Méthodes numériques et optimisation

Idriss Mazari
Équipe pédagogique : Julien Claisse, Lucas Davron,
Zhenjie Ren

Version du 15 septembre 2025

Ce document est un support de cours ; pour les évaluations, seul le cours professé en amphithéâtre fait foi.

Table des matières

Première partie

Introduction & méthodes unidimensionnelles

Chapitre 1

Introduction

1.1 Quel type de problèmes veut-on résoudre ?

Commençons par le commencement : que veut-on faire dans ce cours ? La réponse tient au fait que, quotidiennement, et quel que soit le secteur d'activités où vous exercerez, vous serez régulièrement confrontées et confrontés à des problèmes d'optimisation de toute espèce. Typiquement :

1. Dans le domaine de l'ingénierie : supposez que vous deviez déterminer la structure d'une poutre pour qu'elle soit aussi légère et résistante que possible, quelle configuration adopteriez-vous ?
2. Dans le domaine de l'acoustique : supposez que vous ayez $1m^2$ de peau pour faire un tambour, quelle forme donneriez-vous à ce tambour pour qu'il sonne le plus grave possible ?
3. Dans le domaine médical : supposez qu'un traitement coûte extrêmement cher, quelle est la meilleure manière de l'administrer au cours du temps pour garantir, d'une part, que le traitement soit efficace, mais d'autre part que le coût de ce traitement soit minimal ?
4. Dans le domaine économique : supposez que vous ayez des marchandises à livrer de vos sites de productions à vos sites de distributions. De quelle manière acheminer vos marchandises afin de garantir un coût de transport minimal ?

Dans toutes ces situations (mais également dans une multitude d'autres cas) l'on se retrouve face à un problème éventuellement compliqué, mais qui s'écrit de manière synthétique sous la forme suivante :

$$\min_{x \in X} F(x)$$

où X est l'espace de vos paramètres (par exemples, toutes les structures de poutre possible, toutes les manières possibles d'administrer le traitement...) et où F est le critère à optimiser (la résistance de la poutre, le coût d'un traitement...). Mais dans ces situations, on fait une hypothèse implicite : que

le modèle qui régit le phénomène est connu (typiquement, nous connaissons les paramètres qui nous permettent de déterminer, si une poutre est conçue d'une certaine manière, la résistance de la structure ainsi obtenue).

Un autre cadre applicatif très intéressant et puissant est celui de l'**identific-ation de modèles**. Dans ce contexte, les données à votre disposition, comme l'objectif à résoudre, sont différents : on vous donne un modèle qui, étant donnés certains paramètres $\alpha, \beta, \ldots$, prédit le comportement d'un système. Évidemment, ce modèle est imparfait, et vous devez ajuster les paramètres $\alpha, \beta, \ldots$ de manière à ce que le modèle rende le plus parfaitement compte de la situation observée dans les faits. Pour ce faire, on vous donne un jeu de données observées, représentées par $y_{obs,1}, y_{obs,2}, \ldots$, et des données $y_{mod,1}(\alpha, \beta, \ldots)$,$y_{mod,2}(\alpha, \beta, \ldots)$ prédites par le modèle. Une manière d'ajuster les paramètres du modèle est alors de résoudre le problème

$$\min_{\alpha, \beta, \ldots} \sum_k \|y_{obs,k} - y_{mod,k}(\alpha, \beta, \ldots)\|^2.$$

1.2 Qu'entend-on par "résolution" ?

Dans un monde parfait, il serait possible, partant d'une fonction F quelconque, de déterminer explicitement la solution d'un problème de la forme

$$\min_{x \in X} F(x),$$

et d'en déterminer la valeur numérique. En d'autres termes, il serait possible d'exhiber un point $x^* \in X$ tel que

$$\forall x \in X, F(x^*) \leq F(x).$$

Dans tous les exemples traités dans ce cours, on supposera que l'espace $X = \mathbb{R}^d$ muni de sa structure euclidienne standard. Un tel point x^* est appelé **minimiseur de** F.

En général, il est délicat de déterminer un tel minimum global et, même quand une solution est explicite mathématiquement, il peut être pertinent d'en fournir une bonne approximation numérique.

1. Tout d'abord, notons que toutes les méthodes que nous présentons ont pour finalité de traiter des problèmes plus ou moins concrets, où les applications numériques sont *in fine* le but ; avoir une expression compacte mathématiquement peut ne pas être très utile. Même si elle est moins précise, une approximation peut être plus efficace et largement suffisante.
2. Regardons un cas particulièrement simple : on considère la fonction

   $$f : x \mapsto |x^2 - a|$$

   où $a > 0$ est un réel positif. Clairement,

   $$\forall x \in \mathbb{R}, f(x) \geq 0 \text{ et } f(x) = 0 \text{ ssi } x = \pm\sqrt{a}.$$

On a donc deux minima globaux, $x_1^* = \sqrt{a}\,, x_2^* = -\sqrt{a}$. Si on prend $a = 2$, on n'est pas beaucoup plus avancé pour les applications numériques, car il est nécessaire de fournir une bonne approximation de $\sqrt{2}$ sous forme décimale. Évidemment, dans la plupart des langages, la commande `sqrt()` fournit automatiquement une approximation, mais si on avait besoin de calculer avec une fonction bien plus compliquée que la racine il faudrait réussir à donner une approximation satisfaisante. On peut généraliser cet exemple, en considérant par exemple, pour une matrice inversible $A \in Gl(d;\mathbb{R})$ et un vecteur $b \in \mathbb{R}^d$, la fonction

$$f : \mathbb{R}^d \ni x \mapsto \|Ax - b\|^2.$$

Par les mêmes arguments, l'unique minimiseur de f est $x^* = A^{-1}b$. Mais en général, il est très pénible d'inverser exactement une matrice. Si l'on réussit à trouver une manière plus rapide pour approcher x^*, jusqu'à une certaine précision, on pourra s'estimer satisfait pour les approximations numériques suivantes.

3. Mais maintenant, observons que les deux exemples traités ci-dessus sont très particuliers : un simple examen de la fonction et une comparaison directe permette d'obtenir la réponse souhaitée. Si l'on se donne une fonction beaucoup plus générale $f : \mathbb{R}^d \ni x \mapsto f(x)$, comme par exemple

$$f : x \mapsto e^{-2\cos(x)^{\sin(x^x)}} \cos\left(\frac{1}{1+x^2}\right) \tag{1.1}$$

il y a plusieurs difficultés à lever :

   a) Déjà, **existe-t-il** un minimiseur x^* ? Cette première question est loin d'être triviale en générale et sera l'object d'une section dédiée de ce cours.
   b) Ensuite, admettons que l'on sache qu'un minimiseur x^* existe. Peut-on le caractériser et, si oui, comment ? Ici, on s'appuie sur les conditions classiques d'optimalité, qui sont des conditions issues du calcul différentiel : dérivée première nulle, dérivée seconde positive... qui sont toutes des caractérisations **locales**.
   c) Enfin, une fois que ces conditions d'optimalité ont permis une liste de candidats potentiels à être des minima, il faut les comparer, parfois un à un.

Réfléchissons un peu sur cette dernière liste de conditions et sur cette description schématique de la résolution d'un problème d'optimisation. L'étape cruciale est la seconde du raisonnement : utiliser des conditions d'optimalité locales. Nous reviendrons évidemment sur le cas des dimensions supérieures, mais il est important de toujours garder en tête le cas uni-dimensionnel. On considère une fonction dérivable $f : \mathbb{R} \to \mathbb{R}$ et on veut minimiser f. Ainsi, on cerche un point $x^* \in \mathbb{R}$ tel que

$$\forall x \in \mathbb{R}\,, f(x^*) \leq f(x). \tag{1.2}$$

Or on sait qu'en un point de minimum x^* de f on a nécessairement

$$f'(x^*) = 0. \tag{1.3}$$

On a gagné un peu d'information en remplaçant l'inégalité (1.2), qui fait intervenir toutes les valeurs de $x \in \mathbb{R}$, par l'égalité (1.3) qui ne fait intervenir que l'inconnue x^*. Mais en général (1.3) peut être tout aussi compliquée à résoudre (essayez par exemple d'écrire ce que serait cette équation pour la fonction f donnée en (1.1)) et, répétons le : **peut-être que résoudre exactement** (1.3) **n'est pas nécessaire pour les applications.** Par exemple, si l'on a une méthode qui nous permette de trouver en un temps raisonnable un point x_{appr} tel que

$$|f'(x_{appr})| \leq \varepsilon \tag{1.4}$$

pour $\varepsilon > 0$ petit, ou tel que

$$\forall x \in \mathbb{R}^d \, , f(x_{appr}) \leq f(x) + \varepsilon' \tag{1.5}$$

pour ε' petit, alors on pourra travailler avec x_{appr}.

Exercice 1.2.1. *Discuter l'équivalence entre* (1.4)-(1.5). *Est-elle vraie (quitte à ajuster les valeurs de* $\varepsilon \, , \varepsilon'$*) ? Si oui, l'est-elle uniquement dans un voisinage de* x_{appr} *?*

Dans tout ce cours on cherchera à trouver exactement ce genre de solution approchée des problèmes d'optimisation considérés. Expliquons le type d'algorithme recherché.

1.3 Les algorithmes itératifs

Ttoutes les méthodes de ce cours sont des méthodes *itératives*. Ces méthodes visent à résoudre, ou bien directement

$$\min_{x \in \mathbb{R}^d} f(x) \tag{1.6}$$

ou bien l'analogue multi-dimensionnel de (1.3), à savoir

$$\nabla f(x) = 0. \tag{1.7}$$

Désignons par x^* la solution de celui de ces deux problèmes qui nous intéresse. Un algorithme itératif prend en entrée une initialisation, c'est-à-dire une première approximation grossière de la solution x^*, disons x_0 et une règle de décision qui indique, un point étant donné, une manière de calculer le point suivant. Ainsi, un agorithme génère **une suite de points** $\{x_k\}_{k \in \mathbb{N}}$ qui devrait satisfaire, si tout se passe bien,

$$x_k \underset{k \to \infty}{\to} x^*. \tag{1.8}$$

Remarque 1.3.1. *On devrait en général prendre garde à la topologie de la convergence* (1.8) *; néanmoins, puisque nous travaillons dans tout ce cours en dimension finie, il n'y a pas lieu de distinguer entre convergence en norme (par ailleurs toutes équivalentes) et convergence faible. Cette remarque qui semble faite pour chipoter prend toute son importance dès que l'on commence à traiter de problèmes d'optimisation en dimension infinie, ce qui relève alors du* calcul des variations*.*

Bien sûr, la vie étant ce qu'elle est, il est illusoire d'espérer qu'au bout d'un nombre d'itération, même très grand, l'on puisse tomber exactement sur une solution x^* du problème considéré. Il est donc nécessaire d'imposer un *critère de tolérance*, qui joue le rôle d'une règle d'arrêt. Pour le problème (1.7), fixons un seuil de tolérance $\texttt{tol} > 0$. On peut considérer que si notre itération produit, pour N assez grand, un point $x_N \in \mathbb{R}^d$ tel que

$$\|f'(x_N)\| \leq \texttt{tol}$$

alors x_N peut être considéré comme une bonne approximation d'une solution x^*. Le cas des problèmes de type (1.6) est un peu plus délicat. Mais somme toute, tous les algorithmes considérés dans ce cours peuvent être mis sous la forme : Un algorithme peut bien se comporter, ou, au contraire, faire n'importe quoi. L'objectif est bien évidemment de construire des algorithmes qui convergent, c'est-à-dire tels que (1.8) soit satisfait. On veut également savoir quelle efficacité cet algorithme peut avoir, ce qui revient à demander à quelle vitesse cet algorithme converge. Dans une section ultérieure, nous définirons les vitesses de convergence qui nous intéresseront pour toute la suite de ce cours.

Algorithm 1 Structure type d'un algorithme itératif

```
def algo(f,x0,...,tol=1e-3,IterMax=10)
Initialisation
xn=x0   On définit le premier élément de la liste
L=[]   On définit une liste que l'on augmentera récursivement, et qui contiendra tous les éléments générés
for n in range(IterMax):
if ...< tol : then
  return xn,L
else
  L.append(xn)
  xn=...
end if
print("Erreur, l'algorithme n'a pas convergé
après",IterMax,"itérations")
```

Nous pouvons désormais passer à la liste des pré-requis et à une présentation du plan du cours.

Chapitre 2

Pré-requis

Nous donnons dans ce chapitre le vocabulaire, et les bases théoriques qui seront pleinement exploités dans toute la suite du semestre.

2.1 Études de suite

Comme annoncé dans l'introduction, tous les algorithmes génèrent des **suites de vecteurs**. Il est donc fondamental d'être à l'aise avec l'étude des suites et leurs propriétés de convergence. En d'autres termes, une suite étant donnée, converge-t-elle ? Et si oui, à quelle vitesse ? Rappelons quelques définitions en précisant tout d'abord que dans tout ce qui suit l'espace euclidien $\mathbb{R}^d$ est muni de sa norme euclidienne :

$$\|x\|^2 = \sum_{i=1}^{d} x_i^2.$$

Définition 2.1.1. *Soit $\{x_k\}_{k\in\mathbb{N}} \in (\mathbb{R}^d)^{\mathbb{N}}$ et $x^* \in \mathbb{R}^d$. On dit que $\{x_k\}_{k\in\mathbb{N}}$ converge vers x^* si*

$$\|x_k - x^*\| \underset{k\to\infty}{\to} 0.$$

Comme toujours, cette définition n'est pas la plus pratique puisqu'elle suppose connue la limite x^* de la suite. Le critère de Cauchy est bien plus manipulable. Rappelons d'abord la définition d'une suite de Cauchy :

Définition 2.1.2. *Une suite $\{x_k\}_{k\in\mathbb{N}} \in (\mathbb{R}^d)^{\mathbb{N}}$ est dite de Cauchy si, pour tout $\varepsilon > 0$, il existe $N \in \mathbb{N}$ tel que, pour tout $p\,, q \geq N$,*

$$\|x_p - x_q\| \leq \varepsilon.$$

Par construction des réels, ou comme conséquence d'une autre construction du corps des réels on a la proposition suivante :

Proposition 2.1.1. *Une suite $\{x_k\}_{k\in\mathbb{N}} \in (\mathbb{R}^d)^{\mathbb{N}}$ converge si, et seulement si, elle est de Cauchy.*

2.1.1 Les exemples standards de suite

Il y a plusieurs exemples de suites à savoir manipuler dans le cas scalaire ; la première feuille de TD est là pour vous rafraîchir la mémoire, mais voici les quelques suites essentielles :

1. Les suites géométriques, de la forme
$$x_{k+1} = qx_k.$$
2. Les suites arithmético-géométriques de la forme
$$x_{k+1} = ax_k + b$$
et, de manière plus générale, les suites de type "point fixe"
$$x_{k+1} = \phi(x_k).$$
3. Les suites de la forme
$$x_k = \phi(k^{-\alpha}),$$
pour une fonction ϕ suffisamment lisse.

2.1.2 Les taux de convergence des suites

Si l'on suppose que la suite générée par nos algorithmes convergent, il convient de spécifier le type de convergence, et de les quantifier.

Une première remarque : si la vitesse de convergence d'un algorithme est la vitesse (définie plus loin) à laquelle la suite d'itérations $\{x_k\}_{k\in\mathbb{N}}$ converge vers x^*, cette vitesse de convergence théorique ne coïncide pas forcément avec la vitesse réelle observée sur l'ordinateur, qui doit également prendre en compte le temps dévaluation de certaines quantités par l'ordinateur. Si vous trouvez par exemple un algorithme tel que la suite $\{x_k\}_{k\in\mathbb{N}}$ satisfasse
$$\|x_k - x^*\| \leq e^{-e^k}$$
donc en sur-exponentiel, mais que passer de x_k à x_{k+1} demande à l'ordinateur d'effectuer $e^{e^{e^k}}$ opérations, l'algorithme risque d'être peu utilisable en pratique.

2.1.2.1 Convergence linéaire

Définition de la convergence linéaire Le premier exemple, le plus fréquemment rencontré, est celui de la **convergence linéaire**.

Définition 2.1.3 (Convergence linéaire)**.** *On dit qu'une suite $\{x_k\}_{k\in\mathbb{N}}$* ***converge linéairement vers*** *x^** ***à taux*** *$\alpha \in (0,1)$ s'il existe une constante $C = C_\alpha > 0$ telle que*
$$\|x_k - x^*\| \leq C_\alpha \alpha^k.$$
Si l'infimum des $\alpha \in (0,1)$ vérifiant cette propriété vérifie également cette propriété, cet infimum est appelé ***taux de convergence optimal de la suite****. Si cet infimum est égal à 0, on parle de* ***convergence sur-linéaire****.*

Interprétation de la définition Moralement, que nous donne cette définition ? Si l'on suppose que $\alpha = \frac{1}{10}$ et que $C_{\frac{1}{10}} = 1$, alors, à chaque étape de l'algorithme, on améliore la précision d'une décimale par itération. En d'autres termes, *un algorithme linéaire gagne un nombre fixe de décimales par itérations.* Notez que cela peut être utilisé, *a priori*, comme un critère d'arrêt : on s'arrête une fois déterminée la solution à dix décimales.

Notons également la chose suivante : soit $\{x_k\}_{k\in\mathbb{N}}$ une suite qui converge vers un certain x^* à un taux optimal $\alpha \in (0,1)$, avec une constante optimale C_α. Alors on a

$$\|x_k - x^*\| \sim C_\alpha \alpha^k.$$

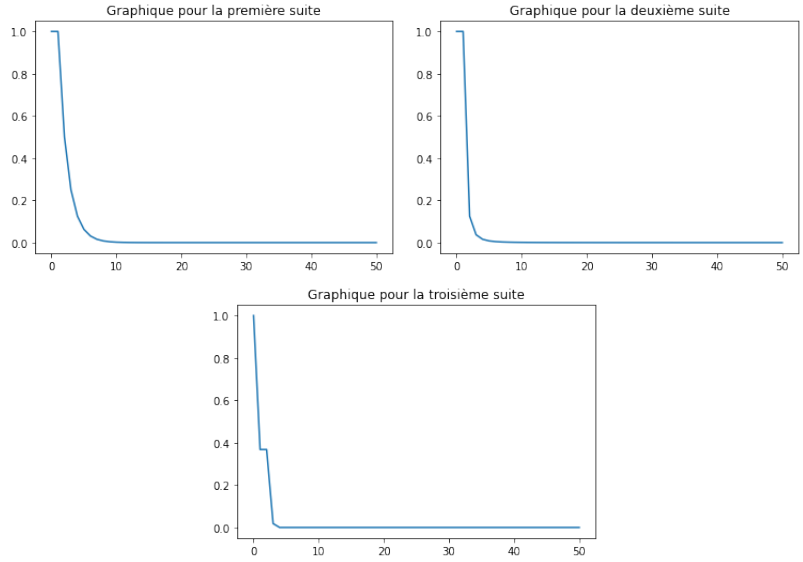
Illustration graphique de la convergence linéaire Maintenant, comment illustrer cette convergence ? Pour illustrer la difficulté considérons les trois suites

$$x_k := 2^{-k}, y_k := \frac{1}{k^3}, z_k = e^{-k^k}.$$

Bien sur, chacune des trois suites converge vers 0 et, intuitivement, la deuxième converge plus lentement que la première, qui converge plus lentement que la troisième. En effet

$$y_k/x_k \underset{k\to\infty}{\to} \infty, x_k/z_k \underset{k\to\infty}{\to} \infty.$$

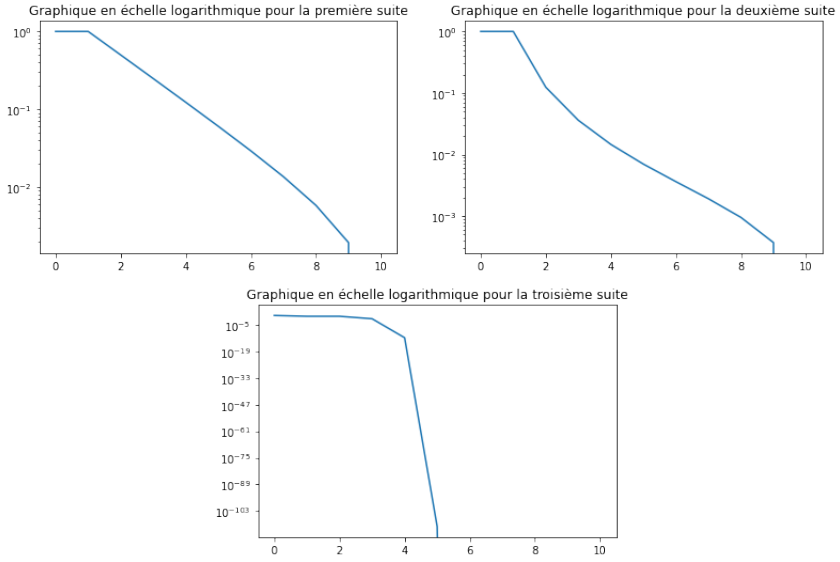
Peut-on observer cela visuellement ? Si on représente les cinquante premiers termes de chacune de ces suites, voici ce que l'on obtient :



La différence n'est pas flagrante. Mais faisons l'observation suivante : si on a une suite $\{x_k\}_{k\in\mathbb{N}}$ qui converge vers x^* à un taux $\alpha \in (0;1)$ optimal, alors on a

$$\ln\left(\|x_k - x^*\|\right) \sim \ln(C_\alpha) + k\ln(\alpha).$$

En particulier, si on représente la suite $\{\ln \|x_k - x^*\|\}_{k\in\mathbb{N}}$ et que l'on observe, asymptotiquement, une droite, on peut lire le taux de convergence sur le coefficient directeur de cette droite. Mais surtout, si l'on a une convergence plus rapide que linéaire, alors, en échelle logarithmique, on observera une courbe **en dessous de toute droite**. On parle de convergence sur-linéaire. De la même manière, si la suite converge plus lentement que linéairement, on observera une courbe **au dessus de toute droite**. On parle de convergence sous-linéaire.



Dit de manière synthétique :

$$\begin{cases} \text{Convergence linéaire } \Rightarrow \text{droite en échelle logarithmique} \\ \text{Convergence sur-linéaire} \Rightarrow \text{ courbe en dessous de toute droite en échelle logarithmique.} \\ \text{Convergence sous-linéaire} \Rightarrow \text{ courbe au dessus de toute droite en échelle logarithmique.} \end{cases}$$

Certes, mais l'on objectera que pour représenter ces taux de convergence, il importe de connaître la limite x^*. En pratique, on calculera les `IterMax` premiers termes de la suite, pour `IterMax` grand, et l'on prendre $x_{\texttt{IterMax}} = x^*$.

