Outils d'optimisation pour les sciences des données et de la décision

6 octobre 2023

Programme

Cours : Méthodes proximales et régularisation

TD : Exercices

MÉTHODES PROXIMALES ET RÉGULARISATION

(1) Optimisation non lisse

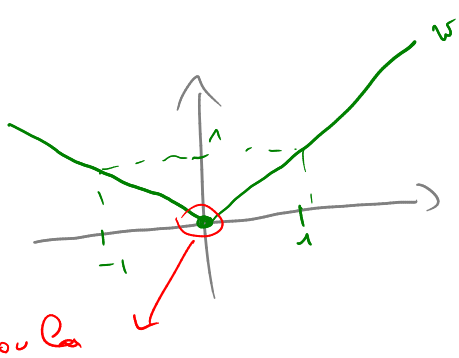
↳ Dans les deux cours précédents, on cherchait à minimiser des fonctions de classe C^1, pour lesquelles il existe un gradient en tout point

⇒ On peut appliquer la descente de gradient dans ce cas

⇒ Que se passe-t-il si f n'est pas C^1 ?

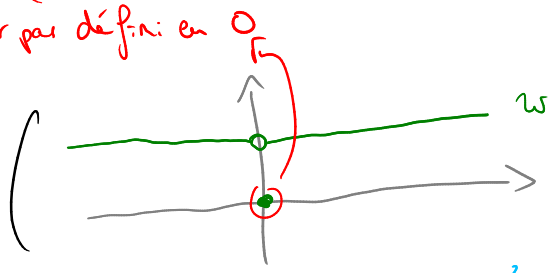
Def : On dit qu'une fonction $f : \mathbb{R}^d \to \mathbb{R}$ est non lisse si il existe $w \in \mathbb{R}^d$ en lequel le gradient de f n'est pas défini mathématiquement

Exemples en dimension 1

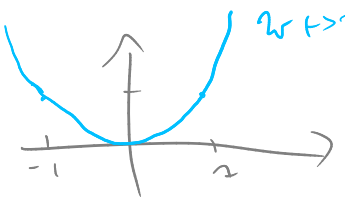


non lisse mais continue et convexe

Le gradient (ou la dérivée) n'est pas défini en 0



non lisse discontinue non convexe



lisse continue convexe

→ On se concentre sur les fonctions non lisses convexes, ce qui garantit en particulier qu'elles sont continues et que tout minimum local est global.

Déf. (Sous-gradient)

Soit $f : \mathbb{R}^d \to \mathbb{R}$ une fonction convexe (non lisse) et $w \in \mathbb{R}^d$

Un vecteur $g \in \mathbb{R}^d$ est un sous-gradient de f en w si il vérifie

$$\forall v \in \mathbb{R}^d, \quad f(v) \geq f(w) + g^T(v-w)$$

L'ensemble des sous-gradients de f en w s'appelle le sous-différentiel de f en w, et on le note $\partial f(w) \subseteq \mathbb{R}^d$

→ Remplace le concept de gradient par un ensemble de vecteurs $\partial f(w)$

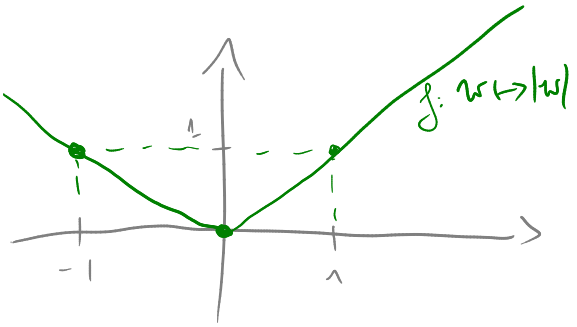
→ Lorsque f est C^1, $\forall (w,v) \in (\mathbb{R}^d)^2$, $f(v) \geq f(w) + \nabla f(w)^T(v-w)$

→ Si $\nabla f(w)$ existe en un point $w \in \mathbb{R}^d$, alors

$$\partial f(w) = \{\nabla f(w)\}$$

⇒ La notion de sous-différentiel généralise le concept de gradient

(Ex)



$$\partial f(w) = \begin{cases} \{1\} & \text{si } w > 0 \\ \{-1\} & \text{si } w < 0 \\ [-1,1] & \text{si } w = 0 \end{cases}$$

} points en lesquels le gradient existe

↑ point en lequel le gradient n'est pas défini

Th Soit $f : \mathbb{R}^d \to \mathbb{R}$ convexe.

