Université de Versailles   Année 2022-2023
http://dumas.perso.math.cnrs.fr/M1-optim.html   semestre 1

TP1: méthodes de desente, problème sans contraintes

L'objectif de cette séance est d'utiliser le logiciel Matlab (ou Python) afin de comparer différents algorithmes de descente avec recherche linéaire pour la minimisation locale d'une fonction définie sur $\mathbb{R}^n$.

Exercice 1

On cherche à programmer tout d'abord la méthode du gradient avec une stratégie de type backtracking avec condition d'Armijo. On rappelle que la condition d'Armijo pour un point de départ X_0 et une direction de descente d s'écrit :

$$J(X_0 + \alpha d) \leq J(X0) + \beta\alpha < d, \nabla J(X_0) >$$

1. Ecrire une fonction Matlab (ou Python) ayant pour arguments ∇J, β, α_{init} , τ (paramètres du procédé de backtracking), X_0 (point initial) et N et renvoyant la valeur de X_N .

2. On cherche à minimiser la fonction de Rosenbrock en dimension 2 :

   $$J(x, y) = 100(y - x^2)^2 + (x - 1)^2$$

   On prend pour cela $\beta = 0.1$, $\alpha_{init} = 1$, $\tau = 0.3$, $X_0 = (0, 1)$.

   Représenter graphiquement sur deux figures séparées :

   (a) les lignes de niveau de la fonction de Rosenbrock sur $[1, 2]^2$ et les 100 premières itérations de l'algorithme précédent

   (b) la fonction $N \mapsto J(N)$ pour $N \in \{0, , ..., 100\}$

Exercice 2

En suivant la même démarche que dans l'exercice 1, écrire une nouvelle fonction Matlab (ou Python) utilisant la méthode de Newton pour minimiser une fonction J et la tester sur le même exemple (fonction de Rosenbrock) avec ou sans recherche linéaire. La méthode de Newton consiste à prendre pour direction de descente la direction d telle que:

$$d = -HJ(X)^{-1}.\nabla J(X)$$

où $HJ(X)$ désigne la matrice Hessienne de J en X.

Exercice 3

Reprendre l'exercice précédent en utilisant cette fois la méthode BFGS. On rappelle que dans ce cas, le hessien de J en X_k est approché par la matrice B_k telle que

$B_0 = I_n$ (matrice identité) et

$$B_k = B_{k-1} + \frac{1}{< s_k, y_k >} y_k({}^t y_k) - \frac{1}{< s_k, B_{k-1} s_k >} B_{k-1} s_k ({}^t s_k) B_{k-1}$$

avec $s_k = X_k - X_{k-1} \in \mathbb{R}^n$ et $y_k = \nabla J(X_k) - \nabla J(X_{k-1}) \in \mathbb{R}^n$ (vecteurs colonnes).

Conclure sur l'efficacité comparée des méthode de gradient, Newton et BFGS.