Critère de convergence linéaire Un critère particulièrement pratique pour la convergence linéaire est le suivant :

Proposition 2.1.2. *Soit $\{x_k\}_{k\in\mathbb{N}}$ une suite de $\mathbb{R}^d$. Supposons qu'il existe $\alpha \in (0,1)$ tel que*

$$\forall k \in \mathbb{N}\,, \|x_{k+1} - x_k\| \leq \alpha \|x_k - x_{k-1}\|.$$

Algorithm 2 Représentation graphique d'une convergence linéaire

```
def algo(f,IterMax=10)
xn=x0   On définit le premier élément de la liste
L=[]   On définit une liste que l'on augmentera récursivement, et qui contiendra tous les éléments générés
for n in range(IterMax):
L.append(xn)
xn=....
print("Erreur, l'algorithme n'a pas convergé
après",IterMax,"itérations")
```

Alors la suite admet une unique valeur d'adhérence $x^* \in \mathbb{R}^d$, et converge linéairement vers x^* avec un taux inférieur à α.

En d'autres termes, une contraction stricte des distances implique une convergence linéaire.

Preuve de la Proposition 2.1.2. Par une récurrence immédiate on obtient

$$\forall k \in \mathbb{N}, \|x_{k+1} - x_k\| \leq \alpha^k \|x_1 - x_0\|.$$

Commençons par montrer que ceci implique que la suite $\{x_k\}_{k\in\mathbb{N}}$ est une suite de Cauchy. On se fixe $(k, p) \in \mathbb{N}^2$. Alors

$$\begin{aligned}\|x_k - x_{k+p}\| &\leq \sum_{i=1}^{p} \|x_{k+i} - x_{k+(i-1)}\| \\ &\leq \alpha^k \|x_1 - x_0\| \sum_{i=1}^{p} \alpha^{i-1} \\ &\leq C\alpha^k\end{aligned}$$

puisque la série $\sum \alpha^i$ converge. En particulier, la suite est de Cauchy et donc converge vers un certain $x^* \in \mathbb{R}^d$. Par ailleurs, on a l'estimation suivante : il existe $C > 0$ tel que pour tout $k \in \mathbb{N}$, pour tout $p \in \mathbb{N}$,

$$\|x_k - x_{k+p}\| \leq C\alpha^k;$$

en passant à la limite $p \to \infty$ on retrouve la convergence linéaire annoncée. □

2.1.2.2 Convergence d'ordre $\beta > 1$

Il se peut que la suite converge bien plus rapidement que linéairement ; un exemple typique est celui de la **convergence quadratique** :

Définition 2.1.4 (Convergence d'ordre $\beta > 1$)**.** *On dit qu'une suite $\{x_k\}_{k\in\mathbb{N}}$* ***converge vers x^* à ordre au moins*** *$\beta > 1$ s'il existe une constante $C > 0$ et $\alpha \in (0, 1)$ tels que*

$$\|x_k - x^*\| \leq C\alpha^{\beta^k}.$$

Le supremum des β vérifiant cette condition est appelé ***ordre de convergence*** *de la suite. Si $\beta = 2$, on parle de* ***convergence quadratique****.*

Nous verrons plus tard, quand nous aborderons la méthode de Newton, l'intérêt de spécifier le taux $\beta = 2$. Si on reprend l'exemple d'un $\alpha = \frac{1}{10}$, avec cette fois $\beta = 2$, alors on se rend compte qu'on double, à chaque itération, le nombre de décimales gagnées. Le problème sur ce type de convergence, c'est qu'il est beaucoup plus difficile de l'observer numériquement (même si l'on peut effectivement essayer de tracer des ln(ln()), les résultats sont rarement probants). Néanmoins si, en échelle logarithmique, on décroît toujours plus vite que des droites, on peut *a minima* dire que la convergence est sur-linéaire.

Nous avons pour la convergence quadratique l'analogue de la Proposition 2.1.2.

Proposition 2.1.3. *Soit $\{x_k\}_{k\in\mathbb{N}}$ une suite de $\mathbb{R}^d$. Supposons qu'il existe $C > 0\,, \beta > 1$ tels que, d'une part*

$$\alpha := C^{\frac{1}{\beta-1}}\|x_1 - x_0\| < 1$$

et que d'autre part

$$\forall k \in \mathbb{N}\,, \|x_{k+1} - x_k\| \leq C\|x_k - x_{k-1}\|^\beta.$$

Alors la suite admet une unique valeur d'adhérence $x^* \in \mathbb{R}^d$, et converge vers x^* à l'ordre β et à taux α.

Preuve de la Proposition 2.1.3. Par une récurrence immédiate on obtient

$$\forall k \in \mathbb{N}\,, \|x_{k+1} - x_k\| \leq C^{\sum_{i=0}^{k-1}\beta^i}\|x_1 - x_0\|^{\beta^k} = C^{-\frac{1}{\beta-1}}\alpha^{\beta^k}.$$

Une fois de plus, ceci implique que la suite $\{x_k\}_{k\in\mathbb{N}}$ est une suite de Cauchy : on se fixe $(k, p) \in \mathbb{N}^2$. Alors

$$\begin{aligned}\|x_k - x_{k+p}\| &\leq \sum_{i=1}^{p}\|x_{k+i} - x_{k+(i-1)}\| \\ &\leq C^{-\frac{1}{\beta-1}}\sum_{i=1}^{p}\alpha^{\beta^{k+i-1}}.\end{aligned}$$

Néanmoins, on observe que pour tout indice i on a l'estimation

$$\alpha^{\beta^{k+i}} \leq \alpha^{\beta^k}\alpha^{i(\beta-1)}.$$

Ceci vient simplement de la majoration

$$(1 + x)^\gamma \geq 1 + \gamma x \text{ si } x \geq 0\,, \gamma \geq 1.$$

Cette estimation implique que, pour tout indice i, on a

$$\beta^i \geq 1 + i\,(\beta - 1)$$

qui, puisque $\alpha < 1$, implique à son tour que

$$\alpha^{\beta^{k+i}} \leq \alpha^{\beta^k(1+i(\beta-1))} \leq \alpha^{\beta^k + i(\beta-1)}$$

On conclut exactement de la même manière qu'avant. □

2.2 Discrétisation de fonctions

Un autre aspect essentiel quand nous traiterons d'algorithmes de minimisation ou de résolution est la discrétisation des fonctions considérées. Rappelons donc que, si l'on sait calculer de manière efficace les zéros d'une fonction f, il sera possible de chercher des optimiseurs de certaines fonctions en cherchant leurs points critiques, c'est-à-dire les points où la dérivée de la fonction s'annule. En pratique, on peut évidemment essayer de calculer ∇F, F étant la fonction que l'on cherche à optimiser, de manière explicite, mais, fréquemment, cela n'est ni possible ni souhaitable. On choisit alors d'approcher les dérivées de la fonction F numériquement, par *différences finies*. Il s'agit de "geler" le quotient définissant la dérivée.

Définition 2.2.1 (Dérivée discrète)**.** *Soit $F \in \mathscr{C}^1(\mathbb{R}^d, \mathbb{R})$. Soit $\delta > 0$ un pas de discrétisation. L'approximation par différences finies (resp. différences finies centrées) de ∇F d'ordre δ est la fonction $\tilde{\nabla}_\delta F : \mathbb{R}^d \to \mathbb{R}^d$ définie par*

$$\forall i \in \{1, \dots, d\} \left(\tilde{\nabla}_\delta F\right)(x)_i := \frac{F(x + \delta \vec{e_i}) - F(x)}{\delta}$$
$$\textit{(resp. } \left(\tilde{\nabla}_{\delta, \textit{centrée}} F\right)(x)_i := \frac{F(x + \delta \vec{e_i}) - F(x - \delta \vec{e_i})}{2\delta}\textit{).}$$

Dans l'expression ci-dessus, $\{\vec{e_i}\}_{i=1,\dots,d}$ est la base canonique de $\mathbb{R}^d$.

Plus $\delta > 0$ est petit, meilleure est l'approximation et on peut la quantifier. Néanmoins, il est inutile de chercher à travailler avec des pas de discrétisation trop petits. En effet, les ordinateurs ont une précision de calcul limitée η, qui quantifie le nombre de chiffres significatifs qu'un ordinateur peut retenir. En pratique, $\eta \sim 10^{-16}$.

Ceci a pour première conséquence qu'une fonction F numérique n'est jamais définie qu'à un ordre η près ; en d'autres termes, toute fonction F est approchée par une fonction numérique F_η telle que

$$\|F_\eta - F\|_\infty \leq \eta.$$

Et en particulier c'est sur F_η que l'on va calculer le gradient discrétisé.

Quand on travaille sur des gradients discrétisés, il faut donc trouver le pas de discrétisation en-deça duquel il n'y a plus d'intérêt à raffiner. Rendons-cela plus quantitatif par un raisonnement assez caractéristique du domaine, en essayant de quantifier par nos différents paramètres la différence entre la véritable dérivée d'une fonction réelle, et sa dérivée approchée. Dans les deux cas, on doit obtenir une borne de la forme

$$\|\tilde{\nabla}\delta / \tilde{\nabla}_{\delta, \text{centrée}} F_\eta - F'\|_\infty \leq \phi(\delta, \eta).$$

Une telle estimation ne peut être que locale. On se place dans le cadre suivant : soit $F \in \mathscr{C}^2([0,1], \mathbb{R})$. On distingue les deux approximations :

1. Approximation par différences finies (fonctions $\mathscr{C}^2$) : dans l'approximation par différences finies, considérons donc $x \in [0,1]$. Alors on a

$$\begin{aligned}\left|\frac{F_\eta(x+\delta)-F_\eta(x)}{\delta} - F'(x)\right| &\leq \left|\frac{F_\eta(x+\delta)-F(x+\delta)}{\delta}\right| + \left|\frac{F_\eta(x)-F(x)}{\delta}\right| \\ &\quad + \left|\frac{F(x+\delta)-F(x)}{\delta} - F'(x)\right| \\ &\leq 2\frac{\eta}{\delta} + \frac{\delta}{2}\|F''\|_{L^\infty([0,1])}.\end{aligned}$$

   Si l'on veut minimiser l'erreur en $\delta > 0$, on aboutit ainsi à choisir

$$\delta \sim \sqrt{\eta},$$

   ce qui donne une erreur d'ordre $\sqrt{\eta}$.

2. Approximation par différences finies centrées (fonctions $\mathscr{C}^3$) : le calcul est similaire, mais on suppose que F est $\mathscr{C}^3$. On a, pour tout $x \in (0,1)$,

$$\begin{aligned}\left|\frac{F_\eta(x+\delta)-F_\eta(x-\delta)}{2\delta} - F'(x)\right| &\leq \left|\frac{F_\eta(x+\delta)-F(x+\delta)}{2\delta}\right| \\ &\quad + \left|\frac{F_\eta(x-\delta)-F(x-\delta)}{2\delta}\right| \\ &\quad + \left|\frac{F(x+\delta)-F(x-\delta)}{2\delta} - F'(x)\right| \\ &\leq \frac{\eta}{\delta} + \left|\frac{F(x+\delta)-F(x-\delta)}{2\delta} - F'(x)\right|.\end{aligned}$$

   Néanmoins, on peut écrire

$$F(x \pm \delta) = F(x) \pm \delta F'(x) + \frac{\delta^2}{2}F''(x) + \frac{\delta^3}{2}\int_0^1 (1-t)^2 F^{(3)}(x+\delta t)dt.$$

   En sommant les deux termes, on aboutit ainsi à l'existence d'une constant $M > 0$ telle que

$$\left|\frac{F_\eta(x+\delta)-F_\eta(x-\delta)}{2\delta} - F'(x)\right| \leq \frac{\eta}{\delta} + M\delta^2.$$

   Si on minimiser cette fois l'erreur obtenue en δ, on tombe sur $\delta \sim \eta^{\frac{1}{3}}$.

Chapitre 3

Méthodes d'optimisation uni-dimensionnelles

Dans ce cours, nous présentons trois méthodes importantes dans le cas unidimensionnel : la méthode de dichotomie, la méthode de la section dorée, et nous faisons une première incursion vers la méthode de Newton. Dans ces trois méthodes, le cadre est le suivant : on travaille avec une fonction $f \in \mathscr{C}^0(\mathbb{R}, \mathbb{R})$, et l'on veut résoudre

$$f(x) = 0. \tag{3.1}$$

En particulier, toutes les méthodes présentées ici sont unidimensionnelles.

3.1 Méthode de dichotomie (méthode d'ordre 0)

3.1.1 Présentation de la méthode

On commence par la méthode la plus classique, la *méthode de dichotomie*. On considère l'équation suivante : f étant une fonction continue, $[a, b] \subset \mathbb{R}$ étant un intervalle réel, déterminer $x^* \in [a, b]$ tel que

$$f(x^*) = 0. \tag{3.2}$$

Pour avoir une chance que cette équation ait une solution, on suppose également que

$$f(a) < 0\,, f(b) > 0$$

ou, plus généralement, que $f(a)f(b) < 0$. Dans ces conditions, le théorème des valeurs intermédiaires garantit l'existence d'une solution x^*.

Remarque 3.1.1. *On n'aborde pas ici de questions d'unicité, mais celle-ci est évidente si la fonction f est strictement monotone.*

La **méthode de dichotomie**, aussi appelée, **méthode de bissection**, consiste, à chaque étape, à réduire l'intervalle de manière assez intuitive. Notons d'abord que, puisque l'on renvoie un intervalle, on a en fait besoin de définir

trois suites : d'abord, $\{y_k^+\}_{k\in\mathbb{N}}, \{y_k^-\}_{k\in\mathbb{N}}$ pour les bornes de l'intervalle, puis y_k comme point milieu de l'intervale. La procédure est la suivante :

1. On initialise à $y_0^- = a\,, y_0^+ = b$. On définit $y_0 := \frac{1}{2}(y_0^- + y_0^+)$.
2. Si $f(y_0) \geq 0$, on pose $y_1^- = y_0^-$ et $y_1^+ = y_0$. Sinon, on pose $y_1^- = y_0$ et $y_1^+ = y_0^+$.
3. Ensuite, pour tout $k \in \mathbb{N}$, on définit

$$\begin{cases} y_{k+1}^- := y_k^-\,, y_{k+1}^+ := y_k \text{ si } f(y_k) > 0\,, \\ y_{k+1}^- := y_k,, y_{k+1}^+ := y_k^+ \text{ si } f(y_k) < 0\,, \\ y_{k+1}^\pm = y_k \text{ si } f(y_k) = 0, \\ y_{k+1} := \frac{y_{k+1}^+ + y_{k+1}^-}{2}. \end{cases}$$

On peut démontrer assez aisément que cet algorithme converge linéairement à taux $\alpha = \frac{1}{2}$:

Lemme 3.1.1 (Convergence de la méthode par dichotomie)**.** *Soit $f \in \mathscr{C}^0([a,b],\mathbb{R})$, $f(a) < 0$ et $f(b) > 0$. La méthode de dichotomie converge linéairement, à taux $\alpha = \frac{1}{2}$, vers une solution de (3.2).*

Preuve du Lemme 3.1.1. On suppose que la suite générée n'est pas stationnaire. On définit pour tout $k \in \mathbb{N}$ l'intervalle $I_k := [x_k^-; x_k^+]$. On note que, d'une part, cette suite est décroissante pour l'inclusion et que, d'autre part,

$$\mathrm{Vol}(I_{k+1}) = \frac{1}{2}\mathrm{Vol}(I_k).$$

Par le théorème des segments emboîtés (ici, simplement des suites adjacentes) on a

$$\cap_{k\in\mathbb{N}} I_k = \{x^*\}$$

pour un certain point x^*. En passant à la limite dans $f(x_k^-) < 0\,, f(x_k^+) \geq 0$ on obtient

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

Par ailleurs, on sait que

$$\forall k \in \mathbb{N}\,, y_k\,, x^* \in I_k \Rightarrow |x^* - y_k| \leq \mathrm{Vol}(I_k) \leq \frac{1}{2^k}\,|b-a|\,.$$

Montrons que le taux $\alpha = \frac{1}{2}$ est optimal quand la suite n'est pas stationnaire : supposons par l'absurde qu'il existe un réel $\alpha \in (0, \frac{1}{2})$ et une constante C tels que

$$|y_k - x^*| \leq C\alpha^k.$$

On a alors :

$$|y_k - y_{k+1}| \leq C\alpha^k + C\alpha^{k+1} \leq 2C\alpha^k.$$

Néanmoins, par construction, on a également, si la suite n'est pas stationnaire,

$$|y_{k+1} - y_k| = \frac{1}{2^k}|b-a|.$$

C'est une contradiction. □

En TD et en TP, nous verrons comment coder cette méthode, et illustrer ses ordres de convergence.

3.1.2 Résumé de l'algorithme

En TP, nous coderons l'algorithme en suivant simplement les grandes lignes suivantes :

1. Initialisation de l'algorithme en choisissant un couple (a, b). On vérifie que l'on est dans les conditions du théorème $(f(a)f(b) < 0)$
2. On calcule le point milieu c, et on choisit le nouvel intervalle grâce à la règle de dichotomie.
3. On intégrera ou bien un critère de tolérance qui correspond à la taille de l'intervalle, ou bien un nombre maximal d'itérations.

3.2 Méthode de Newton en dimension 1 (méthode d'ordre 1)

3.2.1 Présentation de la méthode

On passe au premier algorithme d'ordre 1, c'est-à-dire qui fasse intervenir la dérivée. La convergence de ces algorithmes est à double tranchant : alors que les algorithmes de type dichotomie ou section dorée converge toujours, dans le cas des algorithmes d'ordre 1, *l'initialisation est cruciale* : si on la choisit mal, l'algorithme ne converge pas. En revanche, *si l'initialisation est bien choisie, l'algorithme converge de manière spectaculaire.*

Cet algorithme est fondamental, d'une part par le rôle qu'il a joué dans le développement historique de la théorie de l'approximation, d'autre part parce que son étude constitue une forme de propédeutique à des méthodes plus évoluées, comme celles de descentes de gradient.

Cette méthode est utilisée pour résoudre des équations de la forme

$$f(x) = 0. \tag{3.3}$$

Ici, f est une fonction dérivable. Même s'il est fort probable que vous ayiez déjà vu cette méthode, nous en rappelons les grandes étapes.

3.2.1.1 Description heuristique de la méthode

L'idée, c'est que si l'on se place au voisinage d'un point x_0, on peut approcher la fonction f par sa tangente :

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \underset{x_0 \to x}{o}(|x - x_0|).$$

On définit la partie linéaire de f à x_0 par

$$R_f(x_0, \cdot) : x \mapsto f(x_0) + f'(x_0)(x - x_0).$$

On sait que, dans un petit voisinage de x_0, $R_f(x_0, \cdot) \approx f$, de sorte que résoudre $f(x) = 0$ revient peu ou prou à résoudre $R_f(x_0, x) = 0$. Néanmoins, $R_f(x_0, \cdot)$ étant une fonction linéaire, son unique racine est donnée par

$$x_0^* = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Si on suppose que f' est de signe constant sur l'intervalle où l'on travaille, on s'aperçoit immédiatement qu'alors $f(x_0^*) < f(x_0)$ si $f(x_0) > 0$, et $f'(x_0) < f'(x_0^*) < 0$ si $f(x_0) < 0$. Donc, par cette simple résolution, on réussit à améliorer notre critère.

Remarque 3.2.1. *Simplement à partir de la construction des itérations de la méthode de Newton, on voit que "plus la fonction est linéaire", plus la méthode de Newton risque d'être précise.*

Ceci suggère la construction suivante pour l'algorithme de Newton : on initialise en un point x_0 (dont nous verrons comment il doit être choisi) et, pour passer de x_k à x_{k+1} on définit simplement

$$x_{k+1} := x_k - \frac{f(x_k)}{f'(x_k)}.$$

3.2.2 Quelle vitesse de convergence pour cette méthode ?

Avant de nous lancer dans l'étude de la convergence dans le cas général, faisons deux petites remarques destinées à souligner les différents types de régime auxquels la méthode de Newton peut mener. Dans un premier temps, observons que si l'on travaille avec une fonction linéaire de la forme $f(x) = ax + b$ cette méthode converge en une unique itération. En effet, quel que soit x_0, on a

$$x_1 = x_0 - \frac{ax_0 + b}{a} = -\frac{b}{a}.$$

Dans un second temps, remarquons la convergence peut, à l'inverse, être très lente : par exemple, même avec une bête fonction quadratique de la forme $f(x) = x^2$ on s'aperçoit que

$$x_{k+1} = \frac{1}{2}x_k = 2^{-k}x_0$$

et que donc la méthode, si elle converge bien, ne le fait que linéairement.

3.2.2.1 Convergence de la méthode

En général, la méthode de Newton converge extrêmement rapidement, à condition que l'initialisation soit bien choisie.

Théorème 3.2.1 (Convergence quadratique de la méthode de Newton)**.** *Soit $f \in \mathscr{C}^2(\mathbb{R};\mathbb{R})$ et $x^* \in \mathbb{R}$ tel que :*

$$f(x^*) = 0\,, |f'(x^*)| > 0.$$

Il existe $\varepsilon > 0$ tel que, pour toute initialisation $x_0 \in (x^* - \varepsilon; x^* + \varepsilon)$, la méthode de Newton initialisée en x_0 converge quadratiquement vers x^*.

Preuve du Théorème 3.2.1. Quitte à remplacer f par $-f$ on peut supposer que

$$f'(x^*) > 0.$$

On peut distinguer deux preuves : une qui repose sur les théorèmes de points fixes, mais qui ne donnera pas tellement d'information précise sur le type d'initialisation à choisir, l'autre qui est plus directe, repose simplement sur la formule de Taylor-Lagrange et permet de quantifier. Donnons la première preuve : f étant de classe $\mathscr{C}^3$, on fixe $\varepsilon_0 > 0$ tel que

$$\inf_{[x^*-\varepsilon_0;x^*+\varepsilon_0]} f'(x^*).$$

On choisit $x_0 \in [x^* - \varepsilon_0; x^* + \varepsilon_0]$.

