Outils de Recherche opérationnelle en Génie

MTH 8414

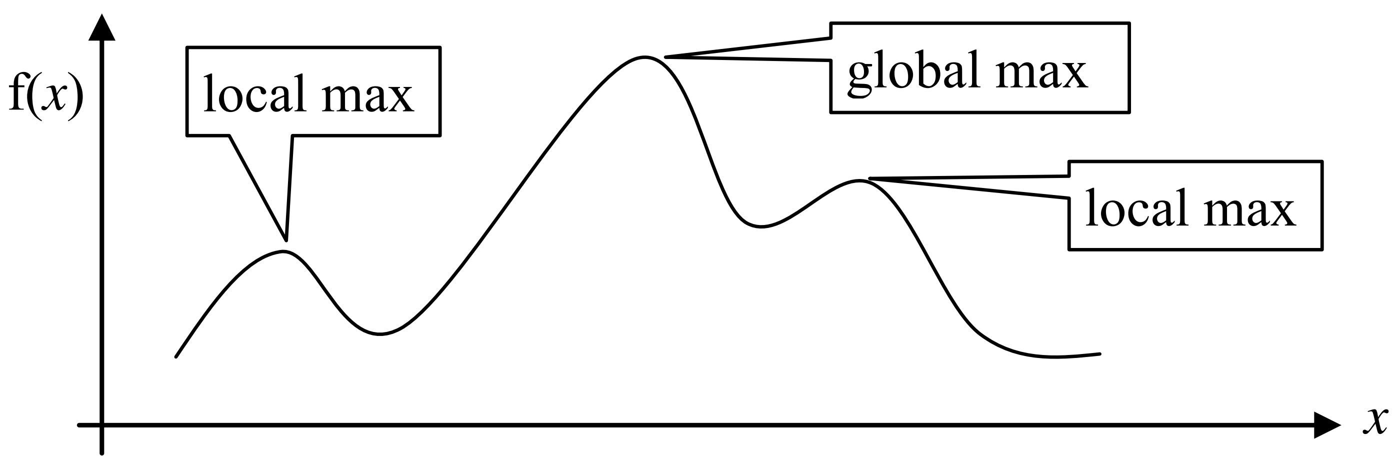
Programmation Non Linéaire

Introduction à la Prog. Non Linéaire

- Un programme non linéaire est similaire à un programme linéaire, car il est constitué:
  - D'une fonction objectif
  - Des contraintes et des bornes sur les variables
- Toutefois ces équations n'ont pas à être linéaires
- Elles sont généralement données comme des fonctions (analytiques)
- Il y a de nombreuses applications pratiques de la PNL:
  - Problème d'optimisation en finance
  - Problème d'optimisation en énergie
  - Problème d'optimisation en (pétro) chimie
  - …
- Un PNL est toutefois beaucoup plus difficile à résoudre qu'un PL.
- Voyons pourquoi…

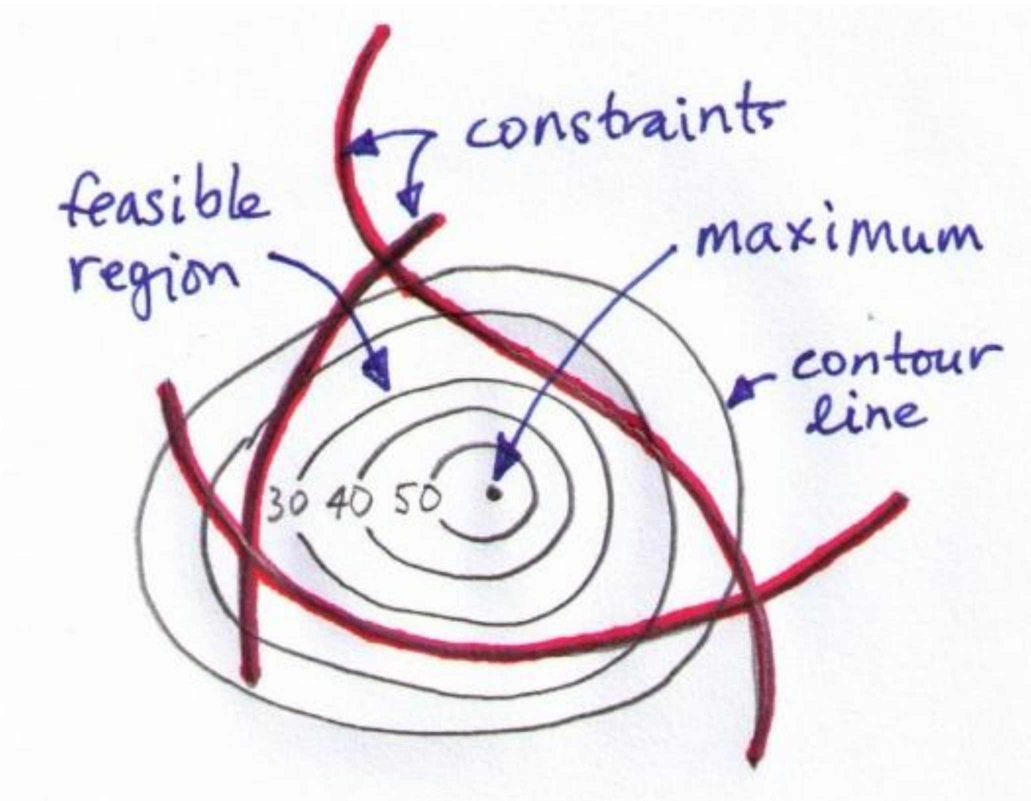
Raison 1: Local ou global cet optimum ?

- En l'absence de convexité (comme pour les PL) les méthodes de résolution de PNL ne peuvent garantir d'identifier la solution optimale
- Elles font généralement une recherche dans l'espace, basée sur:
  - Un point courant
  - La valeur de l'objectif à ce point
  - La valeur du gradient de la fonction objectif
  - La matrice Hessien (dérivée seconde) de la fonction objectif
- Ces informations permettent d'identifier un maximum (ou minimum) local, mais pas de dire s'il est global.



