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

21 octobre 2024

Aujourd'hui : CM

Lundi 28/10 : TD avec notebook
Point sur le projet

Vendredi 08/11 : TD (annales)

Au-delà de la descente de gradient

→ Descente de gradient : algorithme d'optimisation qui est applicable à toute fonction C^1 (qui admet un gradient en tout point), que cette fonction soit convexe ou non convexe

Itération

Pour un problème $\min_{w \in \mathbb{R}^d} f(w)$ avec $f : \mathbb{R}^d \to \mathbb{R}$ C^1,

l'itération k de la descente de gradient s'écrit

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

- $\alpha_k > 0$: taille de pas
- $\nabla f(w_k)$: Gradient de f en w_k (vecteur de $\mathbb{R}^d$)

Complexité

Etant donnée une précision $\varepsilon > 0$, quel est le nombre d'itérations de descente de gradient nécessaire pour atteindre cette précision ?

Ex) Borne de complexité dans le cas convexe

Si f C^1 convexe, alors la descente de gradient calcule w_k tel que $f(w_k) - \min_{w \in \mathbb{R}^d} f(w) \leq \varepsilon$ en au plus $O(\varepsilon^{-1})$ itérations.

- $\min_{w \in \mathbb{R}^d} f(w)$: valeur optimale du problème
- $O(\varepsilon^{-1})$: $C\varepsilon^{-1}$, où $C > 0$ est une constante qui ne dépend pas de ε

→ Indique à quel point le nombre d'itérations augmente lorsque la précision souhaitée diminue

Question :

- Existe-t-il de meilleurs algorithmes que la descente de gradient ?

  ⇒ Oui dans le cas convexe

- Que faire si f n'est pas C^1 ?

  ⇒ Si f est convexe, on peut définir des algorithmes même sans gradient

  ⇒ En sciences des données, le fait que f ne soit pas C^1 provient généralement d'une régularisation. Dans ce cas, on peut définir une famille d'algorithmes qui généralise la descente de gradient : méthodes proximales.

(1) Accélération

Cadre de travail : $\min_{w \in \mathbb{R}^d} f(w)$, $f : \mathbb{R}^d \to \mathbb{R}$ $C^{1,1}_L$

C^1 ↑

∇f L-lipschitz

$\|\nabla f(v) - \nabla f(w)\| \leq L\|v-w\|$

et f convexe.

(f convexe $\Leftrightarrow \forall (v,w) \in (\mathbb{R}^d)^2,\ f(v) \geq f(w) + \nabla f(w)^T(v-w)$)

On suppose que f admet un minimum ($\operatorname{argmin}_{w \in \mathbb{R}^d} f(w) \neq \emptyset$)

↑ ensemble des solutions de $\min_{w \in \mathbb{R}^d} f(w)$

et on considère
$$\begin{cases} w^* \in \operatorname{argmin}_{w \in \mathbb{R}^d} f(w). \\ f^* = f(w^*) = \min_{w \in \mathbb{R}^d} f(w) \end{cases}$$

→ La descente de gradient garantit $f(w_k) - f^* \leq \varepsilon$ en au plus $O(\varepsilon^{-1})$ itérations : est-ce la meilleure complexité

possible ? → NON !

~1970-1980 : résultat théorique "Il existe un algorithme pour l'optimisation convexe qui garantit $f(w_k) - f^* \leq \varepsilon$ en au plus $O(\varepsilon^{-1/2})$ itérations"

↑ mieux que $O(\varepsilon^{-1})$

Car croît moins vite quand ε diminue

← Résultat d'existence : ne décrit pas l'algorithme en question, mais certifie que cet algorithme est optimal en termes de complexité

↳ Cet algorithme a été établi en 1983

Historique

1964 : Polyak propose la méthode "Heavy ball", ou descente de gradient avec momentum

Algorithme :

- Partir de $w_0 \in \mathbb{R}^d$, poser $w_{-1} = w_0$.
- Itération k

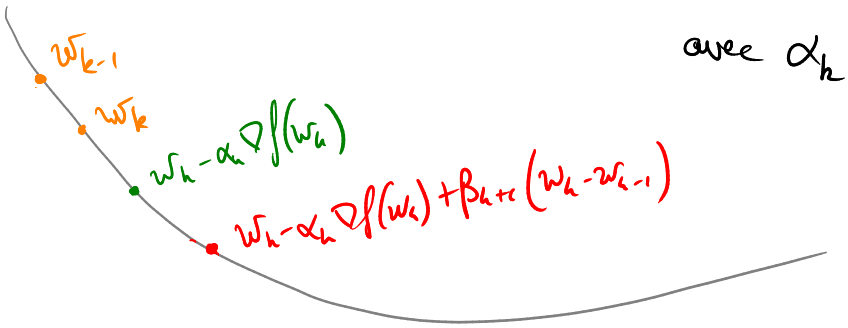
$$w_{k+1} = w_k - \alpha_k \nabla f(w_k) + \beta_{k+1}(w_k - w_{k-1})$$

