L'optimisation pour tout et pour tous

Marcel Mongeau

Laboratoire MAIAA
(Mathématiques Appliquées, Informatique et Automatique pour l'Aérien)
mongeau@recherche.enac.fr

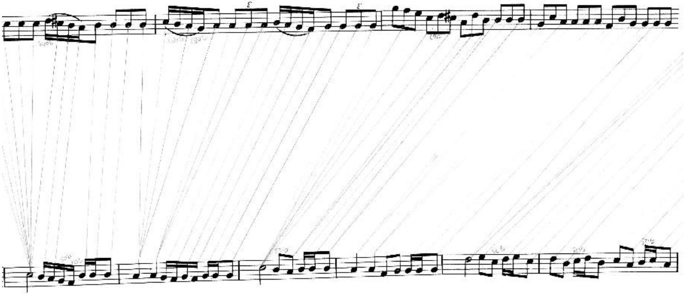
Journée de la recherche, 20 février 2014

Plan de l'exposé

- Introduction
- Modélisation
- Conditions d'optimalité
- Problèmes *boîte noire*
- Optimisation globale

Extension de la théorie de comparaison de séquences (ADN, reconnaissance de la parole etc.)

- calcul d'une dissemblance entre deux partitions
- détection de portions similaires

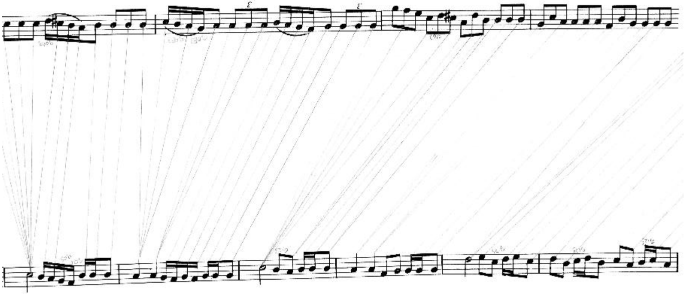


$\rightarrow$ Plus court chemin dans un graphe de toutes les substitutions, insertions/effacement, consolidation/fragmentation.

M. Mongeau, D. Sankoff, Comparison of musical sequences, *Computers & the Humanities*, 1990.

Extension de la théorie de comparaison de séquences (ADN, reconnaissance de la parole etc.)

- calcul d'une dissemblance entre deux partitions
- détection de portions similaires



$\rightarrow$ Plus court chemin dans un graphe de toutes les substitutions, insertions/effacement, consolidation/fragmentation.

M. Mongeau, D. Sankoff, Comparison of musical sequences, *Computers & the Humanities*, 1990.

Le problème d'optimisation

$$\begin{aligned} \text{Minimiser} \quad & f(x) \\ \text{sous la contrainte} \quad & x \in S \end{aligned}$$

où :

- f : fonction-objectif
- S : domaine réalisable le plus souvent représenté par un ensemble d'inégalités $c_i(x) \leq 0$, $i = 1, 2, \ldots, m$

But :

- Trouver x^* tel que $f(x^*) \leq f(x)$, pour tout $x \in V(x^*) \subseteq S$ (optimisation locale).
- Trouver x^* tel que $f(x^*) \leq f(x)$, pour tout $x \in S$ (optimisation globale).

Le problème d'optimisation

$$\begin{array}{rl} \text{Minimiser} & f(x) \\ \text{sous la contrainte} & x \in S \end{array}$$

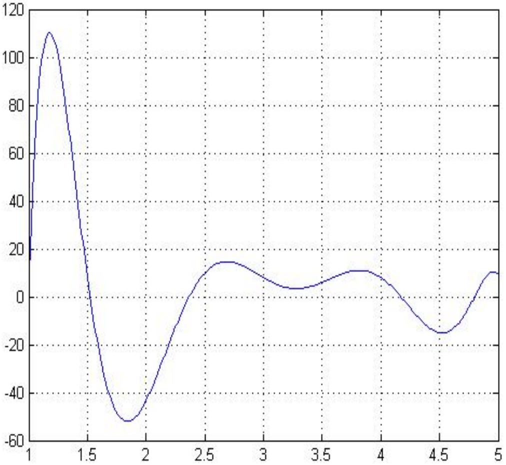
où :