Alors $\quad [\bar{w} \in \mathbb{R}^d \text{ minimum global de } f] \iff [0_{\mathbb{R}^d} \in \partial f(\bar{w})]$

→ Généralisation du cas C^1 $\quad [\bar{w} \text{ minimum global}] \iff [\nabla f(\bar{w}) = 0_{\mathbb{R}^d}]$

(Ex) Régression linéaire robuste $\quad \{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d \quad y_i \in \mathbb{R}$

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \; \|Xw - y\|_1 = \sum_{i=1}^n |x_i^T w - y_i|$$

Rappel : Algorithme de descente de gradient (pour f C^1)

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k) \quad \text{avec } \alpha_k > 0$$

Algorithme de sous-gradient (pour f convexe)

$$w_{k+1} = w_k - \alpha_k g_k \quad \text{avec } \alpha_k > 0 \text{ et } g_k \in \partial f(w_k)$$

⟶ Si f est C^1, l'algorithme de sous-gradient est équivalent à la descente de gradient

Remarques :

→ L'analyse et l'implémentation de l'algorithme du sous-gradient peuvent être plus difficiles que celles de la descente de gradient

→ Mais cet algorithme peut converger avec très peu d'hypothèses sur le problème sous-jacent

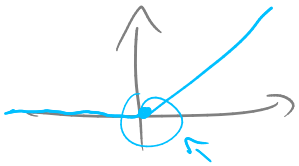
→ En sciences des données, soit les fonctions sont suffisamment simples pour que le sous-différentiel soit calculable explicitement, soit on dispose d'outils automatisés pour calculer les sous-gradients (différentiation automatique ≡ backpropagation)

$$w \mapsto f(w) \underbrace{\quad}_{\text{code}} \xRightarrow[\text{D.A.}]{} g \in \partial f(w)$$

↳ L'optimisation de fonctions non lisses apparaît sous deux formes en sciences des données et au-delà :

- Soit la fonction à optimiser est non lisse ($\|Xw-y\|_1$, entraînement des réseaux de neurones avec activations ReLU, ...)

$$t \mapsto \mathrm{ReLU}(t) = \max(t, 0)$$



$$x^{(0)} \mapsto x^{(1)} \mapsto \dots \mapsto x^{(L)}$$

$$x^{(l+1)} = \mathrm{ReLU}\left(W^{(l)} x^{(l)} + b^{(l)}\right)$$

$\mathrm{ReLU} : \mathbb{R}^d \to \mathbb{R}^d$

$W^{(l)}, b^{(l)}$: paramètres d'optimisation

• Soit la fonction peut s'exprimer comme somme d'un terme lisse et d'un terme non lisse

Ex) LASSO

$$\frac{1}{2}\|Xw - y\|^2 + \lambda \|w\|_1$$

$\lambda >> 0$

"norme ℓ_1" $\|w\|_1 = \sum_{j=1}^{d} |w_j|$

$\frac{1}{2}\|Xw - y\|^2$: somme carré linéaire (C^1)

$\lambda \|w\|_1$: non lisse (pas de gradient en tout vecteur ayant une coordonnée nulle)

$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}$ ↕ échantillons, ↔ d caractéristiques

But : Fournir un modèle linéaire des données qui soit le plus parcimonieux possible, càd (sparse)

w ayant le plus de zéros possible

↳ Pour traiter le premier type de problèmes, on applique un algorithme de sous-gradient

↳ Pour le second, on utilise des méthodes proximales (ou de gradient proximales)

