

# Optimisation-RO

Clément Royer

Certificat Chef de Projet IA - Université Paris Dauphine-PSL

15 novembre 2021





## Clément Royer

- Maître de conférences @ Dauphine-PSL;
- Chercheur permanent @ LAMSADE;
- Chaire tremplin en optimisation @ PRAIRIE;
- Ingénieur @ ENSEEIHT;
- Docteur @ Université de Toulouse.

<https://github.com/clementwroyer/opt-ro-psl>

- Transparents de cours;
- Notebooks Python.

## Déroulé de la formation

- Lundi 15/11 matin+après-midi (C. Royer) : Optimisation douce et optimisation convexe;
- Mardi 16/11 après-midi+Jeudi 18/11 matin (P. Ablin) : Méthodes stochastiques à grande échelle;
- Jeudi 18/11 après-midi (C. Royer) : Optimisation sans dérivées.

- 1 Introduction
- 2 Concepts de base en optimisation
- 3 Optimisation sans contraintes

- 1 Introduction
- 2 Concepts de base en optimisation
- 3 Optimisation sans contraintes

## Ce dont tout le monde parle

- Sciences des données (data science);
- Analyse de données (data analysis);
- Fouille de données (data mining);
- Apprentissage machine/profond (machine/learning);
- Intelligence artificielle (IA);
- Big Data;
- ...

## Ce dont tout le monde parle

- Sciences des données (data science);
- Analyse de données (data analysis);
- Fouille de données (data mining);
- Apprentissage machine/profond (machine/learning);
- Intelligence artificielle (IA);
- Big Data;
- ...

## Ce dont nous allons parler

- Optimisation pour la science des données en général...
- ...et pour l'IA en particulier (ou vice-versa).

## Un ensemble de problèmes basés sur des données

- Extraction d'information à partir de la donnée : *statistiques, attributs principaux, structures*;
- Utilisation de cette information pour la **prédiction du comportement de données futures**.



# Ce que sera l'IA pour nous

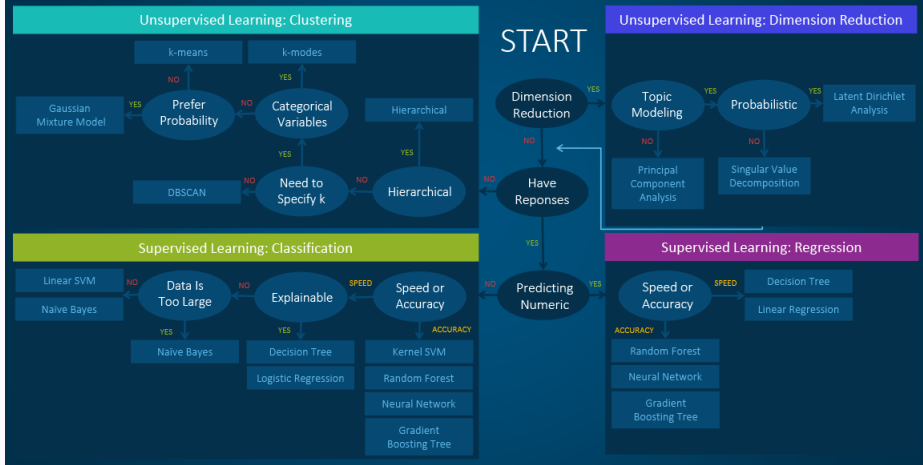
## Un ensemble de problèmes basés sur des données

- Extraction d'information à partir de la donnée : *statistiques, attributs principaux, structures*;
- Utilisation de cette information pour la **prédiction du comportement de données futures**.

## Composantes de l'IA/de la science des données

- Statistiques;
- Informatique (gestion de la donnée, calcul parallèle, etc);
- **Optimisation** pour la modélisation des problèmes et leur résolution par des algorithmes.

## Machine Learning Algorithms Cheat Sheet



Source : <https://blogs.sas.com/content/subconsciousmusings/2017/04/12/machine-learning-algorithm-use/>

## Optimisation numérique

- Montée en puissance en 1970-1980;
- Succès des algorithmes en ingénierie (chimique, aéronautique, etc).
- *Pratique standard en calcul scientifique*: utiliser une méthode de points intérieurs (basée sur Newton, développée dans les années 2000s).

## Optimisation pour l'IA

- Problèmes basés sur de grands volumes de données;
- Les méthodes standard en optimisation ne sont pas les plus efficaces!

*Pratique classique en IA*: Utiliser une approche de gradient stochastique avec momentum (1950s + article théorique de 1983).

## Contexte de données massives/Big Data

- Les calculs usuels (fonction, dérivées) sont très coûteux car ils accèdent à **toute la donnée**.
- La précision souhaitée n'est pas forcément très grande en raison du bruit sur les données.

## Contexte de données massives/Big Data

- Les calculs usuels (fonction, dérivées) sont très coûteux car ils accèdent à **toute la donnée**.
- La précision souhaitée n'est pas forcément très grande en raison du bruit sur les données.

## Communauté de l'IA

- Le problème d'optimisation est souvent un moyen plus qu'une fin;
- Propriétés statistiques des solutions;
- Théorie et pratique différentes de la communauté d'optimisation "classique".

- 1 Introduction
  - Un exemple : classification binaire
- 2 Concepts de base en optimisation
- 3 Optimisation sans contraintes

**Point de départ :** Jeu de données  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ .

- $\mathbf{x}_i$  vecteur d'**attributs** à  $d$  coordonnées;
- $y_i$  **label** binaire égal à 1 ou  $-1$ .

**Point de départ :** Jeu de données  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ .

- $\mathbf{x}_i$  vecteur d'**attributs** à  $d$  coordonnées;
- $y_i$  **label** binaire égal à 1 ou  $-1$ .

**Exemple :** classification de documents

Soit un dictionnaire de  $d$  mots.

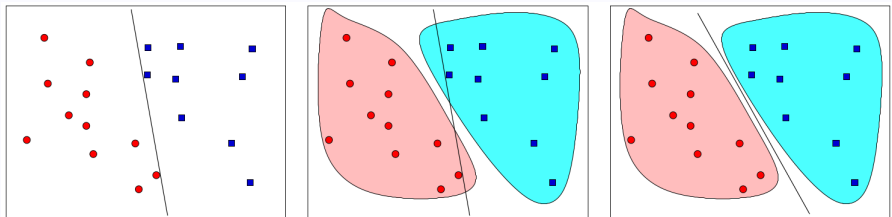
- $\mathbf{x}_i$  représente les mots contenus dans le document  $i$ :

$$[\mathbf{x}_i]_j = \begin{cases} 1 & \text{si le mot } j \text{ est dans le document } i, \\ 0 & \text{sinon.} \end{cases}$$

- $y_i$  égal à  $+1$  si le document traite de l'automobile, à  $-1$  sinon.



# Différentes solutions



Source : S. J. Wright, Optimization Algorithms for Data Analysis, 2018.

- Points :  $\mathbf{x}_i$ , rouges/bleus :  $y_i = 1/y_i = -1$ ;
- Nuages rouges/bleus : distribution des documents;
- Deux techniques de classification linéaires;
- Figure de droite : solution à marge maximale (SVM).

## Optimiseur

- Le problème peut être modélisé comme un programme quadratique convexe, et résolu de manière efficace;
- Potentiellement plusieurs solutions.

# Deux points de vue sur le problème

## Optimiseur

- Le problème peut être modélisé comme un programme quadratique convexe, et résolu de manière efficace;
- Potentiellement plusieurs solutions.

## Applicatif

- Le modèle doit s'appliquer à tous les documents de la distribution  $\Rightarrow$  généralisation;
- Mieux d'avoir une unique solution bien définie, qui ne varie pas trop par rapport aux données.

# Deux points de vue sur le problème

## Optimiseur

- Le problème peut être modélisé comme un programme quadratique convexe, et résolu de manière efficace;
- Potentiellement plusieurs solutions.

## Applicatif

- Le modèle doit s'appliquer à tous les documents de la distribution  $\Rightarrow$  généralisation;
- Mieux d'avoir une unique solution bien définie, qui ne varie pas trop par rapport aux données.