- f : fonction-objectif
- S : domaine réalisable le plus souvent représenté par un ensemble d'inégalités $c_i(x) \leq 0$, $i = 1, 2, \ldots, m$

But :

- Trouver x^* tel que $f(x^*) \leq f(x)$, pour tout $x \in V(x^*) \subseteq S$ (optimisation locale).
- Trouver x^* tel que $f(x^*) \leq f(x)$, pour tout $x \in S$ (optimisation globale).

fonction ayant plusieurs minima locaux ($n = 1$)



Modélisation mathématique

Applications dans les systèmes complexes :
pas si évident de ramener le problème à la forme :

$$\begin{array}{rl} \text{Minimiser} & f(x) \\ \text{sous la contrainte} & x \in S \end{array}$$

Savoir **distinguer** :

1. **Définition** précise du problème (données vs. inconnues)
2. **Formulation** mathématique (sous la forme d'un problème d'optimisation)
3. **Résolution** (avec une méthode spécifiée par certains paramètres à ajuster)

Distinguer :

1. Paramètres (définissant le problème) : **données**
2. Variables de décision (ou d'optimisation) sur lesquelles on peut agir : **inconnues**

3 étapes :

1. Définition des variables de décision (ou d'optimisation) ($x \in \boldsymbol{R}^n$ ou $x_i \in \{0, 1\}$ ...
2. « Expression » de la (ou des) fonction-objectif(s), $f(x)$, en termes de ces variables de décision
3. « Expression » des contraintes en termes de ces variables de décision (généralement sous la forme d'inégalités $c_i(x) \leq 0$, $i = 1, 2, \ldots, m$).

Formulation mathématique du problème d'optimisation

Distinguer :

1. Paramètres (définissant le problème) : **données**
2. Variables de décision (ou d'optimisation) sur lesquelles on peut agir : **inconnues**

3 étapes :

1. Définition des variables de décision (ou d'optimisation) ($x \in \boldsymbol{R}^n$ ou $x_i \in \{0, 1\}$ ...
2. « Expression » de la (ou des) fonction-objectif(s), $f(x)$, en termes de ces variables de décision
3. « Expression » des contraintes en termes de ces variables de décision (généralement sous la forme d'inégalités $c_i(x) \leq 0$, $i = 1, 2, \ldots, m$).

Problèmes du type boîte noire

fonctions objectifs/contraintes données par simulation informatique ou expérimentale

- pas de forme mathématique explicite de f
- dérivées non disponibles
- évaluation coûteuse de f

$\longrightarrow$ mieux si on a les dérivées !

Problèmes du type boîte noire

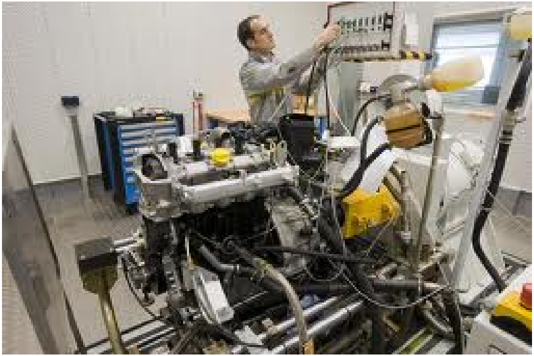
fonctions objectifs/contraintes données par simulation informatique ou expérimentale

- pas de forme mathématique explicite de f
- dérivées non disponibles
- évaluation coûteuse de f

$\longrightarrow$ mieux si on a les dérivées !

