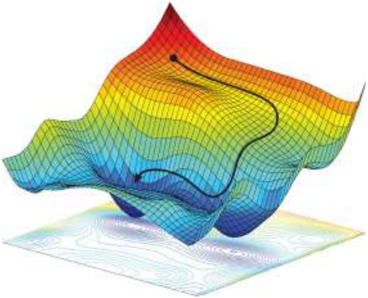
Optimisation convexe

Guillaume Tochon
Laboratoire de Recherche de l'EPITA



Quésaco ?

Optimisation (mathématique)

Optimisation (mathématique)

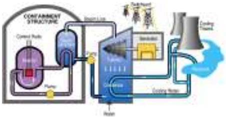
Recherche des conditions d'optimalité (*optimum* = le meilleur) d'un système décrit par un modèle (une fonction) mathématique dépendant de plusieurs paramètres/variables et produisant une sortie observable et quantifiable.

Optimisation (mathématique)

Recherche des conditions d'optimalité (optimum = le meilleur) d'un système décrit par un modèle (une fonction) mathématique dépendant de plusieurs paramètres/variables et produisant une sortie observable et quantifiable.

Tout problème dont le but est de *minimiser* un coût, une erreur, un délai, ... ou de *maximiser* un profit, un rendement, ..., est un problème d'optimisation.

→ omniprésent dans **toutes** les branches de l'ingénierie.




→ également fréquemment rencontré en finance.

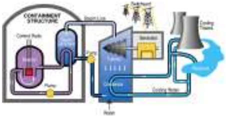
Quésaco ?

Optimisation (mathématique)

Recherche des conditions d'optimalité (*optimum* = le meilleur) d'un système décrit par un modèle (une fonction) mathématique dépendant de plusieurs paramètres/variables et produisant une sortie observable et quantifiable.

Tout problème dont le but est de *minimiser* un coût, une erreur, un délai, ... ou de *maximiser* un profit, un rendement, ..., est un problème d'optimisation.

→ omniprésent dans **toutes** les branches de l'ingénierie.

→ également fréquemment rencontré en finance.

L'optimisation, comme branche à part entière des mathématiques appliqués, se repose largement sur des notions d'algèbre linéaire, de géométrie et de calcul différentiel (calcul variationnel) .

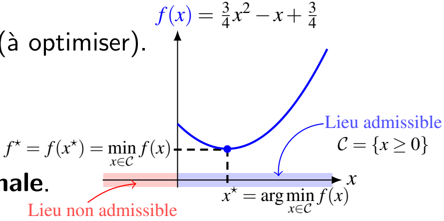
Formulation générale

Un problème d'optimisation s'écrit de manière standard sous la forme :

$$(\text{arg}) \min_{\mathbf{x} \in \mathcal{C}} f(\mathbf{x}) \qquad (\textsc{opt})$$

On appelle :

$\rightarrow$ $f : \begin{array}{ccc} \mathbf{R}^n & \rightarrow & \mathbf{R} \\ \mathbf{x} & \mapsto & f(\mathbf{x}) \end{array}$ la **fonction objective** (à optimiser).



$\rightarrow$ $\mathbf{x}^\star = \arg\min_{\mathbf{x} \in \mathcal{C}} f(\mathbf{x})$ le **point optimal**.

$\rightarrow$ $f^\star = f(\mathbf{x}^\star) = \min_{\mathbf{x} \in \mathcal{C}} f(\mathbf{x})$ la **valeur optimale**.

$\rightarrow$ $\mathcal{C} \subseteq \mathbb{R}^n$ le **lieu admissible**.

$\rightarrow$ défini comme l'intersection de contraintes d'inégalité et d'égalité :

$$\mathbf{x} \in \mathcal{C} \Leftrightarrow \begin{cases} g_i(\mathbf{x}) \leq 0 \ \forall i = 1, \ldots, m & \text{contraintes d'inégalité} \\ h_j(\mathbf{x}) = 0 \ \forall j = 1, \ldots, p & \text{contraintes d'égalité} \end{cases}$$

Bref historique de l'optimisation

Grèce antique : problèmes géométriques (Euclide, Zenorodus), principe du plus cours chemin optique (Héron).

XVIIe siècle : introduction du calcul différentiel (Newton, Leibniz, Fermat), calcul itératif des zéros d'une fonction (méthode de Newton).

XVIIIe siècle : calcul des variations (Euler, Lagrange), résolution des problèmes avec contraintes (méthode des multiplicateurs de Lagrange), applications en mécanique, mécanique céleste et transport optimal (Monge)

XIXe siècle : approfondissement du calcul des variations (Hamilton, Jacobi), méthode des moindres carrés (Legendre, Gauss), formulation de la descente de gradient (Cauchy), prémisses de l'optimisation linéaire (Fourier).

XXe siècle : développement théoriques (Jensen, Minkowski), programmation linéaire (Kantorovitch, Dantzig) et recherche opérationnelle (Von Neumann /) avec applications en logistique militaire et en économie. Démultiplication des algorithmes d'optimisation avec le développement de l'informatique.

Aujourd'hui : domaine de recherche toujours très actif avec de nombreux défis à relever, de nouveaux développements théoriques et améliorations de l'optimisation pratique des algorithmes existants pour s'adapter aux données à grande échelle.

FAQ de l'optimisation

Les questions qui reviennent lors de la résolution d'un problème d'optimisation :

- 🤔 Est-ce qu'il **existe** une solution ?
- 🤨 Si une solution existe, est-elle **unique** ?
- 🤯 Peut-on **décrire analytiquement** l'ensemble des solutions ?
- 💻 Peut-on **calculer numériquement** la ou les solutions ?
- 📏 Peut-on approcher la ou les solutions avec une **précision donnée** ?

À la fin du cours d'OCVX1, vous serez capable de répondre (au moins partiellement) à toutes ces questions.

FAQ de l'optimisation

Les questions qui reviennent lors de la résolution d'un problème d'optimisation :

- 🤔 Est-ce qu'il **existe** une solution ?
- 🤨 Si une solution existe, est-elle **unique** ?
- 🤯 Peut-on **décrire analytiquement** l'ensemble des solutions ?
- 💻 Peut-on **calculer numériquement** la ou les solutions ?
- 📏 Peut-on approcher la ou les solutions avec une **précision donnée** ?

À la fin du cours d'OCVX1, vous serez capable de répondre (au moins partiellement) à toutes ces questions.

Formulez un problème d'optimisation qui :

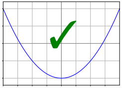
- a un lieu admissible vide.
- admet plusieurs points optimaux.
- n'a pas de valeur optimale mais a un lieu admissible non vide.
- a une valeur optimale mais pas de point optimal.