## Après discussion (data scientist?)

- Prise en compte de certaines problématiques dans la formulation du problème  $\Rightarrow$  Solution unique avec meilleure généralisation.
- Plus de connaissances sur le problème  $\Rightarrow$  Meilleure optimisation.

- Possible de définir des problèmes d'optimisation basés sur des données et de les résoudre efficacement;
- Potentiellement décorrélé du but originel : trouver un modèle sur la distribution des données.

- Possible de définir des problèmes d'optimisation basés sur des données et de les résoudre efficacement;
- Potentiellement décorrélé du but originel : trouver un modèle sur la distribution des données.

## Autres problématiques

- Quantité massive d'attributs (*tous les mots du dictionnaire*) ?
- Quantité massive de données (*articles Wikipedia*) ?
- Classification impossible par modèles linéaires ?

- Possible de définir des problèmes d'optimisation **basés sur des données** et de les résoudre efficacement;
- Potentiellement décorrélé du but originel : trouver un modèle sur la **distribution des données**.

## Autres problématiques

- Quantité massive d'attributs (*tous les mots du dictionnaire*) ?  
Réduction de dimension, recherche de parcimonie.
- Quantité massive de données (*articles Wikipedia*) ?  
Algorithmes stochastiques.
- Classification impossible par modèles linéaires ?  
Optimisation non linéaire.

- Fournir une boîte à outils moderne en optimisation;
- En lien avec la pratique courante en IA et sciences des données.

## Procédure

- Présenter des problématiques et des algorithmes associés;
- De la théorie (mais pas tout);
- Des applications via des notebook Python.



- 1 Introduction
- 2 Concepts de base en optimisation
  - Notations et rappels d'analyse
  - Un problème d'optimisation
  - Algorithmes d'optimisation
- 3 Optimisation sans contraintes

- 1 Introduction
- 2 Concepts de base en optimisation
  - Notations et rappels d'analyse
  - Un problème d'optimisation
  - Algorithmes d'optimisation
- 3 Optimisation sans contraintes

## Cadre restreint

- Optimisation sur des variables réelles;
- En dimension finie;
- On utilisera la structure d'espace classique.

## Mes notations pour aujourd'hui

- Scalaires :  $a, b, c, \dots$
- Vecteurs :  $\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots$
- Matrices :  $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$
- Ensembles :  $\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$

- $\mathbb{R}^d$  : ensemble des vecteurs à  $d \geq 1$  coordonnées réelles;
- Pour tous  $\mathbf{w} \in \mathbb{R}^d$  et  $i \in \{1, \dots, d\}$ ,  $w_i \in \mathbb{R}$  est la  $i$ -ème coordonnée de  $\mathbf{w}$  :  $\mathbf{w} = [w_i]_{1 \leq i \leq d}$ ;
- On représente  $\mathbf{w} \in \mathbb{R}^d$  en colonnes :  $\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$  ;
- On utilise des vecteurs lignes comme “transposés” de vecteurs colonnes :  $\mathbf{w}^T := [w_1 \cdots w_d]$ ;

- $\mathbb{R}^d$  : ensemble des vecteurs à  $d \geq 1$  coordonnées réelles;
- Pour tous  $\mathbf{w} \in \mathbb{R}^d$  et  $i \in \{1, \dots, d\}$ ,  $w_i \in \mathbb{R}$  est la  $i$ -ème coordonnée de  $\mathbf{w}$  :  $\mathbf{w} = [w_i]_{1 \leq i \leq d}$ ;
- On représente  $\mathbf{w} \in \mathbb{R}^d$  en colonnes :  $\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$ ;
- On utilise des vecteurs lignes comme “transposés” de vecteurs colonnes :  $\mathbf{w}^T := [w_1 \cdots w_d]$ ;

## Opérations vectorielles

- *Addition dans  $\mathbb{R}^d$  :  $\mathbf{w} + \mathbf{z} := [w_i + z_i]_{1 \leq i \leq d}$ ;*
- *Multiplication d'un vecteur de  $\mathbb{R}^d$  par un réel :  $\lambda \mathbf{w} := [\lambda w_i]_{1 \leq i \leq d}$ .*

## Norme euclidienne sur $\mathbb{R}^d$

La norme euclidienne (ou norme  $\ell_2$ ) d'un vecteur  $\mathbf{w} \in \mathbb{R}^d$  est donnée par

$$\|\mathbf{w}\| := \sqrt{\sum_{i=1}^d w_i^2}.$$

## Produit scalaire sur $\mathbb{R}^d$

Pour tous  $\mathbf{w}, \mathbf{z} \in \mathbb{R}^d$ , défini par

$$\mathbf{w}^T \mathbf{z} := \sum_{i=1}^d w_i z_i.$$

On a ainsi  $\mathbf{w}^T \mathbf{z} = \mathbf{z}^T \mathbf{w}$  et  $\mathbf{w}^T \mathbf{w} = \|\mathbf{w}\|^2$ .

## Matrices

- $\mathbb{R}^{n \times d}$  : matrices à  $n$  lignes et  $d$  colonnes;
- $\mathbb{R}^{d \times 1} \simeq \mathbb{R}^d$ .

## Matrice transposée

Soit  $\mathbf{A} = [\mathbf{A}_{ij}] \in \mathbb{R}^{n \times d}$ . La *matrice transposée* de  $\mathbf{A}$ , notée  $\mathbf{A}^T$ , est la matrice à  $d$  lignes et  $n$  colonnes telle que

$$\forall i = 1, \dots, n, \forall j = 1, \dots, d, \quad [\mathbf{A}^T]_{ij} = \mathbf{A}_{ji}.$$

## Matrices

- $\mathbb{R}^{n \times d}$  : matrices à  $n$  lignes et  $d$  colonnes;
- $\mathbb{R}^{d \times 1} \simeq \mathbb{R}^d$ .

## Matrice transposée

Soit  $\mathbf{A} = [\mathbf{A}_{ij}] \in \mathbb{R}^{n \times d}$ . La *matrice transposée* de  $\mathbf{A}$ , notée  $\mathbf{A}^T$ , est la matrice à  $d$  lignes et  $n$  colonnes telle que

$$\forall i = 1, \dots, n, \forall j = 1, \dots, d, \quad [\mathbf{A}^T]_{ij} = \mathbf{A}_{ji}.$$

## Matrices carrées

- $\mathbf{A}^T \in \mathbb{R}^{d \times d}$ ;
- $\mathbf{A}$  est dite *symétrique* si  $\mathbf{A} = \mathbf{A}^T$ .



## Inversion de matrice

Une matrice  $\mathbf{A} \in \mathbb{R}^{d \times d}$  est dite *invertible* s'il existe  $\mathbf{B} \in \mathbb{R}^{d \times d}$  telle que  $\mathbf{BA} = \mathbf{AB} = \mathbf{I}_d$ , avec  $\mathbf{I}_d$  matrice identité de  $\mathbb{R}^{d \times d}$ .

Dans ce cas,  $\mathbf{B}$  est l'unique matrice vérifiant cette propriété : on l'appelle *l'inverse de la matrice  $\mathbf{A}$*  et on la note  $\mathbf{A}^{-1}$ .

## Caractère (semi)-défini positif

Une matrice  $\mathbf{A} \in \mathbb{R}^{d \times d}$  est dite *semi-définie positive* si

$$\forall \mathbf{v} \in \mathbb{R}^d, \quad \mathbf{v}^T \mathbf{A} \mathbf{v} \geq 0.$$

Elle est *définie positive* lorsque  $\mathbf{v}^T \mathbf{A} \mathbf{v} > 0$  pour tout vecteur  $\mathbf{v}$  non nul.

## Valeurs propres et vecteur propres

Soit  $\mathbf{A} \in \mathbb{R}^{d \times d}$ . Un réel  $\lambda \in \mathbb{R}$  est une *valeur propre* de  $\mathbf{A}$  si

$$\exists \mathbf{v} \in \mathbb{R}^d, \|\mathbf{v}\| \neq 0, \quad \mathbf{A}\mathbf{v} = \lambda\mathbf{v}.$$

Le vecteur  $\mathbf{v}$  s'appelle alors un *vecteur propre* de  $\mathbf{A}$  (associé à  $\lambda$ ).

