ÉCOLE NORMALE SUPÉRIEURE

INTRODUCTION AU DOMAINE DE RECHERCHE :

Extensions de l'algorithme de Frank-Wolfe pour la recherche de points selles

Gauthier GIDEL

DÉPARTEMENT DE MATHÉMATIQUES ET APPLICATIONS

7 Octobre 2016

Table des matières

1 Introduction

L'optimisation convexe a connu une large résurgence ces 50 dernières années d'une part grâce aux nombreux champs d'application qui sont peu à peu apparus au cours de la seconde partie du $20^{\text{ème}}$ siècle et d'autre part suite aux progrès et à la démocratisation de l'informatique des 50 dernières années. Suivant ces progrès, plusieurs domaines ont commencé à tenter de résoudre concrètement des problèmes d'optimisation ne possédant pas de solution analytique, comme par exemple, le problème de transport de masse [Kantorovich, 1942] en économie ou encore les problèmes de recherche opérationnelle développés par Dantzig [1963]. La physique numérique a utilisé de nombreux algorithmes d'optimisation pour trouver des positions d'équilibre minimisant une énergie. Par exemple, la recherche des points de points d'équilibre du modèle d'Ising [Lucas, 2013] mène à de nombreuses applications comme le débruitage d'images [Nishimori, 2001, Section 6]. Cette dernière application fait partie du champ entier qu'est le traitement d'image. Il forme de nos jours, avec l'allocation de ressources à grande échelle et l'apprentissage statistique, un trio de champs d'application utilisant drastiquement l'optimisation à grande échelle. C'est ainsi que des algorithmes antérieurs à ces nouvelles problématiques ont connus un regain d'intérêt. C'est le cas de l'algorithme de Frank-Wolfe (FW) [Frank and Wolfe, 1956] : créé en 1956, il a été initialement proposé pour résoudre des problèmes quadratiques [Bertsekas, 1999, Luenberger, 1973] en résolvant itérativement plusieurs problèmes linéaires [Dantzig, 1963, Luenberger, 1973]. Dans cette introduction au domaine de recherche, après avoir introduit les définitions de base de l'optimisation convexe, nous développerons les concepts de boîte noire d'optimisation et d'oracle. Nous présenterons ensuite l'algorithme de Frank-Wolfe. Enfin nous nous intéresserons à l'optimisation de points selles (ou point-col) et aux extension possibles des algorithmes d'optimisation convexe.

2 Préliminaires

Les objets centraux de cette introduction au domaine de recherche sont les fonctions et les ensembles convexes. Dans le but de garder en vue les applications décrites dans l'introduction, nous ne nous placerons pas dans le cas le plus général possible. Nous nous restreindrons à la dimension finie et donc au cas des convexes de $\mathbb{R}^d$.

Définition 1 (Ensemble convexe)**.** *Soit $\mathcal{X} \subset \mathbb{R}^d$. $\mathcal{X}$ est convexe si pour tout $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$ et $\lambda \in [0,1]$,*

$$\lambda x + (1-\lambda)\, y \in \mathcal{X}. \tag{1}$$

Définition 2 (Fonction convexe)**.** *Soit $f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ où $\mathcal{X}$ est convexe. On dit que f est une fonction convexe si pour tout $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$ et $\lambda \in [0,1]$,*

$$f(\lambda \boldsymbol{x} + (1-\lambda)\, \boldsymbol{y}) \leq \lambda f(\boldsymbol{x}) + (1-\lambda)\, f(\boldsymbol{y}). \tag{2}$$

Si une fonction est deux fois différentiable, la convexité est équivalente au fait que sa hessienne soit semi-définie positive en tout point. Plus tard nous aurons aussi besoin des notions de convexité forte, une fonction β-fortement convexe est une fonction telle que la différence $f(\cdot) - \frac{\beta}{2}\|\cdot\|_2^2$ est convexe.

Définition 3 (Fonction β-fortement convexe)**.** *Soit $f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ et $\beta > 0$. f est une fonction β-fortement convexe si pour tout $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$ et $\lambda \in [0,1]$,*

$$f(\lambda \boldsymbol{x} + (1-\lambda)\, \boldsymbol{y}) \leq \lambda f(\boldsymbol{x}) + (1-\lambda)\, f(\boldsymbol{y}) - \frac{\beta}{2}\lambda\,(1-\lambda)\,\|\boldsymbol{x} - \boldsymbol{y}\|_2^2. \tag{3}$$

L'étude des fonctions convexes en optimisation est justifiée par le fait que l'on se place dans le cadre d'une boîte noire locale (voir Définition 8). Les algorithmes n'utilisant que les propriétés locales des fonctions ne peuvent qu'assurer la convergence vers un minimum local. Or pour une fonction convexe un minimum local est un minimum global, ainsi la convergence théorique vers le point souhaité qu'est le minimum global de la fonction peut être démontré. Lorsque $\mathcal{X}$ est ouvert, les minimums locaux sont des points stationnaires.