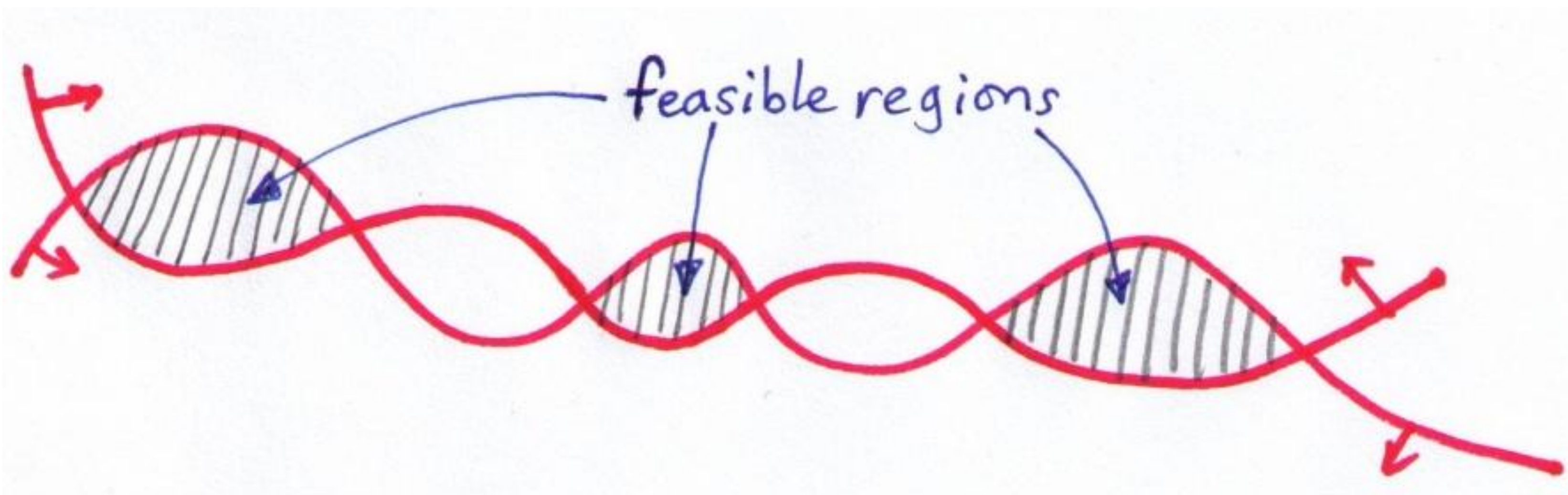
Raison 2: L’optimum et les points extrêmes.

- En PL, s’il existe une solution optimale, alors il existe un point extrême du polyèdre qui est optimal.
- En PNL, l’optimum peut se trouver n’importe où
  - En rouge les lignes de la région réalisables
  - En noir des courbes de niveau de la fonction objectif.



Raison 3: Régions réalisables déconnectées

- Rien ne garantit que l'espace des solutions réalisables est connecté.
- Donc, même si on trouve l'optimum d'une région, rien ne nous dit qu'il n'existe pas d'autres régions réalisables…



Raison 4: L'importance du point de départ

- Comme on ne peut pas garantir l'obtention d'un optimum global, le point d'où on débute la recherche joue un rôle important sur la solution que nous pouvons trouver.

- En effet la plupart des solveurs non linéaires font une recherche locale
  - À partir d'une solution de départ ou solution courante
  - On détermine la meilleure direction (avec le gradient)
  - On fait un « pas » dans cette direction

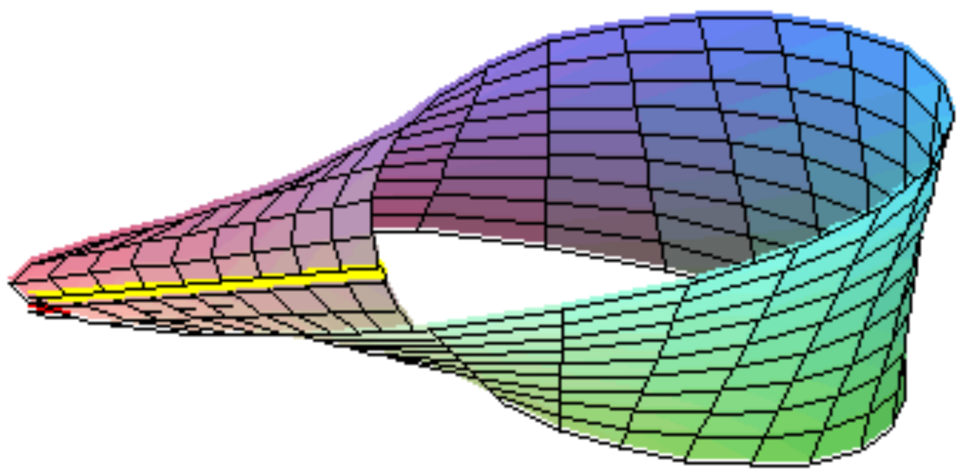
- Une des solutions est de redémarrer le solveur régulièrement à partir de points différents.

Raison 5: Trouver un point de départ

- Il peut être parfois difficile de trouver un point de départ réalisable.
- En PL, il existe une « phase 1 » qui permet toujours de trouver une solution initiale ou de montrer qu'il n'en existe aucun.
- En PNL, on commence en minimisant une distance vis-à-vis de la zone réalisable, comme le carré des violations par exemple.
- Donc si on optimise ce deuxième problème et on trouve une solution optimale dont le coût est 0, alors on a identifié une solution réalisable à notre problème d'origine
- Toutefois, ce deuxième problème est lui-même un problème NL…
- Dans ce cas, les solveurs peuvent donc rester pris dans un minimum local dont la valeur n'est pas 0… ce qui implique
  - Qu'on ne peut pas optimiser le problème
  - On ne peut pas prouver qu'il est irréalisable.

Raison 6: Dures égalités

- Si on a une courbe très « tordue » il peut s’avérer difficile de trouver un point qui tombe « pile » sur la courbe
- De plus, lors de l’optimisation (via recherche locale) il risque d’être encore plus difficile de rester sur cette courbe.



Raison 8: Un zoo de méthodologies

- La PNL regroupe sous une même bannière une vaste gamme de techniques associés à chaque type de fonctions (quadratiques, convexes, etc.)
- Face à un problème, quelle technique appliquée ?
- Les solveurs nécessitent parfois que les fonctions linéaires respectent des conditions particulières… Il peut donc être difficile d'identifier le bon solveur à utiliser.
- Il existe très peu d'outils permettant d'analyser un problème NL, d'identifier ses caractéristiques et de choisir un bon solveur.
- Différents solveurs vont donner différentes solutions…
- Différentes modélisations vont donner différentes solutions…

Raison 9: Difficulté d’usage

- Les solveurs disponibles ont généralement un très grand nombre de paramètres à ajuster (>50).

