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 rapport, dans le format de votre choix, ainsi qu'à une soutenance.*
Un suivi intermédiare ***obligatoire*** *est prévu ; il a pour but d'aiguiller chaque groupe sur les objectifs à atteindre et d'apporter un support au problèmes rencontrés. Il sera également l'occasion de rappeler mes attentes en terme de rendu.*

Rendu

Le rendu doit contenir :
— un rapport ;
— l'ensemble des implémentations ;
— les datasets utilisés et les problèmes d'optimisations générés, si applicable ;
— les slides de la soutenance.

Les dates de soutenances seront fixés le 9 juillet après discussion avec les délégués de classe et l'administration pédagogique. Les contenus hors slides seront à rendre 24 heures avant l'heure de soutenance. Vous êtes libres en ce qui concerne le format de rendu. Votre choix sera jugé quant à son adéquation avec les contraintes de ***clarté***, de ***compréhensibilité*** et d'***éxhaustivité***.

Des répertoirs git seront crée au cours de la semaine du 16 juillet pour accueillir vos rendus.

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 vous seront utiles pour faire des comparatifs.

Vous pouvez utiliser les fonctions de `numpy`, `scipy` et de `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 [1].

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. Je suis seul à pouvoir vous libérer de cette contrainte. Il vous revient de me tenir informé de toute contravention à cette règle.

1.1 Le classique

Question 1-1.

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.

1.2 Accélération de convergence

Dans cette section on étudie des techniques d'accélération et/ou de garantie de convergence liée à la descente de gradient en norme ℓ_2.

Question 1-3. Qu'est-ce ***Learning Rate Scheduling*** ? En implémenter une instance et le tester sur le batch test généré à la section précédente.

Question 1-4. Qu'est-ce que le ***Nesterov Accelerated Gradient*** et la ***Adam Optimization*** ? Implémenter des instances de chacune. Générer des exempes qui permettent de les comparer entre elles et contre une descente de gradient classique.

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 [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 2-5. Générer des problèmes d'optimisations convexes avec contraintes d'égalités.

Question 2-6. Implémenter et tester une méthode de Newton.

Question 2-7. Utiliser la méthode de Newton avec contraintes d'égalités pour minimiser des

2. Tout travail mérite salaire.

fonctions polynômiales dans les valuations d'un flot sur un graphe [a].

a. Me demander des explications.

Question 2-8. Quid d'une méthode de Quasi-Newton ?

2.2 SVM et SMO

Dans cette thématique on cherche à implémenter la ***Sequential Minimal Optimisation*** concernant le traitement des SVMs.

Question 2-9. Expliqer le problème d'optimisation sous-jacent à un ***Support Vector Classifier*** ?

Question 2-10. Implémenter la SMO. Laisser la possibilité de passer le noyau souhaité en argument.

Question 2-11. 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.

Question 2-12. Implémenter une méthode de résolution par barrière logarithmique d'un ***SVM***. Comparer cette démarche à la SMO.

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.

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

Question 2-13. Expliquer la méthode primal-dual du point intérieur exposée dans [BV04, 11.7]. 4 P.

Question 2-14. 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 2-15. Comparer la performance de la méthode précédente à une méthode du simplexe de votre cru. [a]. (+2 P.)

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

Question 2-16. Appliquer les méthodes de résolution précédentes au cas des flots maximaux sur les graphes.

Références

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