Itération de la descente de gradient

Déplacement à l'itération $k-1$ "momentum"

↳ Paramètre de momentum

avec $\alpha_k > 0$ et $\beta_{k+1} > 0$



→ Polyak montre que cette méthode est optimale pour les fonctions quadratiques fortement convexes (sous-ensemble des fonctions convexes)

→ Mais la méthode peut ne pas converger sur des fonctions qui ne sont pas quadratiques (travaux de 1970-1980)

1983 : Nesterov : gradient accéléré / algorithme de Nesterov

Idée. La méthode de Polyak peut être vue comme une double récurrence (2 suites w_k, z_k)

$$z_{k+1} = w_k - \alpha_k \nabla f(w_k) \longrightarrow \text{pas de gradient}$$

$$w_{k+1} = z_{k+1} + \beta_{k+1}(w_k - w_{k-1}) \longrightarrow \text{pas de momentum}$$

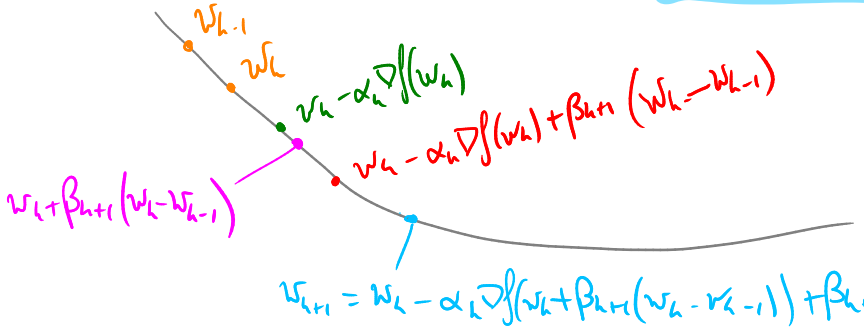
Nesterov inverse les deux équations

$$z_{k+1} = w_k + \beta_{k+1}(w_k - w_{k-1}) \longrightarrow \text{pas de momentum}$$

$$w_{k+1} = z_{k+1} - \alpha_k \nabla f(z_{k+1}) \longrightarrow \text{pas de gradient sur le nouveau point}$$

Ce qui se réécrit

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



↳ Avec les bons choix de α_k et β_{k+1}, l'algorithme de Nesterov possède les garanties de complexité optimales pour les fonctions convexes <u>ainsi que</u> pour les fonctions fortement convexes

⚠ Les choix de (α_k) et (β_{k+1}) sont différents selon que la fonction est convexe ou fortement convexe : en ce sens, l'algorithme de Nesterov est moins versatile que la descente de gradient

<u>Illustration</u> : Pour f convexe (mais fortement convexe), on prend $\alpha_k = \frac{1}{L}$ (où f est C^1_L)

et $\beta_{k+1} = \frac{t_k - 1}{t_{k+1}}$ avec $t_0 = 0$

$t_{k+1} = \frac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right)$

↑ indépendant du problème

- Pour f $C^{1,1}_L$ et μ-fortement convexe ($\mu > 0$) $\left(\Leftrightarrow f(v) \geq f(w) + \nabla f(w)^T(v-w) + \frac{\mu}{2}\|v-w\|^2\right)$

$$\alpha_k = \frac{1}{L} \quad \text{et} \quad \beta_{k+1} = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}$$