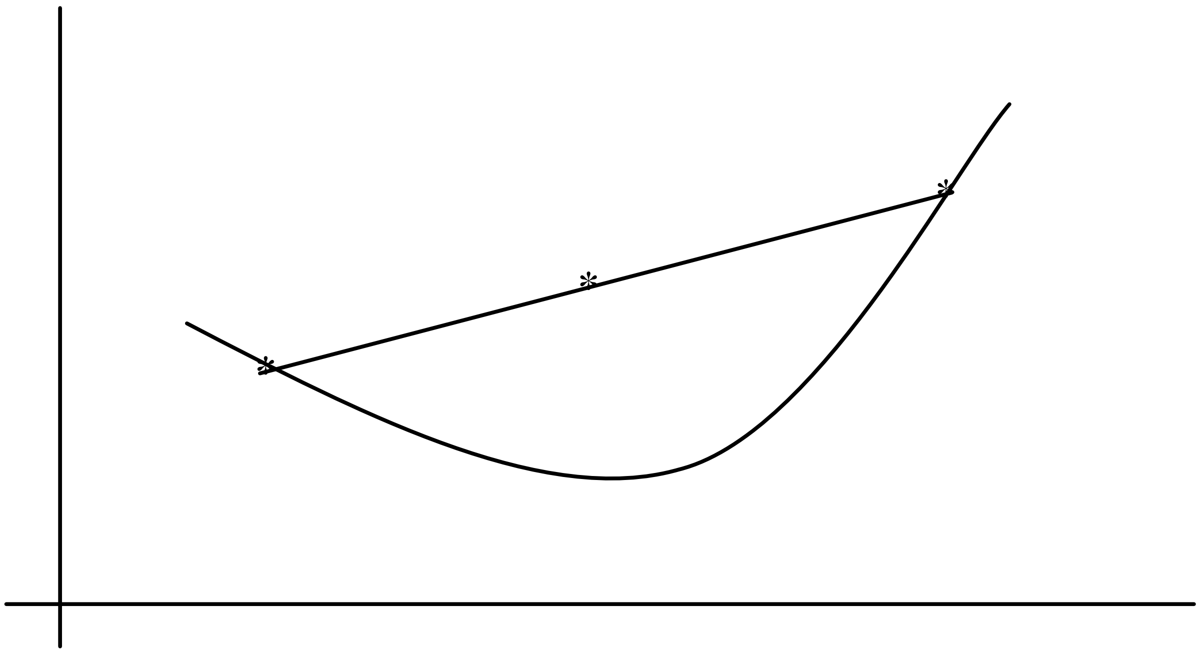
- Optimiser ces paramètres peut s’avérer une tâche complexe
  - Quel type d’optimisation locale, quelle tolérance, inversion de la base, etc.

- Souvent, on a besoin des dérivées des fonctions : elles peuvent être difficiles à trouver.

- Pas de standard pour l'entrée des données : difficile de passer d'un solveur à un autre.

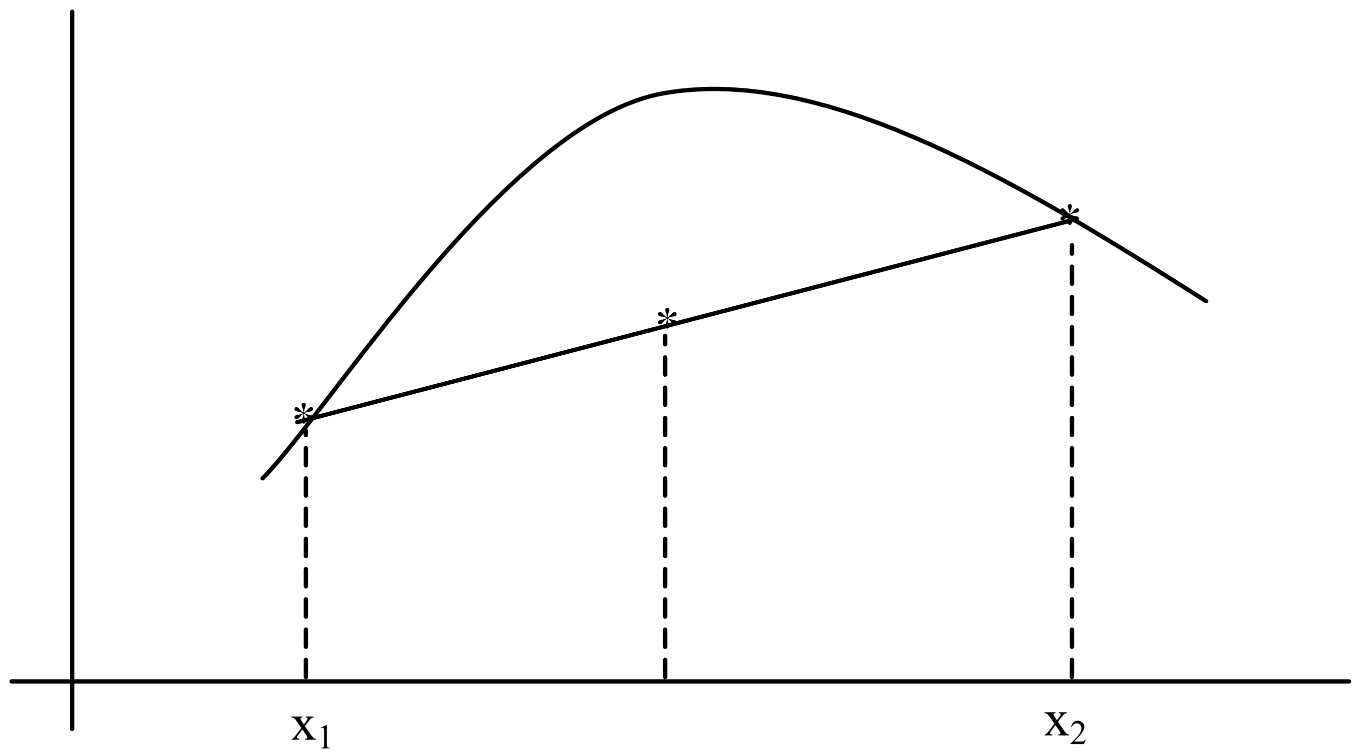
Fonction objectif convexe

$$\lambda f(x_1) + (1-\lambda)\, f(x_2) \geq f\,(\lambda x_1 + (1-\lambda) x_2) \text{ avec } \; 0 \leq \lambda \leq 1$$