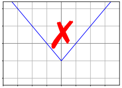
Cadre d'OCVX1

On dit qu'un problème d'optimisation (OPT) est :

différentiable si toutes les fonctions (objective et contraintes) en jeux le sont.

→ propriété de régularité *locale* des fonctions.
→ permet d'introduire des outils mathématiques puissants pour l'étude et la résolution de (OPT).

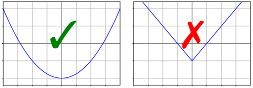




On dit qu'un problème d'optimisation (OPT) est :

différentiable si toutes les fonctions (objective et contraintes) en jeux le sont.

→ propriété de régularité *locale* des fonctions.
→ permet d'introduire des outils mathématiques puissants pour l'étude et la résolution de (OPT).



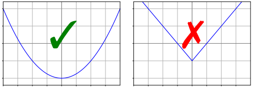
convexe si toutes les fonctions en jeux le sont (les contraintes d'égalité étant de plus affines).

Cadre d'OCVX1

On dit qu'un problème d'optimisation $(\textsc{opt})$ est :

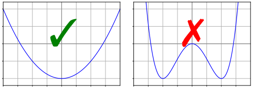
différentiable si toutes les fonctions (objective et contraintes) en jeux le sont.

$\rightarrow$ propriété de régularité *locale* des fonctions.
$\rightarrow$ permet d'introduire des outils mathématiques puissants pour l'étude et la résolution de $(\textsc{opt})$.



convexe si toutes les fonctions en jeux le sont (les contraintes d'égalité étant de plus affines).

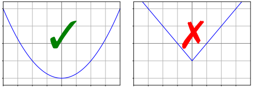
$\rightarrow$ propriété structurelle *globale* des fonctions.
$\rightarrow$ permet de garantir que la solution trouvée de $(\textsc{opt})$ est optimale.



On dit qu'un problème d'optimisation (OPT) est :

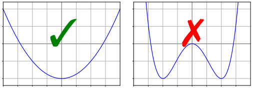
différentiable si toutes les fonctions (objective et contraintes) en jeux le sont.

$\rightarrow$ propriété de régularité *locale* des fonctions.
$\rightarrow$ permet d'introduire des outils mathématiques puissants pour l'étude et la résolution de (OPT).



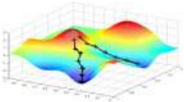
convexe si toutes les fonctions en jeux le sont (les contraintes d'égalité étant de plus affines).

$\rightarrow$ propriété structurelle *globale* des fonctions.
$\rightarrow$ permet de garantir que la solution trouvée de (OPT) est optimale.



sans contrainte s'il n'y a aucune contrainte en jeux ($\mathcal{C} = \mathbb{R}^n$).

$\rightarrow$ permet de construire des algorithmes itératifs se basant sur des propriétés géométriques "simples" pour la résolution de (OPT).

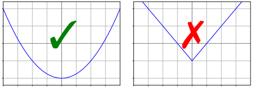


Cadre d'OCVX1

On dit qu'un problème d'optimisation (OPT) est :

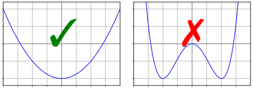
différentiable si toutes les fonctions (objective et contraintes) en jeux le sont.

$\rightarrow$ propriété de régularité *locale* des fonctions.
$\rightarrow$ permet d'introduire des outils mathématiques puissants pour l'étude et la résolution de (OPT).



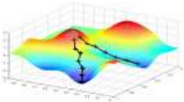
convexe si toutes les fonctions en jeux le sont (les contraintes d'égalité étant de plus affines).

$\rightarrow$ propriété structurelle *globale* des fonctions.
$\rightarrow$ permet de garantir que la solution trouvée de (OPT) est optimale.



sans contrainte s'il n'y a aucune contrainte en jeux ($\mathcal{C} = \mathbb{R}^n$).

$\rightarrow$ permet de construire des algorithmes itératifs se basant sur des propriétés géométriques "simples" pour la résolution de (OPT).



Cadre d'étude

Dans OCVX1, les problèmes d'optimisation abordés seront :

$\rightarrow$ **toujours** sans contraintes (on se revoit en OCVX2 pour les contraintes).
$\rightarrow$ **très souvent** convexes (on se revoit en TP pour comparaison convexe/non convexe).
$\rightarrow$ **toujours** différentiables (il y a déjà suffisamment à raconter sur le sujet).

Formulation du problème

Étant données n points et les $\frac{n(n-1)}{2}$ distances entre chaque paires de points, trouver le plus court chemin qui passe exactement une fois par chaque point et revienne au point de départ.

Formulation du problème

Étant données n points et les $\frac{n(n-1)}{2}$ distances entre chaque paires de points, trouver le plus court chemin qui passe exactement une fois par chaque point et revienne au point de départ.

Problème général NP-complet (complexité $\mathcal{O}(n!)$) :

$\rightarrow$ Un des problèmes les plus étudiés du 20^e siècle (bien que son origine remonte au 19^e siècle).

$\rightarrow$ Applications directes en logistique et planification, ainsi que des domaines plus éloignés tels que la génétique ou le design de circuits intégrés.

$\rightarrow$ Existence de nombreuses heuristiques de résolution.

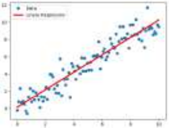
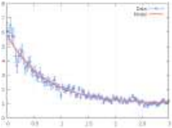
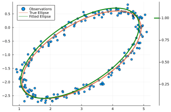
$\rightarrow$ Schéma d'approximation en temps polynomial (prix Gödel 2010) grâce à une formulation comme programme d'optimisation linéaire en nombres entiers.

# villes	# chemins candidats
3	1
4	3
5	12
6	60
7	360
8	2 520
9	20 160
10	181 440
15	4.359×10^{10}
20	6.082×10^{16}
71	5.989×10^{99}

Formulation du problème

Étant données n points $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ ($\mathbf{x}_i$ variables, y_i observations) et un modèle $f(\cdot, \boldsymbol{\alpha})$ paramétré par un vecteur de paramètres $\boldsymbol{\alpha}$, trouver les paramètres optimaux qui minimisent le carré des résidus du modèle :

$$\boldsymbol{\alpha}^\star = \arg\min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \big(y_i - f(\mathbf{x}_i, \boldsymbol{\alpha})\big)^2$$

Méthode utilisée (mais non publiée) par Gauss en 1801 pour la prédiction de l'orbite de Cérès.
$\rightarrow$ dispute avec Legendre qui l'a publié en 1805.



Lorsque $f(\mathbf{x}, \boldsymbol{\alpha}) = \alpha_0 + \alpha_1 x_1 + \cdots + \alpha_p x_p \Rightarrow$ régression linéaire.

