Faculté des sciences et ingénierie (Toulouse III)
Département de mathématiques – M1 MAF
Modélisation en EDO, EDP, optimisation

Année universitaire
2012-2013

Laurent Dietrich

TP – Optimisation sans contraintes (1)

Il est recommandé de créer un nouveau répertoire dans lequel seront placés les fichiers relatifs à ce TP afin de ne pas mélanger, ou pire, écraser, d'anciens fichiers. Il est chaudement recommandé de taper **help** quelquechose dans l'invite de commande de Matlab, lorsque vous voulez des renseignements sur la commande quelquechose et/ou de m'interpeller. Il est aussi chaudement recommandé de ne pas aller recopier du code d'un ancien TP afin d'éviter de recopier au pire des erreurs, au mieux quelque chose dont le code n'aura pas la même structure qu'ici, ce qui vous fera perdre du temps, et de l'entraînement.

1 Premiers pas : dimension 2.

1.1 Préliminaires

On commence par la dimension 2 car on peut représenter les fonctions $\mathbb{R}^2 \to \mathbb{R}$ en $3D$ ou grâce aux lignes de niveau. On s'intéresse à un problème de minimisation d'une fonction quadratique, car on sait l'étudier théoriquement : on pourra donc comparer les résultats numériques aux résultats attendus.

Soit $A = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix}$, $b = (1,1)^T$ et $J_2(x) = \frac{1}{2}Ax \cdot x - b \cdot x$

1. Expliquer pourquoi le problème de minimisation de J_2 sur $\mathbb{R}^2$ a une solution unique.
2. En déterminer la solution.
3. Représenter J_2 sur le pavé $[-10, 10]^2$ grâce aux commandes **meshgrid** et **mesh**.

1.2 Une première méthode de minimisation : le gradient à pas fixe

1. Programmer l'algorithme du gradient à pas fixe $\mu > 0$ pour minimiser J_2. On rappelle que l'itération de l'algorithme en question s'écrit

$$x_0 \in \mathbb{R}^2,\ x_{n+1} = x_n - \mu \nabla J_2(x_n)$$

2. Proposer un critère d'arrêt des itérations de cet algorithme, l'inclure dans votre code.

3. Représenter sur une même figure les lignes de niveau de J_2, le champ de gradients de J_2 ainsi que les positions des itérés successifs x_i. On utilisera les commandes **contour**, **quiver**, **plot** et **hold** on.
4. Constater et expliquer pourquoi il est important de choisir le pas $\mu > 0$ ni trop grand, ni trop petit.

1.3 Une autre méthode : le gradient à pas optimal

Plutôt que de se fixer un pas $\mu > 0$ ni trop grand ni trop petit, on peut vouloir choisir le meilleur pas possible à chaque itération au sens suivant :

$$x_{n+1} = x_n - \mu_n \nabla J_2(x_n)$$

avec μ_n choisi tel que

$$\mu_n = \text{argmin}_{\alpha \in \mathbb{R}}(x_n - \alpha \nabla J_2(x_n))$$

Il faut donc, à chaque itération, résoudre un problème d'optimisation en dimension 1. Expliquer les avantages et inconvénients de cette méthode.

1. Expliquer pourquoi dans le cadre d'une fonction quadratique, ce problème de minimisation $1D$ est soluble analytiquement (le résoudre).
2. En règle générale, proposer des méthodes (simples) pour résoudre ce problème de dimension 1.
3. En déduire une implémentation de l'algorithme du gradient à pas optimal (on pourra utiliser la solution analytique que l'on aura calculée, pour commencer).
4. Ré-itérer les illustrations précédentes.

2 En dimension finie quelconque

1. Implémenter les algorithmes du gradient à pas fixe et à pas optimal en dimension n où A désigne cette fois une matrice tridiagonale $n \times n$ avec des 4 sur la diagonale et des -2 sur les sur et sous-diagonales, et $b = (1, \ldots, 1)^T$. On pourra utiliser **eig** pour se convaincre du caractère défini positif de cette matrice, même si c'est un joli exercice que de le montrer à la main. On pourra bien sûr moins bien représenter les choses.
2. Il est intéréssant de comparer la vitesse de convergence des deux algorithmes selon les paramètres (on n'utilisera pas la solution analytique pour le calcul de μ^n si on veut obtenir un vrai temps de calcul), et le temps de calcul qu'ils nécéssitent en fonction de la dimension n. On pourra proposer des figures qui illustrent cela.