Fonction objectif concave

$$\lambda f(x_1) + (1-\lambda)\, f(x_2) \le f\,(\lambda x_1 + (1-\lambda) x_2) \text{ avec } \quad 0 \le \lambda \le 1$$



Conditions de Karush-Kuhn-Tucker

$$\max \quad f(x)$$

$$s.t. \quad g_i(x) = b_i, \quad i = 1, \ldots, m$$

conditions pour que x soit maximum local

$$\left.\begin{aligned} &\nabla f(x) = \sum_{i}^{m} v_i \nabla g_i(x) \\ &v_i \geq 0, \quad i = 1, \ldots, m \end{aligned}\right\} \quad \text{faisabilité duale}$$

$$g_i(x) \leq b_i \quad i = 1, \ldots, m \qquad \text{faisabilité primale}$$

$$v_i(b_i - g_i(x)) = 0 \quad i = 1, \ldots, m \qquad \text{complémentarité}$$

Programmation Quadratique

- $f(x)$ est quadratique i.e. inclus, les termes x_j^2 et $x_j x_k \quad (j \neq k)_j$
- Les contraintes sont toutes linéaires. $g_i(x)$

$$x = (x_1, x_2, \ldots, x_n) \;\; t.q.$$

$$(PQ) \quad \begin{cases} \text{Max} \quad z = C^T X - \dfrac{1}{2} X^T Q X \\ \text{Sujet :} \\ \qquad\qquad AX \leq b \\ \qquad\qquad X \geq 0 \end{cases}$$

où $Q = \lfloor q_{ij} \rfloor$: symétrique, semi-définie positive.

Si z est concave $(\text{i.e. } X^T Q X \geq 0)$ alors on un optimum unique.

Programmation quadratique avec CPLEX

$$\min \frac{1}{2} x^t Q x + c^t x$$
$$\text{s.c.} \quad Ax \propto b$$
$$l \leq x \leq u$$

Avec α étant toute combinaison de {≤, =, ≥ } pour les contraintes.
Q : matrice de coefficients des termes quadratiques.

CPLEX résoud : min obj convexe avec Q positif semi-défini
$x^tQx \geq 0$
ou max obj. concave avec Q négatif semi-défini
$x^tQx \leq 0$

Problème de programmation convexe (PC)

$z = f(X)$ est concave (car on maximise)

chaque $g_i(X)$ est convexe.

Exemple:

$$(PC)\begin{cases}\text{Max} \quad z = 32x_1 + 50x_2 - 10x_2^2 + x_2^3 - x_1^4 - x_2^4 \\ \text{Sujet :} \\ \qquad 2x_1^2 + x_2 \leq 13 \\ \qquad x_1^2 + x_2 \leq 9 \\ \qquad x_1 \geq 0, \quad x_2 \geq 0\end{cases}$$

Programmation convexe

(max f concave, contraintes $g_i(x)$ convexes)

- Algorithmes de type « gradient »
  - par exemple : gradient réduit généralisé
- Algorithmes d'optimisation non contraint séquentiels
  - transforme le problème contraint en envoyant les contraintes dans l'objectif (méthodes de pénalités ou de barrières)
- Algorithmes d'approximations séquentielles
  - remplace l'objectif par une succession d'approximations linéaires ou quadratiques

Problème de programmation séparable

$f(x)$ est concave et séparable i.e.

$$f(x) = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n) = \sum_{j=1}^{n} f_j(x_j)$$

$g_i(x)$ est convexe et séparable pour chaque i

- On en a déjà discuté dans le cours sur la modélisation en PL

Problème de programmation géométrique

$f(x)$ et les fonctions contraintes sont de la forme $\sum_{j=1}^{p}\left(c_j \prod_{i=1}^{n} x_i^{a_{ij}} \right)$

et sont généralement ni convexes ni concaves, où les c_j et a_{ij} sont des constantes physiques dans les applications de problème d'ingénierie.

Exemple:

$$(PG)\begin{cases} \text{Min} \qquad f(x) = 2x_1^{-2}x_1^{-1} + x_1^{-1}x_2^{-2} \\ \text{sujet à :} \\ \qquad\qquad 4x_1x_2 + x_1^2x_2^2 \leq 12 \\ \qquad\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{cases}$$

Programmation fractionnelle

$$f(x) = f_1(x)/f_2(x) \quad : \qquad \text{ratio de } f_1(x) \text{ et } \mathrm{f}_2(x)$$

$$\text{Max} \quad \mathrm{f(x)} = f_1(x)/f_2(x)$$

- ***Exemple:***

$$(PF) \quad \begin{cases} \text{Max} \quad f(x) = \dfrac{10x_1 + 20x_2 + 10}{3x_1 + 4x_2 + 20} \\ \text{Sujet :} \\ \qquad x_1 + 3x_2 \leq 50 \\ \qquad 3x_1 + 2x_2 \leq 80 \\ \qquad x_1 \geq 0, \quad x_2 \geq 0 \end{cases}$$

Problème de programmation non convexe

Tous problèmes de PNL qui ne remplissent pas les hypothèses des problèmes de programmation convexe.

- ***Exemple:***

$$(PNC)\begin{cases}\text{Max} \quad f(x) = 3x_1x_2 - 2x_2^1 - x_2^2 \\ \text{Sujet :} \\ \qquad x_1^2 + 2x_2^2 \le 4 \\ \qquad 2x_1 - x_2 \le 0 \\ \qquad x_1x_2^2 + x_1^2x_2 = 2 \\ \qquad x_1 \ge 0, \quad x_2 \ge 0. \end{cases}$$

Méthodologie

- On optimise les problèmes à une variables à l'aide des dérivées premières et secondes.

- Nous utiliserons les mêmes concepts pour les problèmes comportant plusieurs variables.

- Le gradient correspond à la dérivée première, alors que la matrice hessienne représente les dérivés secondes.