Formulation du problème

Étant données n points d'entraînement $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, l'apprentissage supervisé d'un modèle $f(\,\cdot\,, \boldsymbol{\alpha})$ cherche le vecteur de paramètres $\boldsymbol{\alpha}$ qui prédit au mieux les sorties y_i en fonctions des entrées $\mathbf{x}_i \to y_i \simeq f(\mathbf{x}_i, \boldsymbol{\alpha})$

Entraîner le modèle pour une fonction de coût $\mathcal{L} \Leftrightarrow$ minimisation des erreurs de prédictions :

$$\boldsymbol{\alpha}^\star = \arg\min_{\boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big)$$

Formulation du problème

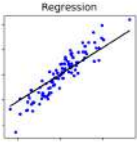
Étant données n points d'entraînement $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, l'apprentissage supervisé d'un modèle $f(\cdot, \boldsymbol{\alpha})$ cherche le vecteur de paramètres $\boldsymbol{\alpha}$ qui prédit au mieux les sorties y_i en fonctions des entrées $\mathbf{x}_i \rightarrow y_i \simeq f(\mathbf{x}_i, \boldsymbol{\alpha})$

Entraîner le modèle pour une fonction de coût $\mathcal{L} \Leftrightarrow$ minimisation des erreurs de prédictions :

$$\boldsymbol{\alpha}^\star = \underset{\boldsymbol{\alpha}}{\arg\min}\ \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big)$$

On parle de

- Régression lorsque $y \in \mathbb{R}$ (valeur quantitative)
  $\rightarrow$ Fonction de coût classique : *erreur quadratique moyenne*
  $$\mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big) = \big(y_i - f(\mathbf{x}_i, \boldsymbol{\alpha})\big)^2$$

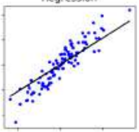


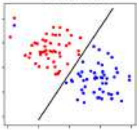
Formulation du problème

Étant données n points d'entraînement $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, l'apprentissage supervisé d'un modèle $f(\cdot, \boldsymbol{\alpha})$ cherche le vecteur de paramètres $\boldsymbol{\alpha}$ qui prédit au mieux les sorties y_i en fonctions des entrées $\mathbf{x}_i \to y_i \simeq f(\mathbf{x}_i, \boldsymbol{\alpha})$

Entraîner le modèle pour une fonction de coût $\mathcal{L} \Leftrightarrow$ minimisation des erreurs de prédictions :

$$\boldsymbol{\alpha}^\star = \arg\min_{\boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big)$$





On parle de

- Régression lorsque $y \in \mathbb{R}$ (valeur quantitative)
  $\to$ Fonction de coût classique : *erreur quadratique moyenne*
  $$\mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big) = \big(y_i - f(\mathbf{x}_i, \boldsymbol{\alpha})\big)^2$$
- Classification lorsque $y \in \{\textcolor{blue}{\bullet}, \textcolor{red}{\bullet}\}$ (valeur qualitative $\equiv$ classe)
  $\to$ Fonction de coût classique : *binary cross-entropy*
  $$\mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big) = -\left[\big(y_i \log(f(\mathbf{x}_i, \boldsymbol{\alpha}))\big) + \big((1 - y_i) \log(1 - f(\mathbf{x}_i, \boldsymbol{\alpha}))\big)\right]$$

Exemples de problèmes d'optimisation dans la vraie vie

Le problème de l'apprentissage supervisé

Formulation du problème

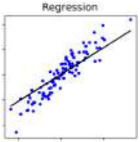
Étant données n points d'entraînement $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, l'apprentissage supervisé d'un modèle $f(\,\cdot\,, \boldsymbol{\alpha})$ cherche le vecteur de paramètres $\boldsymbol{\alpha}$ qui prédit au mieux les sorties y_i en fonctions des entrées $\mathbf{x}_i \to y_i \simeq f(\mathbf{x}_i, \boldsymbol{\alpha})$

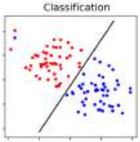
Entraîner le modèle pour une fonction de coût $\mathcal{L} \Leftrightarrow$ minimisation des erreurs de prédictions :

$$\boldsymbol{\alpha}^\star = \arg\min_{\boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big)$$

On parle de

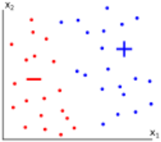
- Régression lorsque $y \in \mathbb{R}$ (valeur quantitative)
  $\to$ Fonction de coût classique : *erreur quadratique moyenne*
  $$\mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big) = \big(y_i - f(\mathbf{x}_i, \boldsymbol{\alpha})\big)^2$$
- Classification lorsque $y \in \{\color{blue}{\bullet}, \color{red}{\bullet}\}$ (valeur qualitative $\equiv$ classe)
  $\to$ Fonction de coût classique : *binary cross-entropy*
  $$\mathcal{L}\big(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha})\big) = -\Big[\big(y_i \log(f(\mathbf{x}_i, \boldsymbol{\alpha}))\big) + \big((1 - y_i) \log(1 - f(\mathbf{x}_i, \boldsymbol{\alpha}))\big)\Big]$$



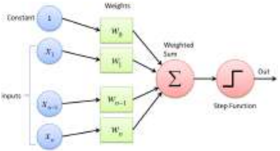


Une fois entraîné, le modèle est capable de prédire une nouvelle sortie en fonction d'une nouvelle entrée : $y_{new} = f(\mathbf{x}_{new}, \boldsymbol{\alpha})$.

Formulation du problème



Trouver une séparatrice linéaire (hyperplan) qui sépare parfaitement les échantillons de la classe • $(+1)$ et ceux de la classe • (-1)

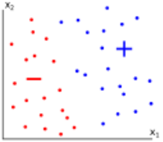


Modèle du perceptron : $(w_0, \mathbf{w}) \in \mathbb{R}^{n+1}$

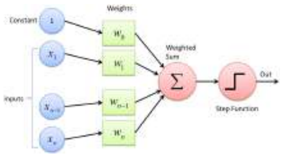
Entrée : $\mathbf{x} \in \mathbb{R}^n$

Sortie : $f(\mathbf{x}, (w_0, \mathbf{w})) = \begin{cases} 1 & \text{si } w_0 + \sum_{i=1}^n w_i x_i > 0, \\ -1 & \text{sinon} \end{cases}$

Formulation du problème



Trouver une séparatrice linéaire (hyperplan) qui sépare parfaitement les échantillons de la classe • $(+1)$ et ceux de la classe • (-1)



Modèle du perceptron : $(w_0, \mathbf{w}) \in \mathbb{R}^{n+1}$

Entrée : $\mathbf{x} \in \mathbb{R}^n$

Sortie : $f(\mathbf{x}, (w_0, \mathbf{w})) = \begin{cases} 1 & \text{si } w_0 + \sum_{i=1}^n w_i x_i > 0, \\ -1 & \text{sinon} \end{cases}$