Exemple : Mise au point moteur (Renault)

- Objectif : minimiser consommation sur un cycle de conduite
- Variables d'optimisation : les $\neq$ réglages moteur
- Contraintes : seuils tolérés pour les $\neq$ polluants émis
- Chaque proposition de solution requiert essais et mesures sur banc moteur



Conditions nécessaires d'optimalité

Revenons au cas classique où on a forme explicite et dérivées

- **Optimisation sans contraintes** :
  x^* est un minimum local de $f \implies \nabla f(x^*) = 0$
  (règle de Fermat)

Condition **nécessaire d'optimalité**

$\implies$ algorithmes itératifs de descente

Une itération d'un algorithme de descente :
Au point courant, $\hat{x}$,
si $\nabla f(\hat{x}) \neq 0 \Longrightarrow$ direction de recherche
$\big($ par exemple : $d := -\nabla f(\hat{x})$ $\big)$

Conditions nécessaires d'optimalité avec contraintes

Optimisation sous contraintes $c_i(x) \leq 0, \; i = 1, 2, \ldots m$:
Un point x^* est dit *de Karush-Kuhn-Tucker* (KKT) si :

il existe un vecteur (*multiplicateurs de Lagrange*) λ tel que :

$$\left\{ \begin{array}{lrcll} \text{i)} & c_i(x^*) & \leq & 0, & i = 1, 2, \ldots m \\ \text{ii)} & -\nabla f(x^*) & = & \displaystyle\sum_{i \in \mathcal{A}(x^*)} \lambda_i \nabla c_i(x^*) & \\ \text{iii)} & \lambda_i & \geq & 0, & i \in \mathcal{A}(x^*) \end{array} \right.$$

où $\mathcal{A}(x^*) := \{i : c_i(x^*) = 0\}$ (contraintes *actives* ou saturées en x^*)

Condition nécessaire de minimalité (sous des hypothèses faibles)
$\Longrightarrow$ algorithmes d'optimisation

Conditions nécessaires d'optimalité avec contraintes

Optimisation sous contraintes $c_i(x) \leq 0,\ i = 1, 2, \ldots m$:
Un point x^* est dit *de Karush-Kuhn-Tucker* (KKT) si :

il existe un vecteur (*multiplicateurs de Lagrange*) λ tel que :

$$\begin{cases} \text{i)} & c_i(x^*) \leq 0, & i = 1, 2, \ldots m \\ \text{ii)} & -\nabla f(x^*) = \sum_{i \in \mathcal{A}(x^*)} \lambda_i \nabla c_i(x^*) & \\ \text{iii)} & \lambda_i \geq 0, & i \in \mathcal{A}(x^*) \end{cases}$$

où $\mathcal{A}(x^*) := \{i : c_i(x^*) = 0\}$ (contraintes *actives* ou saturées en x^*)

Condition nécessaire de minimalité (sous des hypothèses faibles)
$\Longrightarrow$ algorithmes d'optimisation

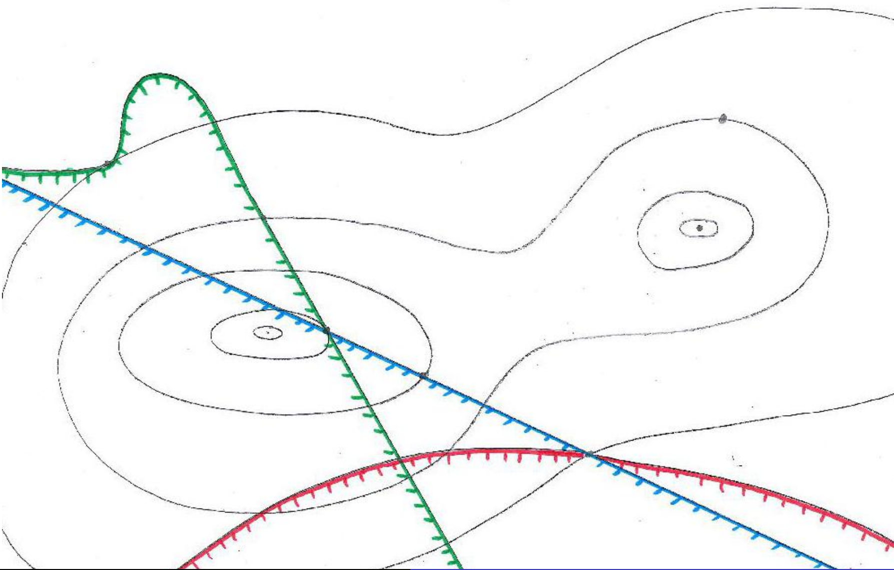
Interprétation en termes d'équilibre de forces