(2) Méthodes proximales

Contexte : minimiser $_{w \in \mathbb{R}^d}$ $f(w) + \lambda \Omega(w)$

- $f(w)$: terme d'attache aux données (C^1)
- $\lambda > 0$: paramètre de régularisation qui contrôle le poids de la régularisation par rapport à l'attache aux données
- $\Omega(w)$: terme de régularisation qui pénalise les vecteurs qui ne satisfont pas une propriété souhaitée

↳ Problème régularisé : on cherche un compromis entre l'attache aux données et une structure souhaitée sur le modèle (w)

En général, Ω est non lisse mais convexe

En somme des données, on veut traiter f et Ω de manière distincte car f dépend des données et pas Ω.

Algorithme : Descente de gradient proximale

DG $w_{k+1} = w_k - \alpha_k \nabla f(w_k)$

$$w_{k+1} \in \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \left\{ f(w_k) + \nabla f(w_k)^T (w - w_k) + \frac{1}{2\alpha_k} \|w - w_k\|^2 + \lambda \Omega(w) \right\}$$

- $\underset{w \in \mathbb{R}^d}{\operatorname{argmin}}$: solution d'un problème de minimisation
- $f(w_k) + \nabla f(w_k)^T (w - w_k) \approx f(w)$
- $\frac{1}{2\alpha_k} \|w - w_k\|^2$: terme proximal : permet de garantir que le problème a une solution ($\alpha_k > 0$)
- $\lambda \Omega(w)$: terme laissé tel quel

→ Si $\Omega(w) = 0 \quad \forall w \in \mathbb{R}^d$, alors l'itération ci-dessus devient

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k)$$: Descente de gradient

→ Pour de nombreux choix classiques de Ω, le problème ci-dessus (sous-problème proximal) peut être résolu de manière explicite et donc plus efficacement que le problème d'origine

Deux exemples importants de régularisation

a) Régularisation ℓ_2 / ridge, "tikhonov" / Tychonov / ...

$$\underset{w \in \mathbb{R}^d}{\text{minimiser}} \quad f(w) + \frac{\lambda}{2} \|w\|^2 \qquad \|w\|^2 = \sum_{j=1}^{d} w_j^2$$

↳ Lorsque $\lambda \nearrow$, la régularisation réduit la norme des solutions en réduisant la valeur des coordonnées de manière uniforme

↳ Dans un contexte de sciences des données, cela conduit à une solution avec une plus faible variance par rapport aux données et souvent une solution qui généralise mieux à d'autres données

(NB : si f convexe, alors $f + \frac{\lambda}{2} \|\cdot\|^2$ est fortement convexe)

GP : $w_{k+1} = w_k - \frac{\alpha_k}{1+\lambda\alpha_k} \nabla f(w_k)$ (forme explicite du sur-problème)

b) Régularisation ℓ_1 / LASSO / parcimonieuse

$$\underset{w \in \mathbb{R}^d}{\text{minimiser}} \; f(w) + \lambda \|w\|_1 \qquad \|w\|_1 = \sum_{j=1}^{d} |w_j|$$

↳ Lorsque $\lambda \nearrow$, les solutions du problème ont de plus en plus de coordonnées nulles

↳ Utilisée pour créer de telles solutions, ce qui en science des données est lié à la sélection d'attributs

↳ Dans ce contexte, l'algorithme du gradient proximal est équivalent à l'algorithme ISTA

$$\forall j = 1..d, \quad [w_{k+1}]_j = \begin{cases} [w_k - \alpha_k \nabla f(w_k)]_j - \lambda\alpha_k & \text{si } [w_k - \alpha_k \nabla f(w_k)]_j > \lambda\alpha_k \\ [w_k - \alpha_k \nabla f(w_k)]_j + \lambda\alpha_k & \text{si } [w_k - \alpha_k \nabla f(w_k)]_j < -\lambda\alpha_k \\ 0 & \text{sinon} \end{cases}$$