Règle de mise à jour du perceptron :

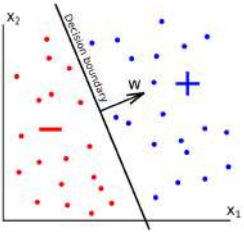
$\rightarrow$ Itérer sur tous les points $\mathbf{x}_i$ jusqu'à convergence

si $y_i(w_0 + \mathbf{w}^T \mathbf{x}_i) \leq 0$:

$$\begin{cases} \mathbf{w} & \leftarrow \mathbf{w} + y_i \mathbf{x}_i \\ w_0 & \leftarrow w_0 + y_i \end{cases}$$ Règle de Rosenblatt

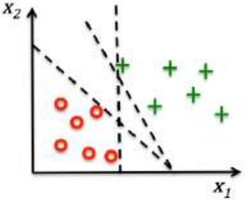
ou

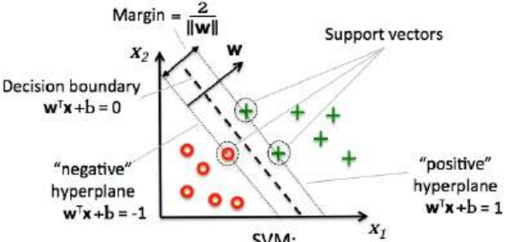
$$\begin{cases} \mathbf{w} & \leftarrow \mathbf{w} - \eta(\mathbf{w}^T \mathbf{x}_i - y_i)\mathbf{x}_i \\ w_0 & \leftarrow w_0 - \eta(w_0 - y_i) \end{cases}$$ Règle de Widrow-Hoff



Formulation du problème

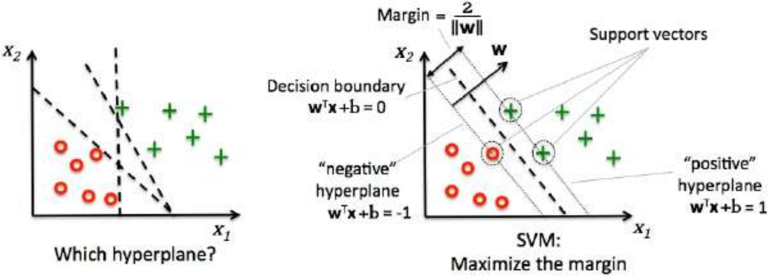
Trouver la séparatrice linéaire $(\mathbf{w}, b) \in \mathbb{R}^n \times \mathbb{R}$ (hyperplan) qui sépare parfaitement les échantillons de la classe $+$ $(+1)$ et ceux de la classe $\circ$ (-1) en maximisant la marge





Formulation du problème

Trouver la séparatrice linéaire $(\mathbf{w}, b) \in \mathbb{R}^n \times \mathbb{R}$ (hyperplan) qui sépare parfaitement les échantillons de la classe $+$ $(+1)$ et ceux de la classe $\circ$ (-1) en maximisant la marge

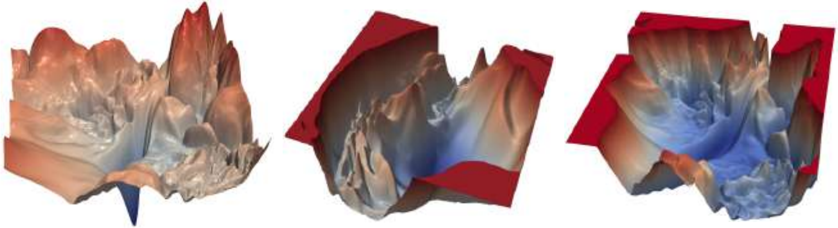


$\Rightarrow$ Trouver $(\mathbf{w}, b) \in \mathbb{R}^n \times \mathbb{R}$ de marge maximale, tel que $\begin{cases} \mathbf{w}^T \mathbf{x}_i + b \geq 1 & \text{if } y_i = 1 \\ \mathbf{w}^T \mathbf{x}_i + b \leq -1 & \text{if } y_i = -1 \end{cases}$

$\Rightarrow$ Minimiser $\frac{1}{2}\|\mathbf{w}\|^2$ sous contrainte que $y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1 \ \forall i = p, \ldots, n$

Formulation du problème

Étant donné une architecture de réseau profond avec des millions de poids à optimiser et des hyperparamètres à la pelle, comment s'assurer que l'apprentissage converge bien vers le minimum global ?



ResNet-56 | ResNet-110 | VGG

Visualisation de la fonction de coût de différentes architectures de réseaux profonds
Source : https://arxiv.org/pdf/1712.09913.pdf

Formulation du problème

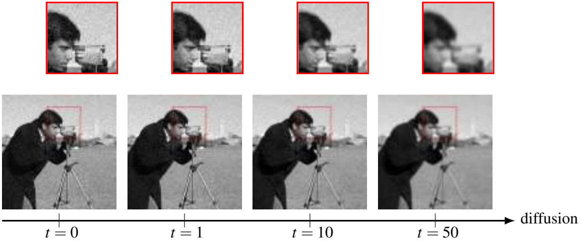
L'image $\mathcal{I}$ s'apparente à un champ de température ($\blacksquare$ = froid, $\square$ = chaud) qui se diffuse au cours du temps selon l'équation de la chaleur, et où les gradients de l'image font barrière au processus de diffusion.

Équation de la chaleur :

$$\frac{\partial \mathcal{I}}{\partial t} = \nabla\big(c(\|\nabla \mathcal{I}\|)\nabla \mathcal{I}\big)$$

Coefficient de diffusion :

$$c\left(\|\nabla I\|\right) = e^{-(\|\nabla I\|/K)^2}$$



La résolution de l'équation de la chaleur est équivalente à minimiser l'énergie $E_{\mathcal{I}}$:

$$E_{\mathcal{I}} = \frac{1}{2}\int g\big(\|\nabla I(x)\|^2\big)dx \text{ avec } c = g' \rightarrow \frac{\partial \mathcal{I}}{\partial t} = -\nabla E_{\mathcal{I}}$$

Exemples de problèmes d'optimisation dans la vraie vie

Les contours actifs (*snake model*)

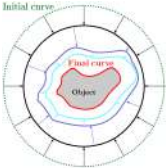
Formulation du problème

Modèle d'une courbe 2D élastique $u(s) = (x(s), y(s))$ (avec x, y coordonnées et $s \in [0, 1]$ paramétrisation de la courbe) qui évolue dynamiquement en fonction de forces internes et externes jusqu'à une situation d'équilibre.

