min}(R_k) \|g_{k+1}\|^2 < +\infty.$$

On en déduit qu'il existe une sous-suite de $\{g_k\}$ qui tend vers zéro, sinon il existerait une constante $\gamma > 0$ tel que $\|g_k\| \geq \gamma$ et l'inégalité ci-dessus contredirait (6.29).

4) Si $\{x_k\}$ est bornée, alors $f_* > -\infty$, car $f(x_k) \rightarrow f_*$ et f a une **minorante affine** (proposition 3.5). Il suffit de montrer que f_* est la valeur minimale de f , puisque $f(x_k) \rightarrow f_*$ (point 1) et qu'alors un point d'adhérence x_* de $\{x_k\}$ sera tel que $f(x_*) = f_*$ (f est fermée), ce qui implique que x_* minimise f . Par le point 3, il existe une sous-suite d'indices \mathcal{K} telle que $\{g_k\}_{k \in \mathcal{K}} \rightarrow 0$. Comme $\{x_k\}$ est bornée, il existe une sous-suite d'indices $\mathcal{K}' \subset \mathcal{K}$ telle que $\{x_k\}_{k \in \mathcal{K}'} \rightarrow x_*$. On peut alors passer à la limite dans $g_k \in \partial f(x_k)$ (résultat de (6.26)) pour obtenir $0 \in \partial f(x_*)$ (point (iii) de la proposition 3.63), c'est-à-dire que x_* minimise f et donc que $f_* = \inf f$. \square

L'algorithme proximal a davantage de propriétés lorsque $R_k = r_k R$, où r_k est un scalaire strictement positif et R est un opérateur auto-adjoint défini positif fixé. Le cas où $R_k = r_k I$ est souvent rencontré. Dans ce cas, la condition de convergence (6.29) devient la condition sur les r_k suivante :

$$\sum_{k \geq 0} r_k = +\infty. \quad (6.32)$$

Comme exemple de propriété supplémentaire : la suite $\{x_k\}$ est nécessairement minimisante, même si elle n'est pas bornée (voir le point 1 ci-dessous).

Proposition 6.17 (algorithme proximal, $R_k = r_k R$) Soient $\{x_k\}$ la suite générée par l'algorithme proximal avec $R_k = r_k R$, où r_k est un scalaire strictement positif et R est un opérateur auto-adjoint défini positif, et $\{g_k\}$ la suite des sous-gradients qui interviennent dans (6.26).

- 1) Si (6.32) a lieu, la suite $\{f(x_k)\}$ décroît vers la valeur minimale de f .
- 2) Si, de plus, $\{r_k\}$ est bornée et f a un minimiseur, alors $\{x_k\}$ converge vers un minimiseur de f .

DÉMONSTRATION. 1) L'argument consiste à examiner comment $\{x_k\}$ dévie d'une **suite de Fejér**, la déviation étant mesurée au moyen de la fonction f . On regarde donc comment évolue l'écart $x_k - x$ en norme R^{-1} , pour un $x \in \mathbb{E}$ pour l'instant arbitraire :

$$\|x_{k+1} - x\|_{R^{-1}}^2 = \|x_{k+1} - x_k\|_{R^{-1}}^2 + 2\langle x_{k+1} - x_k, x_k - x \rangle_{R^{-1}} + \|x_k - x\|_{R^{-1}}^2. \quad (6.33)$$

L'observation-clé est maintenant de montrer que les deux premiers termes (qui font éventuellement dévier $\{x_k\}$ d'une suite de Fejér) peuvent s'estimer par une variation de f . En effet,

$$\begin{aligned} f(x) &\geq f(x_{k+1}) + \langle g_{k+1}, x - x_{k+1} \rangle && \text{[inégalité de convexité]} \\ &= f(x_{k+1}) + \langle R_k^{-1}(x_k - x_{k+1}), x - x_{k+1} \rangle && [(6.26)] \\ &= f(x_{k+1}) + \langle R_k^{-1}(x_k - x_{k+1}), x - x_k \rangle + \langle R_k^{-1}(x_k - x_{k+1}), x_k - x_{k+1} \rangle. \end{aligned}$$