La preuve se ramène à l'étude d'un problème de point fixe : en définissant

$$g(x) := x - \frac{f(x)}{f'(x)}$$

on a immédiatement que $f(y) = 0$ si et seulement si $g(y) = y$. En particulier, il nous suffit de démontrer que la suite récurrente

$$x_{k+1} = g(x_k)$$

initialisée à x_0 converge quadratiquement vers x^*. Ces suites ont été étudiées lors du TD (TD n°1, Exercice 1.9). Il faut vérifier que

$$|g'(x^*)| < 1.$$

Or,

$$g'(x^*) = 1 + \frac{f(x^*)f''(x^*)}{\left(f'(x^*)\right)^2} - 1 = 0 < 1$$

et g est de classe $\mathscr{C}^2$. En particulier, la suite converge quadratiquement (au moins) vers x^*.

Maintenant, donnons l'autre preuve de la convergence quadratique, qui donne une distance maximale d'où initialiser l'algorithme. Cette seconde preuve repose uniquement sur la formule de Taylor-Lagrange : l'on sait que

$$0 = f(x^*) = f(x_0) + (x^* - x_0)f'(x_0) + f''(\xi)\frac{(x^* - x_0)^2}{2}$$

pour un certain $\xi \in (x^*, x_0)$. Noter bien le fait important suivant : pour démontrer la convergence de l'algorithme de Newton, on doit faire un développement de Taylor **autour du point d'initialisation** x_0, ce qui est cohérent avec l'idée sous-jacente de la méthode.

Ainsi on obtient en posant $x_1 := x_0 - (f'(x_0)^{-1})f(x_0)$,

$$\begin{aligned} 0 &= -f'(x_0)\left\{x_0 - \frac{f(x_0)}{f'(x_0)}\right\} + x^* f'(x_0) + \frac{1}{2}f''(\xi)\,(x^* - x_0)^2 \\ &= f'(x_0)\,(x^* - x_1) + \frac{1}{2}f''(\xi)\,(x^* - x_0)^2 \end{aligned}$$

de sorte que

$$|x^* - x_1| \leq \frac{\|f''\|_{L^\infty(I)}}{2\inf_I |f'|}|x^* - x_0|^2,$$

où I est n'importe quel intervalle compact contenant x^* et x_0. Ainsi, par le critère de convergence quadratique (Proposition 2.1.3) on en déduit que la suite des itérées converge quadratiquement si x_0 est choisi de sorte que

$$|x^* - x_0| < \frac{2\inf_I |f'|}{\|f''\|_{L^\infty(I)}}.$$

□

En particulier, par la méthode de Newton, on double le nombre de décimales à chaque itération ; en moyenne, N itérations permettent d'obtenir 2^N décimales, ce qui est prodigieusement rapide.

Insistons une nouvelle fois sur quelques considérations autour de la vitesse de convergence :

1. Il va de soit que l'on ne peut en général pas démontrer que le taux de convergence est au plus quadratique. Pour s'en convaincre, il suffit de considérer la méthode de Newton appliquée à la fonction $f : x \mapsto x$. Quelle que soit l'initialisation x_0 on a

   $$x_1 = x_0 - x_0 = 0$$

   et on converge donc en une étape (ce qui est en particulier plus rapide que n'importe quel ordre de convergence).

2. L'hypothèse $f'(x^*) \neq 0$ est *nécessaire* pour obtenir la convergence quadratique ; à titre d'exemple, la méthode de Newton appliquée à la fonction $f : x \mapsto x^2$ donne, quelle que soit l'initialisation x_0,

   $$x_k = \left(\frac{1}{2}\right)^k x_0$$

   et l'on obtient donc cette fois une convergence linéaire.

3. Mais ce n'est pas le pire que d'avoir une convergence linéaire. Il se peut même que la méthode diverge : prenons l'exemple

$$f : x \mapsto |x|^{\frac{1}{3}}$$

et initialisons notre méthode en n'importe quel point $x_0 > 0$. On obtient

$$x_1 = x_0 - 3\frac{x_0^{\frac{1}{3}}}{x_0^{-\frac{2}{3}}} = -2x_0.$$

On voit, d'une part, qu'on passe dans le demi-plan des x négatifs et d'autre part qu'en itérant la construction on crée une suite divergente. On verra en TD une généralisation de ce phénomène.

3.2.2.2 Résumé de la méthode

Toutes les formules étant explicites, il ne reste qu'à choisir un critère d'arrêt ; on pourra soit prendre un nombre d'itérations arbitrairement grand, soit utiliser, comme critère de tolérance, la valeur de $|f(x_k)|$, et inclure un nombre maximal d'itérations à ne pas dépasser. On observera, numériquement, que la convergence est effectivement sur-linéaire.

3.2.2.3 Derniers commentaires

1. la condition $f'(x^*) \neq 0$ est essentielle. En fait, plus la fonction f est localement linéaire, plus la méthode converge rapidement.
2. Si on pense désormais en termes d'optimisation, et que l'on cherche à déterminer le minimum d'une fonction $F : \mathbb{R} \to \mathbb{R}$, une possibilité est d'appliquer la méthode de Newton à la fonction $f = F'$. La condition $f'(x^*) = 0$ devient alors $F''(x^*) \neq 0$. En d'autres termes, c'est une méthode très pratique si l'on cherche des optima non-dégénérés. Ce type de non-dégénerescence sera également crucial quand nous verrons des algorithmes de **descente de gradients**.

3.3 Méthode de la sécante (méthode d'ordre 0)

3.3.1 Heuristique et ordre de convergence

Quel est l'inconvénient de la méthode de Newton, aussi rapide sa convergence soit-elle ? C'est qu'elle fait appel non seulement à la fonction f, mais également à sa dérivée. La méthode des sécantes, à l'inverse, converge un peu moins bien que quadratiquement, mais n'a besoin de faire qu'un seul appel à la fonction f puisqu'au lieu de la dérivée de f, elle ne fait appel qu'à ses taux d'accroissement. En d'autres termes, le terme $f'(x_n)$ est simplement remplacé par le taux

$$\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Ainsi, les itérations successives de l'algorithme sont données par l'expression

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

La vraie différence est que, si nous n'utilisons pas la fonction f', nous devons choisir deux points d'initialisation, x_0 et x_1 pour définir x_2. Pour cette méthode, on a une convergence sous-quadratique, mais toujours sur-linéaire :

Proposition 3.3.1. *Si $f : \mathbb{R} \to \mathbb{R}$ est une fonction de classe C^3 avec $f(x^*) = 0$ et $f'(x^*) \neq 0$, et si $x_0, x_1 \in \mathbb{R}$ sont suffisamment proches de x^*, alors la suite $\{x_k\}_{k\in\mathbb{N}}$ définie par la méthode de la sécante converge vers x^* à l'ordre au moins $\varphi = \frac{1}{2}(1 + \sqrt{5}) \approx 1.618$ (le nombre d'or).*

Preuve de la Proposition 3.3.1. On ne donne la preuve que dans le cas où la fonction f est quadratique afin de ne pas alourdir la preuve. À la différence de la preuve de la convergence de la méthode de Newton, on fait cette fois un développement limité autour de x^*.

On définit donc le terme qu'il nous faut contrôler, à savoir

$$y_k := x_k - x^*.$$

On observe que, la fonction f étant quadratique, on a un développement de Taylor exact qui nous assure que

$$f(y) = f(x^*) + f'(x^*)(y - x^*) + \frac{1}{2}f''(x^*)(y - x^*)^2.$$

En particulier on a

$$f(x_k) = f'(x^*)y_k + \frac{1}{2}f''(x^*)y_k^2.$$

On peut injecter cette preuve dans la relation de récurrence qui définit la suite des itérations de la méthode de la sécante. Néanmoins, il va falloir être (un peu) plus astucieux que dans le cas de la méthode de Newton et travailler sur la suite des différences $(y_{k+1} - y_k)_{k\in\mathbb{N}}$. En effet, on se rend compte en développant les expressions données par l'algorithme que l'on obtient :

$$\begin{aligned}
y_{k+1} - y_k &= x_{k+1} - x_k \\
&= -\frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})} \\
&= -\frac{f(x_k)(y_k - y_{k-1})}{f'(x^*)(y_k - y_{k-1}) + \frac{1}{2}f''(x^*)(y_k^2 - y_{k-1}^2)} \\
&= -\frac{f'(x^*)y_k + \frac{1}{2}f''(x^*)y_k^2}{f'(x^*) + \frac{1}{2}f''(x^*)(y_k + y_{k-1})}
\end{aligned}$$

d'où l'on déduit que

$$y_{k+1} = y_k - \frac{f(x_k)}{f'(x^*) + \frac{1}{2}f''(x^*)(y_k + y_{k-1})}.$$

Néanmoins, on peut utiliser le fait que

$$f(x_k) = f'(x^*)y_k + \frac{1}{2}f''(x^*)y_k^2$$

pour en déduire que

$$\begin{aligned} y_{k+1} = y_k - \frac{f'(x^*)y_k + \frac{1}{2}f''(x^*)y_k^2}{f'(x^*) + \frac{1}{2}f''(x^*)(y_k + y_{k-1})} &= y_k \frac{1}{2} \cdot \frac{f''(x)y_{k-1}}{f'(x^*) + \frac{1}{2}f''(x^*)(y_k + y_{k-1})} \\ &= y_k T(y_k, y_{k-1}) \end{aligned}$$

où la fonction T est définie par

$$T(x, y) := \frac{1}{2} \cdot \frac{f''(x^*)y}{f'(x^*) + \frac{1}{2}f''(x^*)(x + y)}.$$

Soulignons cette expression : on sait désormais que

$$\boxed{\forall k \in \mathbb{N}, y_{k+1} = y_k T(y_k, y_{k-1}).} \tag{3.4}$$

À partir de cette expression auxiliaire, nous allons déduire deux choses :

1. D'abord, que la suite $\{y_k\}_{k\in\mathbb{N}}$ converge vers 0.
2. Ensuite, qu'elle converge surlinéairement.

Commençons par le convergence vers 0. On observe que la fonction T est continue en (0,0) et que, si (x_0, x_1) sont choisis de sorte que $T(y_0, y_1) < \frac{1}{2}$ alors la suite $(y_k)_{k\in\mathbb{N}}$ est décroissante en norme et converge vers 0. Par ailleurs, ceci implique que

$$\frac{|y_{k+1}|}{|y_k y_{k-1}|} \underset{k\to\infty}{\to} C_0 := \frac{|f''(x^*)|}{2|f'(x^*)|} > 0$$

(si $f''(x^*) = 0$ on est linéaire, et l'on converge en une itération). En effet, on sait que

$$\left|\frac{y_{k+1}}{y_k y_{k-1}}\right| = \frac{1}{2} \cdot \left|\frac{f''(x^*)}{f'(x^*) + \frac{1}{2}f''(x^*)(y_k + y_{k-1})}\right| \underset{k\to\infty}{\to} \frac{1}{2} \cdot \left|\frac{f''(x^*)}{f'(x^*)}\right|.$$

Pour obtenir un taux de convergence précisé, passons au logarithme dans cette inégalité en posant $z_k := -\ln(|y_k|)$. Alors

$$z_{k+1} - z_k - z_{k-1} \underset{k\to\infty}{\to} \ln(C_0).$$

Ainsi, la suite $(z_{k+1} - z_k - z_{k-1})_{k\in\mathbb{N}}$ est bornée. Soit $A > 0$ tel que

$$z_{k+1} - z_k - z_{k-1} \geq -A.$$

Posons $w_k := z_k - A$. Alors

$$w_{k+1} - w_k - w_{k-1} = (z_{k+1} - z_k - z_{k-1}) + A \geq 0$$

et donc

$$w_{k+1} \geq w_k + w_{k-1}.$$

Soit $\varphi := \frac{1}{2}(1 + \sqrt{5})$ le nombre d'or. Le nombre φ vérifie $\varphi > 1$ et $\varphi^2 = 1 + \varphi$. La suite $(w_k)_{k\in\mathbb{N}}$ diverge vers $+\infty$, donc pour n_0 assez grand, on a $w_{n_0} \geq 1$ et $w_{n_0+1} \geq \varphi$. Par une récurrence immédiate, on en déduit que $w_{n_0+n} \geq \varphi^n$. Cela montre que $z_{n_0+n} \geq \varphi^n + A$, et enfin que $|y_{n_0+n}| \leq e^{-A}\alpha^{\varphi^{n_0+n}}$ avec $\alpha := e^{-\varphi^{-n_0}} < 1$, ce qui conclut la preuve.

□

3.3.2 Résumé de l'algorithme

Les mêmes causes menant aux mêmes conséquence, on définit l'algorithme de la manière suivante :

1. Notre algorithme prendra en argument une fonction f, ainsi que deux points d'initialisation x_0 et x_1, une tolérance et un nombre d'itérations maximales.
2. À chaque étape, tant que l'on n'a pas dépassé le nombre maximal d'itérations autorisé, et tant que l'on est au dessus du seuil de tolérance, on itère la construction.

Méthodes multi-dimensionnelles : présentation succinte

Présentation de la descente de gradient

Dans ce mini-cours nous ne faisons que présenter les éléments fondamentaux d'un des algorithmes qui, malgré sa convergence lente, s'avère extrêmement pratique dans les applications, la **descente de gradient**.

Moralement, l'idée est la suivante : on travaille avec une fonction $f : \mathbb{R}^d \to \mathbb{R}$ aussi régulière que souhaité, et l'on désire résoudre (en admettant qu'il admette une solution) le problème

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

On se donne une initialisation $x_0 \in \mathbb{R}^d$. Pour déterminer une construction du point suivant x_1 qui ne soit pas trop bête, une idée possible est d'utiliser un développement de Taylor à l'ordre 1. Expliquons cela : il nous faut une **direction** dans laquelle aller, ainsi que la **taille du pas** dans cette direction. On se fixe done un vecteur $h \in \mathbb{R}^d$, qui définit la direction, et un réel $\tau > 0$ (destiné à être petit).

On peut écrire, au premier ordre

$$f(x_0 + \tau h) \approx f(x_0) + \tau \langle h, \nabla f(x_0) \rangle.$$

Si l'on s'est fixé la taille τ du pas, alors on peut supposer que

$$\|h\| = 1$$

pour la norme euclidienne. Il nous reste à déterminer la direction dans laquelle descendre. Plusieurs possibilités s'offrent à nous, mais l'une d'entre elles est de choisir un vecteur h solution du problème d'optimisation

$$\min_{\|h\|=1} \langle \nabla f(x_0), h \rangle.$$

L'inégalité de Cauchy-Schwarz nous garantit que si $\nabla f(x_0) \neq 0$ alors une solution du problème d'optimisation ci-dessus est donnée par

$$h_{x_0} := -\frac{\nabla f(x_0)}{\|\nabla f(x_0)\|}.$$

L'algorithme de la **descente de gradient à pas constant** est une mise en œuvre de cette idée. Ici, on part d'une initialisation x_0, on se fixe un pas $\tau > 0$ et l'on définit, pour tout $k \in \mathbb{N}$, la suite

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

Une grosse partie du cours qui reste vise à appréhender, à analyser et à maîtriser cet algorithme, ainsi que ses variantes habituelles.

Présentation de la méthode de Newton

Un autre algorithme que nous étudierons amplement dans la suite de ce cours est l'algorithme de Newton. De même qu'en dimension 1 cet algorithme nous permettait de déterminer des points critiques et donc, potentiellement, des minimiseurs, en dimension supérieure, l'algorithme de Newton nous permettra, pour une fonction $f \in \mathscr{C}^2(\mathbb{R}^d;\mathbb{R})$,d e trouver des points critiques de f, c'est-à-dire de résoudre l'équation

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

Puisque, comme en dimension 1, on a l'approximation

$$\nabla f(x) \approx \nabla f(x_0) + \nabla^2 f(x_0) \cdot (x - x_0)$$

on en déduit que la suite des itérations de l'algorithme de Newton est donnée par, d'une part, une initialisation x_0 et, pour tout entier naturel k,

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k).$$

Chapitre 4

La descente de gradient uni-dimensionnelle

4.1 Introduction

On présente dans ce cours quelques aspects de la méthode de descente de gradient en dimension 1, qui servira d'introduction à la méthode en plusieurs dimensions. Cette méthode est utilisée pour résoudre des problèmes d'optimisation

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

où f est une fonction suffisamment régulière. Il est important de noter qu'ici on ne travaille qu'avec un problème d'optimisation sans contraintes, et que les conditions d'optimalité suivent donc, dans un premier temps, la règle de Fermat,

$$f'(x^*) = 0$$

et, dans un second, les règles de Lagrange

$$f''(x^*) \geq 0.$$

Ensuite, la méthode de la descente de gradient est une méthode algorithmique qui prend en argument à la fois la fonction f et sa dérivée. L'avantage considérable par rapport à une méthode de type Newton est que nous n'avons pas à calculer de dérivées secondes, ce qui allège le coût computationnel. L'inconvénient principal est que la vitesse de convergence de l'algorithme est plus faible (linéaire au lieu de quadratique).

4.2 Approche heuristique de la méthode

L'objectif de cette méthode locale est, partant d'un point initial x_0, d'obtenir un point x_1 qui "fasse mieux" que x_0 en effectuant un petit pas dans une

direction bien choisie. Pour cela, le point central est d'utiliser, une fois de plus, les développements de Taylor : en partant d'un point x_0, on a

$$f(x_0 + h) \approx f(x_0) + hf'(x_0).$$

Le paramètre (petit) h quantifie à la fois le sens et l'intensité du pas où l'on doit chercher un minimiseur. Puisque l'on veut minimiser la fonction f, il faut choisir un h tel que

$$hf'(x_0) < 0.$$

Il est a priori loisible de prendre un h très grand en norme, mais cela est en contradiction avec le caractère local de la méthode. Ainsi, plutôt que de tout paramétrer par un même réel h, on distingue (artificiellement en dimension 1) cette direction de descente et la grandeur du pas, en paramétrant plutôt la méthode par un couple (τ, h), où $\tau \in \mathbb{R}_+^*$ est la taille du pas de descente, et $h \in \pm 1$ est défini par

$$h := -\frac{f'}{\|f'\|}(x_0).$$

Cette direction h est appelée *direction de descente pour f en x_0*. On vérifie facilement, grâce à un développement limité, que l'on a bien la résultat suivant :

Proposition 4.2.1. *Soit $f \in \mathscr{C}^2(\mathbb{R}; \mathbb{R})$. Soit $x_0 \in \mathbb{R}$ tel que $f'(x_0) \neq 0$. Soit $h := -f'(x_0)/|f'(x_0)|$. Alors, pour $\tau > 0$ suffisamment petit, si l'on définit x_1 par*

$$x_1 = x_0 + \tau h$$

on a

$$f(x_1) < f(x_0).$$

En particulier, si on l'itère, cette méthode fournit bien une suite qui "devrait" bien s'approcher d'un optimum.

Ceci nous incite à définir un algorithme de la manière suivante :

1. On initialise en un certain x_0.
2. Pour tout $k \in \mathbb{N}$, on définit

$$x_{k+1} := x_k - \tau_k f'(x_k)$$

pour un certain pas τ_k (noter qu'ici on ne normalise pas d'entrée de jeu la direction de descente, pour ne pas s'embêter à distinguer les points critiques).

Néanmoins, il reste à déterminer le pas de descente τ, ce qui peut s'avérer crucial. Afin de bien marquer ce point, évoquons les difficultés principales liées à cette méthode.

4.2.1 Non-convergence pour des pas trop grands

Insistons encore une fois sur la nécessité de choisir un bon pas pour la descente de gradient. On considère, par exemple, la fonction

$$f(x) = \frac{x^2}{2}.$$

Alors $f'(x) = x$ et, avec à l'étape k un pas τ_k, on a, pour les itérations successives de la méthode de gradient

$$x_{k+1} = (1 - \tau_k)x_k = \cdots = x_0 \prod_{i=0}^{k} (1 - \tau_i).$$

On voit que si on prend des τ_i qui sont tous plus grands que 1, la méthode diverge et que, si on prend $\tau_i := 2(-1)^i$, on génère une suite périodique.

4.2.2 Sensibilité aux points critiques et vitesse de convergence

La méthode de descente de gradient est une méthode qui, par nature, est *locale*. En particulier, si on l'initialise en un point critique x_0 (ou si on s'approche d'un point critique x_0) alors on y reste bloqué. En particulier, on fera bien attention à distinguer entre la convergence de la méthode, qui sera vers un point critique, et la caractérisation du point limité comme un minimum global. À titre d'exemple, considérons la fonction

$$f : x \mapsto \frac{x^3}{3}$$

et définissons $x_0 = 1$. On se fixe un pas $\tau \in (0, 1)$. Alors la méthode de descente à pas τ fournit la suite d'itérations

$$x_{k+1} = x_k(1 - \tau x_k).$$

Ainsi, la suite fournie est décroissante, et converge vers 0, qui n'est évidemment pas un minimum (ni global, ni même local), et même plus lentement que linéairement. En fait, nous verrons que le critère naturel pour la descente de gradient est que les points critiques soient non-dégénérés.

Ceci peut avoir une conséquence pratique importante : il se peut que $x_{k+1} - x_k$ soit, en norme, inférieur à la précision numérique de l'ordinateur.

L'autre information que cet exemple fourni est que l'on ne peut espérer qu'un paradigme du type

$$\text{Convergence vers } x^* \Rightarrow \text{convergence vers un minimiseur}$$

ne soit valable que si l'on travaille avec une fonction f pour laquelle points critiques et minima coïncident, ce qui est le cas, par exemple, pour les fonction convexes.

4.2.3 Adaptation en dimensions supérieures

En dimensions supérieures, l'autre difficulté sera de choisir *une direction de descente* et une *taille de pas de descente*, ce qui requérera plusieurs notions fines de calcul différentiel que nous verrons dans la seconde partie de ce cours.

4.3 Démonstration de plusieurs résultats afférents en dimension 1

On donne ici les démonstrations de quelques résultats en dimension 1, qui donnent un échauffement pour les preuves en dimension supérieures. Dans tout ce qui suit, $f \in \mathscr{C}^1(\mathbb{R};\mathbb{R})$.