→ Les forces internes prennent en compte l'élasticité et la courbure du contour : $E_{interne} = \displaystyle\int_0^1 \left(\alpha \left| \frac{\partial u}{\partial s}(s) \right|^2 + \beta \left| \frac{\partial^2 u}{\partial s^2}(s) \right| \right) ds$

→ Les forces externes dépendent du gradient de l'image : $E_{externe} = \displaystyle\int_0^1 \|\nabla \mathcal{I}(u(s))\|^2 ds$

→ Énergie totale du contour à minimiser : $E_{snake} = E_{interne} + E_{externe}$

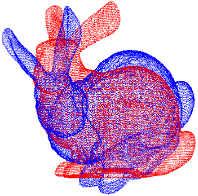


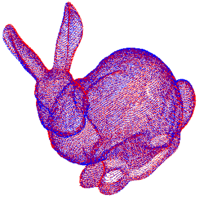
Formulation du problème

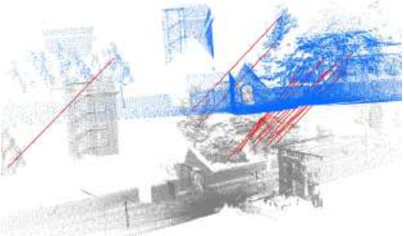
Comment mettre en correspondance deux nuages de points $\mathcal{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_N\}$ et $\mathcal{Q} = \{\mathbf{q}_1, \dots, \mathbf{q}_M\}$ de $\mathbb{R}^d$ qui correspondent en général à deux vues partielles d'un même objet.

$\Rightarrow$ Recherche de la translation $\mathbf{t} \in \mathbb{R}^d$ et la rotation $\mathbf{R} \in SO(d)$ qui minimisent l'erreur de reprojection :

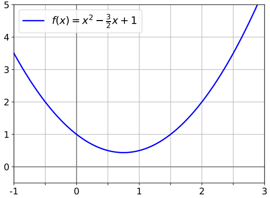
$$\operatorname*{arg\,min}_{\mathbf{t}\in\mathbb{R}^d, \mathbf{R}\in SO(d)} \sum_{i=1}^{N} d_i\big(\mathbf{R}, \mathbf{t}\big)^2 \text{ avec } d_i(\mathbf{R}, \mathbf{t}) = \min_{\mathbf{q}\in\mathcal{Q}} \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}\|$$



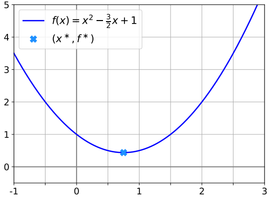




Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^2 - \frac{3}{2}x + 1$?



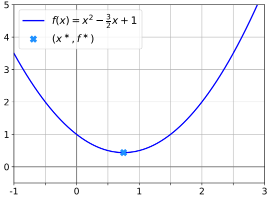
Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^2 - \frac{3}{2}x + 1$?



1. Calcul de la dérivée :
   $f(x) = x^2 - \frac{3}{2}x + 1 \to f'(x) = 2x - \frac{3}{2}$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x = \frac{3}{4}$

$\Rightarrow$ une solution optimale $x^\star = \frac{3}{4}$ et $f^\star = f(x^\star) = \frac{7}{16}$

Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^2 - \frac{3}{2}x + 1$?

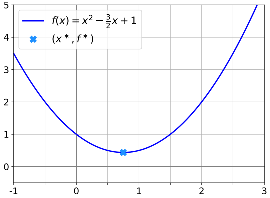


1. Calcul de la dérivée :
   $f(x) = x^2 - \frac{3}{2}x + 1 \to f'(x) = 2x - \frac{3}{2}$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x = \frac{3}{4}$

$\Rightarrow$ une solution optimale $x^\star = \frac{3}{4}$ et $f^\star = f(x^\star) = \frac{7}{16}$

Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^4 - 2x^2$?

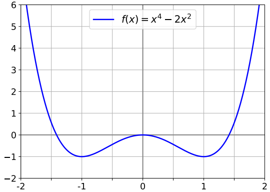
Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^2 - \frac{3}{2}x + 1$?



1. Calcul de la dérivée :
   $f(x) = x^2 - \frac{3}{2}x + 1 \to f'(x) = 2x - \frac{3}{2}$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x = \frac{3}{4}$

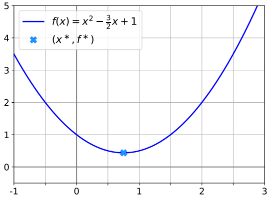
$\Rightarrow$ une solution optimale $x^\star = \frac{3}{4}$ et $f^\star = f(x^\star) = \frac{7}{16}$

Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^4 - 2x^2$?



1. Calcul de la dérivée :
   $f(x) = x^4 - 2x^2 \to f'(x) = 4x^3 - 4x = 4x(x^2 - 1)$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x \in \{-1, 0, 1\}$

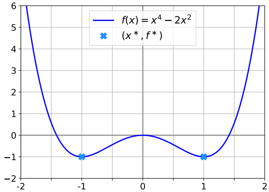
Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^2 - \frac{3}{2}x + 1$?



1. Calcul de la dérivée :
   $f(x) = x^2 - \frac{3}{2}x + 1 \to f'(x) = 2x - \frac{3}{2}$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x = \frac{3}{4}$

$\Rightarrow$ une solution optimale $x^\star = \frac{3}{4}$ et $f^\star = f(x^\star) = \frac{7}{16}$

Quel est le point optimal $x^\star$ et la valeur optimale $f^\star$ de $f : x \mapsto x^4 - 2x^2$?



1. Calcul de la dérivée :
   $f(x) = x^4 - 2x^2 \to f'(x) = 4x^3 - 4x = 4x(x^2 - 1)$
2. Recherche des zéros de la dérivée :
   $f'(x) = 0 \Leftrightarrow x \in \{-1, 0, 1\}$

$\Rightarrow$ deux solutions optimales $x^\star = \pm 1$ et $f^\star = f(x^\star) = -1$

Pour $f : \mathbb{R} \to \mathbb{R}$, les exemples précédents montrent qu'on arrive à s'en sortir si on a accès à la dérivée f' puisqu'il "suffit" d'identifier ses zéros $\{x \in \mathbb{R} \text{ tq } f'(x) = 0\}$.

Mais cette stratégie est bien fragile :

- $\rightarrow$ Comment faire si f n'est pas dérivable ?
- $\rightarrow$ Comment faire si f' a une infinité de zéros ? (exemple : $f(x) = x^2 + cos(2\pi x)$)
- $\rightarrow$ Comment faire si les zéros de f' n'ont pas de solution analytique simple ?