Le Gradient

- Rappel sur le gradient (∇):
- Pour une fonction "f", sur les variables $x_1, x_2, \ldots, x_n$:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Exemple: $f = 15x_1 + 2(x_2)^3 - 3x_1(x_3)^2$

$$\nabla f = \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1x_3 \end{bmatrix}$$

La matrice hessienne

- La matrice hessienne (∇^2) de $f(x_1, x_2, \ldots, x_n)$ est :

$$\nabla^2 f = \begin{bmatrix} \frac{\partial f^2}{\partial x_1^{\,2}} & \frac{\partial f^2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial f^2}{\partial x_1 \partial x_n} \\ \frac{\partial f^2}{\partial x_2 \partial x_1} & \frac{\partial f^2}{\partial x_2^{\,2}} & \cdots & \frac{\partial f^2}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f^2}{\partial x_n \partial x_1} & \frac{\partial f^2}{\partial x_n \partial x_2} & \cdots & \frac{\partial f^2}{\partial x_n^{\,2}} \end{bmatrix}$$

Exemple de matrice hessienne

- Exemple (vue précédemment):

$$f = 15x_1 + 2(x_2)^3 - 3x_1(x_3)^2$$

$$\nabla f = \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1x_3 \end{bmatrix}$$

$$\nabla^2 f = \begin{bmatrix} 0 & 0 & -6x_3 \\ 0 & 12x_2 & 0 \\ -6x_3 & 0 & -6x_1 \end{bmatrix}$$

Optimisation non linéaire sans contraintes

- La procédure d'optimisation sur plusieurs variables fonctionne comme suit:

1. Résoudre le problème qui consiste à trouver les points pour lesquels le gradient est nul.

2. Calculer le Hessien de la fonction objectif et l'évaluer pour chacun des points obtenus en 1.
   - Si la matrice est "définie positive" alors le point est un minimum local.
   - Si la matrice est "définie négative" alors le point est un maximum local.

Positive/Negative Definite

- Une matrice est "définie positive" si toutes ces valeurs propres sont positives. (> 0)

- Une matrice est "définie négative" si toutes ces valeurs propres sont négatives. (< 0)

- Une matrice est "semi-définie positive" si toutes ces valeurs propres sont positives ou nulles. (≥ 0)

- Une matrice est "semi-définie négative" si toutes ces valeurs propres sont négatives nulles. (≤ 0)

Matrice Exemple

Soit la matrice A:

$$A = \begin{bmatrix} 2 & 4 & 5 \\ -5 & -7 & -1 \\ 1 & 1 & 2 \end{bmatrix}$$

Les valeurs propres de A sont:

$\lambda_1 = -3.702$ $\lambda_2 = -2$ $\lambda_3 = 2.702$

Cette matrice est <u>définie négative</u>

Un exemple de PNL sans contrainte

Consider the problem:

Minimiser $f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$

D'abord, trouvons le gradient pour les x_i:

$$\nabla f = \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix}$$

Un exemple de PNL sans contrainte

Ensuite, mettre le gradient = 0:

$$\nabla f = 0 \qquad \Rightarrow \qquad \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Nous avons un système à trois équations et trois inconnues, que nous pouvons résoudre:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Un exemple de PNL sans contrainte

Nous avons donc un candidat à valider

Trouvons la matrice hessienne:

$$\nabla^2 f = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

Un exemple de PNL sans contrainte

Les valeurs propres de cette matrice sont:

$$\lambda_1 = 3.414 \qquad \lambda_2 = 0.586 \qquad \lambda_3 = 2$$

Comme toutes les valeurs propres sont > 0, alors la matrice hessienne est définie positive.

Alors le point $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$ est un minimum

Méthode de résolution

- Dans l’exemple précédent, on avait un système à trois équations linéaires et trois inconnus.
- Pour certains autres problèmes, nous pourrions obtenir des équations non linéaires.
- On doit alors résoudre un système d’équations non linéaires… ce qui n’est pas simple.
- Pour éviter ce problème, les PNL sont généralement résolus numériquement.

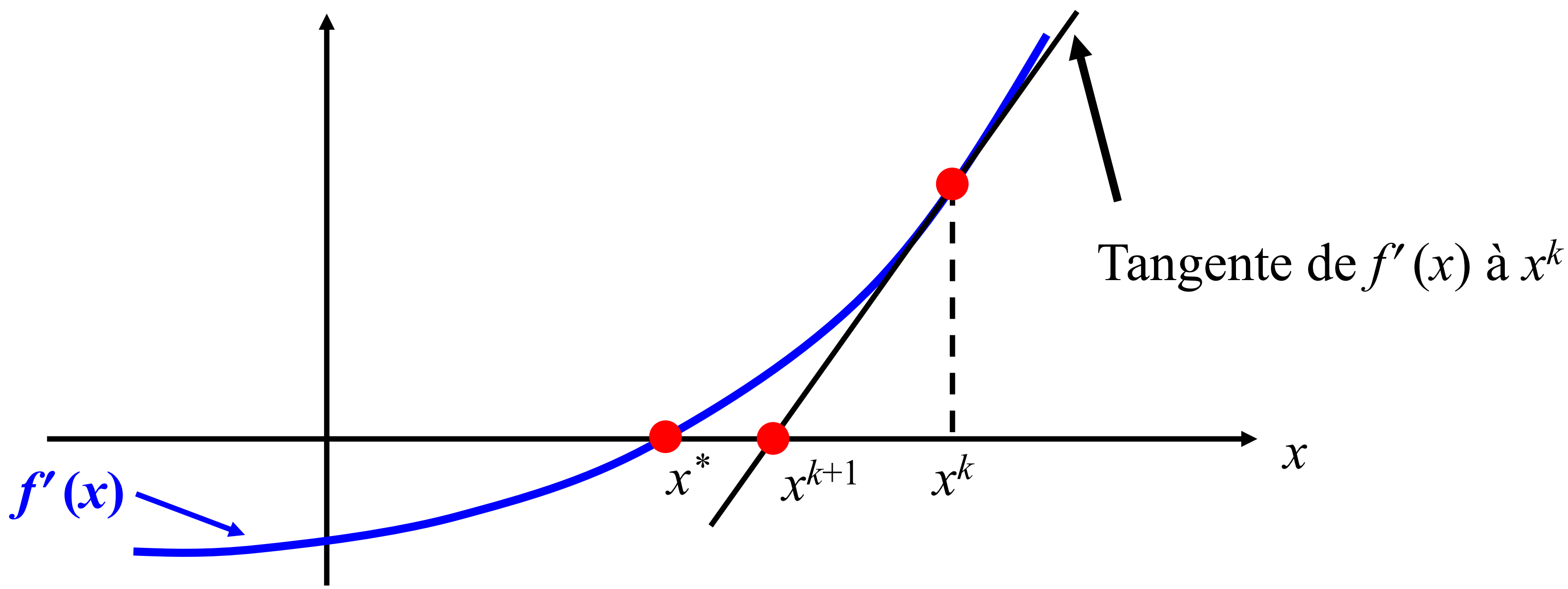
Méthode de Newton

Lors de la résolution de $f'(\mathrm{x}) = 0$ pour trouver un maximum ou un minimum, on peut faire utiliser les itérations suivantes:

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

- k est l'itération courante
- $|x^{k+1} - x^k| < e$ où e est une tolérance spécifiée

Diagramme de la méthode de Newton



La méthode de Newton approxime $f'(x)$ avec une droite au point x^k et on obtient le point (x^{k+1}), qui est utilisé pour approximer la fonction au prochain point. Jusqu'à ce qu'on rapproche suffisamment de x^*.