On fait maintenant apparaître δ_k donné par (6.30) :

$$f(x) \geq f(x_k) + \frac{1}{r_k} \langle x_k - x_{k+1}, x - x_k \rangle_{R^{-1}} + \frac{1}{2r_k} \|x_k - x_{k+1}\|_{R^{-1}}^2 - \delta_k,$$

ce qui donne comme majoration des deux premiers termes de (6.33)

$$\|x_{k+1} - x_k\|_{R^{-1}}^2 + 2\langle x_{k+1} - x_k, x_k - x \rangle_{R^{-1}} \leq 2r_k [f(x) - f(x_k) + \delta_k].$$

L'égalité (6.33) devient alors

$$\|x_{k+1} - x\|_{R^{-1}}^2 \leq \|x_k - x\|_{R^{-1}}^2 + 2r_k [f(x) - f(x_k) + \delta_k]. \quad (6.34)$$

Choisissons maintenant $x \in \mathbb{E}$. On sait d'après le point (i) de la proposition 6.16, que $f(x_k)$ décroît vers une limite f_* . Si $f_* = -\infty$, le résultat est démontré. Supposons maintenant que $f_* > -\infty$. Si $f_* \neq \inf f$, on peut trouver un $x \in \mathbb{E}$ et un $\eta > 0$ tel que $f(x) + \eta \leq f(x_k)$ pour tout indice $k \geq 0$. Alors l'inégalité précédente devient

$$\forall k \geq 0 : \quad \|x_{k+1} - x\|_{R^{-1}}^2 \leq \|x_k - x\|_{R^{-1}}^2 + 2r_k(\delta_k - \eta).$$

Mais $\delta_k \rightarrow 0$ par le point (ii) de la proposition 6.16, si bien que pour k_1 assez grand :

$$\forall k \geq k_1 : \quad \|x_{k+1} - x\|_{R^{-1}}^2 \leq \|x_k - x\|_{R^{-1}}^2 - \eta r_k.$$

En sommant ces inégalités de k_1 à $k_2 \geq k_1$, on obtient

$$0 \leq \|x_{k_2+1} - x\|_{R^{-1}}^2 \leq \|x_{k_1} - x\|_{R^{-1}}^2 - \eta \sum_{k=k_1}^{k_2} r_k,$$

si bien que la série $\sum_k r_k$ serait convergente, en contradiction avec l'hypothèse (6.32).

2) Soit \bar{x} un minimiseur de f . En prenant $x = \bar{x}$ dans (6.34), on obtient

$$\forall k \geq 0 : \quad \|x_{k+1} - \bar{x}\|_{R^{-1}}^2 \leq \|x_k - \bar{x}\|_{R^{-1}}^2 + 2r_k \delta_k. \quad (6.35)$$

Si $\{r_k\}$ est bornée, la série $\sum_k r_k \delta_k$ converge et, en sommant les inégalités précédentes, on voit que $\{x_k\}$ est bornée.

Soit \bar{x} un point d'adhérence de $\{x_k\}$, qui est nécessairement un minimiseur de f (car $f(x_k) \rightarrow \inf f$, par le point 1, et f est s.c.i.). Montrons que toute la suite $\{x_k\}$ converge vers \bar{x} . Soit $\varepsilon > 0$. Il suffit de montrer que $\|x_k - \bar{x}\|_{R^{-1}}^2 \leq \varepsilon$ pour k assez grand. Par définition de \bar{x} , on peut trouver un indice k_1 tel que

$$\|x_{k_1} - \bar{x}\|_{R^{-1}}^2 \leq \frac{\varepsilon}{2} \quad \text{et} \quad 2 \sum_{k \geq k_1} r_k \delta_k \leq \frac{\varepsilon}{2}.$$

En sommant les inégalités (6.35), on trouve que $\|x_k - \bar{x}\|_{R^{-1}}^2 \leq \varepsilon$ pour tout $k \geq k_1$. \square

6.7.3 Versions approchées ▲

Voir Rockafellar [454; 1976], Qi et Chen [436; 1997].