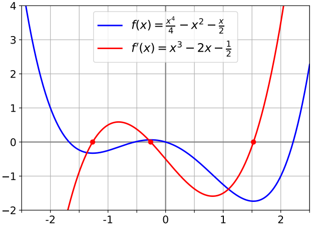
Pour $f : \mathbb{R} \to \mathbb{R}$, les exemples précédents montrent qu'on arrive à s'en sortir si on a accès à la dérivée f' puisqu'il "suffit" d'identifier ses zéros $\{x \in \mathbb{R} \text{ tq } f'(x) = 0\}$.

Mais cette stratégie est bien fragile :

- $\rightarrow$ Comment faire si f n'est pas dérivable ?
- $\rightarrow$ Comment faire si f' a une infinité de zéros ? (exemple : $f(x) = x^2 + cos(2\pi x)$)
- $\rightarrow$ Comment faire si les zéros de f' n'ont pas de solution analytique simple ?

Exemple : $f(x) = \frac{x^4}{4} - x^2 - \frac{x}{2}$

$\rightarrow$ $f'(x) = x^3 - 2x - \frac{1}{2}$



Comment optimiser

Le cas un peu moins facile avec $f : \mathbb{R} \to \mathbb{R}$

Pour $f : \mathbb{R} \to \mathbb{R}$, les exemples précédents montrent qu'on arrive à s'en sortir si on a accès à la dérivée f' puisqu'il "suffit" d'identifier ses zéros $\{x \in \mathbb{R} \text{ tq } f'(x) = 0\}$.

Mais cette stratégie est bien fragile :

$\rightarrow$ Comment faire si f n'est pas dérivable ?

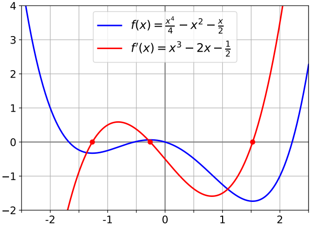
$\rightarrow$ Comment faire si f' a une infinité de zéros ? (exemple : $f(x) = x^2 + cos(2\pi x)$)

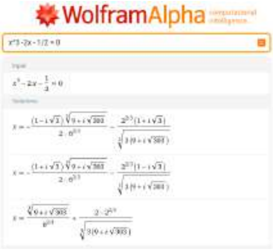
$\rightarrow$ Comment faire si les zéros de f' n'ont pas de solution analytique simple ?

Exemple : $f(x) = \frac{x^4}{4} - x^2 - \frac{x}{2}$

$\rightarrow$ $f'(x) = x^3 - 2x - \frac{1}{2}$

$\rightarrow$ $f'(x) = 0 \Leftrightarrow$ 🤯





Comment optimiser

Le cas un peu moins facile avec $f : \mathbb{R} \to \mathbb{R}$

Pour $f : \mathbb{R} \to \mathbb{R}$, les exemples précédents montrent qu'on arrive à s'en sortir si on a accès à la dérivée f' puisqu'il “suffit” d'identifier ses zéros $\{x \in \mathbb{R} \text{ tq } f'(x) = 0\}$.

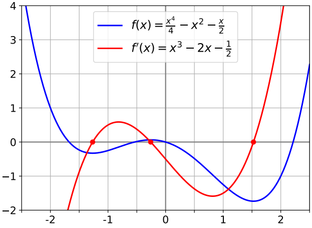
Mais cette stratégie est bien fragile :

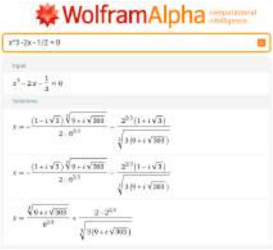
- $\rightarrow$ Comment faire si f n'est pas dérivable ?
- $\rightarrow$ Comment faire si f' a une infinité de zéros ? (exemple : $f(x) = x^2 + cos(2\pi x)$)
- $\rightarrow$ Comment faire si les zéros de f' n'ont pas de solution analytique simple ?

Exemple : $f(x) = \frac{x^4}{4} - x^2 - \frac{x}{2}$

$\rightarrow$ $f'(x) = x^3 - 2x - \frac{1}{2}$

$\rightarrow$ $f'(x) = 0 \Leftrightarrow$ 





$\Rightarrow$ Résolution du problème d'optimisation par une méthode itérative

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \ldots, x_n, \ldots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \ldots, x_n, \ldots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

$\rightarrow$ Comment choisir le point initial x_0 ?

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \dots, x_n, \dots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

$\rightarrow$ Comment choisir le point initial x_0 ?

$\rightarrow$ Comment choisir le nouvel itéré x_{k+1} ?

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \dots, x_n, \dots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

$\rightarrow$ Comment choisir le point initial x_0 ?

$\rightarrow$ Comment choisir le nouvel itéré x_{k+1} ?

$\rightarrow$ Est-ce que la méthode converge ?

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \ldots, x_n, \ldots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

- $\rightarrow$ Comment choisir le point initial x_0 ?
- $\rightarrow$ Comment choisir le nouvel itéré x_{k+1} ?
- $\rightarrow$ Est-ce que la méthode converge ?
- $\rightarrow$ Est-ce que la méthode converge vers le minimum global ?

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \ldots, x_n, \ldots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

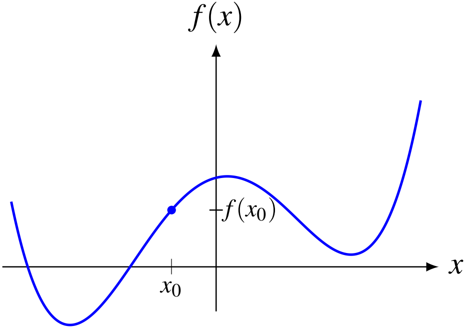
- → Comment choisir le point initial x_0 ?
- → Comment choisir le nouvel itéré x_{k+1} ?
- → Est-ce que la méthode converge ?
- → Est-ce que la méthode converge vers le minimum global ?
- → Comment implémenter efficacement en pratique cette méthode ?
  - → d'un point de vue mémoire
  - → d'un point de vue temps de calcul
  - → d'un point de vue stabilité/précision des résultats

Étant donné un point initial x_0, le but d'une méthode d'optimisation itérative est de construire une suite de points $x_1, x_2, \ldots, x_n, \ldots$ telle que $\forall k,\ f(x_{k+1}) \leq f(x_k)$.