Commentaire sur la méthode de Newton

- Il faut s'assurer que
  - $f(x^{k+1}) < f(x^k)$ pour trouver un minimum.
  - $f(x^{k+1}) > f(x^k)$ pour trouver un maximum.
- Désavantage:
  - Il faut calculer les dérivées premières et secondes
  - La solution initiale est importante, si elle est trop loin de l'optimum la méthode peut ne pas converger.

Méthode de Regula-Falsi

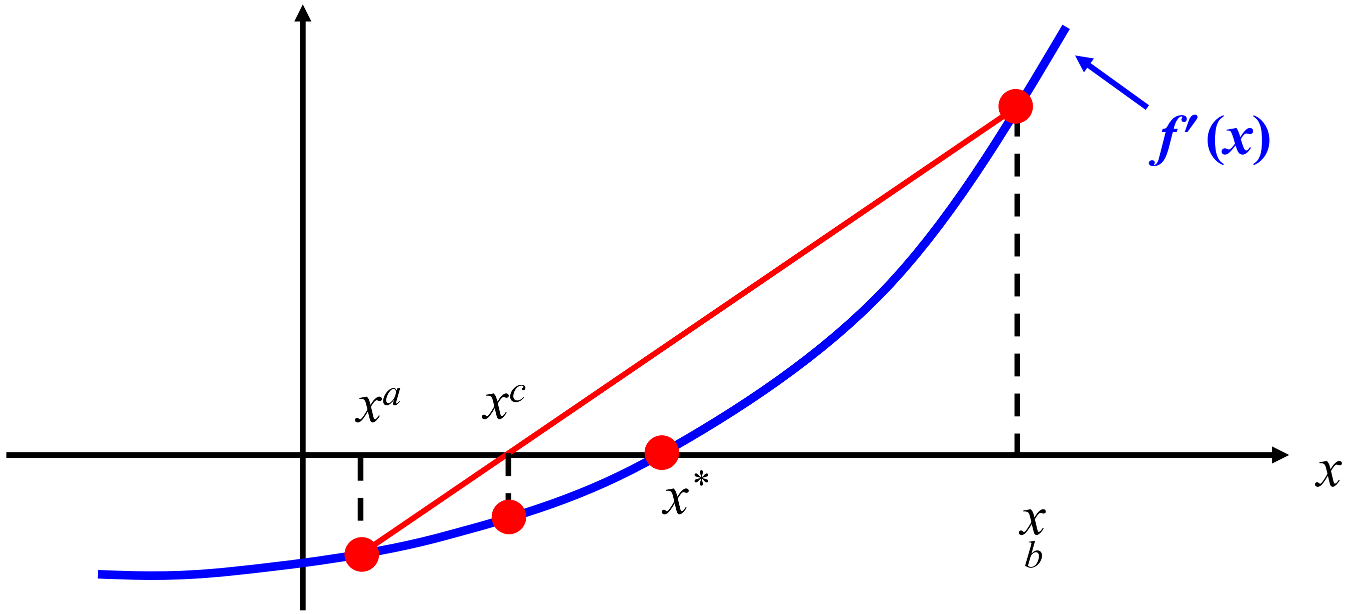
Cette méthode nécessite deux points x^a & x^b qui encadre la solution de l'équation $f\,'(x) = 0$.

$$x^c = x^b - \frac{f'(x^b)\cdot(x^b - x^a)}{f'(x^b) - f'(x^a)}$$

où x^c sera entre x^a & x^b.

Le prochain intervalle sera défini par x^c et soit x^a ou x^b, celui dont le signe sera différent de x^c.

Diagramme de Regula-Falsi



La méthode de Regula-Falsi approxime fonction $f'(x)$ avec une droite et utilise l'interpolation pour trouver la racine.

Commentaires sur Regula-Falsi

- Cette méthode nécessite la connaissance de deux points qui entoure la solution.

- Elle ne nécessite toutefois pas le calcul de la dérivée seconde.

- La méthode de Regula-Falsi nécessite toutefois un peu plus d'itération que la méthode de Newton.

Optimisation multi variables

- Considérons le cas avec plusieurs variables, sans contrainte.
- Pratiquement toutes les méthodes suivent la recette suivante:
  1. Choisir une direction de recherche $\mathbf{d}^k$
  2. Chercher le minimum dans cette direction afin de trouver le prochain point.

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{d}^k$$

Où k est le numéro d'itération courante et a^k est un scalaire positif appeler le pas

Le pas

- Le pas, α^{k}, est calculé de la manière suivante:
  - Nous voulons optimiser la fonction $f(\mathbf{x}^{k+1}) = f(\mathbf{x}^{k} + \alpha^{k}\mathbf{d}^{k})$ où la seule variable est α^{k} parce que $\mathbf{x}^{k}$ & $\mathbf{d}^{k}$ sont connus.

- Nous posons $$\frac{\mathrm{d}f\left(\mathbf{x}^{k} + \alpha^{k}\mathbf{d}^{k}\right)}{\mathrm{d}\alpha^{k}} = 0$$ et résolvons pour α^{k} en utilisant la méthode à une seule variable vue précédemment.