Notes

Pour les règles de recherche linéaire présentées dans ce chapitre, on pourra consulter les travaux originaux de Curry [133 ; 1944], d'Armijo [21 ; 1966], de Wolfe [530, 531 ; 1969-1971], de Zoutendijk [548 ; 1970], de Fletcher [188 ; 1980] et de Lemaréchal [350 ; 1981]. On trouvera une présentation générale des techniques de détermination de pas par encadrement au chapitre 3 de [69].

Si l'on parvient à faire tendre le gradient vers zéro sans trop de difficulté, en général, on n'a pas pour autant nécessairement la convergence *des itérés* vers un minimum de la fonction, même avec l'algorithme du gradient et une recherche linéaire raisonnable. Un contre-exemple est donné par Gonzaga [243 ; 2000].

La suite minimisante spéciale du lemme 6.12 peut aussi être obtenue en utilisant le principe variationnel d'Ekeland [173 ; 1974], au lieu de la recherche linéaire comme dans la démonstration proposée ici (voir Borwein et Zhu [74 ; 2010, lemme 2.4.2]).

De manière à accepter plus rapidement le pas le long de la direction de recherche d_k , il est parfois intéressant de faire ce que l'on appelle de la *recherche linéaire non monotone* : on remplace la condition de descente suffisante (6.9) par

$$f(x_{k+1}) \leq \left(\max_{0 \leq j \leq m(k)} f(x_{k-j}) \right) + \omega_1 \alpha_k \langle g_k, d_k \rangle,$$

qui est donc plus facilement vérifiée que (6.9). On doit imposer des conditions sur le décalage d'indice maximal $m(k)$ pour préserver la convergence : $m(1) = 0$, $0 \leq m(k) \leq \min[m(k-1) + 1, M]$. En pratique, $M \in \mathbb{N}$ est une petite constante entière, souvent prise égale à 3. Donc $\{f(x_k)\}$ n'est plus nécessairement décroissante (c'est dans ce sens que la recherche linéaire est dite « non monotone »). Grippo, Lampariello et Lucidi [265 ; 1986] utilisent cette RL avec l'algorithme de Newton, Lucidi et Roma [358 ; 1994] avec le gradient conjugué non linéaire.

Certains algorithmes, à l'utilité pratique discutable, déterminent un pas le long d'une direction de descente, sans chercher à faire décroître le critère f , mais en utilisant des formules déterminées par des considérations diverses. Ainsi, la règle de Barzilai-Borwein [36] utilise une estimation de la courbure de f suivant la direction de descente; les règles de [334, 2, 433, 434] utilisent des propriétés de minimalité le long de d_k (minimalité de la norme du gradient ou d'une combinaison convexe de celle-ci et du critère) et permettent d'étendre les résultats de convergence de la méthode du gradient de la section 6.6. On ne peut démontrer des résultats de convergence que sous des hypothèses restrictives.

L'idée d'utiliser la régularisation de Moreau-Yosida, comme dans l'algorithme proximal, pour la résolution de problèmes d'optimisation mal conditionnés remonte au moins à Bellman, Kalaba et Lockett [39 ; 1966, chapitre V]. Ce principe fut généralisé par Martinet [365 ; 1970] à la minimisation de fonctions convexes et par Rockafellar [453, 454 ; 1976] à la recherche de zéros d'opérateurs monotones maximaux ainsi qu'à l'optimisation convexe. L'influence des *préconditionneurs* R_k est analysée, par exemple, par Qi et Chen [436 ; 1997], Chen et Fukushima [110 ; 1999]. On pourra aussi consulter [120]. La section 6.7.2 s'inspire largement de [288 ; section XV.4.2].

Exercices

- 6.1.** *Direction de descente.* Les dessins de la figure 6.5 représentent une courbe de niveau (ou iso-valeur) d'une fonction quadratique $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ (f est constante sur cette courbe). Dans quels dessins (A ou B ou les deux) la direction d est-elle de descente au point x ?