Définition 4 (Point stationnaire)**.** *Soit $f : \mathcal{X} \to \mathbb{R}$. On dit que $\boldsymbol{x} \in \mathit{int}(\mathcal{X})$ est un point stationnaire de f si*

$$\nabla f(\boldsymbol{x}) = 0. \tag{4}$$

Néanmoins, cette définition n'est pas satisfaisante lorsque l'on veut traiter des problèmes d'optimisation sous contraintes. En effet si $\mathcal{X}$ est d'intérieur vide ou si un minimum est atteint au bord ce n'est pas forcément un point critique. Dans le premier cas ($\mathcal{X}$ d'intérieur vide) nous pouvons toujours considérer la restriction de notre problème au sous espace affine minimal contenant $\mathcal{X}$, dans lequel ce dernier ne sera pas d'intérieur vide (à moins d'être déjà vide). Ainsi, quitte à projeter on peut considérer que l'on travaille avec des fonctions convexes dont l'ensemble de définition $\mathcal{X}$ est un ensemble convexe d'intérieur non vide. Dans le second cas nous avons besoin d'une nouvelle caractérisation. Grossièrement, l'idée est de considérer les points du bord où le gradient indiquerait une direction orthogonale au bord dans le sens sortant de $\mathcal{X}$, dans ce cas le seul moyen de continuer à minimiser la fonction serait de sortir de l'ensemble $\mathcal{X}$ (voir Figure 1). Pour caractériser rigoureusement un minimum atteint au bord de l'ensemble convexe $\mathcal{X}$ nous aurons besoin du concept de fonction sous différentiable.

Définition 5 (Fonction sous différentiable)**.** *Soit $f : \mathcal{X} \to \mathbb{R}$. On dit que le vecteur $\boldsymbol{g}$ est un sous gradient de f au point $\boldsymbol{x}$ si*

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \langle \boldsymbol{g}, \boldsymbol{y} - \boldsymbol{x} \rangle, \quad \forall \boldsymbol{y} \in \mathbb{R}^d. \tag{5}$$

On appelle alors $\partial f(\boldsymbol{x})$ l'ensemble des sous gradients de f au point $\boldsymbol{x}$.

De cette définition il découle immédiatement que pour tout $\boldsymbol{x} \in \mathbb{R}^d, \partial f(\boldsymbol{x})$ est un convexe fermé. Ainsi, on peut maintenant définir une condition suffisante d'optimalité.

Définition 6 (Condition d'optimalité)**.** *Soit $f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ où $\mathcal{X}$ est convexe. Le point $\boldsymbol{x}^* \in \mathbb{R}^d$ est un point optimal de f si et seulement si il existe un sous gradient $\boldsymbol{g} \in \partial f(\boldsymbol{x})$ tel que*

$$\langle \boldsymbol{g}, \boldsymbol{y} - \boldsymbol{x} \rangle \geq 0, \quad \forall \boldsymbol{y} \in \mathcal{X}. \tag{6}$$

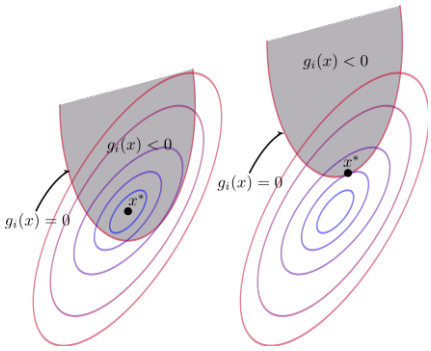

FIGURE 1 – Condition d'optimalité dans le domaine $\mathcal{X} := \{\boldsymbol{x}\,;\ g_i(\boldsymbol{x}) \leq 0\}$: le gradient est orthogonal à la frontière $\{\boldsymbol{x}\,;\ g_i(\boldsymbol{x}) = 0\}$. *source : wikipedia*

Si f est convexe et $\mathcal{X}$ est un convexe compact alors on a existence d'un point optimal.Cette caractérisation est illustrée Figure 1. C'est celle que nous allons utiliser par la suite. Elle peut de plus, habilement se généraliser à la caractérisation de points selles (voir Partie 5). Dans la suite nous utiliserons la dénomination *point selle*.

> Pour résumer, nous considèrerons dans la suite (sans perte de généralité) des fonctions convexes $f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ avec $\mathcal{X}$ ensemble d'intérieur non vide.

3 Oracles et optimisation de boîte noire

Dans son livre, Nesterov [2004] définit les concepts *d'oracle* et de *boîte noire locale*. Ces deux notions sont des notions clés de l'optimisation contemporaine. Ces concepts ont été repris plus récemment par Bubeck [2014].

3.1 Oracle

Tout d'abord, intéressons nous à la notion *d'oracle*, un oracle est un objet assez général qui est difficile à définir rigoureusement sans contexte plus précis. Une définition très large est donnée

par Nesterov [2004] : un oracle $\mathcal{O}_f$ est une application qui répond à une question posée par un algorithme. Ce dernier tente alors de résoudre le problème d'optimisation grâce aux réponses de l'oracle. Plus précisément, en supposant que nous possédons une ressource suffisante de calculs et que le domaine $\mathcal{X} \subset \mathbb{R}^d$ est connu, nous pouvons optimiser la fonction (*inconnue dans sa globalité*) $f : \mathcal{X} \to \mathbb{R}^d$ à l'aide d'un *oracle d'ordre* k.

