Méthodes numériques en Optimisation
T.P.1

Les langages de programmation sont Matlab ou Octave,

1 Objectifs :

Le but de ce TP est d'implémenter plusieurs méthodes numériques de minimisation vues en cours, en construisant des fonctions **Matlab** ou **Octave** qui approchent l'unique point de minimum des fonctions suivantes définies sur $\mathbb{R}^n$ à valeurs dans $\mathbb{R}$, avec $n \in \mathbb{N}^*$:

— Fonction quadratique fortement convexe :

$$f_1(u) = \frac{1}{2}\langle Au\,,\,u\rangle - \langle b\,,\,u\rangle, \quad \forall\, u \in \mathbb{R}^n \tag{1}$$

avec $A \in \mathcal{M}_n(\mathbb{R})$ matrice SDP, $b \in \mathbb{R}^n$ données.

— Fonction non quadratique fortement convexe

$$f_2(u) = \frac{1}{2}\langle Au\,,\,u\rangle - \langle b\,,\,x\rangle + g(u), \quad \forall\, u \in \mathbb{R}^n \tag{2}$$

avec $A \in \mathcal{M}_n(\mathbb{R})$ matrice SDP, $b \in \mathbb{R}^n$ et $g : \mathbb{R}^n \to \mathbb{R}$ une fonction de classe C^2 et convexe (montrer que f_2 est fortement convexe).
Prendre par exemple $g(u) = e^{\lambda_1 u_1} + e^{\lambda_2 u_2} + \cdots e^{\lambda_n u_n}, \quad \forall\, u = (u_1, u_2, \cdots u_n)^T \in \mathbb{R}^n$ avec $\lambda_1, \cdots \lambda_n \in \mathbb{R}$ données ; vérifier d'abord que g est bien une fonction convexe.

— La fonction vue dans le TT1 sur le contrôle optimal en temps discret :

$$f_3(u) = \frac{1}{2}\|u\|^2 + \frac{\alpha}{2}(x_n - d)^2, \quad \forall\, u \in \mathbb{R}^n \tag{3}$$

avec $\alpha > 0,\ d \in \mathbb{R}^n$ et $x =$ le vecteur des états du système contrôlé ; rappelons que x est lié au vecteur u par la loi d'évolution (1) de TT1.

2 Méthodes à implémenter

On se propose d'implémenter quelques **méthodes de gradient**.
Pour une fonction arbitraire $f : \mathbb{R}^n \to \mathbb{R}$ à minimiser, ces méthodes s'écrivent sous la forme :

$$v^{(k+1)} = v^{(k)} - \rho_k \nabla f(v^{(k)}).$$

Les différentes manières de choisir $\rho_k \in \mathbb{R}$ donnent des méthodes de gradient différentes.
Pour chacune des fonctions $f = f_i,\ i = 1, 2, 3$ on propose d'utiliser les méthodes suivantes :

1. **Méthode de gradient à pas fixe**
   prendre $\rho_k = \rho > 0$ constante
2. **Méthode de gradient à pas optimal**
   On cherche $\rho_k \in \mathbb{R}$ tel que

$$f(v^{(k)} - \rho_k \nabla f(v^{(k)})) = \min_{\rho \in \mathbb{R}} f(v^{(k)} - \rho \nabla f(v^{(k)}))$$

   Implémenter cette méthode seulement dans le cas où ρ_k peut se calculer explicitement, c'est à dire dans le cas de la fonction quadratique f_1 (voir un exercice du TD4).

Scanned with CamScanner

3. **Méthode de gradient utilisant la règle d'Armijo :**
   Cette méthode consiste à trouver le plus petit $j \in \mathbb{N}$ tel que

$$f\left(v^{(k)} - \frac{1}{2^j}\nabla f(v^{(k)})\right) \leq f(v^{(k)}) - \frac{\omega}{2^j}\|\nabla f(v^{(k)})\|^2 \tag{4}$$

   avec $0 < \omega < 1$ (par exemple $\omega = \frac{1}{2}$) ; prendre alors $\rho_k = \frac{1}{2^j}$.

3 Quelques conseils pour rédiger le compte rendu

Le compte rendu doit contenir, pour chacune des méthodes utilisées et pour chacune des fonctions f_1, f_2 et f_3 :

1. Une description de la méthode et du rôle de chaque fonction **Matlab / Octave** que vous écrivez.
2. Au moins un test de la méthode sur un cas concret simple (avec $n \geq 2$) pour lequel on connait la solution exacte ; comparer la solution numérique et la solution exacte.
3. Au moins un test sur un cas concret plus compliqué et pour lequel on ne connait pas de solution exacte.
4. Pour chaque test donner l'historique de convergence de l'algorithme respectif (c'est à dire, afficher à chaque itération la valeur de la fonction coût).

Il faut mettre tous les codes dans une annexe à la fin du compte rendu.

Scanned with CamScanner

Méthodes numériques en Optimisation
T.P.2

Le langage de programmation est Matlab ou Octave,

1 Objectifs :

Le but de ce TP est d'implémenter la méthode de gradient avec projection pour résoudre numériquement des problèmes de minimisation avec contraintes, où l'ensemble des contraintes est un pavé. Plus précisément nous considérons l'ensemble suivant des contraintes en $\mathbb{R}^n$, avec $n \in \mathbb{N}^*$:

$$U = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \subset \mathbb{R}^n$$

avec $-\infty < a_i < b_i < +\infty, \quad \forall\, i \in [[1, n]]$.
Pour une fonction $f : \mathbb{R}^n \to \mathbb{R}$ donnée de classe C^1 on considère le problème de minimisation avec contraintes :

$$\min_{x \in U} f(x)$$

c'est à dire, trouver $x^* \in U$ tel que

$$f(x^*) \leq f(x), \quad \forall\, x \in U.$$

Dans ce TP nous allons considérer successivement $f = f_1$ et ensuite $f = f_2$ où f_1, f_2 sont les fonctions considérées dans le TP1 (le cas quadratique et le cas non-quadratique).
Rappelons que la méthode de gradient avec projection consiste à construire une suite $\{v^{(k)}\}_{k \in \mathbb{N}} \subset \mathbb{R}^n$ des approximations de la solution recherchée par une relation de récurrence du type

$$v^{(k+1)} = P_U\left(v^{(k)} - \rho_k \nabla f(v^{(k)})\right).$$

où P_U est la projection sur l'ensemble convexe et fermé U et $\rho_k > 0$ à choisir convenablement.

Pour chacune de deux fonctions f_1, f_2 écrire une fonction **Matlab** *ou* **Octave** *qui calcule une approximation numérique de l'unique point de minimum de la fonction sur U.*

Suivre les conseils donnés en TP1 pour la rédaction du compte rendu.

Scanned with CamScanner