Théorème 4.3.1 (Quelques résultats en dimension 1)**.** *On a les résultats suivants :*

1. *Décroissance de la suite des valeurs : on suppose que f' est M-lipschitzienne. Si $0 < \tau < \frac{1}{M}$, la suite $\{f(x_k)\}_{k\in\mathbb{N}}$ est strictement décroissante tant qu'elle n'est pas stationnaire, $\{x_k\}$ étant la suite générée par descente de gradient à pas fixe τ initialisée en tout x_0.*
2. *Décroissance de la suite des valeurs précisées : on suppose que f est convexe, que f atteint son minimum en x^* et que f' est M-lipschitzienne. Si $\tau < \frac{1}{2M}$, alors*
$$f(x_k) - f(x^*) \leq \frac{|x_0 - x^*|^2}{2\tau k}.$$
3. *Convergence linéaire vers les minima non-dégénérés : on suppose que $f \in \mathscr{C}^2$, que x^* est un point critique de f et que f est strictement convexe sur $[x^* - \varepsilon; x^* + \varepsilon]$ pour $\varepsilon > 0$ suffisamment petit. En d'autres termes, il existe $\ell_0\,, \ell_1 > 0$ tels que*
$$\forall x \in [x^* - \varepsilon; x^* + \varepsilon]\,, f''(x) \in [\ell_0; \ell_1].$$
Si $\tau < \frac{2}{\ell_1}$, si $x_0 \in [x^* - \varepsilon; x^* + \varepsilon]$, la suite $\{x_k\}_{k\in\mathbb{N}}$ converge linéairement vers x^* à taux $\max(|1 - \tau\ell_0|, |1 - \tau\ell_1|)$. On peut optimiser ce taux en choisissant $\tau^* = \frac{2}{\ell_0+\ell_1}$ et le taux optimal de convergence est alors $\frac{\ell_1-\ell_0}{\ell_0+\ell_1}$.

Preuve du théorème 4.3.1. 1. On suppose que f' est M-lipschitzienne. Par les théorèmes des accroissements finis on sait que pour tout entier naturel k on a
$$f(x_{k+1}) - f(x_k) = (x_{k+1} - x_k)f'(\xi)$$
pour un certain $\xi \in [x_k; x_{k+1}]$. Ainsi,
$$f(x_{k+1}) - f(x_k) \leq f'(x_k)(x_{k+1} - x_k) + M(x_{k+1} - x_k)^2.$$
Or, puisque $x_{k+1} = x_k - \tau f'(x_k)$ ceci implique que
$$f(x_{k+1}) - f(x_k) \leq -\tau \left(f'(x_k)\right)^2 + \tau^2 M(f'(x_k))^2$$

$$= \left(f'(x_k)\right)^2 (M\tau - 1)\,\tau,$$

d'où la conclusion annoncée.

2. On reprend l'estimation précédente :

$$f(x_{k+1}) - f(x_k) \leq \left(f'(x_k)\right)^2 (M\tau - 1)\,\tau.$$

On pose $-\delta := (M\tau - 1)\tau < 0$ et l'on choisit τ de sorte que

$$\delta > \frac{\tau}{2}$$

ce qui en gardant en tête que $\delta = (1 - M\tau)\tau$, revient à choisir $\tau < \frac{1}{2M}$. On a alors l'estimation

$$f(x_{k+1}) - f(x_k) \leq -\frac{\tau}{2}(f'(x_k))^2.$$

Maintenant, par convexité de la fonction f, on sait également, une fonction convexe étant au dessus de ses tangentes, que

$$\forall y\,, f(x^*) \geq f(y) + f'(y)(x^* - y),$$

de sorte que l'on obtient en outre

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) - \frac{\tau}{2}(f'(x_k))^2 \\ &\leq f(x^*) + f'(x_k)(x_k - x^*) - \frac{\tau}{2}(f'(x_k))^2 \\ &\Leftrightarrow \\ f(x_{k+1}) - f(x^*) &\leq \frac{1}{\frac{\tau}{2}}\left(-\frac{\tau}{4}^2(f'(x_k))^2 + \frac{\tau}{2}f'(x_k)(x_k - x^*)\right) \\ &\leq \frac{1}{\frac{\tau}{2}}\left(\frac{|x_k - x^*|^2}{4} - \left(\frac{\tau}{2}f'(x_k) - \frac{(x_k - x^*)}{2}\right)^2\right). \end{aligned}$$

On récupère alors dans le terme de droite

$$\frac{\tau}{2}f'(x_k) - \frac{x_k - x^*}{2} = \frac{x^* - x_{k+1}}{2}.$$

Ainsi, on a finalement

$$f(x_{k+1}) - f(x^*) \leq \frac{1}{4\frac{\tau}{2}}\left(|x_k - x^*|^2 - |x_{k+1} - x^*|^2\right) \leq \frac{1}{2\tau}\left(|x_k - x^*|^2 - |x_{k+1} - x^*|^2\right).$$

En sommant ces estimations et en utilisant le premier point du théorème, il vient

$$N\left(f(x_N) - f(x^*)\right) \leq \sum_{k=0}^{N-1}\left\{f(x_{k+1}) - f(x^*)\right\} \leq \frac{1}{2\tau}\left(x_0 - x^*\right)^2.$$

La conclusion s'ensuit.

3. Passons, enfin, au comportement de l'algorithme au voisinage des points critiques non-dégénérés. Ici, la preuve est beaucoup plus simple. Il suffit d'observer que x^* est l'unique point fixe de la fonction $F : x \mapsto x - \tau f'(x)$ dans $[x_0 - \varepsilon; x_0 + \varepsilon]$ si $|1 - \tau\ell_0|, |1 - \tau\ell_1| < 1$. Il suffit ensuite de remarquer que l'on travaille simplement sur une suite de points fixes.

□

En particulier, le dernier point de ce théorème indique, s'il était encore nécessaire, que cette méthode ne peut capturer que des minima locaux.

Deuxième partie

Optimisation : analyse théorique

Chapitre 5

Intermezzo : calcul différentiel, conditions d'optimalité dans les problèmes de minimisation

Dans ce cours plus théorique nous mettons en place les ingrédients et les outils nécessaires à la mise en place des algorithmes de gradient en dimensions supérieures.

5.1 Prolégomène

Il faut faire très attention aux trois étapes essentielles de l'étude d'un problème d'optimisation :

1. L'**existence** d'un minimiseur. Dans ce cours, l'outil essentiel est la coercivité.
2. L'**identification des candidats à être de minimiseurs**. Dans ce cours, nous utiliserons des critères différentiels et nous identifierons des points critiques, ainsi que des minima locaux.
3. La **caractérisation des minimiseurs**. Dans ce cours, nous utiliserons des hypothèses de convexité.

Il faut garder en tête que ces trois étapes sont très liées, mais peuvent être traitées dans plusieurs ordres possibles. Identifier des minima locaux sans être certain qu'ils existent, c'est une quasi garantie de foncer dans le mur.

5.2 Rappels sur les formules de Taylor

Dans toute la suite, f est une fonction C^2 de $\mathbb{R}^d$ dans $\mathbb{R}$. On note $\{e_k\}_{k=1,\dots,d}$ la base canonique de $\mathbb{R}^d$.

Soit $i \in \{1, \dots, d\}$. La i-ème dérivée partielle de f dans la direction e_i est définie (la limite étant supposée exister) par

$$\frac{\partial f}{\partial x_i} := \lim_{t \to 0} \frac{f(x + te_i) - f(x)}{t}.$$

Si l'on suppose que ces dérivées partielles sont continues en x, alors la fonction f est différentiable au sens classique : pour tout $x \in \mathbb{R}^d$, il existe une application $d_x f \in \mathcal{L}(\mathbb{R}^d, \mathbb{R})$ telle que

$$\forall y \in \mathbb{R}\,, f(x + y) = f(x) + d_x f(y) + o(\|y\|).$$

En particulier, une direction $v \in \mathbb{R}^d$ étant fixée, on a

$$d_x f(v) = \lim_{t \to 0} \frac{f(x + tv) - f(x)}{t},$$

limite qui existe si f est différentiable. Observons alors que

$$d_x f(e_i) = \frac{\partial f}{\partial x_i}. \tag{5.1}$$

Par le théorème de représentation de Riesz, on sait que l'application d_x s'identifie à un vecteur $N_x \in \mathbb{R}^d$, dont il suffit de déterminer les coordonnées dans la base canonique. Puisque

$$\langle N_x, e_i \rangle = d_x f(e_i) \text{ (par Riesz) } = \frac{\partial f}{\partial x_i} \text{ (par (5.1))}$$

on en déduit que

$$N_x = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

et l'on note habituellement ce vecteur ∇f. Le **gradient de f en x** est le vecteur

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

et l'on retiendra que pour une application différentiable f on a

$$f(x + h) = f(x) + \langle \nabla f(x), h \rangle + o\left(\|h\|\right).$$

Il est très important d'être à l'aise avec le gradient, qui interviendra dans les méthodes de descente mais également dans les méthodes de Newton.

Enfin, notons que l'on a, outre le développement limité ci-dessus, la formule de Taylor avec reste intégral

$$\forall x\,, h \in \mathbb{R}^d\,, f(x + h) = f(x) + \int_0^1 \langle \nabla f(x + th), h \rangle dt$$

On passe ensuite à la différentielle seconde. Si l'application $x \mapsto d_x f = \nabla f(x)$ (modulo une identification) est elle-même différentiable en un point $x \in \mathbb{R}^d$, on dit que f est deux fois différentiable en x, et on note $d_x^2 f$ cette différentielle seconde. De même que le gradient était associé à une forme linéaire, la différentielle seconde est associée à une **forme quadratique** [1]

$$d_x^2 f : (h_1, h_2) \mapsto d_x^2 f(h_1, h_2).$$

Il est à noter que, par le même type d'arguments que précédemment, on a une description matricielle de cette différentielle seconde : la différentielle seconde s'identifie à la matrice $\nabla^2 f(x)$ appelée **matrice Hessienne de f en x** [2] et donnée, pour tous indices $i, j \leq d$, par

$$\nabla^2 f(x)_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial}{\partial x_i}\left(\frac{\partial f}{\partial x_j}\right).$$

Par "s'identifie", on entend que

$$\forall h_1\,, h_2 \in \mathbb{R}^d\,, d_x^2 f(h_1, h_2) = \langle \nabla^2 f(x) h_1, h_2 \rangle.$$

Dans le cas des fonctions deux fois différentiable en un point x, on a le développement de Taylor à l'ordre 2

$$f(x+h) = f(x) + \langle \nabla f(x), h \rangle + \frac{1}{2}\langle \nabla^2 f(x) h, h \rangle + o\left(\|h\|^2\right).$$

Un théorème fondamental est le **théorème de Schwarz** qui assure que, si f est deux fois différentiable, alors

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i},$$

de sorte que la Hessienne est symétrique.

On a en outre la formule de Taylor avec reste intégral :

$$\forall x\,, h \in \mathbb{R}^d\,, f(x+h) = f(x) + \langle \nabla f(x), h \rangle + \int_0^1 (1-t)\langle \nabla^2 f(x+th)(h)\,, h \rangle dt. \tag{5.2}$$

Dans ce qui suit, on se préoccupe pas outre mesure des questions de différentiabilité des fonctions considérées, questions qui peuvent rapidement devenir cauchemardesques, et l'on travaillera toujours avec des fonctions de classe au moins C^2. Ceci nous permet en particulier d'identifier la différentielle d'une fonction f avec le gradient de la fonction f, et d'utiliser la hessienne de f pour faire des développements limités d'ordre 2.

1. On rappelle qu'une forme quadratique est un polynôme homogène de degré 2.
2. Que l'on note parfois aussi $\mathrm{Hess}(f)(x)$, ou $H(f)(x)$.

5.3 Identification des candidats : règles de Fermat

Dans toute cette section, on fixe une fonction $f : \mathbb{R}^d \to \mathbb{R}$ deux fois différentiable et l'on étudie le problème d'optimisation

$$\min_{x \in \mathbb{R}^d} f(x). \tag{5.3}$$

On suppose que ce problème a une solution et on fixe $x^* \in \mathbb{R}^d$ tel que

$$f(x^*) = \min_{x \in \mathbb{R}^d} f(x).$$

En général, il est illusoire de pouvoir exhiber x^*, et l'on doit se contenter de restreindre notre recherche. L'objectif de ce cours, qui sert de fondement à la méthode de gradient, est d'obtenir des conditions locales qui permettent parfois de caractériser x^*.

Insistons sur un aspect très important : **ces conditions ne sont que des conditions locales** et ne permettent d'identifier que des **minima locaux**. Pour pouvoir dire qu'un minimum local est un **minimum global** il faut une information justement globale. Le cas le plus simple est celui des fonctions convexes.

5.3.1 Critère de Fermat : condition d'optimalité d'ordre 1

Commençons par le critère d'optimalité d'ordre 1. On insiste encore une fois sur le caractère **nécessaire** et non suffisant de cette règle :

Proposition 5.3.1 (Critère de Fermat)**.** *En un point de minimum x^* on a*

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

Il s'agit exactement de l'analogue du critère unidimensionnel

$$x \text{ minimal pour } f : \mathbb{R} \to \mathbb{R} \ \Rightarrow f'(x) = 0.$$

Preuve de la Proposition 5.3.1. Dans cette preuve se trouve déjà l'idée de la descente de gradient.

On raisonne par l'absurde : si $\nabla f(x^*) \neq 0$, considérons le vecteur $h := -\frac{\nabla f(x^*)}{\|\nabla f(x^*)\|}$. Alors, on a

$$f(x + th) - f(x) = -t\|\nabla f(x^*)\| + o(t^2),$$

quantité strictement négative quand $t \to 0$. □

On introduit la définition suivante :

Définition 5.3.1. *Soit $f \in C^1(\mathbb{R}^d; \mathbb{R})$. Un point $x \in \mathbb{R}^d$ est appelé **point critique de** f si*

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

5.3.2 Un exemple fondamental

Un exemple fondamental de point critique est le suivant : si $A \in M_d(\mathbb{R})$ est une matrice et si $b \in \mathbb{R}^d$ un vecteur fixé, la fonction quadratique définie à partir de A et de b est l'application

$$f_{A,b} : x \mapsto \frac{1}{2}\langle x, Ax\rangle - \langle b, x\rangle.$$

Calculons le gradient de f en un point $x \in \mathbb{R}^d$ fixé : h étant une perturbation donnée, on a

$$\begin{aligned} f_{A,b}(x+h) = &\frac{1}{2}\langle x, Ax\rangle - \langle b, x\rangle \\ &+ \frac{1}{2}\langle h, Ax\rangle + \frac{1}{2}\langle x, Ah\rangle - \langle b, h\rangle \\ &+ \frac{1}{2}\langle h, Ah\rangle. \end{aligned}$$

Si l'on définit l'application linéaire

$$L : h \mapsto \frac{\langle Ax, h\rangle + \langle x, Ah\rangle}{2} - \langle b, h\rangle$$

on a donc

$$f_{A,b}(x+h) = f_{A,b}(x) + L(h) + \frac{1}{2}\langle h, Ah\rangle.$$

L est un bon candidat à être la différentielle de f au point x. Il faut pour garantir cela vérifier que le terme $\langle h, Ah\rangle$ est bien un $o(\|h\|)$. Or, rappelons la définition de la norme matricielle de A :

$$\|A\|_{\text{op}} = \sup_{y \neq 0} \frac{\|Ay\|}{\|y\|}.$$

Ainsi, on a

$$\langle h, Ah\rangle \leq \|h\| \cdot \|Ah\| \leq \|h\|^2 \cdot \|A\|_{\text{op}}$$

et on peut en conclure que la différentielle de $f_{A,b}$ en x est l'application L. Pour déterminer le gradient de f en x il suffit de déterminer l'unique vecteur $\xi \in \mathbb{R}^d$ tel que, pour tout $h \in \mathbb{R}^d$,

$$\langle \xi, h\rangle = L(h).$$

Or, remarquons que puisque, pour tout $y, z \in \mathbb{R}^d$ on a

$$\langle y, Az\rangle = \langle A^T y, z\rangle$$

on peut écrire que, pour tout $h \in \mathbb{R}^d$,

$$L(h) = \frac{\langle Ah, x\rangle + \langle Ax, h\rangle}{2} - \langle b, h\rangle = \left\langle \frac{A + A^T}{2}x - b, h \right\rangle.$$

En particulier,

$$\nabla f_{A,b}(x) = \frac{A + A^T}{2} x - b$$

et donc, si la matrice A est symétrique,

$$\nabla f_{A,b}(x) = Ax - b.$$

De la sorte, on obtient la remarque importante suivante : si $A \in S_d(\mathbb{R})$ et $b \in \mathbb{R}^d$, $x \in \mathbb{R}^d$ est un point critique de $f_{A,b}$ si, et seulement si, x est solution de

$$Ax = b.$$

En d'autres termes, si la fonction $f_{A,b}$ admet un minimum, ce minimum est la solution d'une équation linéaire. Ainsi, *on peut dans certains cas résoudre les systèmes linéaires par des méthodes d'optimisation.*

5.3.3 Condition d'optimalité d'ordre 2

Le problème des conditions d'ordre 1, c'est que même si elles permettent de restreindre l'espace où l'on cherche ces points critiques, elles ne permettent pas de distinguer, comme en dimension 1, les minima et les maxima locaux. Il nous faut donc chercher une information d'ordre 2, qui se trouve contenue dans la matrice Hessienne de f. Supposons que f est une fonction de classe C^2 dont la hessienne est continue.

Si l'on repart de la formule de Taylor à l'ordre 2, on voit qu'en un point critique x^* on a

$$\forall h\,, f(x^* + h) - f(x^*) = \frac{1}{2}\langle \nabla^2 f(x^*)h, h\rangle + o\left(\|h\|^2\right).$$

En particulier, pour que x^* soit un minimiseur, il est nécessaire que

$$\forall h \in \mathbb{R}^d\,, \langle \nabla^2 f(x^*)h, h\rangle \geq 0. \tag{5.4}$$

On a ainsi établi la proposition suivante :

Proposition 5.3.2. *En un point de minimum x^* de f on a*

$$\nabla f(x^*) = 0 \text{ et } \nabla^2 f(x^*) \in S_d^+(\mathbb{R})$$

où $S_d^+(\mathbb{R})$ désigne l'ensemble des matrices symétriques positives.

Insistons ici sur le fait qu'il s'agit d'une condition **nécessaire** d'optimalité locale, mais pas d'une condition **suffisante**. Pour s'en convaincre, il suffit de reprendre l'exemple

$$f : x \mapsto x^3$$

qui vérifie $f'(0) = 0\,, f''(0) = 0 \geq 0$, et pour laquelle 0 n'est pas un minimum local.

En revanche, on peut, en la renforçant, faire de cette condition nécessaire une condition suffisante de minimalité locale :

Proposition 5.3.3. *Soit x^* un point critique de f (i.e. $\nabla f(x^*) = 0$). On suppose que $\nabla^2 f(x^*) \in S_d^{++}(\mathbb{R})$, où $S_d^{++}(\mathbb{R})$ est l'ensemble des matrices symétriques définies positives. En d'autres termes, on suppose qu'il existe une constante $\lambda > 0$ telle que*

$$\forall h \in \mathbb{R}^d \, , \langle \nabla^2 f(x^*)h, h\rangle \geq \lambda \|h\|^2.$$

Alors x^* est un minimum local : il existe $\varepsilon > 0$ tel que

$$\forall x \in \mathbb{R}^d \, , \|x - x^*\| \leq \varepsilon \Rightarrow f(x) > f(x^*).$$

On laisse cette proposition à titre d'exercice.

Notons que bien évidemment la matrice $\nabla^2 f$ peut être définie négative, auquel cas x^* est un maximum local, et qu'elle peut n'être ni positive, ni négative : s'il existe $h_1 \, , h_2 \in \mathbb{R}^d$ tels que

$$\langle \nabla^2 f(x^*)h_1, h_1\rangle > 0 \, , \langle \nabla^2 f(x^*)h_2, h_2\rangle < 0$$

alors x^* n'est ni un maximum local, ni un minimum local, et l'on parle de **point selle**. Ceci signifie qu'il existe des directions de perturbations dans lesquelles f décroît, et d'autres où f croît. Si l'on prend par exemple la fonction

$$f : (x, y) \mapsto \frac{1}{2} \cdot (x^2 - y^2)$$

on voit que le point $(0, 0)$ est un point selle. En effet, on a

$$\nabla^2 f(0, 0) = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Enfin, il se peut que la forme quadratique $\nabla^2 f$ soit nulle le long de certaines directions, en d'autres termes, qu'il existe $h \in \mathbb{R}^d \, , h \neq 0$ tel que

$$\langle \nabla^2 f(x^*)h, h\rangle = 0.$$

C'est typiquement le cas de la fonction

$$f : (x, y) \mapsto x^2 - y^4$$

dont la hessienne en 0 vaut

$$\nabla^2 f(0, 0) = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}.$$

Dans ce cas-là, on ne peut rien dire, et l'on doit, ou bien chercher des arguments d'ordre 3, ou bien poursuivre les calculs à la main.

5.3.4 En pratique

Néanmoins, en pratique, travailler avec des matrices symétriques définies positives générales peut s'avérer délicat, et il est certainement plus commode d'avoir affaire à des matrices diagonales.

Rappelons le théorème spectral :

Théorème 5.3.1 (Théorème spectral, admis)**.** *Toute matrice symétrique réelle est diagonalisable en base orthonormée réelle. En d'autres termes, il existe une matrice orthogonale $P \in O_d(\mathbb{R})$ (*i.e. $PP^T = \mathrm{I}_d$*) et une matrice $D \in D_d(\mathbb{R})$, l'ensemble des matrices diagonales, telles que*

$$M = P^T D P.$$

On en déduit donc qu'une matrice symétrique est positive si, et seulement si, toutes ses valeurs propres sont positives, et qu'elle est définie positive si, et seulement si, toutes ses valeurs propres sont strictement positives. Néanmoins, le calcul d'une base de diagonalisation et des valeurs propres d'une matrice peut s'avérer très ardu en toute dimension. En dimension 2 cependant, on a un critère extrêmement simple, qui sera largement utilisé en Travaux Dirigés :

Proposition 5.3.4 (Critère pour l'appartenance à $S_2^+(\mathbb{R})$ et $S_2^{++}(\mathbb{R})$)**.** *Soit $M \in S_2(\mathbb{R})$. Si*

$$\det(M) > 0\,, \mathrm{Tr}(M) > 0$$

alors $M \in S_2^{++}(\mathbb{R})$.

Si

$$\det(M) \geq 0\,, \mathrm{Tr}(M) \geq 0$$

alors $M \in S_2^{+}(\mathbb{R})$.

Cette proposition se démontre facilement en passant par la base d'orthonormalisation donnée par le théorème 5.3.1. En dimensions supérieures, ce critère est évidemment faux : il suffit pour s'en convaincre de considérer la matrice

$$M := \begin{pmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

5.4 Unicité éventuelle des points critiques : notions de convexité

Dans ce paragraphe, nous étudions une condition raisonnable pour avoir unicité des points critiques, la notion de convexité.