Définition 7 (Oracle d'ordre 0,1 [Bubeck, 2014])**.** *Soit $f : \mathcal{X} \to \mathbb{R}$ une fonction.*

- *Un oracle d'ordre zéro prend comme entrée un point $\boldsymbol{x} \in \mathcal{X}$ et renvoie la valeur de f au point $\boldsymbol{x}$.*
- *Un oracle d'ordre un prend en entrée un point $\boldsymbol{x} \in \mathcal{X}$ et renvoie un sous gradient de f.*

Cette notion est essentielle en optimisation et résume les préoccupations pratiques du domaine. En effet, l'oracle est la modélisation de l'interaction que l'on peut avoir avec le problème. Cette interaction est le plus souvent locale pour des raisons pratique puisqu'il semble peu réaliste d'avoir accès à toute la fonction mais de ne pas en connaitre le minimum. Ainsi, l'appel de l'oracle étant la seule manière d'obtenir des informations sur le problème, la complexité d'un algorithme d'optimisation se comptera en nombre d'appel de l'oracle. Néanmoins, il faut noter qu'il peut exister plusieurs oracles pour un même problème et par conséquent la complexité de chaque oracle est donc à prendre en compte. Pour illustrer le fait que la complexité de l'oracle est essentielle considérons le cas suivant : on cherche à minimiser une fonction $f \in \mathcal{C}^2(\mathbb{R}^d, \mathbb{R})$, pour cela, on peut utiliser une simple descente de gradient. Soit $\eta > 0$ un paramètre fixé, le schéma de descente de gradient est le suivant,

$$\begin{cases} \boldsymbol{x}^{(0)} \in \mathbb{R}^d, \\ \boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - \eta \nabla f(\boldsymbol{x}^{(t)}). \end{cases} \tag{7}$$

Ici, l'oracle est un oracle d'ordre 1, sa complexité est celle du calcul d'un gradient : $O(d)$. On peut montrer que avec des hypothèses assez raisonnables cette méthode converge vers un point stationnaire avec une précision de ϵ en $O(1/\epsilon)$ appels de l'oracle (complexité sous linéaire). De même l'algorithme de Newton converge vers un point stationnaire,

$$\begin{cases} \boldsymbol{x}^{(0)} \in \mathbb{R}^d, \\ \boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} - (\nabla^2 f(\boldsymbol{x}^{(t)}))^{-1} \nabla f(\boldsymbol{x}^{(t)}). \end{cases} \tag{8}$$

Ici, l'oracle est d'ordre 2, sa complexité est un peu subtile car il faut considérer que le calcul de la résolution du système en y, $(\nabla^2 f(\boldsymbol{x}))\boldsymbol{y} = \nabla f(\boldsymbol{x})$ est contenue dans l'oracle et que ce dernier renvoie au final une solution y de ce système. La complexité de l'oracle est donc de $O(d^3)$ ce qui rend cet algorithme difficilement utilisable en pratique bien qu'il ne nécessite que $o(1/\log(\epsilon))$ appels de l'oracle (complexité quadratique).

3.2 Boite noire locale

Dans la suite on supposera donc que l'on dispose d'un seul oracle $\mathcal{O}_f$ pour résoudre notre problème. Un cadre naturel, est celui de l'optimisation à l'aide d'une *boîte noire locale*.

Définition 8 (Boite noire locale [Nesterov, 2004])**.** *Une boîte noire locale suit deux principes :*

1. *La seule information disponible pour le schéma d'optimisation est la réponse de l'oracle.*
2. *L'oracle est supposé* local *au sens où une petite variation du problème assez loin du point x ne changera pas la réponse de l'oracle à ce point.*

Ce cadre semble représenter la majorité des cas en pratique puisque qu'un oracle rendant compte des propriété globales du problème semble beaucoup trop couteux en terme de temps de calculs. Le point de vue abordé ici sera d'essayer de tirer parti d'un oracle local et (relativement) peu couteux dans le but de résoudre avec une précision arbitraire le problème associé en un minimum de temps.

4 Optimisation convexe sous contraintes

Le cadre de l'optimisation sous contraintes est le suivant : soit $\mathcal{X}$ un convexe de $\mathbb{R}^d$; on cherche à résoudre le problème,

$$\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) \quad \text{avec } f \text{ convexe et uniquement un accès à un oracle } \mathcal{O}_f(\cdot). \tag{9}$$

Ici, il faut tout d'abord noter que si le minimum appartient à l'intérieur de $\mathcal{X}$ alors les contraintes sont inutiles et $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) = \min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x})$, ce qui n'est pas inintéressant mais sort de notre cadre puisque nous avons décidé de nous focaliser sur l'optimisation sous contraintes. Le cas intéressant est donc celui où la solution appartient au bord du domaine $\mathcal{X}$. De plus nous supposerons aussi que $\mathcal{X}$ est un compact de $\mathbb{R}^d$.

4.1 Projection de descente de gradient

Pour résoudre des problèmes d'optimisation convexe sous contraintes, l'approche standard dans la littérature est l'algorithme de descente de gradient projeté. Le principe est simple : étendre l'algorithme existant déjà en optimisation non contrainte (dont les propriétés principales sont bien connues) à l'optimisation sous contraintes. Pour cela, l'idée est la suivante : puisque suivre la direction du gradient peut faire sortir l'itérée $\boldsymbol{x}^{(t)}$ du domaine $\mathcal{X}$, une projection sur $\mathcal{X}$ est effectuée entre chaque itération. Le problème de cette méthode est le suivant : l'oracle utilisé ici comprend une étape de projection sur un convexe. Cette projection peut être très coûteuse voir impossible à réaliser en temps raisonnable sur certains convexes. Cette idée sera développée plus en détail dans la prochaine partie en comparant les oracles utilisant une projection et celui de Frank-Wolfe.