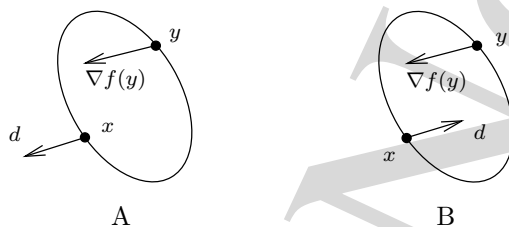


Fig. 6.5. Directions de descente ?

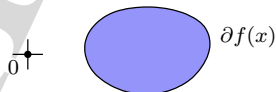
- 6.2.** *Exemples de direction de descente.*

- 1) Soient $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction différentiable, $x \in \mathbb{R}^n$ et $\nabla f(x)$ le gradient de f en x pour un produit scalaire $\langle \cdot, \cdot \rangle$ (on note $\| \cdot \|$ la norme associée). Soit enfin $d \in \mathbb{R}^n$ une direction *non nulle* telle que

$$\|\nabla f(x) + d\| \leq \|\nabla f(x)\|.$$

Montrez que d est une direction de descente de f en x .

- 2) Soient $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction convexe différentiable et x et y deux points de \mathbb{R}^n tels que $f(y) < f(x)$. Montrez que $y - x$ est une direction de descente de f en x .
- 3) Soient $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ une fonction convexe et $D := \{d \in \mathbb{R}^2 : f'(x; d) \leq 0\}$ l'ensemble des directions de descente au sens large (il n'y a pas d'inégalité stricte) de f en un point $x \in \mathbb{R}^2$. On a représenté ci-dessous le sous-différentiel $\partial f(x)$ de f en x pour le produit scalaire euclidien et l'origine 0. Déterminez graphiquement l'ensemble D .



- 4) Soient $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$ une fonction différentiable et $f : \mathbb{R}^n \rightarrow \mathbb{R}$ la fonction définie par $f(x) = \|F(x)\|$, où $\| \cdot \|$ est une norme sur \mathbb{R}^m . Soit $d \in \mathbb{R}^n$ une direction telle que

$$\|F(x) + F'(x) \cdot d\| < \|F(x)\|.$$

Montrez que f a des dérivées directionnelles et que d est une direction de descente de f en x , dans le sens où $f'(x; d) < 0$.

- 6.3.** *Détermination d'un pas de Goldstein.* Donnez un algorithme déterminant un pas de Goldstein (vérifiant donc (6.9) et (6.10)) en un nombre fini d'étapes sous des conditions semblables à celles de la proposition 6.7 (dans ce cas, on n'a besoin de la dérivabilité de f qu'en zéro et de sa continuité).
- 6.4.** *Toute direction de descente est un gradient.* Toute direction de descente d'une fonction $f : \mathbb{E} \rightarrow \mathbb{R}$ en x , définie sur un espace vectoriel \mathbb{E} , est l'opposé du gradient de f en x pour un certain produit scalaire sur \mathbb{E} .

6.5. *Terminaison finie de l'algorithme proximal.* Soient $f \in \overline{\text{Conv}}(\mathbb{E})$ et \tilde{f} sa régularisée de Moreau-Yosida (section 3.6.2). On suppose que $\bar{x} \in (\text{dom } f)^*$ minimise f et que $0 \in \text{int } \partial f(\bar{x})$ (de manière plus précise : il existe $\varepsilon > 0$ tel que $\bar{B}(0, \varepsilon) \subset \partial f(\bar{x})$). On note x_p le point proximal de $x \in \mathbb{E}$. Montrez que

- (i) $g \in \partial f(x)$ et $\|g\| < \varepsilon \implies x = \bar{x}$,
- (ii) $\|x - \bar{x}\| \leq \varepsilon \implies x_p = \bar{x}$,
- (iii) f est quadratique sur $\bar{B}(\bar{x}, \varepsilon)$.