Définition 5.4.1. *Une fonction $f : \mathbb{R}^d \to \mathbb{R}$ est **convexe** si*

$$\forall x, y \in \mathbb{R}^d\,, \forall t \in [0;1]\,, f\left((1-t)x + ty\right) \leq (1-t)f(x) + tf(y).$$

*On dit que f est **strictement convexe** si l'inégalité ci-dessus est stricte pour tout $x \neq y$ et pour tout $t \in]0;1[$.*

Cette définition, qui revient à demander que le graphe de la fonction f soit au dessus de chacune de ses tangentes, est une notion géométrique. Pour obtenir des caractérisations plus tractables, on doit supposer plus de régularité sur la fonction f. Quoiqu'une fonction convexe f soit lipschitzienne, et qu'elle soit en conséquence différentiable presque partout (Théorème de Rademacher), on va travailler directement avec des fonctions f de classe $\mathscr{C}^2$.

Il est bien connu qu'en dimension 1, une fonction dérivable f est convexe si, et seulement si, sa fonction dérivée f' est une fonction monotone croissante, ce qui est équivalent, dès que f est de classe $\mathscr{C}^2$, à la positivité de f''. En dimensions supérieures, les notions de convexité peuvent vite devenir pénibles à manipuler, mais nous avons la caractérisation suivante :

Proposition 5.4.1. *Soit $f \in \mathscr{C}^2(\mathbb{R}^d;\mathbb{R})$. Les trois propriétés suivantes sont équivalentes :*

i) f est convexe.

ii) ∇f est monotone au sens suivant :

$$\forall x\,, y \in \mathbb{R}^d\,, \langle \nabla f(y) - \nabla f(x), y - x\rangle \geq 0.$$

iii) Pour tout $x \in \mathbb{R}^d$, $\nabla^2 f(x) \in S_d^+(\mathbb{R})$.

Ce fait est classique et ne sera pas démontré ici.

(Stricte) positivité de la hessienne, (stricte) convexité de la fonction Se pose la question de savoir si l'on peut caractériser une fonction strictement convexe à l'aide d'une propriété de stricte positivité de sa hessienne. Une implication est claire, au vu du lemme suivant :

Lemme 5.4.1. *Soit $f \in \mathscr{C}^2(\mathbb{R}^d;\mathbb{R})$ telle que, pour tout $x \in \mathbb{R}^d$, $\nabla^2 f(x) \in S_d^{++}(\mathbb{R})$. Alors f est strictement convexe.*

La réciproque est néanmoins fausse : si l'on considère la fonction $x \mapsto x^4$, alors la fonction f est strictement convexe, et pourtant $f''(0) = 0$. On pourra objecter que c'est un exemple particulier, puisque la hessienne de f (ici sa dérivée seconde) ne s'annule qu'en un unique point. Néanmoins il est facile de construire, à partir des exercices du TD n°2, un exemple d'une fonction strictement convexe telle que f'' s'annule une infinité de foix. Si l'on définit

$$f : x \mapsto \frac{x^2}{2} - \cos(x)$$

alors il a été démontré dans le TD n°2 que $f'(x) = x + \sin(x)$ était une fonction strictement croissante. En particulier, f' est strictement croissante, et f est ainsi strictement convexe. Par ailleurs, $f''(x) = 1 + \cos(x)$ s'annule une infinité de fois.

Du point de vue de l'optimisation, la propriété fondamentale des fonctions convexes est la suivante :

Proposition 5.4.2. *Soit $f \in \mathscr{C}^2(\mathbb{R}^d;\mathbb{R})$ une fonction strictement convexe telle que, pour tout $x \in \mathbb{R}^d$, $\nabla^2 f(x) \in S_d^{++}(\mathbb{R})$. Alors f admet au plus un point critique x^*.*

Remarque 5.4.1. *La stricte convexité est évidemment nécessaire, il suffit pour s'en rendre compte d'étudier la fonction*

$$f : x \mapsto \begin{cases} e^{x^2} - 1 \ si \ x \leq 0 \\ 0 \ si \ 0 \leq x \leq 1 \\ e^{(x-1)^2} - 1 \ si \ x \geq 1 \end{cases}$$

Alors tout point $x \in [0;1)$ est critique.

Par ailleurs, ce résultat ne donne évidemment pas de résultat d'existence des points critiques. En effet, si l'on considère la fonction $x \mapsto e^x$, c'est évidemment une fonction strictement convexe qui n'admet pas de point critique.

Ainsi, dans le cas d'une fonction strictement convexe, identifier un point critique revient à identifier l'unique candidat possible pour être un extremum local. En outre, pour une fonction suffisamment convexe et de classe $\mathscr{C}^2$, tout point critique x^* est un minimiseur global, comme indiqué par la proposition suivante :

Proposition 5.4.3. *Soit $f \in \mathscr{C}^2(\mathbb{R}^d;\mathbb{R})$ telle que, pour tout $x \in \mathbb{R}^d$, $\nabla^2 f(x) \in S_d^{++}(\mathbb{R})$. Si x^* est un point critique de f, alors x^* est un minimiseur global de f.*

Preuve de la Proposition 5.4.3. On applique la formule de Taylor avec reste intégral (5.2), et on utilise simplement le fait que la hessienne de f est toujours symétrique définie positive. □

Néanmoins, observons les choses suivantes :

1. Il est possible qu'une fonction convexe n'admette pas de point critique, comme montré par l'étude de la fonction $x \mapsto e^{-x}$.
2. Plus important, même si l'équation $\nabla f(x) = 0$ a une unique solution x^*, il ne s'ensuit pas que x^* est un extrémum local. Par exemple, la fonction $f : x \mapsto x^3$ a un unique point critique, qui n'est pas un extremum local.

5.5 Problèmes de minimisation dans $\mathbb{R}^n$: existence de minimiseurs

Dans ce dernier paragraphe, nous présentons finalement les conditions les plus simples à requérir de la fonction f pour que le problème d'optimisation

$$\inf_{x \in \mathbb{R}^d} f(x)$$

admette une solution. Dans tout ce qui suit, f est une fonction continue.

La méthode naturelle pour obtenir l'existence d'un minimiseur est d'employer ce qui dans le jargon est appelé **méthode directe du calcul des variations** ; derrière ce nom se cache simplement l'utilisation de la définition de l'infimum d'une fonction : en notant

$$m := \inf_{\mathbb{R}^d} f$$

on sait qu'il existe une suite $\{x_k\}_{k\in\mathbb{N}} \in \left(\mathbb{R}^d\right)^{\mathbb{N}}$ telle que

$$f(x_k) \underset{k\to\infty}{\to} m.$$

Puisque la fonction f est continue, il suffit pour obtenir l'existence d'un minimiseur d'obtenir l'existence d'une valeur d'adhérence x^* de la suite $\{x_k\}_{k\in\mathbb{N}}$. Mais l'existence d'une telle valeur d'adhérence est évidemment garantie si l'on sait *a priori* que les éléments de la suite vivent dans un ensemble compact. Par exemple, ce n'est pas le cas si l'on prend la fonction $f : x \mapsto e^{-x}$: aucune suite minimisante n'a de valeur d'adhérence.

L'hypothèse de **coercivité** permet de répondre à cette question de manière synthétique :

Définition 5.5.1 (Fonction coercive)**.** *On dit qu'une fonction continue $f : \mathbb{R}^d \to \mathbb{R}$ est **coercive** si, pour tout $M \in \mathbb{R}$, l'ensemble de niveau*

$$\Omega_M := \{x\,, f(x) \leq M\}$$

est compact.

Puisque f est une fonction continue, Ω_M est compact si et seulement si il est borné, son caractère fermé étant évident.

On a la proposition suivante :

Proposition 5.5.1. *Soit $f \in \mathscr{C}^0(\mathbb{R}^d;\mathbb{R})$ une fonction coercive. Le problème d'optimisation*

$$\inf_{x\in\mathbb{R}^d} f(x)$$

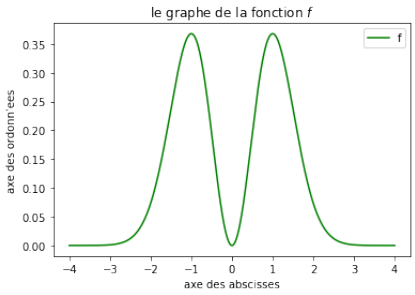
admet une solution x^*.

Preuve de la Proposition 5.5.1. On considère une suite minimisante $\{x_k\}_{k\in\mathbb{N}}$. Si x_0 n'est pas un minimiseur de f (auquel cas on aurait en fait terminé) alors, pour tout $k \in \mathbb{N}$ suffisamment grand, on a

$$f(x_k) \leq f(x_0).$$

En d'autres termes, pour tout k suffisamment grand, $\{x_k\}_{k\in\mathbb{N},k\gg 1} \subset \Omega_{f(x_0)} := \left\{x \in \mathbb{R}^d\,, f(x) \leq f(x_0)\right\}$ qui, par hypothèse, est un ensemble compact. On peut donc extraire une sous-suite qui converge vers x^*. Puisque f est continue, x^* est un minimiseur de f. □

La coercivité est une condition suffisante pour avoir l'existence d'un minimiseur, mais elle n'est pas nécessaire : certaines suites minimisantes peuvent avoir des valeurs d'adhérence et d'autres non. C'est le cas par exemple de la fonction $f : x \mapsto x^2 e^{-x^2}$, dont le graphe est donné sur la figure suivante :



Un tableau récapitulatif

Si l'on récapitule tout ce que l'on a vu jusqu'à présent, on a le tableau suivant :

Propriété de la fonction	Propriété obtenue	Propriété manquante
Strictement convexe, $\mathscr{C}^1$	Unicité des points critiques	Existence des points critiques
Coercive	Existence d'un minimiseur	Caractérisation univoque des points critiques

Ainsi, on a le schéma suivant :

f coercive+f strictement convexe $\Rightarrow$ existence et unicité d'un minimiseur x^*.

5.6 L'exemple paradigmatique

L'exemple le plus courant et le plus utile de fonction coercive et strictement convexe est celui d'une fonction modelée sur une matrice symétrique définie positive : si $A \in S_d^{++}(\mathbb{R})$, si $b \in \mathbb{R}^d$ est un vecteur fixé, si l'on pose

$$f_{A,b} : x \mapsto \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$$

alors la fonction $f_{A,b}$ est strictement convexe, et elle est coercive.

1. Preuve de la convexité : il est immédiat devoir que $\nabla^2 f = A$, et donc que f est strictement convexe.

2. Preuve de la coercivité : la matrice A étant définie positive, sa plus petite valeur propre $\lambda_1(A)$ est strictement positive. En particulier, par l'inégalité de Cauchy-Schwarz,

$$\forall x \in \mathbb{R}^d, f_{A,b}(x) \geq \frac{\lambda_1(A)}{2}\|x\|^2 - \|b\| \cdot \|x\|,$$

de sorte que la fonction est coercive.

Si nous précisons cette fonction, c'est qu'elle est la plus important à partir du chapitre suivant, où nous attaquons les descentes de gradient et les algorithmes de gradient conjugué, en particulier pour les fonctions de la forme $f_{A,b}$.

Troisième partie

Méthode de gradients

Chapitre 6

Descente de gradient II : première analyse en dimensions supérieures

Nous étudions dans ce cours les méthodes de descente de gradient à pas constant, pour la minimisation de fonctions $f : \mathbb{R}^d \to \mathbb{R}$. Toutes les illustrations seront faites dans le cas $d = 2$, qui permet de visualiser ce qui se produit, mais l'étude théorique est valable en toute dimension.

6.1 Heuristique de la méthode

6.1.1 Choix de la direction de plus grande descente

Nous l'avons vu dans le cadre de la descente de gradient unidimensionnel, l'idée est de suivre une direction dans laquelle la fonction décroît. En d'autres termes, partant d'un point initial x_0, on veut choisir une direction $v \in \mathbb{S}^{d-1}$ et un pas $\tau > 0$ tel que

$$f(x_0 + \tau v) < f(x_0).$$

Si en dimension 1 on n'avait le choix qu'entre "aller à gauche" et "aller à droite", en dimensions supérieures il est nécessaire de choisir entre d directions possibles. Déterminer cette direction est plutôt aisé : si l'on suppose que l'on part d'un point x_0 et qu'un petit pas $\tau > 0$ est donné, si l'on prend l'approximation

$$f(x_0 + \tau v) \approx f(x_0) + \tau \langle v, \nabla f(x_0) \rangle$$

et que l'on a envie de faire décroître la fonction f le plus rapidement possible, on voit que l'on doit choisir comme direction v la solution du problème d'optimisation

$$\inf_{v \in \mathbb{S}^{d-1}} \langle v \, , \nabla f(x_0) \rangle. \tag{6.1}$$

Si $\nabla f(x_0) = 0$, en d'autres termes, si l'on se trouve en un point critique, alors ce problème est trivial. Si en revanche $\nabla f(x_0) \neq 0$, l'inégalité de Cauchy-Schwarz

assure que l'unique solution v^* de (6.1) est

$$v^* := -\frac{\nabla f(x_0)}{\|\nabla f(x_0)\|}.$$

Il semble donc pertinent de choisir cette direction-là et de s'y déplacer. Autrement dit, nous créerons une suite d'itérations de la manière suivante :

$$\forall k \in \mathbb{N}\,, x_{k+1} = x_k - \tau \nabla f(x_k).$$

Attention : dans tout ce cours, nous supposerons la taille du pas fixe, c'est-à-dire indépendante de l'itération. Dans les cours suivants, nous nous intéresserons à des descentes de gradient à pas variables.

6.1.2 Interprétation géométrique de la direction de plus grande descente

Pour déterminer la direction dans laquelle nous allons nous déplacer et représenter, géométriquement, cette direction, il nous faut donner une interprétation géométrique des ensembles de niveau, et du gradient de la fonction f.

Définition 6.1.1 (Surfaces de niveau)**.** *Si $\lambda \in \mathbb{R}$ est un réel fixé, la **surface de niveau λ de f** est l'ensemble $\Sigma_\lambda := f^{-1}(\{\lambda\})$. En d'autres termes,*

$$\Sigma_\lambda = \{x \in \mathbb{R}^d\,, f(x) = \lambda\}.$$

On a déjà vu en travaux pratiques comment représenter ces surfaces de niveau pour une fonction de deux variables. L'observation qualitative clé est la suivante : **la direction de plus grande descente est orthogonale aux lignes de niveau** de la fonction f. Ceci s'exprime par la proposition suivante :

Proposition 6.1.1. *Pour tout $\lambda \in \mathbb{R}$, pour tout $x \in \Sigma_\lambda$ tel que $\nabla f(x) \neq 0$, le vecteur $\nabla f(x)$ est perpendiculaire à la surface de niveau Σ_λ.*

Preuve de la proposition 6.1.1. Il suffit de remarquer que pour toute application $\gamma : \mathbb{R} \to \Sigma_\lambda$ de classe $\mathscr{C}^1$, la différentiation de l'équation $f(\gamma(t)) = \lambda$ implique

$$\langle \nabla f(\gamma(t)), \gamma'(t) \rangle = 0;$$

en particulier, le gradient est orthogonal à tous les vecteurs tangents à la surface, ce qui est la définition de l'orthogonalité. □

6.2 Description de l'algorithme

Une fois ces quelques petites idées mises en place, on peut passer à la définition de l'algorithme ; cet algorithme prend en argument une point initial x_0, un pas $\tau > 0$ fixé, un nombre maximal d'itérations `Niter` et une tolérance `tol`.

Algorithm 3 Structure type de l'algorithme de direction de plus grande descente

```
def algo(f,df,x0,alpha,tol=1e-3,Niter=10)
Initialisation
xn=x0   On définit le premier élément de la liste
L=[]   On définit une liste que l'on augmentera récursivement, et qui contiendra tous les éléments générés
for n in range(Niter):
if ...< tol : then
  return xn,L
else
  L.append(xn)
  xn=f(xn)-alpha*df(xn)
end if
print("Erreur, l'algorithme n'a pas convergé
après",IterMax,"itérations")
```

La structure typique de l'algorithme est donc la suivante : Notez qu'ici on a implicitement supposé que le gradient de f, noté df, était explicitement connu. Dans le TP n°3, nous verrons comment coder le gradient discrétisé quand l'expression explicite du gradient n'est pas accessible.

6.3 Convergence de l'algorithme : le cas quadratique

Nous divisons l'étude de la convergence de l'algorithme en deux parties. La philosophie est la suivante : si l'on se rappelle qu'en dimension 1 le caractère **non-dégénéré** des minima locaux de f (en d'autres termes, l'hypothèse que $f''(x^*) > 0$) nous permettait d'obtenir des vitesses et des taux de convergence, on se doute que la non-dégénerescence des minima de f en dimensions d, c'est-à-dire le caractère défini positif de la hessienne $\nabla^2 f(x^*)$ va également être crucial. Localement, au voisinage de x^*, nous pourrons approcher f par son développement de Taylor à l'ordre 2, c'est-à-dire que nous utiliserons des développements qui disent essentiellement que

$$f(x) \approx f(x^*) + \frac{1}{2}\langle \nabla^2 f(x^*)(x - x^*), (x - x^*)\rangle.$$

Il est donc naturel de commencer l'étude de la convergence de l'algorithme de descente de gradient à pas fixe dans le cas d'une fonction quadratique.

6.3.1 Une première analyse de la convergence : identification des éléments constitutifs

Dans toute la suite de ce paragraphe, on fixe une matrice $A \in S_d^{++}(\mathbb{R})$, un vecteur $b \in \mathbb{R}^d$ et l'on définit

$$f_{A,b} : x \mapsto \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

Nous avons vu lors du cours précédent que cette fonction était strictement convexe. En outre, son gradient est

$$\nabla f_{A,b}(x) = Ax - b$$

et sa hessienne

$$\nabla^2 f_{A,b}(x) = A.$$

Remarque 6.3.1 (Descente de gradient et résolution de systèmes linéaires)**.** *Un autre intérêt de la minimisation des fonctions de la forme $f_{A,b}$ par descente de gradient est le suivant : le minimiseur x^* de $f_{A,b}$ (qui existe, par coercivité, et qui est unique, par convexité) est l'unique solution de*

$$Ax^* = b.$$

Les algorithmes de gradient peuvent en pratique s'avérer beaucoup plus rapide que l'implémentation d'un pivot de Gauss pour la résolution de ce système linéaire.

De la remarque 6.3.1 on retient que le minimiseur x^* de $f_{A,b}$ est la solution de

$$Ax^* = b \Leftrightarrow x^* = A^{-1}b.$$

Identifions désormais les éléments constitutifs de la preuve de convergence de l'algorithme. On sait que si l'on se fixe un pas $\tau > 0$ la suite des itérés est donnée par

$$x_{k+1} = x_k - \tau A x_k + \tau b$$

de sorte que

$$x_{k+1} - x^* = x_k - \tau A x_k + \tau A x^* - x^* = x_k(\mathrm{I}_d - \tau A) - (\mathrm{I}_d - \tau A)x^* = (\mathrm{I}_d - \tau A)\,(x_k - x^*)\,.$$

Si l'on veut utiliser les critères de convergence linéaire ou quadratique vus lors du premier cours, il est donc nécessaire de pouvoir contrôler la **norme d'opérateur** de la matrice $\mathrm{I}_d - \tau A$.

On rappelle à toutes fins utiles la définition de la norme d'opérateur d'une matrice :

Définition 6.3.1 (Norme d'opérateur)**.** *Soit $M \in M_d(\mathbb{R})$ et $\|\cdot\|$ une norme sur $\mathbb{R}^d$. La norme d'opérateur de M induite par $\|\cdot\|$ est*

$$\|M\|_{\mathrm{op}} := \sup_{x \in \mathbb{R}^d \setminus \{0\}} \frac{\|Mx\|}{\|x\|}.$$

Nous étudierons en TD des normes d'opérateurs induites par des normes autres que la norme euclidienne standard. Néanmoins, dans toute la suite du cours, nous ne travaillerons qu'avec la norme d'opérateur induite par la norme euclidienne standard, et nous noterons simplement $\|\cdot\|$ pour la norme euclidienne, et $\|\cdot\|_{\mathrm{op}}$ pour la norme induite.

Dans le cas des matrices symétriques réelles et pour la norme d'opérateur induite par la norme euclidienne, cette norme d'opérateur est contrôlée par les valeurs propres de la matrice M. On rappelle la proposition suivante, qui sera démontrée en travaux dirigés :

Proposition 6.3.1. *Soit $M \in S_d(\mathbb{R})$ et soient $\lambda_1(M) \leq \lambda_2(M) \leq \cdots \leq \lambda_d(M)$ ses valeurs propres. Alors on a :*

1. *Principe de Courant-Fisher :*
$$\lambda_1(M) = \inf_{\|x\|=1} \langle Mx, x\rangle \text{ et } \lambda_d(M) = \sup_{\|x\|=1} \langle Mx, x\rangle.$$
2. *Norme d'opérateur et valeurs propres : la norme d'opérateur de la matrice M est donnée par*
$$\|M\|_{\mathrm{op}} = \max\left(|\lambda_1(M)|, |\lambda_d(M)|\right).$$

Le résultat de cette analyse, c'est qu'une maîtrise des rudiments de l'algèbre linéaire est nécessaire avant d'aller plus loin dans l'analyse des algorithmes de minimisation.

Revenons au problème de la convergence de la descente de gradient à pas constant τ. On rappelle que

$$x_{k+1} - x^* = (\mathrm{I}_d - \tau A)(x_k - x^*),$$

de sorte que

$$\|x_{k+1} - x^*\| \leq \max\left(|1 - \tau\lambda_1(A)|, |1 - \tau\lambda_d(A)|\right) \|x_k - x^*\|.$$

On en déduit que la suite $\{x_k\}_{k\in\mathbb{N}}$ converge vers x^* si

$$\max\left(|1 - \tau\lambda_1(A)|, |1 - \tau\lambda_d(A)|\right) < 1,$$

ce qui implique

$$\tau\lambda_d(A) < 2.$$

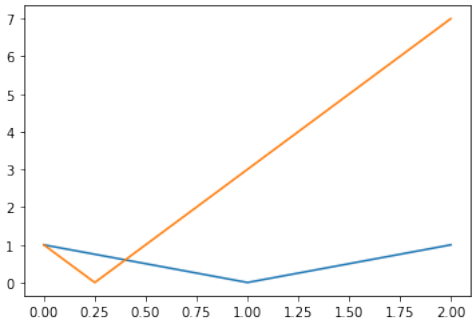
On a donc établi le théorème suivant :

Théorème 6.3.1. *[Convergence de la descente de gradient à pas fixe] La descente de gradient à pas fixe converge linéairement vers un minimiseur x^* de $f_{A,b}$ si le pas τ vérifie*

$$\tau\lambda_d < 2.$$

Cette convergence linéaire est à taux

$$\alpha(\tau) := \max\left(|1 - \tau\lambda_1(A)|, |1 - \tau\lambda_d(A)|\right).$$



On a donc obtenu un résultat général, qui donne un taux de convergence explicite. C'est une convergence lente que l'on peut essayer d'optimiser. Pour cela, il suffit de choisir un τ qui minimise $\alpha(\tau)$, en d'autres termes, de déterminer τ^* tel que

$$\alpha(\tau^*) = \min \alpha(\tau).$$

Or, la fonction α est définie comme le maximum de deux fonctions. La première de ces fonctions vaut

$$\beta_1(\tau) = |1 - \tau\lambda_1(A)|$$

et la seconde

$$\beta_d(\tau) = |1 - \tau\lambda_d(A)|.$$

Pour $k = 1, d$, la fonction β_k est décroissante sur $[0; 1/\lambda_k(A)]$ puis croissante.