Table de complexité

	f $C^{1,1}_L$ convexe	f $C^{1,1}_L$ μ-fortement convexe
Descente de gradient $\alpha_k = 1/L$	$f(w_k) - f^* \leq \varepsilon$ en au plus $O(\varepsilon^{-1})$ itérations	$f(w_k) - f^* \leq \varepsilon$ en au plus $O\left(\frac{L}{\mu} \ln(\varepsilon^{-1})\right)$ itérations ; $L/\mu \geqslant 1$ par définition
Heavy-ball (Polyak) $\alpha_k = 1/L$	×	Si f est quadratique, alors $f(w_k) - f^* \leq \varepsilon$ en au plus $O\left(\sqrt{\frac{L}{\mu}} \ln(\varepsilon^{-1})\right)$ itérations ; $\beta_{k+1} = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}$; $\sqrt{\frac{L}{\mu}} \leq L/\mu$
Nesterov $\alpha_k = 1/L$	$f(w_k) - f^* \leq \varepsilon$ en au plus $O(\varepsilon^{-1/2})$ itérations $\beta_{k+1} = \frac{t_k - 1}{t_{k+1}}$ ...	$f(w_k) - f^* \leq \varepsilon$ en au plus $O\left(\sqrt{\frac{L}{\mu}} \ln(\varepsilon^{-1})\right)$ itérations avec $\beta_{k+1} = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}$

A venir : Notebook

- Les méthodes avec accélération peuvent conduire à augmenter la valeur de f !

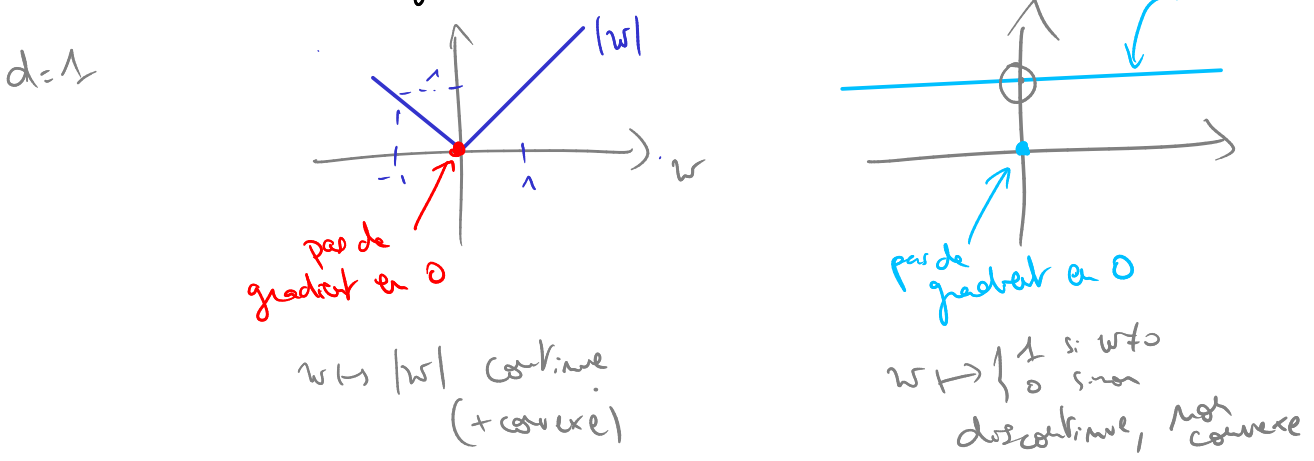
- La méthode de Nesterov pour fonctions convexes n'est pas la meilleure sur les problèmes fortement convexes, idem pour la variante pour fonctions fortement convexes (pas la meilleure sur les problèmes convexes)
- Une implémentation efficace des algorithmes de Polyak et Nesterov requiert 3 vecteurs de stockage (contre 1 ou 2 pour la descente de gradient)

Remarque : Dans le cas non convexe, le momentum / l'accélération ne marche pas en général (mais on peut l'implémenter !)
⇒ Par contre, lorsque l'on fait du gradient stochastique, utiliser du momentum donne de bons résultats pratiques.

(2) Régularisation et méthodes proximales

↳ Q : Que faire si f n'est pas C^1 ?

⇒ On dira que f est non lisse s'il existe des points en lesquels le gradient de f n'est pas défini



→ En sciences des données, et notamment en traitement du signal et de l'image, les fonctions non lisses apparaissent souvent à cause de la régularisation.

