Évaluation optimisation convexe II

L'évaluation est un travail d'analyse, à faire en groupe d'un ***maximum de 3 personnes et d'un minimum de 2****. Ce travail doit donner lieu à un exposé. Ces exposés auront lieu entre le* ***24 et le 31 janvier****.*
Il revient à chaque groupe de me transmettre les noms de ses membres par mail à l'addresse ***bashar.dudin@epita.fr*** *avant le* ***12 janvier****.*

Exposé

L'exposé est d'une durée de ***20*** minutes suivies de ***5*** à ***10*** minutes de discussion avec le jury [1]. Vous serez évalués sur la ***compréhension*** que vous aurez du contenu que vous abordez mais également sur la ***clarté*** de l'exposé. Rappelez-vous qu'un dessin vaut parfois mieux qu'un long discours.

Rendu

L'ensemble des implémentations ainsi que les datasets et les problèmes d'optimisations générés sont à rendre au plus tard le ***23 janvier***, il en va de même de vos slides.

Contraintes techniques

Les implémentations seront à faire en `python`. L'environement `python` permet un prototypage agréable et les bibliothèques de ML qui y sont disponibles nous seront utiles pour faire des comparatifs. On ajoute à cela le fait que vous n'avez pas à vous préoccuper de dépassement de taille maximal des flottants, vu le comportement par défaut de `python`.

Vous pouvez utiliser les fonctions `numpy`, `scipy` et `pandas` liées au calcul différentiel, à l'algèbre linéaire et au pré-traitement et traitement des datasets que vous seriez amenés à manipuler. Tout ce qui n'est pas explicitement mentionné est à proscrire.

Attendus

Des ***dessins*** pour résumer les études de performances. On attend de vous des tests fiables, donc en nombre suffisamment important et de préférence avec une estimation du risque. Aucun comparatif ou analyse de performance d'un algo n'a de sens sinon. À vous de vous mettre dans la peau de quelqu'un qui participe par son expertise à une decision importante sur un choix de techno.

Quand on parle de comparaison, on entend batch de tests et résumé des résultats en sortie.

1 Descentes de gradient sans contraintes

Cette section ***n'est pas optionnelle***, ce qui n'est pas en bonus doit être traité par chaque groupe.

1. Moi :).

Question 1-1. 6 P.

1. Construires des familles de fonctions qui ont des nombres de conditionnements quelconques (≥ 1).
2. Tracer le nombre d'itérations d'une descente de gradient à pas constant contre le nombre de conditionnement d'une même famille de fonctions.
3. Effectuer l'étude précédente pour différents pas.

Question 1-2. Comparer les descentes de gradients en normes ℓ_2 et ℓ_1. 2 P.

Question 1-3. Procéder de même en comparant l'une des deux descentes précédentes à celle de la descente de gradient stochastique. Pour quelle raison nous ne l'avons pas aabordé en cours ? (+2 P.)

2 Thématiques plus avancées

Les thématiques de cette section sont aux choix, il vous revient de choisir lesquelles et combien d'entre elles vous souhaitez traiter.

2.1 Méthode de Newton

Cette thématique est centrée autour de la méthode de Newton dans le cas des contraintes d'égalité. Dans les implémentations que vous abordez il n'est pas nécessaire de traiter le cas d'un point initial non-admissible. Son traitement sera comptabilisé s'il est fait par des points bonus [2].

Il vous est également autorisé d'utiliser les méthodes de calcul numérique ou symbolique des gradients, hessiennes et les solveurs de systèmes linéaires disponibles dans les bibliothèques `numpy` et `scipy`.

Question 2-4. Implémenter et tester une méthode de Newton sur un problème d'optimisation convexe avec contraintes d'égalité. 4 P.

Question 2-5. Comparer la méthode de Newton précédente avec une approche qui procède par élimination des contraintes d'égalité. 3 P.

Question 2-6. Expliquer une méthode de quasi-Newton. Comparer celle implémentée par `scipy` à la méthode de Newton précédente. 3 P.

2.2 SVM et SMO

Dans cette thématique on cherche à implémenter la ***Sequential Minimal Optimisation*** concernant le traitement des SVMs. C'est une implémentation du contenu abordé en cours.

Question 2-7. Implémenter la SMO. Laisser la possibilité de passer le noyau souhaité en argument. 5 P.

2. Tout travail mérite salaire.

Question 2-8. Tester l'implémentation précédente sur le problème de classification proposé par le dataset ***MNIST***. Dans le but de traiter un problème de classification binaire on se limite au repérage d'un chiffre.
Garder le noyau qui vous donne les meilleurs résultats suivant la métrique de votre choix. — 3 (+1) P.

Question 2-9. Comparer votre modèle précédent au classificateur SVM proposé par `sklearn`. — 2 P.

2.3 Méthode du point intérieur

Cette thématique est plus théorique que les précédentes. La méthode du point intérieur n'a pas été abordée en cours ; elle est évaluée en conséquence. On part ici sur le prolongement naturel des problématiques abordées en cours ; étudier une méthode générale de résolution de problèmes d'optimisation convexe. Elle sera intégrée dès cette année au contenu des cours d'OCVX.

Le but est de comparer la performance de la méthode du point intérieur à celle du simplexe.

Question 2-10. Expliquer la méthode primal-dual du point intérieur exposée dans [BV04, 11.7]. — 4 P.

Question 2-11. En donner une implémentation en présupposant que l'origine est toujours un point admissible [a]. — 6 P.

a. Il y a des méthodes standards pour se ramener à ce cas.

Question 2-12. Comparer la performance de la méthode précédente à celle de la méthode du simplexe implémentée par `scipy` [a]. — (+2 P.)

a. Bien entendu, dans le cas des programmes linéaires.

Références

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.