Algorithme 1 Descente de gradient projetée

1: Soit $\boldsymbol{x}^{(0)} \in \mathcal{X}$
2: **for** $t = 0 \dots T$ **do**
3: $\quad \boldsymbol{r}^{(t)} := \nabla f(\boldsymbol{x})$
4: $\quad \boldsymbol{x}^{(t+1)} := P_{\mathcal{X}}(\boldsymbol{x}^{(t)} - \eta\, \boldsymbol{r}_t)$
5: **end for**

4.2 Oracle de Frank-Wolfe

L'oracle utilisé dans l'algorithme de Frank-Wolfe est lui même un problème d'optimisation. C'est l'un des problèmes d'optimisation sous contrainte non triviales les plus simple à résoudre. C'est ce qui fait son intérêt puisque pour certains ensembles $\mathcal{X}$ très complexes c'est le seul problème que l'on sait résoudre en temps raisonnable. Cet aspect sera développé plus en détail dans la suite. L'oracle est donc défini par

$$\mathcal{O}_f(\boldsymbol{x}) := \boldsymbol{s}(\boldsymbol{x}) \in \arg\min_{\boldsymbol{s}\in\mathcal{X}} \langle \nabla f(\boldsymbol{x}), \boldsymbol{s} \rangle \tag{10}$$

La valeur retournée par l'oracle s'interprète comme suit : L'oracle calcule le gradient de la fonction au point $\boldsymbol{x}$ ce qui en fait un oracle d'ordre 1. Ensuite grâce à ce gradient, il peut calculer une linéarisation de la fonction au point $\boldsymbol{x}$. L'oracle cherche donc ensuite à minimiser cette approximation de la fonction f au point $\boldsymbol{x}$. Par exemple si $\mathcal{X}$ est la boule unité pour la norme 2, $\mathcal{O}_f(\boldsymbol{x}) = -\nabla f(\boldsymbol{x})$ et l'étape de minimisation de l'oracle n'apporte ici pas plus que le gradient qu'il avait calculé auparavant. De manière plus intéressante, si $\mathcal{X}$ est la boule unité pour la norme infinie, $(\mathcal{O}_f(\boldsymbol{x}))_i = -\operatorname{sign}((\nabla f(\boldsymbol{x}))_i)$. Enfin si $\mathcal{X}$ est un d-simplexe, alors l'oracle renvoie un vecteur nul sauf en une des coordonnées maximales du gradient. Plus précisément, $\mathcal{O}_f(\boldsymbol{x})_i = \delta_{i,i^*}$, où $(\nabla f(\boldsymbol{x}))_{i^*} = \max_i (\nabla f(\boldsymbol{x}))_i$.

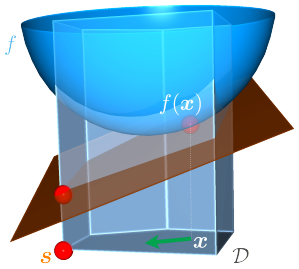

FIGURE 2 – Une étape de l'algorithme de Frank-Wolfe : la valeur retournée par l'oracle minimise la linéarisation de la fonction au point $\boldsymbol{x}$. *source : wikipedia*

L'oracle de Frank-Wolfe est particulièrement efficace pour l'optimisation de relaxations convexes de problèmes combinatoires [Joulin et al., 2014, Chari et al., 2015], i.e., lorsque l'on cherche $\min_{\boldsymbol{x}\in\mathcal{A}} f(\boldsymbol{x})$ avec $\mathcal{A}$ un ensemble fini de points de $\mathbb{R}^d$. Lorsque le nombre de points de $\mathcal{A}$ est trop grand (par exemple lorsqu'il croît exponentiellement avec d et que l'on se place en grande dimension), une solution pour résoudre ce problème est de se ramener au calcul du minimum de f dans l'enveloppe convexe de $\mathcal{A}$

$$\min_{\boldsymbol{x}\in\operatorname{conv}(\mathcal{A})} f(\boldsymbol{x}). \tag{11}$$

Cette méthode peut être ensuite résolue avec n'importe quel algorithme d'optimisation convexe. Néanmoins, il est souvent difficile de projeter sur un polytope avec de nombreuses faces (ce qui est souvent le cas lorsque le cardinal de $\mathcal{A}$ est grand) ce qui rend les méthodes utilisant des projections difficiles à utiliser. De manière opposée, la minimisation d'une fonction linéaire sur ce polytope

est souvent beaucoup moins couteuse. Plus concrètement, de nombreux exemples d'oracles et leur complexité respective sont donnés par Jaggi [2013].

4.3 Algorithme de Frank-Wolfe

Nous allons maintenant nous intéresser à la description rigoureuse de l'algorithme et à ses propriétés. Plaçons nous après t étapes, l'itérée actuelle est le point $\boldsymbol{x}^{(t)}$, une fois que l'oracle a renvoyé le point $\boldsymbol{s}^{(t)}$ qui minimise la linéarisation de la fonction f au point $\boldsymbol{x}^{(t)}$, nous possédons une estimation de la direction $\boldsymbol{d}^{(t)} := \boldsymbol{s}^{(t)} - \boldsymbol{x}^{(t)}$ vers laquelle se trouve le minimum de f. Nous effectuons donc l'opération,

$$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} + \gamma \boldsymbol{d}^{(t)}, \qquad (12)$$

qui n'a pas besoin d'être suivie d'un projection puisqu'en injectant l'expression de $\boldsymbol{d}^{(t)}$ dans la définition de $\boldsymbol{x}^{(t+1)}$ on obtient que ce dernier est une combinaison convexe des points $\boldsymbol{x}^{(t)}$ et $\boldsymbol{s}^{(t)}$ appartenant tout deux au convexe $\mathcal{X}$,

$$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} + \gamma \boldsymbol{d}^{(t)} = \gamma \boldsymbol{s}^{(t)} + (1-\gamma)\, \boldsymbol{x}^{(t)} \in \mathcal{X}. \qquad (13)$$