Méthode de la descente la plus rapide

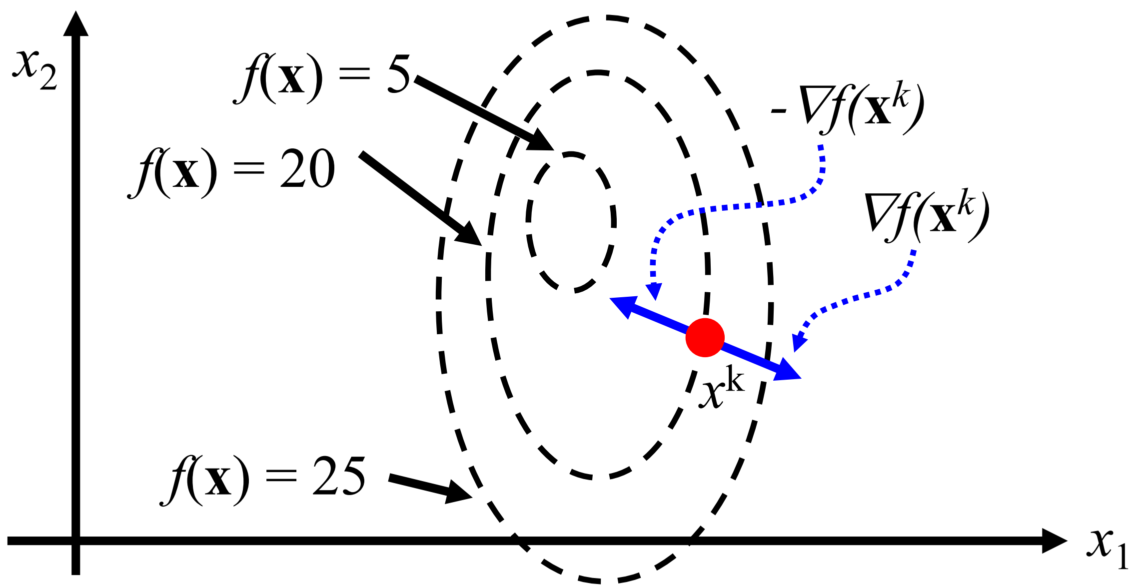
- Cette méthode est très simple
  - Pour donner une direction, elle utilise le gradient (pour maximiser) ou le gradient négatif (pour minimiser)

$$\mathbf{d}^k = \begin{Bmatrix} + \\ - \end{Bmatrix} \nabla f(\mathbf{x}^k) \quad \text{pour} \quad \begin{Bmatrix} \max \\ \min \end{Bmatrix}$$

$$\mathbf{x}^{k+1} = \mathbf{x}^k \begin{Bmatrix} + \\ - \end{Bmatrix} \alpha^k \nabla f(\mathbf{x}^k)$$

Méthode de la descente la plus rapide

- L'opposé du gradient nous indique la direction de descente la plus rapide.



Méthode de la descente la plus rapide

Les étapes sont:

1. Choisir un point initial $\mathbf{x}^0$
2. Calculer le gradient $\nabla f(\mathbf{x}^k)$ où k est un numéro d'itération
3. Calculer la direction de recherche: $\mathbf{d}^k = \pm\nabla f(\mathbf{x}^k)$
4. Calculer le prochain point $\mathbf{x}$:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{d}^k$$

en utilisant une méthode d'optimisation à une variable sur α^{k}.

Méthode de la descente la plus rapide

Pour déterminer la convergence, utiliser une tolérance ε_1 et arrêter:

$$\left| f(\mathbf{x}^{k+1}) - f(\mathbf{x}^{k}) \right| < \varepsilon_1$$

Où utiliser une tolérance ε_2 et arrêter:

$$\left\| \nabla f(\mathbf{x}^{k}) \right\| < \varepsilon_2$$

- Deux critères d'arrêt sont valides pour la majorité des méthodes d'optimisation à plusieurs variables.
- Rappelons que la norme d'un vecteur, $\|\mathbf{x}\|$ est donné par:

$$\left\| \mathbf{x} \right\| = \sqrt{\mathbf{x}^T \cdot \mathbf{x}} = \sqrt{(x_1)^2 + (x_2)^2 + \cdots + (x_n)^2}$$

Exemple

Minimiser $f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$

Avec le point de départ $\mathbf{x}^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

Exemple

$$\nabla f(\mathbf{x}) = \begin{bmatrix} 2x_1 + (1 - x_2) & -x_1 + 2x_2 - x_3 & -x_2 + 2x_3 + 1 \end{bmatrix}$$

$$\begin{aligned} \mathbf{d}^0 &= -\nabla f(\mathbf{x}^0) = -\begin{bmatrix} 2(0) + 1 - 0 & -0 + 0 - 0 & -0 + 0 + 1 \end{bmatrix} \\ &= -\begin{bmatrix} 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & -1 \end{bmatrix} \end{aligned}$$

$$\mathbf{x}^1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} + \alpha^0 \cdot \begin{bmatrix} -1 & 0 & -1 \end{bmatrix}$$

Il faut maintenant trouver α^0

Exemple

$$f(\mathbf{x}^1) = (\alpha^0)^2 + (-\alpha^0)(1) + 0 - 0 + (\alpha^0)^2 + (-\alpha^0)$$

$$= 2(\alpha^0)^2 - 2(\alpha^0)$$

$$\frac{df(\mathbf{x}^1)}{d\alpha^0} = 4(\alpha^0) - 2$$

On met =0 et on résout:

$$4(\alpha^0) = 2 \;\Rightarrow\; \alpha^0 = {}^2\!/_4 = {}^1\!/_2$$

Exemple

$$\mathbf{x}^1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} + \alpha^0 \cdot \begin{bmatrix} -1 & 0 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix}$$

$$\therefore \mathbf{x}^1 = \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix}$$

Exemple

On choisit le gradient négatif comme prochaine direction de descente.

$$\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) = -\begin{bmatrix} -1+1+0 & \dfrac{1}{2}+0+\dfrac{1}{2} & 0-1+1 \end{bmatrix}$$

$$\therefore \mathbf{d}^1 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}$$

Exemple

On met à jour les données d'itération

$$\mathbf{x}^2 = \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix} + \alpha^1 \cdot \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{1}{2} & -\alpha^1 & -\frac{1}{2} \end{bmatrix}$$