## Valeurs propres et vecteur propres

Soit  $\mathbf{A} \in \mathbb{R}^{d \times d}$ . Un réel  $\lambda \in \mathbb{R}$  est une *valeur propre* de  $\mathbf{A}$  si

$$\exists \mathbf{v} \in \mathbb{R}^d, \|\mathbf{v}\| \neq 0, \quad \mathbf{A}\mathbf{v} = \lambda\mathbf{v}.$$

Le vecteur  $\mathbf{v}$  s'appelle alors un *vecteur propre* de  $\mathbf{A}$  (associé à  $\lambda$ ).

Une matrice symétrique de  $\mathbb{R}^{d \times d}$  possède  $d$  valeurs propres réelles. Étant données deux matrices symétriques  $(\mathbf{A}, \mathbf{B}) \in \mathbb{R}^{d \times d}$ , on utilisera les notations suivantes :

- $\lambda_{\min}(\mathbf{A})/\lambda_{\max}(\mathbf{A})$  : plus petite/plus grande valeur propre de  $\mathbf{A}$ ;
- $\mathbf{A} \stackrel{n}{\succeq} \mathbf{B} \Leftrightarrow \lambda_{\min}(\mathbf{A}) \geq \lambda_{\max}(\mathbf{B})$ ;
- $\mathbf{A} \stackrel{n}{\succ} \mathbf{B} \Leftrightarrow \lambda_{\min}(\mathbf{A}) > \lambda_{\max}(\mathbf{B})$ .

Avec ces notations  $\mathbf{A}$  sera semi-définie positive (resp. définie positive) si et seulement si  $\mathbf{A} \succeq 0$  (resp.  $\mathbf{A} \succ 0$ ).

On considère une fonction **lisse** (ou douce, ou *smooth*)  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ .

On considère une fonction **lisse** (ou douce, ou *smooth*)  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ .

### Dérivée à l'ordre 1

Si  $f$  est continûment dérivable sur  $\mathbb{R}^d$ , on définit pour tout  $\mathbf{w} \in \mathbb{R}^d$  le **gradient de  $f$  en  $\mathbf{w}$**  par

$$\nabla f(\mathbf{w}) := \left[ \frac{\partial f}{\partial w_i}(\mathbf{w}) \right]_{1 \leq i \leq d} \in \mathbb{R}^d.$$

L'ensemble des fonctions continûment dérivables est noté  $\mathcal{C}^1 \stackrel{n}{=} \mathcal{C}^1(\mathbb{R}^d, \mathbb{R})$ .  
On parle de fonction de classe  $\mathcal{C}^1$ .

On considère une fonction **lisse** (ou douce, ou *smooth*)  $f : \mathbb{R}^d \rightarrow \mathbb{R}^m$ .

## Matrice jacobienne

$f : \mathbb{R}^d \rightarrow \mathbb{R}^m$  est dérivable en  $\mathbf{w} \in \mathbb{R}^d$  si il existe une matrice  $\mathbf{J}_f(\mathbf{x}) \in \mathbb{R}^{m \times d}$  telle que

$$\lim_{\substack{\mathbf{z} \rightarrow \mathbf{w} \\ \mathbf{z} \neq \mathbf{w}}} \frac{\|f(\mathbf{z}) - f(\mathbf{w}) - \mathbf{J}_f(\mathbf{w})(\mathbf{z} - \mathbf{w})\|}{\|\mathbf{z} - \mathbf{w}\|} = 0.$$

$\mathbf{J}_f(\mathbf{w})$  s'appelle la (matrice) **jacobienne** de  $f$  en  $\mathbf{w}$ .

## Cas particuliers

- $m = 1$  : la jacobienne se ramène à un vecteur  $\nabla f(\mathbf{x}) \equiv \mathbf{J}_f(\mathbf{x})^T$ , que l'on appelle le **vecteur gradient**;
- $n = m = 1$  : la jacobienne est équivalente à un scalaire  $f'(w)\mathbf{J}_f(w)$ , que l'on appelle la dérivée de  $f$  en  $\mathbf{w}$ .

On considère une fonction **lisse** (ou douce, ou *smooth*)  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ .

On considère une fonction **lisse** (ou douce, ou *smooth*)  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ .

## Dérivée d'ordre 2

Si  $f$  est *deux fois* continûment dérivable sur  $\mathbb{R}^d$ , on définit pour tout  $\mathbf{w} \in \mathbb{R}^d$  la **matrice hessienne de  $f$  en  $\mathbf{w}$**  par

$$\nabla^2 f(\mathbf{w}) := \left[ \frac{\partial^2 f}{\partial w_i \partial w_j}(\mathbf{w}) \right]_{1 \leq i, j \leq d} \in \mathbb{R}^{d \times d}.$$

Cette matrice est **symétrique**.

L'ensemble des fonctions deux fois continûment dérivables est noté  $\mathcal{C}^2$  (on dira que  $f$  est de classe  $\mathcal{C}^2$ ).



## Développement de Taylor à l'ordre 1

Si  $f \in \mathcal{C}^1$ , pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) = f(\mathbf{w}) + \int_0^1 \nabla f(\mathbf{w} + t\mathbf{h})^T \mathbf{h} dt.$$

## Développement de Taylor à l'ordre 1

Si  $f \in \mathcal{C}^1$ , pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) = f(\mathbf{w}) + \int_0^1 \nabla f(\mathbf{w} + t\mathbf{h})^T \mathbf{h} dt.$$

## Développement de Taylor à l'ordre 2

Si  $f \in \mathcal{C}^2$ , pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) = f(\mathbf{w}) + \nabla f(\mathbf{w})^T \mathbf{h} + \frac{1}{2} \int_0^1 \mathbf{h}^T \nabla^2 f(\mathbf{w} + t\mathbf{h}) \mathbf{h} dt.$$

## Définition

Une fonction  $\mathbf{g} : \mathbb{R}^d \rightarrow \mathbb{R}^m$  est dite  $L$ -lipschitzienne si il existe  $L > 0$  telle que

$$\forall (\mathbf{w}, \mathbf{z}) \in (\mathbb{R}^d)^2, \quad \|\mathbf{g}(\mathbf{w}) - \mathbf{g}(\mathbf{z})\| \leq L \|\mathbf{w} - \mathbf{z}\|.$$

La valeur  $L$  s'appelle une constante de Lipschitz pour  $\mathbf{g}$ .

- Ex) Toute fonction linéaire est Lipschitzienne;
- $\mathcal{C}_L^{1,1}$  : sous-ensemble de  $\mathcal{C}^1$  des fonctions avec dérivée première  $L$ -lipschitzienne;
- $\mathcal{C}_L^{2,2}$  : sous-ensemble de  $\mathcal{C}^2$  des fonctions avec dérivée seconde  $L$ -lipschitzienne;

## Approximation de Taylor à l'ordre 1

Soit  $f \in \mathcal{C}_L^{1,1}$ . Pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^T \mathbf{h} + \frac{L}{2} \|\mathbf{h}\|^2.$$

## Approximation de Taylor à l'ordre 1

Soit  $f \in \mathcal{C}_L^{1,1}$ . Pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^T \mathbf{h} + \frac{L}{2} \|\mathbf{h}\|^2.$$

⇒ Une des inégalités majeures en optimisation non linéaire.

## Approximation de Taylor à l'ordre 1

Soit  $f \in \mathcal{C}_L^{1,1}$ . Pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^T \mathbf{h} + \frac{L}{2} \|\mathbf{h}\|^2.$$

⇒ Une des inégalités majeures en optimisation non linéaire.