Algorithme 2 Algorithme de Frank-Wolfe

1: Soit $\boldsymbol{x}^{(0)} \in \mathcal{X}$
2: **for** $t = 0 \dots T$ **do**
3: $\quad \boldsymbol{r}^{(t)} := \nabla f(\boldsymbol{x}^{(t)})$
4: $\quad \boldsymbol{s}^{(t)} := \arg\min_{\boldsymbol{s} \in \mathcal{X}} \left\langle \boldsymbol{x}, \boldsymbol{r}^{(t)} \right\rangle$
5: $\quad g_t := \left\langle \boldsymbol{x}^{(t)} - \boldsymbol{s}^{(t)}, \boldsymbol{r}^{(t)} \right\rangle$
6: $\quad$ **if** $g_t \leq \epsilon$ **then return** $\boldsymbol{x}^{(t)}$
7: $\quad \gamma = \frac{2}{2+t}$ **ou** $\gamma = \arg\min_{\gamma' \in [0,1]} f((1-\gamma)\boldsymbol{x}^{(t)} + \gamma \boldsymbol{s}^{(t)})$
8: $\quad \boldsymbol{x}^{(t+1)} := (1-\gamma)\boldsymbol{x}^{(t)} + \gamma \boldsymbol{s}^{(t)}$
9: **end for**

Cet algorithme possède de nombreux avantages (en dehors de celui de ne nécessiter qu'un oracle de minimisation linéaire) Tout d'abord il est simple à implémenter, la seule difficulté étant l'implémentation de l'oracle, qui dépend du problème.

Un autre avantage de cet algorithme est qu'il est invariant par transformation affine [Jaggi, 2013], c'est à dire que toute transformation affine du problème $\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$ par la fonction affine surjective $A : \mathcal{X}' \to \mathcal{X}$ en $\min_{\hat{\boldsymbol{x}} \in \mathcal{X}'} f(M\hat{\boldsymbol{x}})$ ne change pas le résultat de l'algorithme. Plus précisément si l'on note $\left(\boldsymbol{x}^{(t)}\right)_{t \in \mathbb{N}}$ la suite d'itérées obtenue en résolvant le premier problème avec comme condition initiale $\boldsymbol{x}^{(0)}$ et $\left(\hat{\boldsymbol{x}}^{(t)}\right)_{t \in \mathbb{N}}$ la suite d'itérées obtenues en résolvant le second problème comme condition initiale $\hat{\boldsymbol{x}}^{(0)} := M\boldsymbol{x}^{(0)}$ alors pour tout $t \in N,\ \boldsymbol{x}^{(t)} = M\hat{\boldsymbol{x}}^{(t)}$. Cette propriété n'est pas vérifiée par la descente de gradient.

Une troisième propriété utile est la possibilité d'avoir un critère d'arrêt de l'algorithme pour un coût calculatoire négligeable. En effet le *gap* de Frank-Wolfe est la quantité g_t définie Ligne 4 de l'Algorithme 2.

$$g(\boldsymbol{x}) := \max_{\boldsymbol{s} \in \mathcal{X}} \langle \boldsymbol{x} - \boldsymbol{s}, \nabla f(\boldsymbol{x}) \rangle = \langle \boldsymbol{x} - \boldsymbol{s}(\boldsymbol{x}), \nabla f(\boldsymbol{x}) \rangle\,. \qquad (14)$$

Par un simple argument de convexité (Illustré Figure 2) ce *gap* majore la sous-optimalité au point $\boldsymbol{x}$,

$$g(\boldsymbol{x}) := \max_{\boldsymbol{s} \in \mathcal{X}} \langle \boldsymbol{x} - \boldsymbol{s}, \nabla f(\boldsymbol{x}) \rangle \geq \langle \boldsymbol{x} - \boldsymbol{x}^*, \nabla f(\boldsymbol{x}) \rangle \geq f(\boldsymbol{x}) - f(\boldsymbol{x}^*). \qquad (15)$$

Le coût calculatoire de cette quantité est faible puisque le gradient et le résultat de l'oracle $\boldsymbol{s}^{(t)}$ ont déjà été calculés pour obtenir $\boldsymbol{x}^{(t+1)}$.

Enfin une dernière propriété de cet algorithme est qu'il permet d'avoir une représentation *parcimonieuse* des ses itérées. En effet après t itérations, le point $\boldsymbol{x}^{(t)}$ est une combinaison convexe des t points retournés par l'oracle. En grande dimension cela permet d'éviter d'avoir une représentation trop coûteuse en mémoire. De plus, dans le cas de l'optimisation sous contraintes à un polytope de grande dimension, l'intuition est que la solution appartient à une face de dimension faible et a donc elle aussi une représentation parcimonieuse. Ce type de représentation est impossible si l'on utilise des projections.

Théorème 9 ([Jaggi, 2013])**.** *Soit f une fonction convexe dont le gradient est L-Lipschitz et $\mathcal{X}$ un compact convexe de diamètre $D_{\mathcal{X}}$. Pour tout $t \geq 1$, les itérées de l'Algorithme 2 appliquées à la minimisation de f dans $\mathcal{X}$ vérifient,*

$$f(\boldsymbol{x}^{(t)}) - f^* \leq \frac{2LD_{\mathcal{X}}^2}{t+2}, \qquad (16)$$

où $f^* \in \mathcal{X}$ est le minimum de f sur $\mathcal{X}$.