Le pas qui assure un taux optimal est

$$\tau^* = \frac{2}{\lambda_1 + \lambda_d}$$

qui correspond à un taux

$$\alpha(\tau^*) = \frac{\lambda_d(A) - \lambda_1(A)}{\lambda_d(A) + \lambda_1(A)}.$$

Représentons graphiquement la fonction $\max(\beta_1, \beta_d)$ dans le cas $\lambda_1(A) = 1$ et $\lambda_d(A) = 4$:

On voit que le minimum du maximum de ces deux fonctions est atteint au point τ^* où

$$\beta_1(\tau^*) = \beta_d(\tau^*) \Leftrightarrow (1 - \tau^*\lambda_1(A))^2 = (1 - \tau^*\lambda_d(A))^2.$$

En développant ces deux polynômes on voit qu'il faut avoir

$$-\frac{2}{\tau^*}(\lambda_1(A) - \lambda_d(A)) = \lambda_d(A)^2 - \lambda_1(A)^2 \Leftrightarrow \tau^* = \frac{2}{\lambda_d(A) + \lambda_1(A)}.$$

Pour ce τ^* optimal on a

$$\alpha(\tau^*) = \frac{\lambda_d(A) - \lambda_1(A)}{\lambda_1(A) + \lambda_d(A)}.$$

Par cette petite analyse nous avons établi le théorème suivant :

Théorème 6.3.2 (Convergence de la descente de gradient à pas fixe optimal)**.** *Le pas $\tau^* = \frac{2}{\lambda_1(A)+\lambda_d(A)}$ est optimal pour la descente de gradient à pas fixe. L'algorithme converge linéairement vers un minimiseur au taux (optimal)*

$$\alpha(\tau^*) = \frac{\lambda_d(A) - \lambda_1(A)}{\lambda_d(A) + \lambda_1(A)}.$$

6.4 Petit aparté : conditionnement des matrices

Dans le théorème précédent, nous avons établi le taux de convergence optimal pour une descente de gradient à pas fixe. On voit que, pour que la descente de gradient converge bien, il faut que $\lambda_d(A)$ et $\lambda_1(A)$ soient du même ordre de grandeur, autrement dit que

$$\frac{\lambda_d(A)}{\lambda_1(A)} \approx 1.$$

Le quotient de ces deux valeurs propres porte un nom :

Définition 6.4.1 (Conditionnement d'une matrice)**.** *Soit $A \in S_d^{++}(\mathbb{R})$. Le* ***nombre de conditionnement*** *de la matrice A est la quantité*

$$\mathrm{cond}(A) := \frac{\lambda_d(A)}{\lambda_1(A)}.$$

Cette notion sera vue sous toutes les coutures dans la feuille de TD n°5 mais retenons qu'en général plus la matrice A est mal conditionnée plus la convergence de l'algorithme de gradient sera mauvaise.

Remarque 6.4.1. *On a l'expression alternative*

$$\mathrm{cond}(A) = \|A\|_{\mathrm{op}} \cdot \|A^{-1}\|_{\mathrm{op}}.$$

Cette simple réecriture permet de généraliser la notion de conditionnement à celle de "conditionnement induit par une norme sur $\mathbb{R}^d$".

Donnons une inteprétation géométrique du mauvais conditionnement d'une matrice, en fonction des lignes de niveau, en considérant, en deux dimensions, une matrice $A \in S_2^{++}(\mathbb{R})$ et

$$f_{A,0} : \mathbb{R}^2 \ni x \mapsto \frac{1}{2}\langle Ax, x\rangle.$$

On peut, à changement de base orthonormale près, supposer que A est une matrice diagonale :

$$A = \begin{pmatrix} \lambda_1(A) & 0 \\ 0 & \lambda_2(A) \end{pmatrix}.$$

L'ensemble de niveau disons $r_0 > 0$ de la fonction $f_{A,0}$ est donné par

$$\Sigma(A, r_0) = \{(x_1, x_2) \in \mathbb{R}^2\,, \lambda_1(A)x_1^2 + \lambda_2(A)x_2^2 = r_0\}.$$

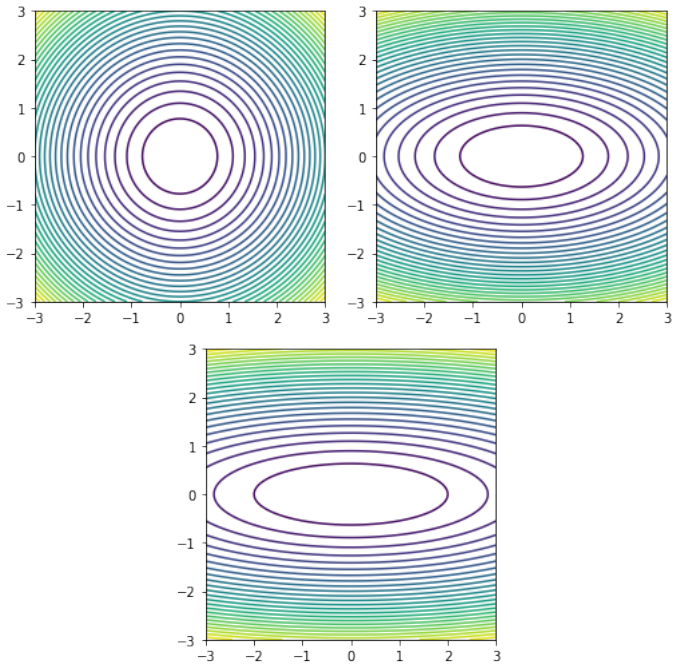


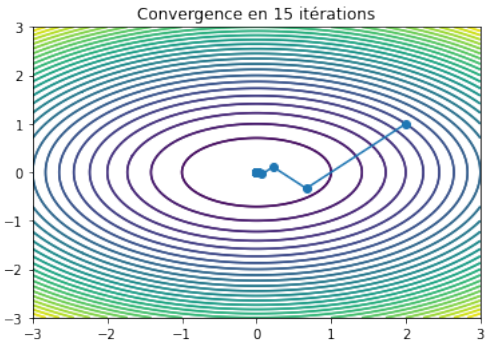
FIGURE 6.1 – De droite à gauche : $\lambda_1(A) = \lambda_2(A) = 1$, $\lambda_1(A) = 1\,, \lambda_2(A) = 4$, $\lambda_1(A) = 1\,, \lambda_2(A) = 10$.

On reconnaît l'équation d'une ellipse d'excentricité $\sqrt{1 - \frac{\lambda_1(A)^2}{\lambda_d(A)^2}}$. Voyons ce que cela donne géométriquement en représentant les ensembles de niveau de la fonction $f_{A,b}$: dans tous les exemples suivant on prend $b = 0$.

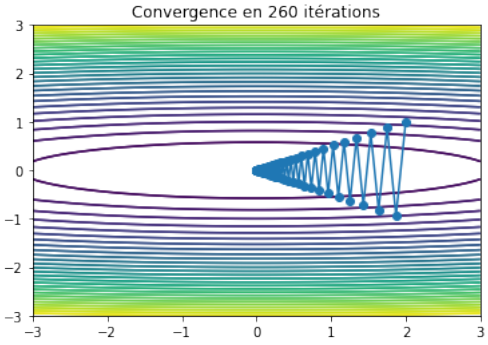
On voit qu'un mauvais conditionnement correspond à des lignes de niveau plus aplaties. Ainsi, on peut interpréter le conditionnement comme le "degré d'applatissement" des ensembles de niveau de la fonction. Plus ils sont homogènes (*i.e.* proche d'un cercle) meilleure est la convergence de l'algorithme. À l'inverse, plus ils sont plats, plus la convergence est désastreuse.

6.5 Illustration numérique du bon et du mauvais conditionnement d'une matrice

Numériquement, on observe bien ce phénomène : voici la suite des itérations pour les paramètres $\lambda_1(A) = 1, \lambda_2(A) = 2$ (donc une matrice plutôt bien conditionnée), $b = 0$ et un point d'initialisation $x_0 = (2, 1)$:



Si on prend maintenant $\lambda_1(A) = 1$ et $\lambda_2(A) = 30$ (donc, une matrice très mal conditionnée) on observe un ralentissement dramatique :



6.6 Convergence de l'algorithme : le cas général

Une petite précision : quand nous disons "cas général" nous parlons simplement d'un cas plus général que celui évoqué précédemment, qui est celui d'une fonction non-linéaire, mais dont les minima locaux sont dégénérés au sens vu au cours précédent.

Autrement dit, le problème d'intérêt ici est le suivant : on considère une fonction $f \in \mathscr{C}^3(\mathbb{R}^d; \mathbb{R})$, un point x^* qui est un **minimum local non dégénéré de** f et l'on travaille sur la suite des itérations fournie par la méthode de descente de gradient issue d'une point initial x_0.

On a en particulier supposé que

$$\nabla f(x^*) = 0\,, \nabla^2 f(x^*) \in S_d^{++}(\mathbb{R}).$$

La première idée qui vient (et qui fonctionne) pour établir la convergence de l'algorithme est la suivante (nous énoncerons le théorème recherché au terme de notre analyse) : on écrit que pour tout $x, y \in \mathbb{R}^d$ on a

$$\nabla f(y) = \nabla f(x) + \int_0^1 \nabla^2 f(x + t(y - x)) \cdot (y - x) dt.$$

En particulier, avec $x = x^*$ on obtient

$$\nabla f(y) = \int_0^1 \nabla^2 f(x^* + t(y - x^*)) \cdot (y - x^*) dt.$$

Grâce à cette réécriture, observons que la suite des itérations de la méthode de descente de gradient vérifie, pour tout $k \in \mathbb{N}$,

$$x_{k+1} - x^* = (x_k - x^*) - \tau \nabla f(x_k) = (x_k - x^*) - \tau \int_0^1 \nabla^2 f(x^* + t(x_k - x^*)) \cdot (x_k - x^*) dt.$$

Cette expression se réécrit, matriciellement

$$x_{k+1} - x^* = \left(\mathrm{I}_d - \tau \int_0^1 \nabla^2 f(x^* + t(x_k - x^*)) dt \right) (x_k - x^*) \,.$$

On introduit la matrice

$$M_k := \mathrm{I}_d - \tau \int_0^1 \nabla^2 f(x^* + t(x_k - x^*)) dt.$$

On voudrait utiliser les informations que l'on a sur $\nabla^2 f(x^*)$. On utilise alors le lemme clé suivant :

Lemme 6.6.1. *Pour tout $A, B \in S_d(\mathbb{R})$, on a*

$$\max(|\lambda_1(A) - \lambda_1(B)|, |\lambda_d(A) - \lambda_d(B)|) \leq \|A - B\|_{\mathrm{op}}.$$

Preuve du Lemme 6.6.1. Soit x_B un vecteur propre de norme 1 de la matrice B associé à la valeur propre $\lambda_1(B)$. Alors

$$\lambda_1(A) \leq \langle A x_B, x_B \rangle = \langle (A - B) x_B, x_B \rangle + \lambda_1(B) \leq \|A - B\|_{\mathrm{op}} \cdot \|x\|^2 + \lambda_1(B),$$

ce qui conclut la preuve. En utilisant, inversement, la formulation de $\lambda_d(B)$ comme un quotient de Rayleigh, on obtient la même conclusion pour la d-ième valeur propre. □

Mais cette continuité pour la norme d'opérateur des valeurs propres est alors cruciale ! En effet, notons

$$\ell_0(x^*) := \frac{\lambda_1(\nabla^2 f(x^*))}{2}, \ell_1(x^*) := 2\lambda_d(\nabla^2 f(x^*)).$$

Alors, par le Lemma 6.6.1 on en déduit qu'il existe $\varepsilon > 0$ tel que, pour tout $x \in \mathbb{B}(x^*, \varepsilon)$,

$$\ell_0(x^*) \leq \lambda_1(\nabla^2 f(x)) \leq \lambda_d(\nabla^2 f(x)) \leq \ell_1(x^*).$$

En particulier, si l'on pose

$$M(x) := \mathrm{Id} - \tau \int_0^1 \nabla^2 f(x^* + t(x - x^*)) dt$$

on récupère que, pour le même $\varepsilon > 0$, et pour tout $x \in \mathbb{B}(x^*, \varepsilon)$, on a

$$1 - \tau\ell_1(x^*) \leq \lambda_1(M(x)) \leq \lambda_d(M(x)) \leq 1 - \tau\ell_0(x^*),$$

et ainsi on en déduit que, si $x_k \in \mathbb{B}(x^*, \varepsilon)$,

$$\|x_{k+1} - x^*\| \leq \max\left(|1 - \tau\ell_1(x^*)|, |1 - \tau\ell_0(x^*)|\right) \|x_k - x^*\|.$$

Ainsi, si $\tau < \frac{1}{\ell(x^*)}$ on récupère bien un taux de convergence linéaire de la descente de gradient.

Bien évidemment, on peut préciser toutes ces informations comme nous l'avons fait auparavant dans le cas d'une fonction quadratique, et l'on s'attend, quand on optimise le pas, à avoir une convergence à taux

$$\frac{4\lambda_d(\nabla^2 f(x^*)) - \lambda_1(\nabla^2 f(x^*))}{4\lambda_d(\nabla^2 f(x^*)) + \lambda_1(\nabla^2 f(x^*))}.$$

On retrouve encore une fois le problème du conditionnement de la hessienne $\nabla^2 f(x^*)$.

On a donc établi le théorème suivant :

Théorème 6.6.1. *Soit $f \in \mathscr{C}^2(\mathbb{R}^d; \mathbb{R})$ et $x^* \in \mathbb{R}^d$ un minimum local de f. On suppose que $\nabla^2 f(x^*) \in S_d^{++}(\mathbb{R})$. Il existe $\varepsilon > 0$ tel que, si $x_0 \in \mathbb{B}(x^*, \varepsilon)$ et si $\tau < \frac{1}{\lambda_d(\nabla^2 f(x^*))}$, la descente de gradient à pas τ converge linéairement vers x^*.*

6.7 La question de la taille des pas et des directions de descente

Remarquons que, jusqu'ici, nous avons considéré une version plutôt simple de l'idée générale des méthodes de descente : nous avons fixé un pas taux, et nous avons simplement effectué le pas $-\tau\nabla f(x_k)$. Cette méthode n'est peut-être pas la plus intelligente, et il se peut que l'on choisisse, à chaque étape, une nouvelle direction d_k dans la quelle se diriger, et une taille de pas α_k. On peut espérer que ceci donne de meilleures convergences des algorithmes considérés. Pour ces raisons, posons la définition suivante :

Définition 6.7.1 (Direction de descente)**.** *Soit $f \in \mathscr{C}^1(\mathbb{R}^d; \mathbb{R})$. Soit $x \in \mathbb{R}^d$. Un vecteur $d \in \mathbb{R}^d$ est une **direction de descente** de f en x si*

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

Une fois que l'on est un point x et que l'on veut minimiser la fonction f, il faut choisir une direction de descente et une taille de pas. Il faut bien sûr trouver un compromis : si on prend un pas τ très petit, alors passer de x à $x + \tau d$ améliorera certainement le critère, mais la convergence risque d'être extrêmement lente si l'on n'y prend pas garde, comme nous l'avons montré dans le cas des fonctions quadratiques. À l'inverse, si l'on prend un pas trop grand, on peut ne plus améliorer du tout le critère... Il faut donc trouver un compromis entre ces deux exigences. C'est l'objectif du cours suivant que de présenter une batterie de méthodes permettant de satisfaire à ces deux exigences.

6.8 Une dernière remarque sur les critères d'arrêt

Une dernière remarque, qui a son importance, et qui repose également sur le conditionnement de la matrice A : mettons que l'on considère une fonction quadratique de la forme

$$f : x \mapsto \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

Comme toujours, on autorisera notre ordinateur à effectuer un nombre maximal d'itérations mais on aimerait s'assurer que, ou bien la solution trouvée soit "satisfaisante", ou bien que l'algorithme s'arrête s'il a déjà trouvé une solution satisfaisante.

Une idée, pour ce faire, est de dire que "satisfaisant" signifie $\|\nabla f(x)\| < \varepsilon$ où $\varepsilon > 0$ est un (petit) paramètre fixé à l'avance. Mais ce critère, très naturel, répond-il à nos exigences ? En général, **non**, et il faut le manier avec beaucoup de délicatesse. En effet, dans ce cas, le gradient de f vaut

$$\nabla f(x) = Ax - b.$$

Mais, si la matrice A est mal conditionnée, il se peut que $\|Ax - b\|$ soit faible, et que $\|x - x^*\|$ soit, au contraire, très important. On y prendra donc gare, quand on implémentera de tels critères d'arrête en séances de travaux pratiques.

Chapitre 7

Descente de gradient III : quelques règles de détermination des pas de descente

Comme annoncé à la fin du cours précédent, nous nous intéressons dans ce cours à des méthodes qui nous permettent de contourner les difficultés inhérentes aux descentes de gradient à pas constants *i.e.* le fait que la vitesse de convergence soit assez faible, et que l'on doive choisir à l'avance un bon pas. Pour cela, une bonne idée est de chercher, à chaque étape, le "bon" pas τ_k. Mais attention, ce choix ne peut pas être arbitraire : considérons la fonction $f : x \mapsto x^2$. On définit un pas adaptatif de taille

$$\tau_k := \frac{2 + 3 \cdot 2^{-(k+1)}}{2|x_k|}$$

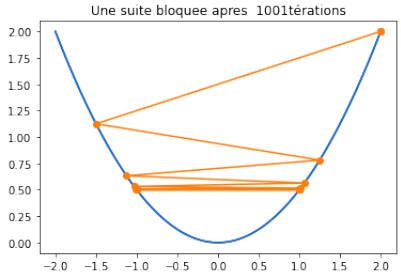
et l'on initialise la descente de gradient en $x_0 = 2$. On peut vérifier à la main que les itérations de la descente de gradient satisfont la relation de récurrence

$$x_{k+1} = x_k - \operatorname{sgn}(x_k) \left(2 + 3 \cdot 2^{-k-1}\right)$$

puis démontrer par récurrence (exercice !) que, pour tout $k \in \mathbb{N}$, on a

$$x_k = (-1)^k (1 + 2^{-k}).$$

On peut voir sur cette expression explicite que la suite reste *in fine* bloquée et oscille perpétuellement entre $+1$ et -1. Pourtant, on a bien choisi une direction de descente, et la valeur de la fonction à optimiser décroît bien à chaque itération, comme on peut le voir sur la figure suivante :



Une analyse possible de ce problème est que le pas est toujours bien trop grand. Voyons à présent comment remédier à cela, en présentant trois règles fondamentales : la recherche du **pas optimal exact**, la **règle de Wolfe** et la **règle d'Armijo**.

7.1 La descente de gradient à pas optimal

7.1.1 Présentation de la méthode

Une première idée (d'ailleurs plutôt bonne) pour améliorer la convergence de la descente de gradient est, une fois une direction de descente d_k en une itération x_k choisie, de résoudre le problème d'optimisation unidimensionnel

$$\min_{\tau \geq 0} f(x_k + \tau d_k). \tag{7.1}$$

La résolution de ce problème, disons τ_k^*, est alors choisie comme taille de pas, et l'on pose

$$x_{k+1} = x_k + \tau_k^* d_k.$$

Dans ce cas, par définition même, on a bien

$$f(x_{k+1}) < f(x_k).$$

Mais (7.1) est un problème unidimensionnel, et l'on peut donc utiliser toutes les méthodes précédemment vues (méthode de la sécante, méthode de la section doré...) pour le calculer. C'est faisable, mais cela peut considérablement alourdir le coût des calculs et le temps que l'ordinateur met à générer une itération.

Une propriété importante de la descente de gradient à pas optimal est la suivante :

Lemme 7.1.1. *Soit $(x_k)_{k\in\mathbb{N}}$ une suite de points générées par une descente de gradient à pas optimal où, pour tout $k \in \mathbb{N}$, la direction de descente est $d_k \in \mathbb{R}^d$. Alors :*

$$\forall k \in \mathbb{N}\,, \langle \nabla f(x_{k+1}), d_k \rangle = 0.$$

Preuve du Lemme 7.1.1. On considère le problème d'optimisation (7.1). En un τ^* optimal, la condition d'optimalité d'ordre 1 donne

$$\langle \nabla f(x_k + \tau^* d_k), d_k \rangle = 0,$$

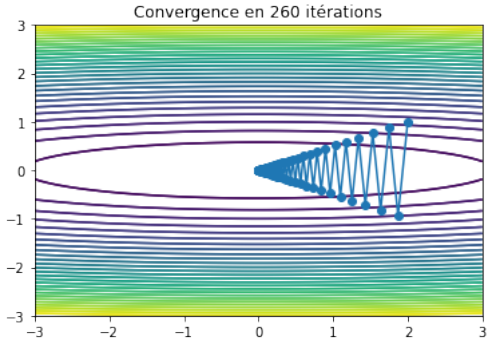
ce qui est exactement le propriété d'orthogonalité désirée. □

7.1.2 Quelques exemples numériques

On fixe comme choix de direction de descente $d_k = -\nabla f(x_k)$. Pour se convaincre que le choix d'un pas optimal peut effectivement améliorer la convergence de l'algorithme, reprenons le cas, vu au cours précédent, de la minimisation de

$$f : x \mapsto \frac{1}{2}\langle Ax, x\rangle \text{ avec } A = \begin{pmatrix} 1 & 0 \\ 0 & 30 \end{pmatrix},$$

donc, pour une matrice très mal conditionnée. On rappelle que la descente de gradient à pas constant donnait une convergence très mauvaise :



Si maintenant, au lieu de choisir un pas constant, on optimise la taille du pas on obtient

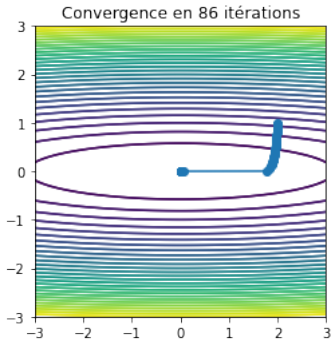


FIGURE 7.1 – Amélioration de la convergence

ce qui est une amélioration significative. Mais notons que ceci est fortement lié à la position du point initial, qui n'est pas situé "trop loin" d'un des axes de l'ellipse. Encore une fois, ceci est lié au mauvais conditionnement de la matrice. Pour forcer le trait, si l'on prend un point qui est situé presque exactement sur le plus long des axes de l'ellipse on obtient une convergence en deux itérations !