Évidemment, une telle stratégie vient avec son lot de questions :

- $\rightarrow$ Comment choisir le point initial x_0 ?
- $\rightarrow$ Comment choisir le nouvel itéré x_{k+1} ?
- $\rightarrow$ Est-ce que la méthode converge ?
- $\rightarrow$ Est-ce que la méthode converge vers le minimum global ?
- $\rightarrow$ Comment implémenter efficacement en pratique cette méthode ?
  - $\rightarrow$ d'un point de vue mémoire
  - $\rightarrow$ d'un point de vue temps de calcul
  - $\rightarrow$ d'un point de vue stabilité/précision des résultats

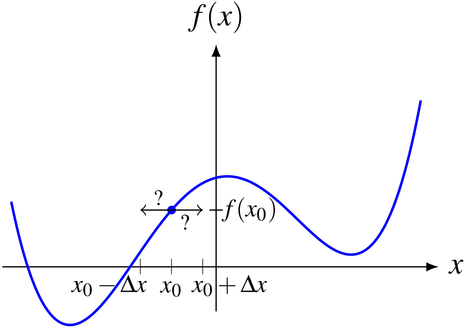
L'objectif d'OCVX1 est de répondre (au moins partiellement) à ces questions.



Étant donné un point initial x_0

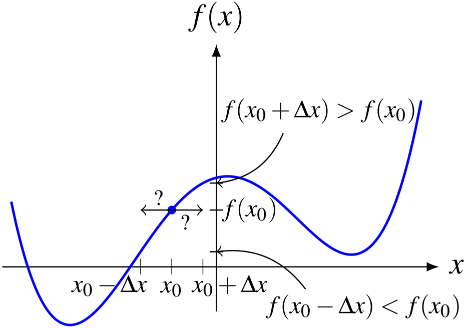
Comment décider dans quelle direction partir (⬅️ ou ➡️) pour minimiser f ?

Exemple pratique de minimisation itérative



Étant donné un point initial x_0

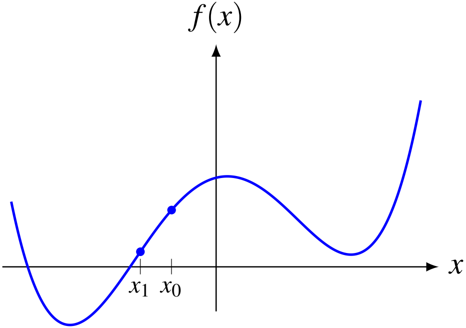
On se donne un pas Δx :
$x_0 + \Delta x$ pour le prochain itéré si ➡️
$x_0 - \Delta x$ pour le prochain itéré si ⬅️



Étant donné un point initial x_0

On se donne un pas Δx :
$x_0 + \Delta x$ pour le prochain itéré si ➡️
$x_0 - \Delta x$ pour le prochain itéré si ⬅️

SI $f(x_0 + \Delta x) < f(x_0)$
➡️ : $x_1 \leftarrow x_0 + \Delta x$
SINON
⬅️ : $x_1 \leftarrow x_0 - \Delta x$



Étant donné un point initial x_0

On recommence cette procédure sur le nouvel itéré :

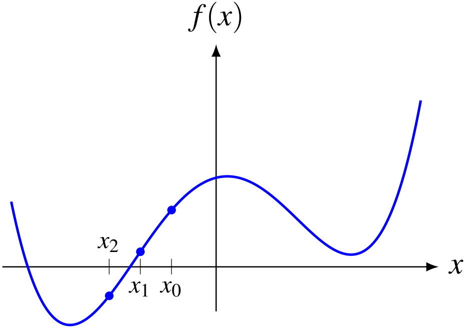
RÉPÉTER jusqu'à convergence

SI $f(x_k + \Delta x) < f(x_k)$

➡️ : $x_{k+1} \leftarrow x_k + \Delta x$

SINON

⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$



Étant donné un point initial x_0

On recommence cette procédure sur le nouvel itéré :

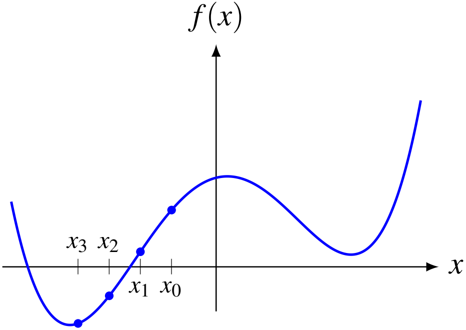
RÉPÉTER jusqu'à convergence

SI $f(x_k + \Delta x) < f(x_k)$

➡️ : $x_{k+1} \leftarrow x_k + \Delta x$

SINON

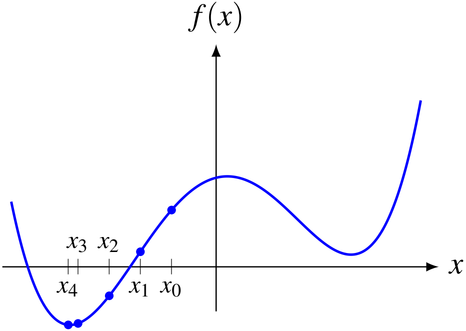
⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$



Étant donné un point initial x_0

On recommence cette procédure sur le nouvel itéré :

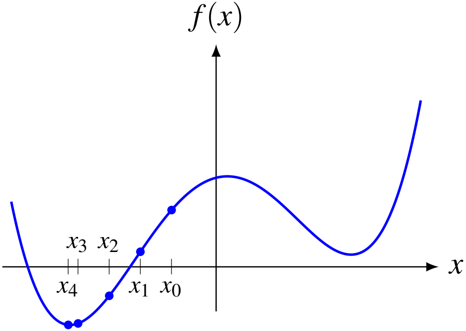
RÉPÉTER jusqu'à convergence
  SI $f(x_k + \Delta x) < f(x_k)$
    ➡️ : $x_{k+1} \leftarrow x_k + \Delta x$
  SINON
    ⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$



Étant donné un point initial x_0

On recommence cette procédure sur le nouvel itéré :

RÉPÉTER jusqu'à convergence
- SI $f(x_k + \Delta x) < f(x_k)$
  - ➡️ : $x_{k+1} \leftarrow x_k + \Delta x$
- SINON
  - ⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$



Étant donné un point initial x_0

On recommence cette procédure sur le nouvel itéré :

RÉPÉTER jusqu'à convergence
SI $f(x_k + \Delta x) < f(x_k)$
➡️ : $x_{k+1} \leftarrow x_k + \Delta x$
SINON
⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$

La condition $f(x_k+\Delta x) < f(x_k)$ peut se réécrire à moindres frais $\frac{f(x_k+\Delta x)-f(x_k)}{\Delta x} < 0$, qui fait apparaître le taux d'accroissement de f en x_k pour un pas Δx.

SI $f(x_k + \Delta x) < f(x_k)$
➡️ : $x_{k+1} \leftarrow x_k + \Delta x$
SINON
⬅️ : $x_{k+1} \leftarrow x_k - \Delta x$

indique donc que la direction de descente choisie à un taux d'accroissement négatif : on part dans la direction où la fonction décroît.