## Approximation de Taylor à l'ordre 2

Soit  $f \in \mathcal{C}_L^{2,2}$ . Pour tous  $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ ,

$$f(\mathbf{w} + \mathbf{h}) \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^T \mathbf{h} + \frac{1}{2} \mathbf{h}^T \nabla^2 f(\mathbf{w}) \mathbf{h} + \frac{L}{6} \|\mathbf{h}\|^3,$$

- De nombreuses notes de cours disponibles;
- **Annexes** de nombreux ouvrages d'optimisation (voire d'IA)!

## Mes recommandations

- En français :  
<https://www.lpsm.paris/pageperso/bolley/poly-cdiff.pdf>  
<https://www.lpsm.paris/pageperso/bolley/poly-algebre3.pdf>
- En anglais :  
<http://vmls-book.stanford.edu/vmls.pdf> (Chapters 1-3)  
[https://sebastianraschka.com/pdf/books/dlb/appendix\\_d\\_calculus.pdf](https://sebastianraschka.com/pdf/books/dlb/appendix_d_calculus.pdf).

- 1 Introduction
- 2 Concepts de base en optimisation
  - Notations et rappels d'analyse
  - Un problème d'optimisation
  - Algorithmes d'optimisation
- 3 Optimisation sans contraintes



- Recherche opérationnelle;
- Prise de décision;
- Sciences de la décision;
- Programmation mathématique;
- Optimisation mathématique.

⇒ Tous ces concepts peuvent correspondre à de l'optimisation.

- Recherche opérationnelle;
- Prise de décision;
- Sciences de la décision;
- Programmation mathématique;
- Optimisation mathématique.

⇒ Tous ces concepts peuvent correspondre à de l'optimisation.

## Ma définition

*Le but de l'optimisation est de prendre la meilleure décision parmi un ensemble de possibilités.*

Un **problème** de minimisation sur  $d$  paramètres réels s'écrit sous la forme :

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) \quad \text{s. c. } \mathbf{w} \in \mathcal{F}$$

Un **problème** de minimisation sur  $d$  paramètres réels s'écrit sous la forme :

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) \quad \text{s. c. } \mathbf{w} \in \mathcal{F}$$

- $\mathbf{w}$  représenté les **variables de décision**;
- $d$  est la dimension du problème (on prendra toujours  $d \geq 1$ );
- $f(\cdot)$  est la fonction **objectif/de coût/de perte**;
- $\mathcal{F}$  est l'ensemble réalisable/admissible regroupant les contraintes sur les variables de décision.

Un **problème** de minimisation sur  $d$  paramètres réels s'écrit sous la forme :

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) \quad \text{s. c. } \mathbf{w} \in \mathcal{F}$$

- $\mathbf{w}$  représenté les **variables de décision**;
- $d$  est la dimension du problème (on prendra toujours  $d \geq 1$ );
- $f(\cdot)$  est la fonction **objectif/de coût/de perte**;
- $\mathcal{F}$  est l'ensemble réalisable/admissible regroupant les contraintes sur les variables de décision.

*Maximiser  $f$  revient à minimiser  $-f$ .*

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) \quad \text{s. c. } \mathbf{w} \in \mathcal{F}$$

## Minimum local

- Un point  $\mathbf{w}^*$  est un **minimum local** du problème s'il existe un voisinage  $\mathcal{N}$  de  $\mathbf{w}^*$  sur lequel  $f(\mathbf{w}^*) \leq f(\mathbf{w}) \forall \mathbf{w} \in \mathcal{N} \cap \mathcal{F}$ ;
- Cas sans contraintes : il existe  $\epsilon > 0$ ,  
 $f(\mathbf{w}^*) < f(\mathbf{w}) \forall \mathbf{w}, \|\mathbf{w} - \mathbf{w}^*\| \leq \epsilon$ .

## Minimum global

Un point  $\mathbf{w}^*$  est un **minimum global** du problème si  $f(\mathbf{w}^*) \leq f(\mathbf{w}) \forall \mathbf{w} \in \mathcal{F}$ .

- Trouver des minima globaux est difficile en général;
- Trouver et certifier des minima locaux peut aussi être difficile.

- Trouver des minima globaux est difficile en général;
- Trouver et certifier des minima locaux peut aussi être difficile.

## Cas “faciles”

- Bonnes propriétés de la fonction objectif;
- Bonne géométrie de l'ensemble des contraintes, le cas échéant.

⇒ L'optimisation douce conduit à certains cas faciles.



**Problème sans contraintes** minimiser  $_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^1$ .

**Problème sans contraintes** minimiser  $_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^1$ .

Condition nécessaire à l'ordre 1

**Si**  $\mathbf{w}^*$  est un minimum local du problème, **alors**

$$\|\nabla f(\mathbf{w}^*)\| = 0.$$

**Problème sans contraintes** minimiser  $\mathbf{w} \in \mathbb{R}^d$   $f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^1$ .

Condition nécessaire à l'ordre 1

Si  $\mathbf{w}^*$  est un minimum local du problème, **alors**

$$\|\nabla f(\mathbf{w}^*)\| = 0.$$

- Cette condition est seulement nécessaire;
- Un point tel que  $\|\nabla f(\mathbf{w}^*)\| = 0$  peut aussi être un maximum local ou un point selle.

**Problème sans contraintes** minimiser  $_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^2$ .

**Problème sans contraintes** minimiser  $\mathbf{w} \in \mathbb{R}^d$   $f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^2$ .

Condition nécessaire à l'ordre 2

Si  $\mathbf{w}^*$  est un minimum local du problème, **alors**

$$\|\nabla f(\mathbf{w}^*)\| = 0 \quad \text{et} \quad \nabla^2 f(\mathbf{w}^*) \succeq 0.$$

**Problème sans contraintes** minimiser  $\mathbf{w} \in \mathbb{R}^d$   $f(\mathbf{w})$ ,  
 $f$  de classe  $\mathcal{C}^2$ .

Condition nécessaire à l'ordre 2

Si  $\mathbf{w}^*$  est un minimum local du problème, **alors**

$$\|\nabla f(\mathbf{w}^*)\| = 0 \quad \text{et} \quad \nabla^2 f(\mathbf{w}^*) \succeq 0.$$

Condition suffisante à l'ordre 2

Si  $\mathbf{w}^*$  vérifie

$$\|\nabla f(\mathbf{w}^*)\| = 0 \quad \text{and} \quad \nabla^2 f(\mathbf{w}^*) \succ 0,$$

**alors** c'est un minimum local du problème.

- 1 Introduction
- 2 Concepts de base en optimisation
  - Notations et rappels d'analyse
  - Un problème d'optimisation
  - Algorithmes d'optimisation
- 3 Optimisation sans contraintes

- **Mathématique** : Prouver l'existence de solutions, le côté bien posé d'une formulation souvent complexe.
- **Logicielle** : Programmer un algorithme pour résoudre une classe de problèmes en pratique.
- **Numérique** : Élaborer des algorithmes, établir des garanties théoriques et valider leur implémentation.



- **Mathématique** : Prouver l'existence de solutions, le côté bien posé d'une formulation souvent complexe.
- **Logicielle** : Programmer un algorithme pour résoudre une classe de problèmes en pratique.
- **Numérique** : Élaborer des algorithmes, établir des garanties théoriques et valider leur implémentation.

On traitera essentiellement de la dernière catégorie.

# Comment résoudre un problème d'optimisation douce ?

## Approche idéale

- Trouver les solutions de  $\|\nabla f(\mathbf{w})\| = 0$ ;
- Prendre celle avec la plus faible valeur à l'objectif.

# Comment résoudre un problème d'optimisation douce ?

## Approche idéale

- Trouver les solutions de  $\|\nabla f(\mathbf{w})\| = 0$ ;
- Prendre celle avec la plus faible valeur à l'objectif.

## Problème

- Résoudre l'équation n'est pas trivial;
- Il peut y avoir une infinité de solutions;
- L'implémentation de cette procédure doit être possible.

## De manière itérative

- Idée de base : étant donné un point courant, se déplacer vers un point potentiellement meilleur;
- Une itération représente l'ensemble des calculs nécessaires pour ce déplacement.

## Notre but dans le reste du cours

- Proposer des algorithmes;
- Décrire leurs garanties théoriques;
- Vérifier leur intérêt pratique (notebooks).