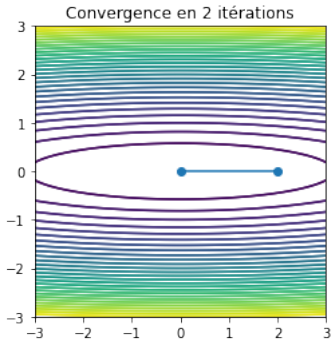


FIGURE 7.2 – Une convergence très rapide si l'on initialise sur le grand axe de l'ellipse

À l'inverse, si l'on prend une initialisation situé sur le plus petit axe de l'ellipse, on fait presque aussi mauvais qu'avec une descente de gradient à pas constant :

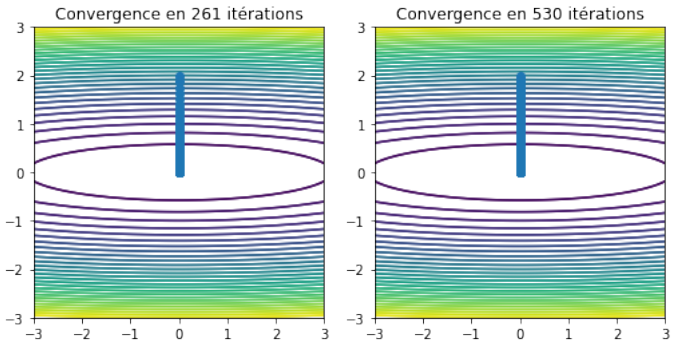


FIGURE 7.3 – À gauche, une descente de gradient à pas constant, à droite une descente de gradient à pas optimal

Observons finalement que, si la matrice A est colinéaire à l'identité alors, quelle que soit l'initialisation choisie, la méthode converge en une ou deux itérations :

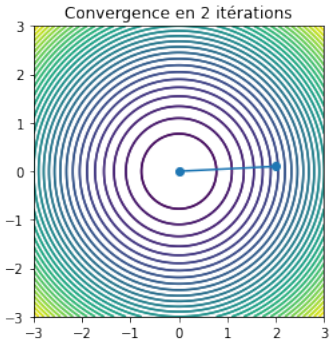


FIGURE 7.4 – Convergence très rapide pour des matrices A colinéaires à l'identité

7.1.3 Quelle convergence pour la descente de gradient à pas optimal ?

Il reste à voir si cet algorithme de descente à pas optimal converge effectivement, et s'il fait mieux, ou non, que la descente de gradient à pas constant.

Théorème 7.1.1 (Convergence de la descente de gradient à pas optimal)**.** *Soit* $A \in \mathcal{S}_d^{++}(\mathbb{R}), b \in \mathbb{R}^d$ *et*

$$f : x \mapsto \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

*La descente de gradient (*i.e. *à chaque itération* k *la direction de descente est* $-\nabla f(x_k)$*) à pas optimal converge linéairement à taux*

$$\left(\frac{\mathrm{cond}(A)-1}{\mathrm{cond}(A)+1}\right).$$

On a l'estimation précisée

$$\langle A(x_k - x^*), x_k - x^*\rangle \leq \langle A(x_0 - x^*), x_0 - x^*\rangle \left(\frac{\mathrm{cond}(A)-1}{\mathrm{cond}(A)+1}\right)^{2k}.$$

On voit donc que le taux minimal de convergence pour la descente de gradient à pas optimal est le taux optimal pour la descente de gradient à pas constant. En particulier, *la descente de gradient à pas optimal peut être aussi mauvaise que la descente de gradient à pas constant*. Donnons néanmoins une preuve de ce théorème de convergence. Au delà de l'aspect "complétude" du cours, le point positif de cette preuve est qu'elle utilise l'inégalité de Kantorovich, fondamentale dans l'étude des méthodes de descente de gradient.

Proposition 7.1.1 (Inégalité de Kantorovich)**.** *Soit* $A \in S_d^{++}(\mathbb{R})$ *et* $0 < \lambda_1(A) \leq \cdots \leq \lambda_d(A)$ *ses valeurs propres ordonnées. Alors, pour tout* $x \in \mathbb{R}^d$ *de norme 1, on a*

$$1 \leq \langle Ax, x\rangle \cdot \langle A^{-1}x, x\rangle \leq \frac{1}{4} \cdot \frac{(\lambda_1(A) + \lambda_d(A))^2}{\lambda_1(A)\lambda_d(A)}.$$

La preuve de cette inégalité est un joli exercice de convexité. Nous donnons ici la preuve due à Newman (qui traite d'une généralisation de cette inégalité. Nous renvoyons à la fiche de TD).

Preuve de la Proposition 7.1.1. La preuve repose sur deux ingrédients : l'inégalité arithmético-géométrique, et l'inégalité de Jensen. Que ce soit pour la borne inférieure ou la borne supérieure, on travaille sous l'hypothèse que A est une matrice diagonale. En toute généralité, il suffit d'utiliser le théorème spectral et de faire un changement de base orthogonale. En notant simplement, pour des raisons de simplicité, λ_k au lieu de $\lambda_k(A)$ pour la k-ème valeur propre de A, on a

$$\langle Ax, x\rangle \cdot \langle A^{-1}x, x\rangle = \sum_{k=1}^{d} x_k^2 \lambda_k \sum_{k=1}^{d} \frac{x_k^2}{\lambda_k}.$$

Commençons par la borne inférieure. Par l'inégalité de Cauchy-Schwarz, on a

$$1 = \left(\sum_{k=1}^{d} x_k^2\right)^2 = \left(\sum_{k=1}^{d} |\sqrt{\lambda_k} x_k| \cdot \left|\frac{x_k}{\sqrt{\lambda_k}}\right|\right)^2 \leq \sum_{k=1}^{d} x_k^2 \lambda_k \sum_{k=1}^{d} \frac{x_k^2}{\lambda_k}.$$

Passons à la borne supérieure. Par l'inégalité arithmético-géométrique on a, pour tout $\delta > 0$,

$$\begin{aligned} 2\left(\sum_{k=1}^{d} x_k^2 \lambda_k \sum_{k=1}^{d} \frac{x_k^2}{\lambda_k}\right)^{\frac{1}{2}} &\leq \sum_{k=1}^{d} \delta x_k^2 \lambda_k + \sum_{k=1}^{d} \frac{x_k^2}{\delta \lambda_k} \\ &\leq \sum_{k=1}^{d} x_k^2 \phi(\delta \lambda_k), \end{aligned}$$

avec $\phi : x \mapsto x + \frac{1}{x}$. Observons à présent que la fonction ϕ est convexe, donc qu'en particulier

$$\sum_{k=1}^{d} x_k^2 \phi(\delta \lambda_k) \leq \max(\phi(\delta \lambda_1), \phi(\delta \lambda_d)).$$

Choisissons δ tel que

$$\phi(\delta \lambda_1) = \phi(\delta \lambda_d).$$

Par des calculs explicites, on obtient

$$\delta = \frac{1}{\sqrt{\lambda_1 \lambda_d}} \text{ et ainsi } \phi(\delta \lambda_1) = \sqrt{\frac{\lambda_1}{\lambda_d}} + \sqrt{\frac{\lambda_d}{\lambda_1}}.$$

En particulier, nous avons démontré

$$2\left(\sum_{k=1}^{d} x_k^2 \lambda_k \sum_{k=1}^{d} \frac{x_k^2}{\lambda_k}\right)^{\frac{1}{2}} \leq \sqrt{\frac{\lambda_1}{\lambda_d}} + \sqrt{\frac{\lambda_d}{\lambda_1}}.$$

En élevant chaque membre de cette inégalite au carré, on obtient la conclusion désirée. □

Enfin, afin de donner une preuve efficace, calculons le pas optimal pour la fonction

$$f : x \mapsto \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle.$$

Proposition 7.1.2. *Si $A \in S_d^{++}(\mathbb{R})$, le pas optimal pour la descente de gradient, partant d'un point x, est*

$$\tau^* = \frac{\|Ax - b\|^2}{\langle A^2 x - Ab, Ax - b\rangle}.$$

En particulier, l'algorithme de descente de gradient à pas optimal admet la description explicite suivante :

1. *Initialisation en $x_0 \in \mathbb{R}^d$.*
2. *Pour tout $k \in \mathbb{N}$,*

$$x_{k+1} = x_k - \frac{\|Ax_k - b\|^2}{\langle A^2 x_k - Ab, Ax_k - b\rangle}(Ax_k - b)$$

Preuve de la proposition 7.1.2. Le gradient de f en x est donné par

$$\nabla f(x) = Ax - b := p_x.$$

Considérons la fonction d'une variable réelle

$$g : \mathbb{R}_+ \ni \tau \mapsto f(x - \tau p_x).$$

En un point de minimum τ^* on a

$$g'(\tau^*) = 0.$$

Puisque

$$g'(\tau) = \langle \nabla f(x - \tau p_x), -p_x \rangle$$

on en déduit qu'en un minimum τ^* (nécessairement unique par stricte convexité et coercivité de la fonction f) on a

$$\langle A(x - \tau p_x) - b, p_x \rangle = 0,$$

ce qui est équivalent à

$$\tau \langle Ap_x, p_x \rangle = \langle Ax - b, p_x \rangle = \langle Ax - b, Ax - b \rangle = \|Ax - b\|^2,$$

ou encore

$$\tau^* = \frac{\|Ax - b\|^2}{\langle A^2 x - Ab, Ax - b \rangle}.$$

□

Preuve du Théorème 7.1.1. Définissons, pour tout $k \in \mathbb{N}$,

$$y_k := A(x_k - x^*).$$

Dans la mesure où x^* est l'unique solution de $Ax = b$ on peut réécrire grâce à la Proposition 7.1.2 la suite des itérations de la descente de gradient comme

$$x_{k+1} = x_k - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} y_k.$$

Ainsi,

$$\begin{cases} x_{k+1} - x^* = (x_k - x^*) - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} y_k = \left(A^{-1} - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} \mathrm{I}_d \right) y_k \,, \\ y_{k+1} = \left(\mathrm{I}_d - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} A \right) y_k \end{cases}$$

de sorte que

$$\langle y_{k+1}, x_{k+1} - x^* \rangle = \left\langle \left(\mathrm{I}_d - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} A \right) y_k, \left(A^{-1} - \frac{\|y_k\|^2}{\langle Ay_k, y_k \rangle} \mathrm{I}_d \right) y_k \right\rangle$$

$$
\begin{aligned}
&= \left\langle y_k, \left(\mathrm{I}_d - \frac{\|y_k\|^2}{\langle Ay_k, y_k\rangle} A\right)\left(A^{-1} - \frac{\|y_k\|^2}{\langle Ay_k, y_k\rangle}\mathrm{I}_d\right) y_k \right\rangle \\
&\text{car } A \text{ est symétrique} \\
&= \left\langle y_k, A\left(A^{-1} - \frac{\|y_k\|^2}{\langle Ay_k, y_k\rangle}\mathrm{I}_d\right)^2 y_k \right\rangle \\
&= \left\langle y_k, \left(A^{-1} - 2\frac{\|y_k\|^2}{\langle Ay_k, y_k\rangle}\mathrm{I}_d + \frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle^2} A\right) y_k \right\rangle \\
&= \langle y_k, A^{-1} y_k\rangle \\
&- 2\frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle} \\
&+ \frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle} \\
&= \langle A(x_k - x^*), x_k - x^*\rangle \\
&- \frac{\|y_k\|^4 \langle y_k, A^{-1} y_k\rangle}{\langle Ay_k, y_k\rangle\langle A^{-1} y_k, y_k\rangle} \\
&= \langle A(x_k - x^*), x_k - x^*\rangle \left(1 - \frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle\langle A^{-1} y_k, y_k\rangle}\right).
\end{aligned}
$$

En définissant la fonction

$$G(x) := \langle A(x - x^*), x - x^*\rangle$$

nous avons donc établi

$$G(x_{k+1}) = G(x_k)\left(1 - \frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle\langle A^{-1} y_k, y_k\rangle}\right).$$

Par l'inégalité de Kantorovich (Proposition 7.1.1) on sait que

$$\left(1 - \frac{\|y_k\|^4}{\langle Ay_k, y_k\rangle\langle A^{-1} y_k, y_k\rangle}\right) \leq 1 - 4\frac{\lambda_1(A)/\lambda_d(A)}{\left(\lambda_1(A)/\lambda_d(A) + 1\right)^2} = 1 - 4\frac{\mathrm{cond}(A)}{(\mathrm{cond}(A) + 1)^2},$$

et donc

$$G(x_{k+1}) \leq G(x_k)\left(1 - 4\frac{\mathrm{cond}(A)}{(\mathrm{cond}(A) + 1)^2}\right) = G(x_k)\left(\frac{\mathrm{cond}(A) - 1}{\mathrm{cond}(A) + 1}\right)^2.$$

On en tire la conclusion par une simple récurrence. □

7.2 Règle d'Armijo, règle de Wolfe

Comme nous l'avons vu, les algorithmes d'optimisation en dimension 1 peuvent être gourmands en temps. Nous allons donc voir dans la suite de ce cours deux règles pour déterminer des pas qui *sont suffisamment satisfaisants du point de vue de l'optimisation* tout en étant plus faciles à calculer.

7.2.1 Discussion heuristique

L'idée des deux règles que nous présentons maintenant est qu'à l'étape k, on choisisse un pas qui ne soit ni trop grand, ni trop petit. Nous l'avons vu, c'est cela qui peut poser problème quand on travaille sur $f : x \mapsto x^2$.

Pour vérifier que l'on ne choisit pas des pas trop grands, qui nous enverraient trop loin de la solution x^* (auquel cas nos approximations de Taylor au premier ordre n'auraient plus aucun sens), on utilise la règle d'Armijo suivante : on sait que, si l'on definit

$$g_k : \tau \mapsto f(x_k + \tau d_k)$$

alors on veut choisir τ_k qui garantisse que $g_k(\tau_k)$ soit "suffisamment" inférieur à $g_k(0) = f(x_k)$. Il nous faut quantifier cela. Fixons donc un paramètre $\gamma > 0$. L'idéal serait d'obtenir un pas de descente $\tau > 0$ tel que

$$f(x_k + \tau d_k) \leq f(x_k) - \gamma\tau.$$

Par ailleurs, il est naturel de quantifier γ en fonction de la pente de la fonction f au point x_k, dans la direction d_k. On choisira donc plutôt

$$\gamma = \gamma_1 \langle \nabla f(x_k), d_k \rangle$$

où $\gamma_1 \in]0; 1[$ sera un paramètre réel à choisir. Ce type de condition va s'appeler **condition d'Armijo**.

À l'inverse, on ne veut pas que les pas soient trop petits. En effet, quelle que soit la fonction f, quelle que soit l'initialisation x_0, si l'on fait des pas microscopiques, on n'a aucune chance de converger vers un minimiseur (il suffit de prendre une suite de taille de pas τ_k telle que $\sum_{k=0}^{\infty} \tau_k \|\nabla f(x_k)\| \leq \frac{\|x_0 - x^*\|}{2}$). Comment quantifier ce critère-là ? L'idée de la **condition de Wolfe** est de travailler sur le gradient de f. En effet, d_k étant une direction de descente, on a

$$\langle \nabla f(x_k), d_k \rangle < 0.$$

Si l'on choisissait pour τ le τ^* du pas optimal exact, on aurait

$$\langle \nabla f(x_k + \tau^* d_k), d_k \rangle = 0. \tag{7.2}$$

La condition de Wolfe, c'est relaxer la condition d'égalité dans (7.2) pour la remplacer par une condition de type "le produit scalaire $\langle \nabla f(x_k + \tau^* d_k), d_k \rangle$ s'est suffisamment rapproché de 0". En choisissant un paramètre $\gamma_2 \in]0; 1[$ il faudra donc garantir une condition du type

$$\langle \nabla f(x_k + \tau^* d_k), d_k \rangle \geq \gamma_2 \langle \nabla f(x_k), d_k \rangle.$$

7.2.2 Formalisation des règles d'Armijo et de Wolfe

Ces considérations nous amènent à poser la définition suivante :

Définition 7.2.1. *Soit $f \in \mathscr{C}^1(\mathbb{R}^d; \mathbb{R})$, soit $x_k \in \mathbb{R}^d$, d_k une direction de descente en x_k (on suppose en particulier que $\nabla f(x_k) \neq 0$). Soit $\tau > 0$ une longueur de pas.*

1. *Si $\gamma_1 \in (0;1)$, on dit que le pas τ satisfait* ***la condition d'Armijo*** *en x_k pour γ_1 si*

$$f(x_k + \tau d_k) \leq f(x_k) + \gamma_1 \tau \langle \nabla f(x_k), d_k \rangle.$$

2. *Si $0 < \gamma_1 < \gamma_2 < 1$, on dit que le pas τ vérifie la* ***condition de Wolfe*** *en x_k pour (γ_1, γ_2) si τ vérifie la condition d'Armijo en x_k pour γ_1 et si, en outre, on a*

$$\langle \nabla f(x_k + \tau d_k), d_k \rangle \geq \gamma_2 \langle \nabla f(x_k), d_k \rangle.$$

Il nous reste à vérifier deux choses : la première, c'est que, si l'on se fixe un couple (γ_1, γ_2) alors, à chaque itération, si l'on se fixe une direction de descente d_k, on peut trouver un pas τ_k qui vérifie la condition de Wolfe en x_k et, en outre, que l'algorithme ainsi modifié a de meilleures propriétés de convergence. Commençons par le prmeier point, à savoir que, si l'on se fixe un couple (γ_1, γ_2), alors on peut trouver à chaque itération un pas admissible :

Proposition 7.2.1. *On fixe $0 < \gamma_1 < \gamma_2 < 1$ et $f \in \mathscr{C}^2(\mathbb{R}^d; \mathbb{R})$. Soit $x_k \in \mathbb{R}^d$ et d_k une direction de descente de f en x_k. On suppose que f est bornée inférieurement. Alors il existe $0 < \tau_1 < \tau_2$ tels que, pour tout $\tau \in]\tau_1; \tau_2[$, le pas τ vérifie la condition de Wolfe en x_k pour (γ_1, γ_2). On prendra garde au fait que τ_1 et τ_2 dépendent de l'indice k.*

Le point important dans cette proposition, c'est que les deux paramètres γ_1 , γ_2 sont uniformes, c'est-à-dire qu'ils ne dépendent pas de l'itération.

Preuve de la proposition. Commençons par construire τ_1 , τ_2 tels que tout pas $\tau \in]\tau_1; \tau_2[$ est admissible pour la règle d'Armijo. À cet effet, définissons la fonction

$$\Phi : t \mapsto f(x_k + t d_k).$$

Puisque $\Phi'(0) = \langle \nabla f(x_k), d_k \rangle < 0$ et que f est de classe $\mathscr{C}^1$ on en déduit qu'il existe $\varepsilon_1 > 0$ tel que,

$$\forall t \in [0; \varepsilon_1[\,, \Phi'(t) \leq \gamma_1 \Phi'(0).$$

En particulier, pour tout $t \in [0; \varepsilon_1[$, on a

$$\Phi(t) \leq \Phi(0) + t\gamma_1 \Phi'(0).$$

Par la définition de Φ, ceci implique que

$$\forall t \in [0; \varepsilon_1[\,, f(x_k + t d_k) \leq f(x_k) + t\gamma_1 \langle \nabla f(x_k), d_k \rangle,$$

ou encore que tout pas $\tau \in]0; \varepsilon_1[$ vérifie la règle d'Armijo en x_k.

Remarque 7.2.1. *De manière cohérente avec l'intuition, on ne voit pas dans cette construction apparaître de borne inférieure sur la taille du pas que l'on construit. Cette borne inférieure, qui garantit que l'on ne fait pas de pas trop petits, vient de la règle de Wolfe.*

Maintenant, pour construire un pas qui satisfasse également la règle de Wolfe, saturons l'inégalité précédente ; en d'autres termes, définissons

$$\varepsilon_1^* := \sup\{\varepsilon > 0 \text{ tel que } \forall t < \varepsilon\,, f(x_k + td_k) \leq f(x_k) + t\gamma_1\langle \nabla f(x_k), d_k\rangle\}\,.$$

Puisque f est bornée inférieurement, ε_1^* est bien défini et on a, en ε_1^*,

$$f(x_k + \varepsilon_1^* d_k) = f(x_k) + \varepsilon_1^*\gamma_1\langle \nabla f(x_k), d_k\rangle.$$

En reprenant la fonction Φ introduite précédemment, cette identité s'écrit

$$\frac{\Phi(\varepsilon_1^*) - \Phi(0)}{\varepsilon_1^*} = \gamma_1\Phi'(0) > \gamma_2\Phi'(0).$$

Or, par le théorème des accroissements finis, il existe $\overline{\tau} \in (0; \varepsilon_1)$ tel que

$$\Phi'(\overline{\tau}) = \frac{\Phi(\varepsilon_1^*) - \Phi(0)}{\varepsilon_1^*} = \gamma_1\Phi'(0) > \gamma_2\Phi'(0),$$

et donc, il existe un voisinage $]\tau_1; \tau_2[$ de $\overline{\tau}$ tel que, pour tout $\tau \in]\tau_1; \tau_2[$, on a

$$\gamma_2\Phi'(0) = \gamma_2\langle \nabla f(x_k), d_k\rangle \leq \langle \nabla f(x_k + \tau d_k), d_k\rangle,$$

ce qui est exactement la condition de Wolfe. Par construction, on $\overline{\tau} \in]0; \varepsilon_1^*[$ et l'on peut donc choisir $\tau_2 < \varepsilon_1^*$. En particulier, $\tau_1\,, \tau_2$ satisfont aux conclusions du théorème. □

7.2.3 Peut-on estimer la taille des pas ?

Montrons que la règle de Wolfe garantit effectivement que les pas que l'on construit ne sont pas trop petits. On suppose le paramètre γ_2 fixés dans la suite. On se donne une fonction f, et l'on suppose que ∇f est L-lipschitzien. On suppose que l'on travaille en une itération x_k, que l'on a choisi une direction de descente d_k, non nécessairement égale à $-\nabla f(x_k)$, et que l'on a un pas τ qui vérifie le critère Wolfe.