Il faut noter ici que bien que L et $D_{\mathcal{X}}$ soient des constantes sensibles aux transformations affines on peut définir de nouvelles constantes rendant compte des mêmes propriétés de f et $\mathcal{X}$ mais invariantes par transformation affine. En effet si l'on définit la *courbure* C_f de f par

$$C_f := \sup_{\substack{\boldsymbol{x},\boldsymbol{s}\in\mathcal{X} \\ \gamma\in[0,1] \\ \boldsymbol{y}=\gamma\boldsymbol{s}+(1-\gamma)\boldsymbol{x}}} \frac{2}{\gamma^2}\left(f(\boldsymbol{y}) - f(\boldsymbol{x}) - \gamma\left\langle \boldsymbol{x}-\boldsymbol{s}, \nabla f(\boldsymbol{x})\right\rangle\right), \tag{17}$$

on remarque que cette quantité est invariante par transformation affine et vérifie $C_f \leq LD_{\mathcal{X}}$. De plus il a été démontré [Jaggi, 2013] que l'on peut remplacer $LD_{\mathcal{X}}$ par C_f dans le Théorème 9.

5 Optimisation de point selle sous contraintes

La recherche de minimum ou de maximum n'est pas le seul type de problème d'optimisation que l'on peut rencontrer. En effet, certain problèmes peuvent se formuler comme la combinaison d'une minimisation et d'une maximisation. Plus précisément, soient $\mathcal{X}$ et $\mathcal{Y}$ deux ensembles convexes compacts et $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, on cherche à calculer

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{y}\in\mathcal{Y}} \mathcal{L}(\boldsymbol{x},\boldsymbol{y}) \qquad \text{à l'aide d'un oracle } \mathcal{O}_{\mathcal{L}}(\cdot), \tag{18}$$

où $\mathcal{L}$ est une fonction *convexe-concave*, i.e., $\forall \boldsymbol{y} \in \mathcal{Y}, \mathcal{L}(\cdot,\boldsymbol{y})$ est convexe et $\forall \boldsymbol{x} \in \mathcal{X}, \mathcal{L}(\boldsymbol{x},\cdot)$ est concave. Tout d'abord il faut bien comprendre que ce problème peut quand même être formulé comme un problème d'optimisation convexe puisque

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{y}\in\mathcal{Y}} \mathcal{L}(\boldsymbol{x},\boldsymbol{y}) = \min_{\boldsymbol{x}\in\mathcal{X}} p(\boldsymbol{x}) = \max_{\boldsymbol{y}\in\mathcal{Y}} q(\boldsymbol{y}), \tag{19}$$

où $p(\boldsymbol{x}) := \max_{\boldsymbol{y}\in\mathcal{Y}} \mathcal{L}(\boldsymbol{x},\boldsymbol{y})$ est convexe (car un supremum de fonctions convexes est convexe) et $q(\boldsymbol{y}) := \min_{\boldsymbol{y}\in\mathcal{Y}}$ est concave. Néanmoins, la subtilité ici porte sur le fait que les fonction p et q sont en général trop difficiles d'accès au sens où un oracle sur ces fonctions serait trop coûteux en terme de complexité temporelle.

5.1 Définition et exemples de points selles

Nous donnons ici la définition la plus générale de *point selle* donnée par Hiriart-Urruty and Lemaréchal [2013, VII.4]

Définition 10 (Point selle)**.** *Soit $\mathcal{X}$ et $\mathcal{Y}$ deux ensembles non vides et $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ une fonction. Un couple $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ est appelé* point selle *si*

$$\mathcal{L}(\boldsymbol{x}^*,\boldsymbol{y}) \leq \mathcal{L}(\boldsymbol{x}^*,\boldsymbol{y}^*) \leq \mathcal{L}(\boldsymbol{x},\boldsymbol{y}^*) \quad \forall \boldsymbol{x} \in \mathcal{X}, \forall \boldsymbol{y} \in \mathcal{Y}. \tag{20}$$

Notre cadre est plus restreint puisque nous prendrons les mêmes hypothèses que celles utilisées pour décrire le problème Équation 18. Dans ce cadre là on peut montrer que la définition de *point selle* (Définition 10) coïncide avec les points optimaux de l'Équation 18.

Les exemples de problèmes de recherche de point selle sont nombreux dans la littérature. On en trouve par exemple en théorie des jeux :

Jeu à somme nulle. Considérons un jeu à deux joueurs où le but est de maximiser son gain. C'est un jeu à somme nulle au sens où à chaque tour la somme gagnée par le premier joueur est égale à celle perdue par le second. À chaque tour les règles sont les suivantes :

- Le joueur 1 choisit une action $i \in \{1,\dots,I\}$.
- Le joueur 2 choisit une action $j \in \{1,\dots,J\}$.
- Le gain du joueur 1 suite à ces deux choix est $a_{ij} \in \mathbb{R}$.
- Chaque joueur joue selon une distribution de probabilité (respectivement $\boldsymbol{x}$ et $\boldsymbol{y}$ pour les joueurs 1 et 2).
- Ainsi l'espérance du profit du joueur 1 est

$$\mathbb{E}\left[a_{ij}\right] = \boldsymbol{x}^\top A \boldsymbol{y}. \tag{21}$$

- Chaque joueur cherche donc la distribution minimisant le profit de son adversaire (ce qui revient à maximiser le sien). Ainsi le problème pour le joueur 2 est

$$\min_{\boldsymbol{x}\in\Delta_I}\max_{\boldsymbol{y}\in\Delta_J} \boldsymbol{x}^\top A\boldsymbol{y}. \tag{22}$$

où les Δ sont les simplexes de probabilité, $\Delta_I := \{\boldsymbol{x}\in\mathbb{R}^I \mid \sum_{i=1}^I x_i = 1,\ \boldsymbol{x}\geq 0\}$.