Pour résoudre minimiser  $\mathbf{w} \in \mathbb{R}^d$   $f(\mathbf{w})$ , l'algorithme devrait satisfaire les propriétés suivantes :

- 1 Les points calculés tendent vers une solution;
- 2 Les valeurs de l'objectif tendent vers la valeur optimale;
- 3 Une condition d'optimalité est satisfaite à la limite.

Pour résoudre minimiser  $\mathbf{w} \in \mathbb{R}^d$   $f(\mathbf{w})$ , l'algorithme devrait satisfaire les propriétés suivantes :

- 1 Les points calculés tendent vers une solution;
- 2 Les valeurs de l'objectif tendent vers la valeur optimale;
- 3 Une condition d'optimalité est satisfaite à la limite.

## Convergence des itérés

L'algorithme génère une suite  $\{\mathbf{w}_k\}_k$  telle que

$$\|\mathbf{w}_k - \mathbf{w}^*\| \rightarrow 0 \quad \text{lorsque } k \rightarrow \infty,$$

où  $\mathbf{w}^* \in \operatorname{argmin}_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$  est une solution globale du problème.

### Convergence en valeur de fonction

$$f(\mathbf{w}_k) \rightarrow f^* \quad \text{lorsque } k \rightarrow \infty,$$

où  $f^* = \min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ .

## Convergence en valeur de fonction

$$f(\mathbf{w}_k) \rightarrow f^* \quad \text{lorsque } k \rightarrow \infty,$$

où  $f^* = \min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ .

## Convergence vers un point stationnaire d'ordre 1

$$\|\nabla f(\mathbf{w}_k)\| \rightarrow 0 \quad \text{lorsque } k \rightarrow \infty.$$

*Condition plus générale.*



# Pourquoi ces conditions ?

Pratique  $\neq$  théorie :

- Solution optimale inconnue;
- Valeur à l'optimum inconnue.

# Pourquoi ces conditions ?

Pratique  $\neq$  théorie :

- Solution optimale inconnue;
- Valeur à l'optimum inconnue.

Du point de vue algorithmique, on peut observer:

- le comportement des itérés;
- le comportement des valeurs de la fonction objectif;
- le comportement de la norme du gradient.

- En optimisation, les résultats classiques sont asymptotiques

$$Ex : \|\nabla f(\mathbf{w}_k)\| \rightarrow 0 \quad \text{lorsque } k \rightarrow \infty.$$

- En optimisation, les résultats classiques sont asymptotiques

$$Ex : \|\nabla f(\mathbf{w}_k)\| \rightarrow 0 \quad \text{lorsque } k \rightarrow \infty.$$

- Les **vitesses de convergence** sont plus à la mode

$$Ex : \|\nabla f(\mathbf{w}_k)\| = \mathcal{O}\left(\frac{1}{k}\right) \Leftrightarrow \exists C > 0, \|\nabla f(\mathbf{w}_k)\| \leq \frac{C}{k} \quad \forall k.$$

tout comme la notion équivalente de **complexité au pire cas**:

$$Ex : \text{Algorithm } \mathcal{O}(\epsilon^{-1}) \Leftrightarrow \exists C > 0, \|\nabla f(\mathbf{w}_k)\| \leq \epsilon \text{ quand } k \geq \frac{C}{\epsilon}.$$

- Common in convex optimization;
- Standard in theoretical computer science/statistics.

## Langages typiques des optimiseurs

- C/C++/Fortran (calcul à hautes performances)
- Matlab, Python, Julia.(interprétés).

## Langages typiques des optimiseurs

- C/C++/Fortran (calcul à hautes performances)
- Matlab, Python, Julia.(interprétés).

## Langages de modélisation

- GAMS, AMPL, CVX, Pyomo sont génériques;
- MATPOWER, PyTorch sont spécifiques à un domaine;
- La plupart peuvent être interfacés avec les langages ci-dessus.

## Modélisation

- Objectif, variables et contraintes;
- Solutions locales et globales.

## Modélisation

- Objectif, variables et contraintes;
- Solutions locales et globales.

## Outils d'analyse

- Dérivées et développements de Taylor;
- Conditions d'optimalité.



## Modélisation

- Objectif, variables et contraintes;
- Solutions locales et globales.

## Outils d'analyse

- Dérivées et développements de Taylor;
- Conditions d'optimalité.

## Principe algorithmique

- Processus itératifs : trouver une suite de points qui conduit à la solution;
- Étude de la vitesse de convergence.

- 1 Comment s'écrit la condition d'optimalité à l'ordre 1 pour la minimisation d'une fonction de classe  $\mathcal{C}^1$  ?
- 2 Quel objet caractérise la dérivée à l'ordre 2 d'une fonction de classe  $\mathcal{C}^2$  ?
- 3 Quelle est la différence entre un minimum local et un minimum global ?

- 1 Introduction
- 2 Concepts de base en optimisation
- 3 Optimisation sans contraintes
  - Moindres carrés linéaires
  - Méthode de descente de gradient

## Problème

minimiser  $f(\mathbf{w})$ .  
 $\mathbf{w} \in \mathbb{R}^d$

## Hypothèses

- $f$  minorée par  $f^*$ ;
- $f$  douce/lisse  $\Rightarrow$  les dérivées de  $f$  peuvent être utilisées pour résoudre ce problème.

## Moindres carrés linéaires

- Reposent sur de l'**algèbre linéaire**;
- Caractérisation explicite des solutions possible;
- Application orientée données : Régression linéaire.

## Optimisation douce sans contraintes

- Outil d'analyse : le gradient;
- But : Convergence itérative vers une solution;
- Application orientée données : Régression logistique.

- 1 Introduction
- 2 Concepts de base en optimisation
- 3 **Optimisation sans contraintes**
  - Moindres carrés linéaires
  - Méthode de descente de gradient

## Données

- Jeu de données à  $n$  éléments (individus, échantillons, etc);
- Chaque élément  $i$  est caractérisé par un vecteur  $\mathbf{x}_i \in \mathbb{R}^d$  d'attributs ainsi qu'un label  $y_i \in \mathbb{R}$ .

$$\Rightarrow \text{Matrice } \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \text{ et vecteur } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

## Données

- Jeu de données à  $n$  éléments (individus, échantillons, etc);
- Chaque élément  $i$  est caractérisé par un vecteur  $\mathbf{x}_i \in \mathbb{R}^d$  d'attributs ainsi qu'un label  $y_i \in \mathbb{R}$ .

$$\Rightarrow \text{Matrice } \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \text{ et vecteur } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

## But

On cherche un modèle linéaire  $h : \mathbf{x} \mapsto \mathbf{x}^T \mathbf{w}$  qui prédise correctement les  $y_i$  d'après les  $\mathbf{x}_i$ .

- Les modèles linéaires (dans le bon espace) sont souvent une bonne approximation;
- Utilisation d'algèbre linéaire (intérêt à la fois théorique et numérique).



## Prédiction idéale

- Un vecteur  $\mathbf{w} \in \mathbb{R}^d$  tel que  $\mathbf{x}_i^T \mathbf{w} = y_i$  pour tout  $i$ ;
- Ces  $n$  equations peuvent s'écrire sous la forme d'un système linéaire:  
 $\mathbf{X}\mathbf{w} = \mathbf{y}$ .

## Prédiction idéale

- Un vecteur  $\mathbf{w} \in \mathbb{R}^d$  tel que  $\mathbf{x}_i^T \mathbf{w} = y_i$  pour tout  $i$ ;
- Ces  $n$  equations peuvent s'écrire sous la forme d'un système linéaire:  
 $\mathbf{X}\mathbf{w} = \mathbf{y}$ .

## Résoudre des systèmes d'équations linéaires

- Une histoire d'algèbre linéaire;
- L'existence de solutions ne dépend que de  $\mathbf{X}$  et  $\mathbf{y}$ .

## A dataset

- $\mathbf{x}_1 = \mathbf{x}_2 = \dots = \mathbf{x}_n = 1$  ( $d = 1$ );
- $y_1, \dots, y_n$  sont distincts (typique de mesures bruitées).