Puisque d_k est une direction de descente et que $\gamma_2 < 1$ on a

$$\begin{aligned}
(\gamma_2 - 1)\langle \nabla f(x_k), d_k\rangle &= \gamma_2\langle \nabla f(x_k), d_k\rangle \\
&\quad - \langle \nabla f(x_k), d_k\rangle \\
&\leq \langle \nabla f(x_{k+1}), d_k\rangle - \langle \nabla f(x_k), d_k\rangle \\
&\quad \text{car le pas vérifie la règle de Wolfe} \\
&\leq L\|x_{k+1} - x_k\| \cdot \|d_k\| \text{ par Cauchy-Schwarz} \\
&\leq \tau L\|d_k\|^2.
\end{aligned}$$

Ainsi, on a établi

$$\tau \geq \frac{1 - \gamma_2}{L}\left|\frac{\langle \nabla f(x_k), d_k\rangle}{\|d_k\|^2}\right| \tag{7.3}$$

En particulier, si l'on a choisi la direction de plus grande descente ou, en d'autres termes, si l'on a choisi $d_k = -\nabla f(x_k)$ (c'est-à-dire que l'on fait une descente de gradient standard) on obtient

$$\tau \geq \frac{1-\gamma_2}{L}.$$

7.2.4 Convergence de l'algorithme pour des pas satisfaisant les critères d'Armijo et de Wolfe

Nous allons à présent voir que, si le pas τ satisfait à la fois les critères d'Armijo et de Wolfe, alors la méthode converge en un certain sens.

Proposition 7.2.2 (Zoutendjik)**.** *Soit $f \in \mathscr{C}^1(\mathbb{R}^d;\mathbb{R})$ une fonction minorée et telle que ∇f soit L-lipschitzienne. Soient $0 < \gamma_1 < \gamma_2 < 1$ deux paramètres fixés. Soit $\{x_k\}_{k\in\mathbb{N}}$ la suite définie par $x_0 \in \mathbb{R}^d$ et, pour tout $k \in \mathbb{N}$, $x_{k+1} = x_k + \tau_k d_k$, où d_k est une direction de descente de f en x_k et où τ_k satisfait les critères d'Armijo et de Wolfe pour les paramètres (γ_1, γ_2) en x_k. Alors*

$$\sum_{k=0}^{\infty} \frac{|\langle \nabla f(x_k), d_k\rangle|^2}{\|d_k\|^2} < \infty.$$

En particulier, si $d_k = -\nabla f(x_k)$, on a

$$\nabla f(x_k) \underset{k\to\infty}{\to} 0.$$

Preuve de la proposition. Repartons de l'estimation (7.3) : pour tout $k \in \mathbb{N}$, on a

$$\tau_k \geq \frac{1-\gamma_2}{L} \left| \frac{\langle \nabla f(x_k), d_k\rangle}{\|d_k\|^2} \right|.$$

Rappelons que ceci est valable car τ_k vérifie la règle de Wolfe. Puisque τ_k satisfait également la règle d'Armijo, on a également

$$f(x_k) - f(x_{k+1}) \geq \tau_k \gamma_1 \left| \langle \nabla f(x_k), d_k\rangle \right|.$$

Ainsi,

$$f(x_k) - f(x_{k+1}) \geq \frac{\gamma_1(1-\gamma_2)}{L} \cdot \left| \frac{\langle \nabla f(x_k), d_k\rangle}{\|d_k\|} \right|^2.$$

En somment ces estimations, on obtient l'estimation

$$\frac{\gamma_1(1-\gamma_2)}{L} \sum_{k=0}^{\infty} \frac{|\langle \nabla f(x_k), d_k\rangle|^2}{\|d_k\|^2} \leq f(x_0) - \inf f.$$

Ceci conclut la preuve. □

Mais attention, nous avons bien dit que cette convergence n'avait lieu qu'en un certain sens. En effet, dans le cas $d_k = -\nabla f(x_k)$, on n'obtient "que" l'information

$$\nabla f(x_k) \underset{k\to\infty}{\to} 0.$$

Or, il se peut que ceci se produise, sans toutefois que la suite $\{x_k\}_{k\in\mathbb{N}}$ elle-même ne converge. Pensons par exemple à la fonction e^{-x^2}. Néanmoins, si l'on suppose en outre que la fonction $\|\nabla f\|^2$ est coercive, alors cette information permet de conclure que la suite $\{x_k\}_{k\in\mathbb{N}}$ est bornée, et donc de commencer à utiliser des arguments de compacité.

7.2.5 Détermination numérique de pas satisfaisants les critères d'Armijo et de Wolfe

Exposons brièvement comment ces pas peuvent être déterminés numériquement. On se fixe dans toute la suite une fonction f, un point x, une direction de descente d en x, deux paramètres $0 < \gamma_1 < \gamma_2 < 1$. On construit deux suites $\{\tau_{1,k}, \tau_{2,k}\}_{k\in\mathbb{N}}$ telles que :

1. Pour tout $k \in \mathbb{N}$, $\tau_{1,k} < \tau_{2,k}$.
2. Pour tout $k \in \mathbb{N}$, $\tau_{1,k}$ vérifie la règle d'Armijo, mais pas $\tau_{2,k}$.
3. Les deux suites convergent, et ont la même limite τ_∞.

Notons d'abord le fait suivant : on peut bien initialiser l'algorithme **si f est bornée inférieurement**. Pour $\tau_{1,0}$, il suffit de choisir un pas extrêmement petit (on peut le tester à la main sur ordinateur). Pour $\tau_{2,0}$ en revanche il faut s'assurer que l'on peut choisir de sorte à ce qu'il ne satisfasse pas la règle d'Armijo. Mais observons que, f étant bornée inférieurement, la fonction

$$t \mapsto f(x + td) - t\gamma_1\langle\nabla f(x), d\rangle - f(x)$$

tend vers $-\infty$ quand $t \to \infty$. Maintenant, supposons cette initialisation donnée. On définit l'itération suivante en posant $x_1 := \frac{\tau_{1,0}+\tau_{2,0}}{2}$ et

$$(\tau_{1,1}, \tau_{2,1}) := \begin{cases} (x_1, \tau_{2,0}) \text{ si } x_1 \text{ vérifie la règle d'Armijo,} \\ (\tau_{1,0}, x_1) \text{ sinon.} \end{cases}$$

On itère cette construction, et on l'arrête dès que $\tau_{1,k}$ vérifie également le critère de Wolfe, ce qui nous mène à énoncer l'algorithme sous la forme suivante :

Il faut nous assurer que cet algorithme converge en un nombre fini d'étapes. C'est le cas, comme on peut le voir par l'argument (élémentaire) suivant : il est clair que, par les mêmes arguments que ceux utilisés pour la méthode de dichotomie, les deux suites $\{\tau_{1,k}\}_{k\in\mathbb{N}}$ et $\{\tau_{2,k}\}_{k\in\mathbb{N}}$ sont adjacentes. En particulier, elles convergent vers une même limite τ^*. Mais on a, d'une part,

$$f(x + \tau_{2,k}d) - f(x) > \gamma_1\tau_{2,k}\langle\nabla f(x), d\rangle$$

Algorithm 4 Recherche d'un pas vérifiant à la fois Armijo et Wolfe

```
def pasWolfe(F, dF, x, h, tau0, Tau0, gamma1=0.2, gamma2=0.7,
Niter=1000):
 Fx, dFx = F(x), dF(x)
Initialisation
taun, Taun, xn = tau0, Tau0, (tau0 + Tau0)/2
taun satisfait Wolfe
 if dot(dF(x + taun*h), h) >= gamma2*dot(dFx,h):
return   taun
taun satisfait Armijo
if F(x + xn*h) <= Fx + gamma1*xn*dot(dFx,h):
 taun, Taun = taun, xn
else:
 taun,Taun=xn,Taun
print("Erreur, l'algorithme n'a pas convergé
après",IterMax,"itérations")
```

et d'autre part

$$f(x + \tau_{1,k}d) - f(x) < \gamma_1 \tau_{1,k}.$$

Formant le quotient différentiel $(f(x + \tau_{2,k}d) - f(x + \tau_{1,k}d))/(\tau_{2,k} - \tau_{1,k})$ il apparaît que

$$\langle \nabla f(x + \tau^* d), d\rangle \geq \gamma_1 \langle \nabla f(x), d\rangle > \gamma_2 \langle \nabla f(x), d\rangle.$$

En d'autres termes, τ^* vérifie la condition de Wolfe. Ainsi, pour k suffisamment grand, $\tau_{1,k}$ satisfait également la condition de Wolfe.

Chapitre 8

Descentes de gradient IV : gradient conjugué

8.1 Première discussion et principe de la méthode

Passons au dernier aspect de la descente de gradient en discutant la méthode du **gradient conjugué**. Le problème de la descente de gradient, même à pas optimal, c'est qu'elle converge très lentement si elle n'est pas bien initialisée. Même si c'est un exemple que nous connaissons désormais bien, considérons le cas d'une matrice

$$A := \begin{pmatrix} a_0 & 0 \\ 0 & a_1 \end{pmatrix}$$

avec $a_0 \neq a_1$, et posons

$$f : \mathbb{R}^2 \ni x = (x_0, x_1) \mapsto \frac{1}{2} a_0 x_0^2 + \frac{1}{2} a_1 x_1^2.$$

Pour trouver la solution en disons une itération, il faudrait, si l'on choisit de suivre la direction de descente du gradient, qu'il existe un réel $\tau \in \mathbb{R}$ tel que

$$\tau \nabla f(x_0) = x_0.$$

Or ceci se réécrit

$$x_{0,0} = \tau a_0 x_{0,0}\,, x_{0,1} = \tau a_1 x_{0,1}.$$

On voit donc que si $x_{0,0} x_{0,1} \neq 0$, il est impossible de remplir cette condition. Est-il néanmoins possible, si l'on itère suffisamment de fois l'algorithme de descente de gradient à pas optimal, d'arriver à un point x_k tel que x_{k+1} soit égal à 0 ? Reprenons la proposition 7.1.2 : on voit que les itérations de la descente de gradient à pas optimal, que nous noterons $\{x_k = (x_{0,k}, x_{1,k})\}_{k \in \mathbb{N}}$ sont données de manière itérative par

$$x_{0,k+1} = \frac{a_1^2 (a_1 - a_0) x_{0,k} x_{1,k}^2}{a_0^3 x_{0,k}^2 + a_1^3 x_{1,k}^2}\,, x_{1,k+1} = \frac{a_0^2 (a_0 - a_1) x_{0,k}^2 x_{1,k}}{a_0^3 x_{0,k}^2 + a_1^3 x_{1,k}^2}.$$

En particulier, si $x_{0,0}x_{0,1} \neq 0$ alors, pour tout $k \in \mathbb{N}$, $x_{k,0}x_{k,1} \neq 0$, et donc la descente de gradient à pas optimal ne converge pas en un nombre fini d'itérations.

Observons néanmoins qu'il n'y a *a priori* aucune raison pour que ce choix de direction de descente soit le meilleur. Arrivant en effet au point x_0, il faudrait plutôt choisir comme direction de descente celle qui nous emmène directement de x_1 à 0, c'est-à-dire choisir

$$d_1 = \frac{x_1}{\|x_1\|}.$$

Observons alors (ceci sera systématisé) que les vecteurs $d_0 = \nabla f(x_0)$ et d_1 sont A-orthogonaux :

$$\langle Ad_0, d_1 \rangle = 0.$$

En particulier, dans ce cas très particulier où l'on cherche en fait à calculer 0, le choix de deux directions de descentes A-orthogonales permet d'obtenir un algorithme qui converge en deux itérations. Isolons cette notion d'orthogonalité :

Définition 8.1.1. *Soit $A \in S_d^{++}(\mathbb{R})$. Deux vecteurs $x, y \in \mathbb{R}^d$ sont dits* A-orthogonaux *si, et seulement si,*

$$\langle Ax, y \rangle = 0.$$

Il existe plusieurs manières d'attaquer la construction des méthodes de gradient conjugué. L'une, par laquelle nous allons commencer, est la plus naturelle mais ne fait pas forcément apparaître la nature hilbertienne de la méthode. La seconde, par projection, est plus absconse mais à de nombreux égards très instructive.

Dans toute la suite, $A \in S_d^{++}(\mathbb{R})$ et $b \in \mathbb{R}^d$ sont fixés, et l'on définit

$$f : \mathbb{R}^d \ni x \mapsto \frac{1}{2}\langle Ax, x \rangle - \langle b, x \rangle.$$

8.2 Une première approche de la méthode du gradient conjugué

Une première manière d'étudier le gradient conjugué est de renforcer le principe de la descente de gradient à pas optimal. Rappelons que cette méthode, déjà étudiée lors de l'un des cours précédents, consiste à choisir, une itération x_k étant donnée (telle que $\nabla f(x_k) \neq 0$), x_{k+1} par la formule

$$x_{k+1} = x_k - \tau^* \nabla f(x_k) \text{ avec } \tau^* \text{ défini par } \tau^* := \operatorname*{argmin}_{\tau > 0} f(x_k - \tau \nabla f(x_k)).$$

Ce faisant, on a déjà complètement oublié tout ce qui avait été calculé aux étapes précédentes. Il est *a priori* bien meilleur de plutôt choisir x_{k+1} sous la

forme

$$x_{k+1} = x_k - \sum_{i=0}^{k} \tau_i^* \nabla f(x_i) \text{ avec } \tau_0^*, \dots, \tau_k^* = \underset{\tau_0, \dots, \tau_n > 0}{\operatorname{argmin}} f\left(x_k - \sum_{i=0}^{k} \tau_i \nabla f(x_i) \right). \tag{8.1}$$

Si l'on définit ainsi les itérations, on voit que les conditions d'optimalité s'écrivent

$$\forall i \in \{0, \dots, k\}, \langle \nabla f(x_{k+1}), \nabla f(x_i) \rangle = 0. \tag{8.2}$$

La condition (8.2) est bien plus forte que celle de la simple descente de gradient à pas optimal, qui ne faisait que générer des gradients qui étaient successivement orthogonaux.

Une conséquence immédiate de cette simple observation est la suivante :

Lemme 8.2.1. *L'algorithme décrit ci-dessus par la relation* (8.1) *converge en au plus d itérations.*

Démonstration. Notons $k^* := \max\{k \in \mathbb{N}, \nabla f(x_k) \neq 0\} \in \mathbb{N} \cup \{\infty\}$. Évidemment, la fonction f étant strictement convexe on sait que, pour tout $i \leq k^*$ on a

$$\nabla f(x_i) \neq 0.$$

Si, par l'absurde, on a

$$d < k^*$$

alors la famille $\{\nabla f(x_i)\}_{i=0,\dots,k^*}$ est une famille de vecteurs non-nuls et deux à deux orthogonaux. En particulier, c'est une famille libre. Or, en dimension d, les familles libres sont de cardinal au plus d, une contradiction. □

Le problème malgré tout, c'est qu'il n'est pas tout à fait évident de savoir si l'on a vraiment amélioré notre approche, en remplaçant un problème d'optimisation à d variables en une suite de problèmes d'optimisation en k variables. Montrons que cette méthode peut naturellement se réecrire comme une méthode itérative simple. Pour cela, il nous faut commencer par analyser les directions de descente générées par (8.1).

Une première propriété des directions de descente dans (8.1) Introduisons, pour $k \in \mathbb{N}$ tel que $\nabla f(x_k) \neq 0$, la direction de descente d_k choisie par la règle (8.1), c'est-à-dire

$$d_k := x_{k+1} - x_k.$$

Puisque pour tout $x \in \mathbb{R}^d$ on a

$$\nabla f(x) = Ax - b$$

on en déduit que

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

Par ailleurs, par définition même de x_{k+1}, il existe $(k+1)$-coefficients $\{\alpha_{k,j}\}_{j=0,\dots,k}$ tels que

$$d_k = \sum_{j=0}^{k} \alpha_{k,j} \nabla f(x_j).$$

Isolons ces deux informations :

$$\begin{cases} d_k = \sum_{j=0}^{k} \alpha_{k,j} \nabla f(x_j)\,, \\ \nabla f(x_{k+1}) = \nabla f(x_k) + A d_k. \end{cases} \tag{8.3}$$

Par la condition d'orthogonalité (8.2) on en déduit que, pour tout $i \leq k-1$,

$$0 = \langle \nabla f(x_{k+1}), \nabla f(x_i) \rangle = \langle \nabla f(x_k), \nabla f(x_i) \rangle + \langle A d_k, \nabla f(x_i) \rangle = \langle A d_k, \nabla f(x_i) \rangle.$$

Mais, puisque, si $j \leq k-1$, on peut écrire

$$d_j = \sum_{i=1}^{j} \alpha_{j,i} \nabla f(x_i)$$

c'est-à-dire comme une somme de vecteurs A-orthogonaux à d_k, on en déduit la propriété cruciale du gradient conjugué :

$$\boxed{\forall k \neq j\,, \langle A d_j, d_k \rangle = 0.} \tag{8.4}$$

Cette propriété de A-orthogonalité donne son nom à la méthode du gradient conjugué. Notons que la morale de ce que cet algorithme se termine bien en un nombre fini d'étapes, c'est qu'il n'était pas pertinent de choisir la plus profonde direction de descente pour le produit scalaire euclidien, mais plutôt pour le produit scalaire induit par la matrice A. Nous retrouverons ce point de vue dans notre présentation alternative de la méthode du gradient conjugué.

Par ailleurs, en prenant cette fois le produit scalaire avec $\nabla f(x_k)$ dans l'expression de $\nabla f(x_{k+1})$ on obtient

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

En particulier, si l'algorithme n'a pas terminé en x_k, $\langle A d_k, \nabla f(x_k) \rangle \neq 0$. Mais on obtient en fait mieux !

Calcul des coefficients $\alpha_{k,i}$ Mais toutes ces considérations ne nous avancent pas plus sur le calcul effectif des coefficients $\{\alpha_{k,j}\}_{k,j}$, calcul qui nous permettrait d'exhiber une structure itérative simple.

Commençons par le lemme suivant :

Lemme 8.2.2. *Tant que l'algorithme du gradient conjugué n'a pas convergé, on a*

$$\forall k\,, \langle d_k, \nabla f(x_k) \rangle \neq 0.$$

Preuve du Lemme 8.2.2. On démontre cette propriété par récurrence. Pour $k = 0$, d_0 est, par définition, colinéaire à $\nabla f(x_0)$. Supposons la propriété vraie au rang k. Supposons par l'absurde, l'algorithme n'ayant pas terminé à l'itération $k+1$, que l'on ait

$$\langle d_{k+1}, \nabla f(x_{k+1})\rangle = 0.$$

Par construction, on obtient

$$d_{k+1} = \sum_{j=0}^{k} \alpha_{k+1,j} \nabla f(x_j).$$

Néanmoins, la condition (8.4) garantit la liberté de la famille $(d_i)_{i\leq k}$. Par un simple argument de dimension, on en déduit que
$\mathrm{Vect}(d_1, \ldots, d_k) = \mathrm{Vect}(\nabla f(x_1), \ldots, \nabla f(x_k))$. On obtient ainsi que $d_{k+1} \in \mathrm{Vect}(d_1, \ldots, d_k)$, en contradiction avec la propriété (8.4) qui garantit la liberté de la famille $(d_1, \ldots, d_{k+1})$. □

Ainsi, quitte à multiplier d_k par une constante, on peut supposer que $\alpha_{k,k} = 1$, et l'on adoptera donc la convention

$$d_k = \sum_{j=0}^{k-1} \alpha_{k,j} \nabla f(x_j) + \nabla f(x_k).$$

Attention, la descente effective se fera avec cette convention dans la direction $-d_k$. Mais utilisons de nouveau la relation d'orthogonalité

$$\forall k \neq j\,, \langle d_k, Ad_j\rangle = 0.$$

Puisque $Ad_j = -(\nabla f(x_{j+1}) - \nabla f(x_j))$ il apparaît que

$$\langle d_k, \nabla f(x_{j+1})\rangle = \langle d_k, \nabla f(x_j)\rangle.$$

Si $j = k-1$ on en tire

$$\|\nabla f(x_k)\|^2 = \langle d_k, \nabla f(x_k)\rangle = \alpha_{k,k-1}\|\nabla f(x_{k-1})\|^2.$$

Si $j < k-1$ on en tire

$$\alpha_{k,j+1}\|\nabla f(x_{j+1})\|^2 = \alpha_{k,j}\|\nabla f(x_k)\|^2.$$

En particulier, on obtient, pour tout $j \leq k-1$,

$$\alpha_{k,j} = \frac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_j)\|^2},$$

et, ainsi, la décomposition explicite

$$d_k = \sum_{j=0}^{k} \frac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_j)\|^2} \nabla f(x_j),$$

de sorte que

$$d_{k+1} = \frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2} d_k + \nabla f(x_{k+1}). \tag{8.5}$$

Nous sommes désormais en bonne voie pour réussir à mettre la méthode du gradient conjugué sous forme itérative. Pour conclure, il nous reste à calculer le pas optimal τ_k^* à faire dans la direction d_k.

On rappelle que τ_k^* est défini comme le minimiseur de la fonction

$$\tau \mapsto f(x_k + \tau d_k).$$

Par les mêmes arguments que ceux qui nous ont permis de calculer le pas optimal dans un cours précédent, on obtient immédiatement pour τ_k^* l'expression

$$\tau_k^* = \frac{-\langle \nabla f(x_k), d_k \rangle}{\langle A d_k, d_k \rangle}.$$

8.3 Résumé de la méthode du gradient conjugué

Si l'on résume toute l'analyse que nous venons de mener, la méthode du gradient conjugué peut être décrite de la manière suivante :

1. Initialisation en un point $x_0 \in \mathbb{R}^d$. La première direction de descente est $d_0 = \nabla f(x_0)$, le premier pas est $\tau_0^* = \frac{\langle \nabla f(x_0), d_0 \rangle}{\langle A d_0, d_0 \rangle}$ et $x_1 = x_0 - \tau_0^* d_0$.
2. Pour tout $k \in \mathbb{N}^*$, on définit

$$d_k = \frac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_{k-1})\|^2} d_{k-1} + \nabla f(x_k)\,, \tau_k^* := \frac{\langle \nabla f(x_k), d_k \rangle}{\langle A d_k, d_k \rangle}$$

   et on pose

$$x_{k+1} = x_k - \tau_k^* d_k.$$

3. L'algorithme converge en au plus d itérations.