L'équilibre de ce jeu est atteint lorsque chaque joueur joue la meilleur stratégie possible sachant que l'adversaire essaye lui aussi de maximiser son profit. Ce point d'équilibre est un point selle de la fonction définie Équation (22).

Apprentissage robuste. La notion de robustesse en apprentissage est elle aussi intimement liée à celle de point selle. Par exemple, la littérature considère des modèles de prédiction linéaire où l'on minimise une moyenne de perte empirique à laquelle on ajoute une régularisation,

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{k=1}^n \ell(f(x_k), y_k) + \lambda\Omega(\theta), \tag{23}$$

où la fonction de perte ℓ est convexe et $\mathcal{F}$ est une classe de fonctions. Maintenant on peut considérer qu'il existe un incertitude concernant les données fournies : en effet, on peut penser que les données ont été altérées ou que quelqu'un en a modifié légèrement la nature pour fausser nos prédictions. Ainsi de meilleures prédictions seraient faites sur les données de test en résolvant le problème,

$$\min_{f\in\mathcal{F}} \max_{\substack{\tilde{x}_i\in B_\delta(x_i)\\ i\in\{1,\dots,n\}}} \frac{1}{n}\sum_{k=1}^n \ell(f(x_k), y_k) + \lambda\Omega(\theta) \quad \text{où} \quad B_\delta(x) := \{\boldsymbol{y} \mid \|y-x\|\leq\delta\}. \tag{24}$$

Ce problème est un problème de point selle.

5.2 Algorithme de Frank Wolfe appliqué aux points selles

L'algorithme généralement cité dans la littérature pour rechercher des points selles est un extension de l'algorithme de descente de gradient projetée. L'idée est d'effectuer simultanément une descente de gradient dans $\mathcal{X}$ et une montée de gradient dans $\mathcal{Y}$. Les itérées sont actualisées de la manière suivante

$$\boldsymbol{x}^{(t+1)} = P_{\mathcal{X}}(\boldsymbol{x}^{(t)} - \eta\nabla_x\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}))$$
$$\boldsymbol{y}^{(t+1)} = P_{\mathcal{Y}}(\boldsymbol{y}^{(t)} + \eta\nabla_y\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})).$$

La même idée va être utilisée pour la généralisation de l'algorithme de Frank-Wolfe. Le vecteur $\boldsymbol{r}^{(t)}$ qui était égal au gradient est remplacé par la quantité ligne 3 de l'Algorithme 3. Ligne 7 les constantes τ et $C_{\mathcal{L}}$ sont des constantes invariantes par transformation affine dépendant de la fonction $\mathcal{L}$ et des ensembles $\mathcal{X}$ et $\mathcal{Y}$. Nous avons démontré le théorème suivant. La définition de certaines quantités n'y est volontairement pas rigoureuse afin d'éviter un trop gros formalisme. Pour une construction complète des constantes invariantes par transformation affine et un énoncé sans ambigüité du théorème on pourra consulter [Gidel et al., 2016].

Algorithme 3 FW pour points selles (SP-FW)

1: Let $\boldsymbol{z}^{(0)} = (\boldsymbol{x}^{(0)}, \boldsymbol{y}^{(0)}) \in \mathcal{X}\times\mathcal{Y}$
2: **for** $t = 0\dots T$ **do**
3: $\quad \boldsymbol{r}^{(t)} := \begin{pmatrix}\nabla_x\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})\\ -\nabla_y\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})\end{pmatrix}$
4: $\quad \boldsymbol{s}^{(t)} := \operatorname{argmin}_{\boldsymbol{z}\in\mathcal{X}\times\mathcal{Y}} \left\langle \boldsymbol{z}, \boldsymbol{r}^{(t)}\right\rangle$
5: $\quad g_t := \left\langle \boldsymbol{z}^{(t)} - \boldsymbol{s}^{(t)}, \boldsymbol{r}^{(t)}\right\rangle$
6: $\quad$ **if** $g_t \leq \epsilon$ **then return** $\boldsymbol{z}^{(t)}$
7: $\quad \gamma = \min\left(1, \frac{\tau}{2C_{\mathcal{L}}}g_t\right)$ **or** $\gamma = \frac{2}{2+t}$
8: $\quad \boldsymbol{z}^{(t+1)} := (1-\gamma)\boldsymbol{z}^{(t)} + \gamma\boldsymbol{s}^{(t)}$
9: **end for**

Théorème 11. *Soit $\mathcal{L}$ une fonction convexe-concave et $\mathcal{X}\times\mathcal{Y}$ un compact convexe.. Supposons que le gradient de $\mathcal{L}$ est Lipschitz, que $\boldsymbol{x}\mapsto\mathcal{L}(\boldsymbol{x},\boldsymbol{y}) - \mu\|\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{y}\|_2^2$ est convexe-concave et que nous sommes dans l'un des deux cas suivant :*

Le point selle appartient à l'intérieur de $\mathcal{X}\times\mathcal{Y}$. (I)