# Juste de l'algèbre linéaire ?

## A dataset

- $\mathbf{x}_1 = \mathbf{x}_2 = \dots = \mathbf{x}_n = 1$  ( $d = 1$ );
- $y_1, \dots, y_n$  sont distincts (typique de mesures bruitées).

## Expliquer les données par un modèle linéaire

- On cherche  $\mathbf{w} = w \in \mathbb{R}$  tel que  $\mathbf{x}_i^T \mathbf{w} = x_i w = y_i \forall i$ ;
- Système linéaire :

$$\begin{cases} w = y_1 \\ w = y_2 \\ \vdots \\ w = y_n \end{cases}$$

# Juste de l'algèbre linéaire ?

## A dataset

- $\mathbf{x}_1 = \mathbf{x}_2 = \dots = \mathbf{x}_n = 1$  ( $d = 1$ );
- $y_1, \dots, y_n$  sont distincts (typique de mesures bruitées).

## Expliquer les données par un modèle linéaire

- On cherche  $\mathbf{w} = w \in \mathbb{R}$  tel que  $\mathbf{x}_i^T \mathbf{w} = x_i w = y_i \forall i$ ;
- Système linéaire :

$$\begin{cases} w = y_1 \\ w = y_2 \\ \vdots \\ w = y_n \end{cases}$$

- Ce système n'a pas de solution !
- En revanche, il existe une solution au problème de “coller aux données” (“data fitting”).

## Formulation du problème

Étant donné  $\{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq n}$  avec  $\mathbf{x}_i \in \mathbb{R}^d$ , on considère le problème d'optimisation :

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 = \frac{1}{2} (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}),$$

$$\text{avec } \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \text{ et } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n.$$

## Formulation du problème

Étant donné  $\{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq n}$  avec  $\mathbf{x}_i \in \mathbb{R}^d$ , on considère le problème d'optimisation :

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 = \frac{1}{2}(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}),$$

$$\text{avec } \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \text{ et } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n.$$

## Propriétés

- Problème sans contraintes;
- Fonction objectif minorée par 0;
- Objectif doux/lisse : polynomial en les coordonnées de  $\mathbf{w}$ .

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$



$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

- Si  $\mathbf{w}^*$  est une solution du système  $\mathbf{X}\mathbf{w} = \mathbf{y}$ , c'est aussi une solution du problème d'optimisation !

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

- Si  $\mathbf{w}^*$  est une solution du système  $\mathbf{X}\mathbf{w} = \mathbf{y}$ , c'est aussi une solution du problème d'optimisation !
- Quid du cas où le problème n'a pas de solution ?

## Système carré

$\mathbf{X} \mathbf{w} = \mathbf{y}$ , avec  $\mathbf{X} \in \mathbb{R}^{n \times d}$  et  $n = d$ .

## Cas 1 : $\mathbf{X}$ est inversible

$$\mathbf{X} \mathbf{w} = \mathbf{y} \Leftrightarrow \mathbf{w} = \mathbf{X}^{-1} \mathbf{y}.$$

Le système possède une unique solution  $\mathbf{w}^* = \mathbf{X}^{-1} \mathbf{y}$ , qui est aussi le minimum **global** du problème d'optimisation minimiser  $\mathbf{w} \in \mathbb{R}^d \frac{1}{2} \|\mathbf{X} \mathbf{w} - \mathbf{y}\|_2^2$ .

## Exemple avec $d = n = 2$

$$\begin{cases} w_1 + w_2 &= 0, \\ 3w_1 + 2w_2 &= 1. \end{cases}$$

L'unique solution est  $\mathbf{w} = [1 \ -1]^T$ .

## Système carré

$\mathbf{X} \mathbf{w} = \mathbf{y}$ , avec  $\mathbf{X} \in \mathbb{R}^{n \times d}$  et  $n = d$ .

## Cas 2: $\mathbf{X}$ non inversible

- Il peut ne pas y avoir de solution;
- Il peut en avoir une infinité.

Dans les deux cas, on peut calculer une solution au sens des moindres carrés!

## Système carré

$\mathbf{X} \mathbf{w} = \mathbf{y}$ , avec  $\mathbf{X} \in \mathbb{R}^{n \times d}$  et  $n = d$ .

## Cas 2: $\mathbf{X}$ non inversible

- Il peut ne pas y avoir de solution;
- Il peut en avoir une infinité.

Dans les deux cas, on peut calculer une solution au sens des moindres carrés!

## Autres cas

- $\mathbf{X} \mathbf{w} = \mathbf{y}$ , avec  $\mathbf{X} \in \mathbb{R}^{n \times d}$ ,  $n \neq d$ ;
- Pas de solution, une ou une infinité !

## Ce que l'on voudrait

- L'équivalent d'une inverse;
- Qui donne une solution qu'il en existe ou non.

## Ce que l'on voudrait

- L'équivalent d'une inverse;
- Qui donne une solution qu'il en existe ou non.

## Pseudo-inverse

Pour toute matrice  $\mathbf{X} \in \mathbb{R}^{n \times d}$ , il existe  $\mathbf{A} \in \mathbb{R}^{d \times n}$  qui vérifie les équations de Moore-Penrose

$$\begin{cases} \mathbf{AXA} = \mathbf{A} \\ \mathbf{XAX} = \mathbf{X} \end{cases} \quad \text{and} \quad \begin{cases} (\mathbf{AX})^T = \mathbf{AX} \\ (\mathbf{XA})^T = \mathbf{XA} \end{cases}$$

Cette matrice s'appelle la pseudo-inverse de  $\mathbf{X}$  et on la note  $\mathbf{A} = \mathbf{X}^\dagger$ .  
Si  $\mathbf{X}$  est inversible,  $\mathbf{X}^\dagger = \mathbf{X}^{-1}$ .

$$\mathbf{X} \in \mathbb{R}^{n \times d}, \quad \mathbf{y} \in \mathbb{R}^n.$$

## Théorème

Pour tout  $\mathbf{y} \in \mathbb{R}^n$ ,  $\mathbf{X}^\dagger \mathbf{y}$  est la solution du problème aux moindres carrés

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) = \frac{1}{2} \|\mathbf{X} \mathbf{w} - \mathbf{y}\|^2.$$

**de norme minimale.** Pour tout  $\hat{\mathbf{w}} \in \operatorname{argmin}_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ , on a :



$$\mathbf{X} \in \mathbb{R}^{n \times d}, \quad \mathbf{y} \in \mathbb{R}^n.$$

## Théorème

Pour tout  $\mathbf{y} \in \mathbb{R}^n$ ,  $\mathbf{X}^\dagger \mathbf{y}$  est la solution du problème aux moindres carrés

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) = \frac{1}{2} \|\mathbf{X} \mathbf{w} - \mathbf{y}\|^2.$$

**de norme minimale.** Pour tout  $\hat{\mathbf{w}} \in \operatorname{argmin}_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ , on a :

- $f(\mathbf{X}^\dagger \mathbf{y}) = f(\hat{\mathbf{w}})$ ;
- $\|\mathbf{X}^\dagger \mathbf{y}\| \leq \|\hat{\mathbf{w}}\|$ .

$$\mathbf{X}\mathbf{w} = \mathbf{y}, \quad \mathbf{X} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

## Une solution

- Le problème minimiser  $\mathbf{w} \in \mathbb{R}^d$   $\frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$  possède une infinité de solutions;
- Parmi celles-ci  $\mathbf{w}^* = \mathbf{X}^\dagger \mathbf{y}$  est de norme minimale;
- Cette solution est la moyenne  $\mathbf{w}^* = \frac{1}{n} \sum_{i=1}^n y_i!$

## Points clés

- Calcul de la pseudo-inverse;
- Ou son approximation!