Exemple

ÉCOLE POLYTECHNIQUE MONTRÉAL

Insérez dans la fonction originale et dérivez pour trouver α^1:

$$f(\mathbf{x}^2) = \frac{1}{4} + \left(-\frac{1}{2}\right)\left(1+\alpha^1\right) + (\alpha^1)^2 - (\alpha^1)\left(\frac{1}{2}\right) + \frac{1}{4} - \frac{1}{2}$$

$$= (\alpha^1)^2 - \alpha^1 - 1/2$$

$$\frac{df(\mathbf{x}^1)}{d\alpha^1} = 2(\alpha^1) - 1$$

Exemple

Mettre la dérivée = 0 et résoudre pour α^1:

$$2(\alpha^1) = 1 \;\Rightarrow\; \alpha^1 = 1/2$$

Exemple

Calculez maintenant $\mathbf{x}^2$:

$$\mathbf{x}^2 = \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix} + \alpha^1 \cdot \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix} + \begin{bmatrix} 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

$$\therefore \mathbf{x}^2 = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

Exemple

$$\mathbf{d}^2 = -\nabla f(\mathbf{x}^2) = -\begin{bmatrix} -1+1+\dfrac{1}{2} & \dfrac{1}{2}-1+\dfrac{1}{2} & \dfrac{1}{2}-1+1 \end{bmatrix}$$

$$\therefore \mathbf{d}^2 = \begin{bmatrix} -\dfrac{1}{2} & 0 & -\dfrac{1}{2} \end{bmatrix}$$

$$\mathbf{x}^3 = \begin{bmatrix} -\dfrac{1}{2} & -\dfrac{1}{2} & -\dfrac{1}{2} \end{bmatrix} + \alpha^2 \cdot \begin{bmatrix} -\dfrac{1}{2} & 0 & -\dfrac{1}{2} \end{bmatrix}$$

$$= \begin{bmatrix} -\dfrac{1}{2}(\alpha^2+1) & -\dfrac{1}{2} & -\dfrac{1}{2}(\alpha^2+1) \end{bmatrix}$$

Exemple

Trouvez α^2:

$$f(\mathbf{x}^3) = \frac{1}{2}(\alpha^2 + 1)^2 - \frac{3}{2}(\alpha^2 + 1) + \frac{1}{4}$$

$$\frac{df(\mathbf{x}^3)}{d\alpha^2} = (\alpha^2 + 1) - \frac{3}{2}$$

Mettre la dérivée = 0 et résoudre

$$(\alpha^2 + 1) = \frac{3}{2} \Rightarrow \alpha^2 = 1/2$$

Exemple

Calculaler $\mathbf{x}^3$:

$$\mathbf{x}^3 = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix} + \alpha^2 \cdot \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix} + \begin{bmatrix} -\frac{1}{4} & 0 & -\frac{1}{4} \end{bmatrix}$$

$$\therefore \mathbf{x}^3 = \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2} & -\frac{3}{4} \end{bmatrix}$$

Exemple

Trouvez la prochaine direction:

$$\mathbf{d}^3 = -\nabla f(\mathbf{x}^3) = -\begin{bmatrix} 0 & \frac{1}{2} & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

$$\mathbf{x}^4 = \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2} & -\frac{3}{4} \end{bmatrix} + \alpha^3 \cdot \begin{bmatrix} 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2}(\alpha^3 + 1) & -\frac{3}{4} \end{bmatrix}$$

Exemple

Trouvez α^3:

$$f(\mathbf{x}^4) = \frac{1}{4}(\alpha^3 + 1)^2 - \frac{3}{2}(\alpha^3) - \frac{3}{2}$$

$$\frac{df(\mathbf{x}^4)}{d\alpha^3} = \frac{1}{2}(\alpha^3 + 1) - \frac{9}{8} = 0$$

$$\Rightarrow \alpha^3 = \frac{5}{4}$$

Exemple

$\mathbf{x}^4$ devient:

$$\mathbf{x}^4 = \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2} & -\frac{3}{4} \end{bmatrix} + \begin{bmatrix} 0 & -\frac{5}{8} & 0 \end{bmatrix}$$

$$\therefore \mathbf{x}^4 = \begin{bmatrix} -\frac{3}{4} & -\frac{9}{8} & -\frac{3}{4} \end{bmatrix}$$

Exemple

La prochaine direction :

$$\mathbf{d}^4 = -\nabla f(\mathbf{x}^4) = -\begin{bmatrix} \frac{5}{8} & -\frac{3}{4} & \frac{5}{8} \end{bmatrix} = \begin{bmatrix} -\frac{5}{8} & \frac{3}{4} & -\frac{5}{8} \end{bmatrix}$$

$$\mathbf{x}^5 = \begin{bmatrix} -\frac{3}{4} & -\frac{9}{8} & -\frac{3}{4} \end{bmatrix} + \alpha^4 \cdot \begin{bmatrix} -\frac{5}{8} & \frac{3}{4} & -\frac{5}{8} \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{1}{4}(3+\frac{5}{2}\alpha^4) & -\frac{3}{4}(\frac{3}{2}-\alpha^4) & -\frac{1}{4}(3+\frac{5}{2}\alpha^4) \end{bmatrix}$$

Exemple

Trouvez α^4:

$$f(\mathbf{x}^5) = \frac{73}{32}(\alpha^4)^2 - \frac{43}{32}\alpha^4 - \frac{51}{64}$$

$$\frac{df(\mathbf{x}^5)}{d\alpha^4} = \frac{73}{16}\alpha^4 - \frac{43}{32} = 0$$

$$\Rightarrow \alpha^4 = {}^{43}\!/_{146}$$

Exemple

Mettre à jour $\mathbf{x}^5$:

$$\mathbf{x}^5 = \begin{bmatrix} -\frac{3}{4} & -\frac{9}{8} & -\frac{3}{4} \end{bmatrix} + \frac{43}{146} \cdot \begin{bmatrix} -\frac{5}{8} & \frac{3}{4} & -\frac{5}{8} \end{bmatrix}$$

$$\therefore \mathbf{x}^5 = \begin{bmatrix} -\frac{1091}{1168} & -\frac{66}{73} & -\frac{1091}{1168} \end{bmatrix}$$

Steepest Descent Example

Vérifions le critère de convergence sur $||\nabla f(\mathbf{x}^5)||$:

$$\nabla f(\mathbf{x}^5) = \begin{bmatrix} \dfrac{21}{584} & \dfrac{35}{584} & \dfrac{21}{584} \end{bmatrix}$$

$$\left\|\nabla f(\mathbf{x}^5)\right\| = \sqrt{\left(21/584\right)^2 + \left(35/584\right)^2 + \left(21/584\right)^2} = 0.0786$$

Steepest Descent Example

Puisque $||\nabla f(\mathbf{x}^5)|| = 0.0786$, est suffisamment proche de 0, nous décidons d'arrêter.

Noter que la réponse $\mathbf{x} = \begin{bmatrix} -\dfrac{1091}{1168} & -\dfrac{66}{73} & -\dfrac{1091}{1168} \end{bmatrix}$

Est très proche de la solution obtenue analytiquement $\mathbf{x}^* = \begin{bmatrix} -1 & -1 & -1 \end{bmatrix}$