Université Paris Descartes
UFR de Mathématiques et Informatique
45, rue des Saints-Pères 75270 Paris cedex 06

MASTER 1 INFO 2014–2015

Optimisation Algorithmique

Polycopié et notes autorisés. Durée 1h15.

Fichiers disponibles :

1. Cours, fiches de TD/TP et corrigé des TP dans le répertoire COURS_TP ;
2. dans le répertoire SCILAB le polycopié Scilab ;
3. les fichiers CC_Ex1.sci, CC_Ex1.sce, CC_Ex2.sci et CC_Ex2.sce sont à compléter et contiennent déjà une partie de code à utiliser !

À remettre :

1. les fichier `CC_Ex1.sce`, `CC_Ex1.sce`, `CC_Ex2.sci` et `CC_Ex2.sce` des fonctions respectivement des commandes Scilab avec
votre nom en commentaire ;
2. la copie double sur laquelle vous pouvez expliquer ce que vous avez fait : calculs, formules, graphiques, problèmes rencontrés,...

Exercice I

Pour $x = (x_1, x_2) \in \mathbb{R}^2$ on définit $f(x) = \dfrac{1}{2}(20x_1^2 + x_2^2)$.

1. Calculer $\nabla f(x)$ et $H_f(x)$. Où se trouve le minimum x^* de f ?
2. En choisissant quatre valeurs initiales distinctes $x^{(0)}$ illustrer le comportement de convergence de la méthode de descente du gradient.
On tracera : lignes de niveaux, parcours de la suite $(x^{(\mathrm{k})})_k$, précision du résultat,... dans chaque cas. Commentez.
3. Illustrez le comportement de la méthode de Newton appliqué à la minimisation de la fonction f. Commentez.

Exercice II

On s'intéresse à la minimisation de la fonctionnelle coût :

$$f(x) = \sum_{j=1}^{k} \left(\|x - y_j\|_2^2 - d_j^2 \right)^2 ,$$

où $y_1, \ldots, y_m \in \mathbb{R}^n$ et $d_1, \ldots, d_m \in \mathbb{R}$.

On veut comparer les performances de la méthode de descente de gradient et de la méthode de Newton sur les données ($n = 2$ et $m = 5$) :

$$y_1 = \begin{pmatrix} 1.8 \\ 2.5 \end{pmatrix}, y_2 = \begin{pmatrix} 2.0 \\ 1.7 \end{pmatrix}, y_3 = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}, y_4 = \begin{pmatrix} 1.5 \\ 2.0 \end{pmatrix}, y_5 = \begin{pmatrix} 2.5 \\ 1.5 \end{pmatrix}$$

et $(d_1, \ldots, d_5) = (2.00,\ 1.24,\ 0.59,\ 1.31,\ 1.44)$

Compléter les fichiers `CC_Ex2.sci` et `CC_Ex2.sce` afin de tester les deux méthodes avec les valeurs initiales : `x0 = [ 0 ; 0]`, `x0 = [3.5; 3.5]` et `x0 = [2.5; 2.7]`.

On tracera : lignes de niveaux, parcours de la suite $(x^{(\mathrm{k})})_k$, précision du résultat,... dans chaque cas. Commentez.