## Utilisation de solveurs d'algèbre linéaire

- Décomposition de  $\mathbf{X}$  (QR, LU, SVD, etc) pour faciliter le calcul;
- Utilisation d'algèbre linéaire itératif (LSQR, LSLQ, etc) pour calculer une solution approchée, prise en compte des erreurs d'arrond;
- **Exécution dans des environnements parallèles/distribués.**

## Least-squares formulations

- Naturally arise in a plurality of fields that try to minimize the error between a **model** and some **data**  
*Ex) weather forecasting, statistics, economy.*
- Some problems can also be formulate or **reformulated** as linear least squares.

# Two applications among many more

## Least-squares formulations

- Naturally arise in a plurality of fields that try to minimize the error between a **model** and some **data**  
*Ex) weather forecasting, statistics, economy.*
- Some problems can also be formulate or **reformulated** as linear least squares.

## Two illustrations

- 1 Rewrite an optimization problem as a linear least-squares problem;
- 2 Form a linear least-squares formulation of a problem.

- Données :  $\{(\mathbf{x}_i, y_i)\}_i$ ,  $\mathbf{x}_i \in \mathbb{R}^d$ ,  $y_i \in \mathbb{R}$ .
- But : calculer un modèle linéaire  $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$  tel que  $h(\mathbf{x}_i) \approx y_i$  for  $i = 1, \dots, n$ .
- Objectif : minimiser les carrés des erreurs  $|h(\mathbf{x}_i) - y_i|$ .

- Données :  $\{(\mathbf{x}_i, y_i)\}_i$ ,  $\mathbf{x}_i \in \mathbb{R}^d$ ,  $y_i \in \mathbb{R}$ .
- But : calculer un modèle linéaire  $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$  tel que  $h(\mathbf{x}_i) \approx y_i$  for  $i = 1, \dots, n$ .
- Objectif : minimiser les carrés des erreurs  $|h(\mathbf{x}_i) - y_i|$ .

## Problème d'optimisation

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} \frac{1}{2n} \sum_{i=1}^n (h(\mathbf{x}_i) - y_i)^2 = \frac{1}{2n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

- On retrouve les moindres carrés linéaires !

## Qualité de la solution des moindres carrés

- Meilleure approximation possible en termes d'erreurs;
- Solution déterministe quand  $\{(\mathbf{x}_i, y_i)\}_i$  fixés.



## Qualité de la solution des moindres carrés

- Meilleure approximation possible en termes d'erreurs;
- Solution déterministe quand  $\{(\mathbf{x}_i, y_i)\}_i$  fixés.

## En présence de données bruitées

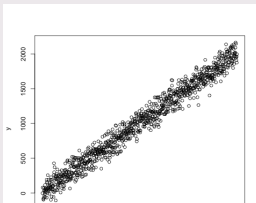
- Approche statistique : supposer que  $\mathbf{y} = \mathbf{X}\mathbf{w}^* + \epsilon$  avec  $\epsilon$  vecteur aléatoire de loi connue;
- Calculer l'estimateur du **maximum de vraisemblance** en résolvant un problème d'optimisation;
- Équivalent aux moindres carrés pour  $\epsilon$  gaussien.

## Qualité de la solution des moindres carrés

- Meilleure approximation possible en termes d'erreurs;
- Solution déterministe quand  $\{(\mathbf{x}_i, y_i)\}_i$  fixés.

## En présence de données bruitées

- Approche statistique : supposer que  $\mathbf{y} = \mathbf{X}\mathbf{w}^* + \epsilon$  avec  $\epsilon$  vecteur aléatoire de loi connue;
- Calculer l'estimateur du **maximum de vraisemblance** en résolvant un problème d'optimisation;
- Équivalent aux moindres carrés pour  $\epsilon$  gaussien.



## Buts

- Trouver une relation linéaire entre composantes d'un jeu de données;
- Fonctionne même en l'absence d'un modèle sous-jacent !

## Techniques

- Faire le lien avec les systèmes linéaires;
- Utilisation de solveurs d'algèbres linéaires pour le calcul explicite;
- Interprétation statistique possible, comme dans le contexte de la **régression linéaire**.

- 1 Introduction
- 2 Concepts de base en optimisation
- 3 **Optimisation sans contraintes**
  - Moindres carrés linéaires
  - **Méthode de descente de gradient**

minimiser  $f(\mathbf{w})$ .  
 $\mathbf{w} \in \mathbb{R}^d$

minimiser  $f(\mathbf{w})$ .  
 $\mathbf{w} \in \mathbb{R}^d$

## Hypothèses

- $f$  est minorée par  $f_{\text{low}}$ ;
- $f$  est lisse (au moins de classe  $\mathcal{C}^1$ ).

# Exemple : Nonlinear regression

## Contexte

- Jeu de données  $\{(\mathbf{x}_i, y_i)\}_i$ ,  $\mathbf{x}_i \in \mathbb{R}^d$ ,  $y_i \in \{0, 1\}$ ;
- But : Classification via un modèle linéaire  $\mathbf{x} \mapsto \mathbf{w}^T \mathbf{x}$ .

# Exemple : Nonlinear regression

## Contexte

- Jeu de données  $\{(\mathbf{x}_i, y_i)\}_i$ ,  $\mathbf{x}_i \in \mathbb{R}^d$ ,  $y_i \in \{0, 1\}$ ;
- But : Classification via un modèle linéaire  $\mathbf{x} \mapsto \mathbf{w}^T \mathbf{x}$ .

## Fonction objectif

- Basée sur la perte sigmoïde :  $\phi(\mathbf{x}_i; \mathbf{w}) = \left(1 + e^{-\mathbf{x}_i^T \mathbf{w}}\right)^{-1}$ ;
- On pénalise l'erreur  $(y_i - \phi(\mathbf{x}_i; \mathbf{w}))^2$ .



### Problème d'optimisation

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \left( y_i - \frac{1}{1 + e^{-\mathbf{x}_i^T \mathbf{w}}} \right)^2.$$

### Problème d'optimisation

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \left( y_i - \frac{1}{1 + e^{-\mathbf{x}_i^T \mathbf{w}}} \right)^2.$$

- Structure de moindres carrés **non linéaires**;
- Problème lisse : on applique la **descente de gradient**.

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}), \quad f \in \mathcal{C}^1.$$

Pour tout  $\mathbf{w} \in \mathbb{R}^d$ ,

- 1 Soit  $\mathbf{w}$  est un minimum local et donc  $\nabla f(\mathbf{w}) = 0$ ;
- 2 Soit  $f$  décroît **localement** depuis  $\mathbf{w}$  dans la direction de  $-\nabla f(\mathbf{w})$ .  
*Preuve basée sur Taylor.*

**Entrées :**  $\mathbf{w}_0 \in \mathbb{R}^d$ ,  $\alpha_0 > 0$ ,  $\varepsilon > 0$ ,  $k_{\max} \in \mathbb{N}$ .

Set  $k = 0$ .

- 1 Evaluer  $\nabla f(\mathbf{w}_k)$ ; si  $\|\nabla f(\mathbf{w}_k)\| < \varepsilon$  terminer.
- 2 Poser  $\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k \nabla f(\mathbf{w}_k)$ .
- 3 Incrémenter  $k$  de 1; si  $k = k_{\max}$  terminer, sinon aller à l'étape 1.

**Entrées :**  $\mathbf{w}_0 \in \mathbb{R}^d$ ,  $\alpha_0 > 0$ ,  $\varepsilon > 0$ ,  $k_{\max} \in \mathbb{N}$ .

Set  $k = 0$ .

- 1 Evaluer  $\nabla f(\mathbf{w}_k)$ ; si  $\|\nabla f(\mathbf{w}_k)\| < \varepsilon$  terminer.
- 2 Poser  $\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k \nabla f(\mathbf{w}_k)$ .
- 3 Incrémenter  $k$  de 1; si  $k = k_{\max}$  terminer, sinon aller à l'étape 1.

## Critères d'arrêt

- Convergence :  $\|\nabla f(\mathbf{w}_k)\| < \varepsilon$ ;
- Budget :  $k = k_{\max}$ .

## Pas constant

Si  $f \in \mathcal{C}_L^{1,1}$ , poser  $\alpha_k = \frac{1}{L}$ :

- Garantit une décroissance à chaque itération;
- Mais demande de connaître  $L$ .

