1 Exercices avec Julia

Exercice 1 *[Noyaux et contraintes linéaires]* Considérez les données suivantes:

$$\bar{x} = (1\ 2\ 3\ 1\ 2\ 3)^t \tag{1}$$

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 3 & 5 & 2 & 4 & 6 \\ 1 & 1 & 3 & 3 & 5 & 5 \end{pmatrix} \tag{2}$$

$$b = (10\ 10\ 10)^t \tag{3}$$

et le problème de trouver le point le plus proche de $\bar{x}$ dans les contraintes $Ax = b$, $\min_{Ax=b} \frac{1}{2}\|x - \bar{x}\|^2$. Je vous fournis un calcul de x_0 et Z en utilisant les outils de Julia ainsi qu'un modèle LLSModel qui calcule $\phi(\alpha) = f(x_0 + Z\alpha)$. Les étapes qui suivent consistent à ajouter les calculs pertinents dans le fichier `test0.jl`.

a) Obtenez la solution $x^* = x_0 + Z\alpha^*$ en minimisant la fonction $\phi(\alpha) = f(x_0 + Z\alpha)$. Fourni en guise d'exemple `test0.jl` qui utilise le gradient conjugué linéaire.

b) Obtenez directement la solution x^* en utilisant les conditions d'optimalité de la fonction $\phi(\alpha)$, i.e. $0 = \nabla\phi(\alpha) = \nabla f(x_0 + Z\alpha)Z$.

c) Obtenez directement la solution x^* en résolvant un système de deux blocs: $\nabla f(x^*)Z = 0$ et $Ax^* = b$ totalisant $n = 6$ variables et $3 + 3$ équations.

d) Obtenez la solution x^* et λ^* à partir des conditions avec les multiplicateurs de Lagrange. Cette fois, on aura $n + m = 9$ variables et équations. Fournissez λ^* et vérifiez que $\nabla f(x^*) + \lambda^* A = 0$. Il est commode de tout ramener à des vecteurs colonnes pour uniformiser les systèmes d'équations à résoudre.

e) Pour une solution obtenue à l'aide des matrices Z, trouvez une formulation pour obtenir λ^* et testez-là dans Julia.

□

Exercice 2 *[Contraintes d'inégalité]* Considérez les mêmes données que dans l'exercice 1 sauf que les contraintes sont maintenant des contraintes d'inégalité inférieur ou égal. La solution obtenue dans l'exercice 1 comporte un $\lambda_3 < 0$ et donc ne constitue pas la solution. Pour obtenir la solution sous contraintes d'inégalités, il faut identifier correctement les contraintes satisfaites avec égalité. Nous explorons deux méthodes pour ce faire.

a) La solution naïve va tester tous les choix possibles. L'exercice 1 a déjà testé le choix $I^* = \emptyset$. Je vous fournis un code qui utilise l'itérateur `subsets` pour générer toutes les combinaisons possible d'ensembles actifs. Vous devez y insérer le test que la combinaison est la bonne. Il s'agit d'ajouter le bon test d'arrêt au fichier `test1.jl`.

b) Maintenant, essayons une solution un peu plus efficace. Pour une combinaison donnée (on démarre avec $I^* = \emptyset$), on résout le problème en utilisant le système d'équations primal-dual. Si au moins une contrainte inactive (celles pas dans I^*) n'est pas satisfaite, on ajoute à I^* la plus violée et on résout à nouveau; autrement si au moins un multiplicateur λ_{I^*} est négatif, on enlève de I^* l'indice du plus négatif et on résout à nouveau. Autrement, la solution est optimale car toutes les contraintes et tous les multiplicateurs sont du bon signe.Vous devez compléter `test2.jl`.

□

2 Liste des fichiers

- `Exemple1.jl`: l'exemple de base.
- `Exemple2.jl`: un générateur de problèmes pour n et m variables.
- `GC_lin.jl`: un gradient conjugué linéaire de base.
- `test0.jl`: l'exemple avec contraintes d'égalité.
- `test1.jl`: l'exemple simple résolu par énumération des contraintes actives.
- `test2.jl`: exemple d'algo.
- `compare1.jl et compare2.jl` comparent les deux solveurs avec inégalités à l'exemple simple (compare1) et à un problème aléatoire (compare2).