Rappel : le nombre dérivé

On appelle nombre dérivé de f en x, et on note $f'(x)$, la limite (si elle existe) du taux d'accroissement $\frac{f(x+h)-f(x)}{h}$ quand $h \to 0$

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Comment optimiser

Si seulement c'était aussi simple pour $f : \mathbb{R}^n \to \mathbb{R}$ ...

Rappel : le nombre dérivé

On appelle nombre dérivé de f en x, et on note $f'(x)$, la limite (si elle existe) du taux d'accroissement $\frac{f(x+h)-f(x)}{h}$ quand $h \to 0$

$$f'(x) = \lim_{h \to 0} \frac{f(x+h)-f(x)}{h}$$

Si $f : \mathbb{R} \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ et $h \in \mathbb{R} \Rightarrow$ ✓✓pas de problème ✓✓

Si $f : \mathbb{R}^n \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ **mais** $h \in \mathbb{R}^n$ 🚫🚫 division par un vecteur 🚫🚫

Rappel : le nombre dérivé

On appelle nombre dérivé de f en x, et on note $f'(x)$, la limite (si elle existe) du taux d'accroissement $\frac{f(x+h)-f(x)}{h}$ quand $h \to 0$

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Si $f : \mathbb{R} \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ et $h \in \mathbb{R} \Rightarrow$ ✓✓ pas de problème ✓✓

Si $f : \mathbb{R}^n \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ **mais** $h \in \mathbb{R}^n$ 🚫🚫 division par un vecteur 🚫🚫

Pour des fonctions de $f : \mathbb{R}^n \to \mathbb{R}$, le rôle du nombre dérivé dans le processus de minimisation (c'est-à-dire, indiquer la direction du prochain itéré) est délégué au gradient $\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T \Rightarrow$ on parle alors de **descente de gradient** .

Comment optimiser

Si seulement c'était aussi simple pour $f : \mathbb{R}^n \to \mathbb{R}$ ...

Rappel : le nombre dérivé

On appelle nombre dérivé de f en x, et on note $f'(x)$, la limite (si elle existe) du taux d'accroissement $\frac{f(x+h)-f(x)}{h}$ quand $h \to 0$

$$f'(x) = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h}$$

Si $f : \mathbb{R} \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ et $h \in \mathbb{R} \Rightarrow$ ✓✓pas de problème ✓✓
Si $f : \mathbb{R}^n \to \mathbb{R}$, $f(x+h) - f(x) \in \mathbb{R}$ **mais** $h \in \mathbb{R}^n$ 🚫🚫 division par un vecteur 🚫🚫

Pour des fonctions de $f : \mathbb{R}^n \to \mathbb{R}$, le rôle du nombre dérivé dans le processus de minimisation (c'est-à-dire, indiquer la direction du prochain itéré) est délégué au gradient $\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right)^T \Rightarrow$ on parle alors de descente de gradient .

Malheureusement

- certaines certitudes chèrement acquises pour les fonctions d'une variable réelle s'effondrent brutalement quand on passe dans le domaine des fonctions multi-variées.
- on ne peut plus ignorer les problèmes de géométrie induits par la dimensionnalité de l'espace de travail.

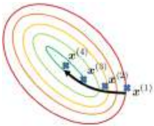
Descente de gradient

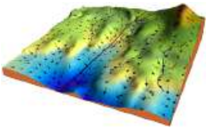
Oui, mais laquelle ?

L'algorithme de descente de gradient est conceptuellement simple

Algorithm 1 Gradient descent

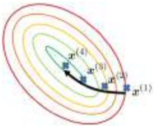
1: Choose initial point $\mathbf{x}_0$ and learning rate η
2: **repeat**
3: Compute gradient $\nabla f(\mathbf{x}_k)$ at $\mathbf{x}_k$
4: Update iterate $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k)$
5: $k \leftarrow k + 1$
6: **until** convergence

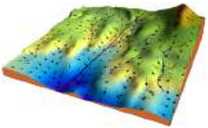




L'algorithme de descente de gradient est conceptuellement simple

Algorithm 1 Gradient descent
1: Choose initial point $\mathbf{x}_0$ and learning rate η
2: **repeat**
3: Compute gradient $\nabla f(\mathbf{x}_k)$ at $\mathbf{x}_k$
4: Update iterate $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k)$
5: $k \leftarrow k + 1$
6: **until** convergence





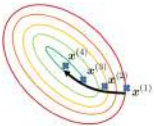
Trop simple pour être vrai ? 😱😱

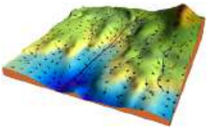
Descente de gradient

Oui, mais laquelle ?

L'algorithme de descente de gradient est conceptuellement simple

Algorithm 1 Gradient descent
1: Choose initial point $\mathbf{x}_0$ and learning rate η
2: **repeat**
3: Compute gradient $\nabla f(\mathbf{x}_k)$ at $\mathbf{x}_k$
4: Update iterate $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k)$
5: $k \leftarrow k+1$
6: **until** convergence





Trop simple pour être vrai ? 😱😱

La descente de gradient peut être :

→ À pas fixe / à pas variable / avec *backtracking* ...
→ À plus forte pente / par coordonnées / stochastique ...
→ Conjuguée / projetée / proximale ...
→ Avec accélération de type momentum / Nesterov / Adam ...
→ Méthode de Newton / de quasi-Newton / de Gauss-Newton ...

État des lieux du cours

Plan du cours

Introduction

→ Introduction générale du cours	2h CM

L'optimisation en théorie

→ Rappels et préliminaires géométriques	2h CM + 1h TD
→ Convexité à l'ordre 0	2h CM + 1h TD
→ Un peu de calcul différentiel	3h CM + 3h TD
→ Convexité à l'ordre $\{1, 2\}$ et points critiques	3h CM + 3h TD

L'optimisation en pratique

→ Anatomie d'une méthode de descente	3h CM
→ Au delà de la descente de gradient	3h CM
→ Comparaison des méthodes de descente	$2\times$3h TP

Les acquis d'apprentissage visés 👩‍🎓👨‍🎓

Enfin, en théorie...

Savoir

- 🔍 Identifier les différents éléments composant un problème d'optimisation, ainsi que des éléments géométriques et analytiques nécessaires à son **étude qualitative**.
- 🧩 **Comprendre les algorithmes** à disposition pour résoudre un problème d'optimisation et les hyperparamètres qui déclinent et gouvernent ceux-ci.
- ✅ Décrire le **domaine de validité** d'un algorithme d'optimisation.

Savoir-faire

- 🧑‍💻 **Implémenter des algorithmes** d'optimisation sans contraintes.
- 📊 Effectuer des **analyses comparatives** entre des différentes algorithmes d'optimisation sans contraintes.