Les ensembles $\mathcal{X}$ et $\mathcal{Y}$ sont des polytopes. (P)

Dans ces deux cas, si la constante μ est assez grande devant les dérivées croisées de $\mathcal{L}$, alors une extension de l'algorithme de Frank-Wolfe avec $\gamma_t = \min\{\gamma_{\max}, \frac{\tau}{2C}g_t\}$ ($\gamma_{\max}$ est le pas maximum pour ne pas sortir de l'ensemble convexe. Il peut être plus petit que 1) converge géométriquement et

$$\min_{s\leq t} g_s = O\left((1-\rho)^{t/2}\right) \text{ pour (I)} \quad \text{et} \quad \min_{s\leq t} = O\left((1-\rho)^{t/3}\right) \text{ pour (P)} \tag{25}$$

où $\rho := \frac{\tau\mu}{2C}$ *et* $C := L\max\{D_{\mathcal{X}}, D_{\mathcal{Y}}\}$ *($D_{\mathcal{X}}$ est le diamètre de l'ensemble $\mathcal{X}$). La constante* $\tau \in [-\infty, 1]$ *représente à quel point* μ *est suffisamment grand devant la norme des dérivées croisées de* $\nabla_y \nabla_x \mathcal{L}$ *et* $\nabla_y \nabla_x \mathcal{L}$*. Notre preuve de convergence se restreint au cas où* $\tau \in]0, 1]$.

5.3 Problèmes ouverts

La conjecture de Hammond. Dans sa thèse Hammond [1984] a conjecturé la convergence de l'algorithme de Frank-Wolfe pour les points selles. Nous répondons partiellement à cette conjecture dans notre Théorème 11. Cette conjecture reste néanmoins ouverte lorsque la condition $\tau \leq 0$. Cette hypothèse est assez forte (bien que raisonnable) et ne semble pas être nécessaire. Nous avons effectué un grand nombre d'expériences que tendent à confirmer le fait que l'algorithme devrait converger pour tout $\tau \in]-\infty, 1]$.

La conjecture de Karlin. La conjecture de Karlin [1960] est maintenant une conjecture relativement ancienne provenant de la théorie des jeux. Elle affirme que le *fictitious play algorithm* converge en $O(1/\sqrt{t})$. Cet algorithme sert à la recherche d'équilibre de jeux à deux joueurs comme celui présenté Équation (22). Nous avons démontré des liens étroits entre le *fictitious play algorithm* [Brown, 1951] et l'algorithme de Frank-Wolfe. En effet, lorsque le problème à résoudre est celui décrit Équation (22), l'Algorithme 3 avec $\gamma = \frac{1}{1+t}$ et le *fictitious play algorithm* sont strictement équivalents. Ainsi prouver la convergence de l'algorithme de Frank-Wolfe dans ce cadre résoudrait la conjecture.

6 Conclusion

Comme nous l'avons vu, les problèmes de point selle permettent de modéliser des problématiques nouvelles généralisant souvent des problèmes d'optimisation convexe et sont d'un grand intérêt dans de nombreux domaines nécessitant de l'optimisation. Nous avons étendu un algorithme dont l'intérêt n'est plus à prouver tant il a été utilisé ces dernières années dans la communauté de l'apprentissage statistique. L'extension que nous proposons possède les mêmes propriétés avantageuses que son homologue convexe. Nous avons réussi à prouver la convergence de ce nouvel algorithme dans certain cas particuliers résolvant partiellement une conjecture vieille de 30 ans. Nous avons aussi effectué un profond travail expérimental afin de vérifier nos affirmations théoriques et vérifier les conjectures encore en suspens.

Références

D. P. Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

G. Brown. Iterative solution of games by fictitious play. *Activity analysis of production & allocation*, 1951.

S. Bubeck. Convex optimization : Algorithms and complexity. *arXiv preprint arXiv :1405.4980*, 2014.

V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic. On pairwise costs for network flow multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5537–5545, 2015.

G. B. Dantzig. *Linear programming and extensions*. Princeton university press, 1963.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Nav. Res. Logist. Q.*, 1956.

G. Gidel, T. Jebara, and S. Lacoste-Julien. Frank-Wolfe Algorithms for Saddle Point Problems. In *arXiv*, 2016.

J. H. Hammond. *Solving asymmetric variational inequality problems and systems of equations with generalized nonlinear programming algorithms*. PhD thesis, Massachusetts Institute of Technology, 1984.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms I : Fundamentals*. Springer, 2013.

M. Jaggi. Revisiting Frank-Wolfe : Projection-free sparse convex optimization. In *ICML*, 2013.

A. Joulin, K. Tang, and L. Fei-Fei. Efficient image and video co-localization with frank-wolfe algorithm. In *European Conference on Computer Vision*, pages 253–268. Springer, 2014.

L. V. Kantorovich. On the translocation of masses. In *Dokl. Akad. Nauk SSSR*, volume 37, pages 199–201, 1942.

S. Karlin. Mathematical methods and theory in games, programming and economics, 1960.

A. Lucas. Ising formulations of many np problems. *arXiv preprint arXiv :1302.5843*, 2013.

D. G. Luenberger. *Introduction to linear and nonlinear programming*, volume 28. Addison-Wesley Reading, MA, 1973.

Y. Nesterov. *Introductory lectures on convex optimization : A basic course*, volume 87. Springer, 2004.

H. Nishimori. *Statistical physics of spin glasses and information processing : an introduction*, volume 111. Clarendon Press, 2001.