## Pas décroissant

Choisir  $\alpha_k$  tel que  $\alpha_k \rightarrow 0$ .

- Garantit une décroissance à partir d'un certain rang;
- Mais force la valeur à décroître.

## En optimisation classique

- Recherche linéaire : À chaque itération,  $\alpha_k$  obtenue par retour arrière (*backtracking*) sur un ensemble de valeurs en ordre décroissants (ex:  $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \dots$ ).
- La valeur renvoyée vérifie une condition type décroissance de la valeur de l'objectif.

## En apprentissage (notamment profond)

$\alpha_k = \text{Learning rate}$

- Utiliser une valeur fixe pendant un certain nombre d'itérations;
- Diminuer progressivement cette valeur selon une règle fixée (*scheduling*).

## Exemple : Minimiser une fonction quadratique

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{A} \mathbf{w} - \mathbf{b}^T \mathbf{w}$$

- Pour tout  $\mathbf{w} \in \mathbb{R}^d$ ,  $\nabla f(\mathbf{w}) = \mathbf{A} \mathbf{w} - \mathbf{b}$ ;
  - Si  $\mathbf{w}$  est un minimum local,  $\nabla f(\mathbf{w}) = 0 \dots$
  - ...donc  $\mathbf{w}$  est une solution de  $\mathbf{A} \mathbf{w} = \mathbf{b}$ .
- 
- Descente de gradient : Algorithme itératif pour résoudre  $\mathbf{A} \mathbf{w} = \mathbf{b}$ ;
  - Bien défini même quand le problème n'a pas de solution (càd lorsque  $\mathbf{A} \not\prec 0$ ).



$$\min_{\mathbf{x} \in \mathbb{R}^d} f(\mathbf{x}), \quad f \in \mathcal{C}_L^{1,1}.$$

## Rappels : Descente de gradient

- Itération :  $\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k \nabla f(\mathbf{w}_k)$ , terminer si  $\nabla f(\mathbf{w}_k) = 0$ .
- Choix de base en théorie :  $\alpha_k = \frac{1}{L}$ .

## Résultats théoriques

- Convergence : Montrer que  $\|\nabla f(\mathbf{w}_k)\| \rightarrow 0$ ;
- Vitesse de convergence : Décroissance de  $\|\nabla f(\mathbf{w}_k)\|$ .
- Complexité au pire cas : Effort requis pour obtenir  $\|\nabla f(\mathbf{w}_k)\| \leq \epsilon$  pour  $\epsilon > 0$ .

## Théorème

Si  $f \in \mathcal{C}_L^{1,1}$  et  $\alpha_k = \frac{1}{L}$ , la descente de gradient produit  $\mathbf{w}_k$  tel que  $\|\nabla f(\mathbf{w}_k)\| \leq \epsilon$  en au plus

$$2L(f(\mathbf{w}_0) - f_{\text{low}})\epsilon^{-2} \text{ itérations.}$$

- Même résultat pour d'autres choix pour  $\alpha_k$ , dont la recherche linéaire.
- On dit que la complexité de la descente de gradient est en  $\mathcal{O}(\epsilon^{-2})$ .

## Théorème

Si  $f \in \mathcal{C}_L^{1,1}$  et  $\alpha_k = \frac{1}{L}$ , alors pour tout  $K \geq 1$ , si  $\{\mathbf{w}_k\}$  est la suite des itérés produite par l'algorithme de descente de gradient, on a

$$\min_{0 \leq k \leq K-1} \|\nabla f(\mathbf{w}_k)\| \leq \frac{\sqrt{2L(f(\mathbf{w}_0) - f_{\text{low}})}}{\sqrt{K}}.$$

## Interpretation

- On dit que la vitesse de convergence de la descente de gradient est  $\mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$ .
- Il existe une fonction telle que cette vitesse correspond exactement au comportement de la méthode !

- 1 Quel est l'avantage des moindres carrés linéaires sur les systèmes linéaires ?
- 2 Écrire l'itération de la descente de gradient.
- 3 Quelle est la différence entre une garantie de complexité, une vitesse de convergence et une garantie de convergence ?

## Optimisation

- Outil de modélisation;
- Outil de résolution de problèmes;
- Outil numérique!

## Optimisation en IA

- Algorithmes classique
- Problèmes structurés et spécifiques;
- Descente de gradient comme méthode de base;
- Autres méthodes (régularisation, second ordre) peuvent aussi être employées).

## Ouvrages:

- J. Nocedal et S. J. Wright, *Numerical Optimization, 2nd Ed.*, Springer-Verlag, 2006.  
⇒ Ouvrage de référence en optimisation non linéaire. See chapters 1 to 3.
- S. J. Wright, *Optimization Algorithms for Data Analysis*. In *The Mathematics of Data*, M. Mahoney, J. C. Duchi and A. Gilbert (eds), AMS, IAS/Park City Mathematics Institute and SIAM, 2018.  
⇒ Tutoriel avec analyse complète de la descente de gradient et d'autres méthodes.

## Codes :

- `scipy` : bibliothèque de base en calcul scientifique pour Python.
- MANOPT/PyMANOPT : codes MATLAB et Python.

## Notebook

- Quelques résultats sur la descente de gradient...
- ...et les moindres carrés.

**Cet après-midi :** Optimisation convexe.

## Notebook

- Quelques résultats sur la descente de gradient...
- ...et les moindres carrés.

**Cet après-midi :** Optimisation convexe.

**Merci beaucoup !**



## Problème lisse

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w})$$

où  $f$  est de classe  $\mathcal{C}^2$ .

Descente de gradient applicable mais :

- Pas invariante par transformation linéaire du problème;
- Pas d'utilisation de la dérivée seconde;
- Pas de vitesse de convergence pour un point stationnaire d'ordre deux.

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{d}_k, \quad \nabla^2 f(\mathbf{w}_k) \mathbf{d}_k = -\nabla f(\mathbf{w}_k).$$

- Algorithme dit de Newton appliqué à  $\nabla f(\mathbf{w}) = 0$ ;
- Pas de garanties autres qu'au voisinage d'une solution en général;
- Peut être modifié (notamment en introduisant un pas  $\alpha_k$ ) pour obtenir des garanties de complexité/des vitesses de convergence.

## Résultats récents

- Même complexité que la descente de gradient pour des problèmes  $\mathcal{C}^2$  génériques ( $\epsilon^{-2}$ );
- Bien meilleurs résultats en pratique et sous d'autres hypothèses (voir séance suivante).

## Problèmes sous contraintes

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser}} f(\mathbf{w}) \quad \text{s. c.} \quad \mathbf{w} \in \mathcal{F},$$

où  $\mathcal{F} \subset \mathbb{R}^d$  est l'ensemble des valeurs admissibles pour  $\mathbf{w}$ .

## Deux techniques modernes

- Utiliser la structure des contraintes;
- Régulariser plutôt que contraindre.

$$\underset{\mathbf{w} \in \mathbb{R}^d}{\text{minimiser } f(\mathbf{w})} \quad \text{s. c.} \quad \mathbf{w} \in \mathcal{F} := \left\{ \mathbf{w} \in \mathbb{R}^d \mid \mathbf{c}(\mathbf{w}) = 0 \right\},$$

où  $\mathbf{c} : \mathbb{R}^d \rightarrow \mathbb{R}^m$ .

- Lorsque  $\mathbf{c}$  est une fonction lisse/douce avec  $m \leq d$ , l'ensemble réalisable est typiquement une variété riemannienne de  $\mathbb{R}^d$  (qui ressemble à  $\mathbb{R}^{d-m}$ )  
*Ex*) Sphère  $\mathbf{c}(\mathbf{w}) = \|\mathbf{w}\|^2 - 1$ .
- On peut définir une notions de gradient sur des variétés  
 $g_f(\mathbf{w}) = (\mathbf{I}_d - \mathbf{w}\mathbf{w}^T)\nabla f(\mathbf{w})$ .
- De nombreux algorithmes se généralisent alors à ce contexte.