Problème régularisé

$$\underset{w \in \mathbb{R}^d}{\text{minimiser}} \left\{ f(w) + \lambda \Omega(w) \right\}$$

- λ : Paramètre de régularisation : donne le poids de la régularisation par rapport à l'attache aux données
- $f(w)$: Terme d'attache aux données, dépend des données du problème
  - ⇩
  - But : Déterminer $w \in \mathbb{R}^d$ qui donne le meilleur modèle des données possible
- $\Omega(w)$: Terme de régularisation, ne dépend pas des données
  - → Typiquement convexe
  - → Souvent non lisse
  - ⇩
  - Favorise les $w \in \mathbb{R}^d$ qui possèdent des propriétés souhaitées

Exemple : Régression linéaire

Données : $\{(x_i, y_i)\}_{i=1..n}$ avec $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

On cherche $w \in \mathbb{R}^d$ tel que $x_i^T w - y_i \approx 0$

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

↳ $\underset{w \in \mathbb{R}^d}{\text{minimiser}} \; \frac{1}{2n} \|Xw - y\|^2 = \frac{1}{2n} \sum_{i=1}^{n} (x_i^T w - y_i)^2$

$\frac{1}{2n} \|Xw - y\|^2$: Terme d'attache aux données

↳ Deux choix de régularisation classiques

- **Régularisation ℓ_2 / "ridge"**

Terme de régularisation ℓ_2 : $\|w\|^2 = \sum_{j=1}^{d} (w_j)^2$

$$\underset{w \in \mathbb{R}^d}{\text{minimise}} \; \frac{1}{2n} \|Xw - y\|^2 + \frac{\lambda}{2} \|w\|^2 \quad \text{avec } \lambda > 0$$

⇒ Pour $\lambda \gg 1$, ce problème équivaut à $\underset{w \in \mathbb{R}^d}{\text{minimiser}} \; \frac{\lambda}{2} \|w\|^2$ dont la solution est $w = 0_{\mathbb{R}^d}$

⇒ Pour des valeurs modestes de λ, la solution du problème régularisé (dont on peut montrer qu'elle est unique) est moins sensible aux perturbations des données que la solution du problème d'origine ($\lambda = 0$)

⇒ Autres propriétés de la régularisation ℓ_2

- Si f est convexe, alors $w \mapsto f(w) + \frac{\lambda}{2}\|w\|_2^2$ est λ-fortement convexe ($\lambda > 0$) ⇒ la fonction régularisée a un unique minimum
- Lorsque λ augmente, les composantes de la solution diminuent de manière uniforme (on dit que la régularisation ℓ_2 "lisse" le problème)
- Contrainte implicite sur la norme de la solution

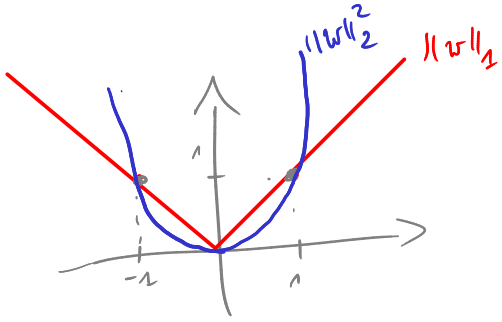
• Régularisation ℓ_1 / LASSO (~1996)

Terme de régularisation ℓ_1

"Norme ℓ_1"

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

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \quad \frac{1}{2n}\|Xw - y\|_2^2 + \lambda \|w\|_1$$



↳ Plus λ est important, plus la solution du problème aura de coordonnées nulles

(≠ de la régularisation ℓ_2 : les coordonnées ne sont nulles que pour $\lambda \gg 1$, et dans ce cas elles sont toutes nulles)

↳ La régularisation ℓ_1 conserve uniquement les coordonnées les plus importantes pour l'attache aux données ⇒ on dit que la régularisation permet de sélectionner les attributs (càd les dimensions de x_i les plus importantes pour l'apprentissage)

↳ Mathématiquement, on peut montrer que la solution du problème régularisé a toujours moins (ou autant) de coordonnées nulles qu'une solution du problème non régularisé

⇒ $\|w\|_1$ est convexe et le problème LASSO est fortement convexe.

↳ On parle parfois de la solution du LASSO comme d'une solution parcimonieuse (sparse en anglais)

⇒ On voudrait construire un algorithme qui s'applique à la fois à la régression linéaire, à la régression ℓ_2 et à la régression LASSO.

différence de structure que l'on peut vouloir exploiter

$\|w\|_1$ non lisse (pas de gradient en tout point qui a une coordonnée nulle)

La descente de gradient ne fait pas de distinction entre ces deux problèmes !

⇓

on ne peut pas appliquer la descente de gradient !

→ Les méthodes proximales sont une classe d'algorithmes dédiée aux problèmes de la forme $\min_{w \in \mathbb{R}^d} f(w) + \lambda \Omega(w)$, qui s'appliquent même quand Ω est non lisse

(NB: elles s'appliquent aussi quand f est non lisse)

Algorithme principal : Le gradient proximal

Hypothèse : $\min_{w \in \mathbb{R}^d} f(w) + \lambda \Omega(w)$

$f : \mathbb{R}^d \to \mathbb{R}$ C^1 (peut seulement convexe)

$\Omega : \mathbb{R}^d \to \mathbb{R}$ convexe (peut seulement C^1)

Itération

$$w_{k+1} \in \operatorname*{argmin}_{w \in \mathbb{R}^d} \left\{ \underbrace{f(w_k) + \nabla f(w_k)^T (w - w_k)}_{\approx f(w) \text{ pour } w \text{ proche de } w_k} + \underbrace{\frac{1}{2\alpha_k} \|w - w_k\|^2}_{\text{terme proximal : force la solution à être proche de } w_k \ (\alpha_k > 0)} + \lambda \Omega(w) \right\}$$

terme de régularisation "tel quel"

→ A chaque itération, l'algorithme du gradient proximal résout un problème d'optimisation auxiliaire (dit "sous-problème") défini par f, λ, Ω et l'itéré courant w_k.

→ L'idée est de remplacer le problème de départ par une suite de sous-problèmes pour lesquels on remplace f par une approximation en utilisant son gradient ⟹ N'a d'intérêt que lorsque le sous-problème est plus simple à résoudre que le problème de départ

→ Si on considère $\Omega(w) = 0 \quad \forall w \in \mathbb{R}^d$, alors l'itération de gradient proximal donne

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k)$$: c'est l'itération de la descente de gradient

→ Pour Ω convexe, on peut montrer que le gradient proximal converge avec une complexité du même ordre que celle de la descente de gradient

(*) Si f convexe, on obtient une valeur de fonction à ε de l'optimum

$$\left((f + \lambda\Omega)(w_k) - \min_w (f + \lambda\Omega)(w) \leq \varepsilon \right)$$

en au plus $O(\varepsilon^{-1})$ itérations

⚠ Une itération de gradient proximal a un coût différent (en général plus grand) de celui d'une itération de descente de gradient

→ Pour des régularisations ℓ_2 et ℓ_1, l'itération du gradient proximal s'écrit de manière explicite

Gradient proximal pour minimiser $f(w) + \frac{\lambda}{2}\|w\|^2$ $\quad f \in C^1$
$w \in \mathbb{R}^d$

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

→ $\lambda = 0$: on retrouve la descente de gradient

→ $\lambda \gg 1 \Rightarrow w_{k+1} \approx 0$

→ Le facteur $\frac{1}{1+\lambda\alpha_k}$ réduit les composantes de w_k de manière uniforme à chaque itération

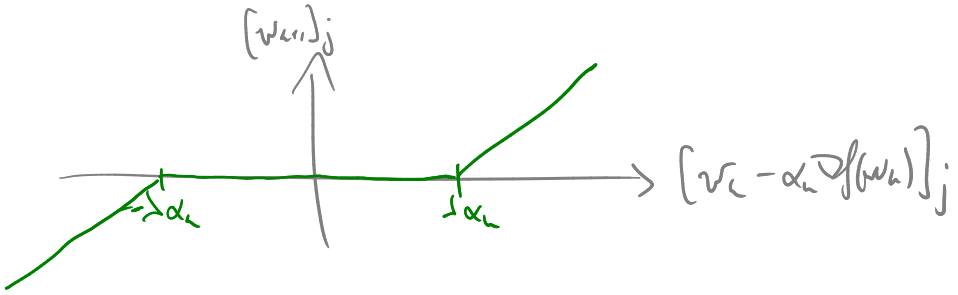
$\Rightarrow$ Lien avec la "Batch normalization" en apprentissage profond

Gradient proximal pour minimiser $f(w) + \lambda \|w\|_1$ $f \in C^1$, $w \in \mathbb{R}^d$

aussi appelé ISTA (Iterative Soft-Thresholding Algorithm)

$\forall j = 1..d$,

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



→ $\lambda = 0$: on retrouve l'itération de descente de gradient

→ plus λ est grand, plus $[-\lambda\alpha_k, \lambda\alpha_k]$ sera un grand intervalle et plus $[w_{k+1}]_j$ aura de chances d'être mis à zéro

→ Par rapport au pas de descente de gradient $w_k - \alpha_k \nabla f(w_k)$, w_{k+1} aura nécessairement plus ou autant de coordonnées nulles.

NB: Si f est convexe, on peut accélérer le gradient proximal (pour régularisation ℓ_1: FISTA, 2009 13999 citations dans Google Scholar)