Un point x^* est dit *de Karush-Kuhn-Tucker* (KKT) si :

il existe un vecteur (*multiplicateurs de Lagrange*) λ tel que :

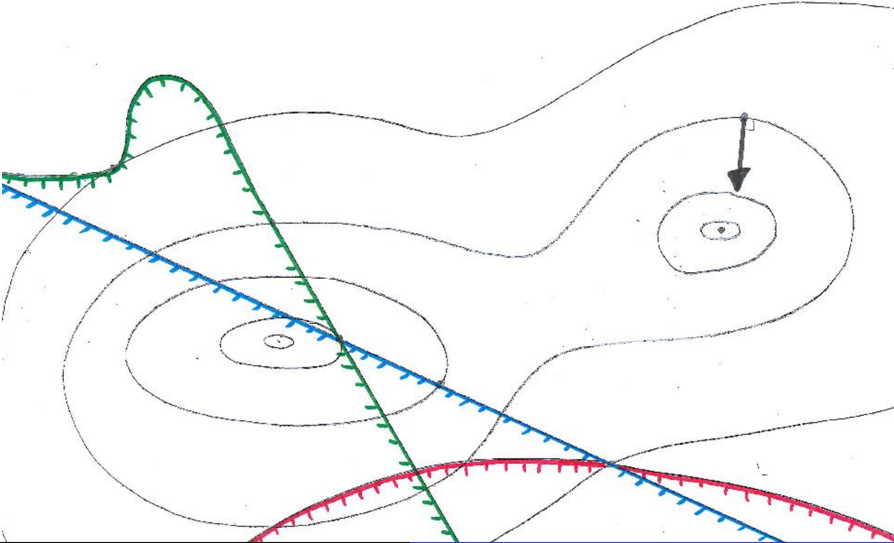
$$\left\{\begin{array}{llcll} \text{i)} & c_i(x^*) & \leq & 0, & i = 1, 2, \ldots m \\ \text{ii)} & -\nabla f(x^*) & - & \displaystyle\sum_{i \in \mathcal{A}(x^*)} \lambda_i \nabla c_i(x^*) & = 0 \\ \text{iii)} & \lambda_i & \geq & 0, & i \in \mathcal{A}(x^*) \end{array}\right.$$

Exemple 2D avec 3 contraintes d'inégalité



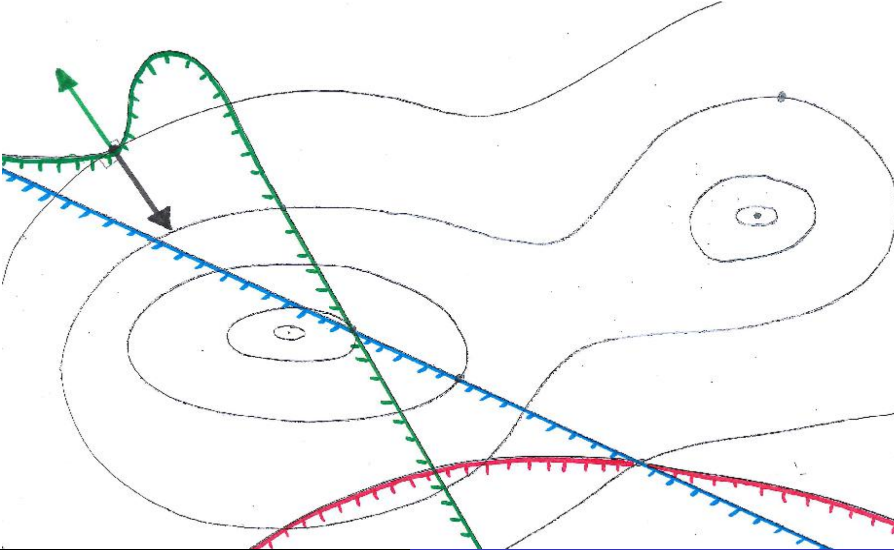
Exemple 2D avec 3 contraintes d'inégalité

Un point qui n'est pas minimum local
(avec aucune contrainte active)



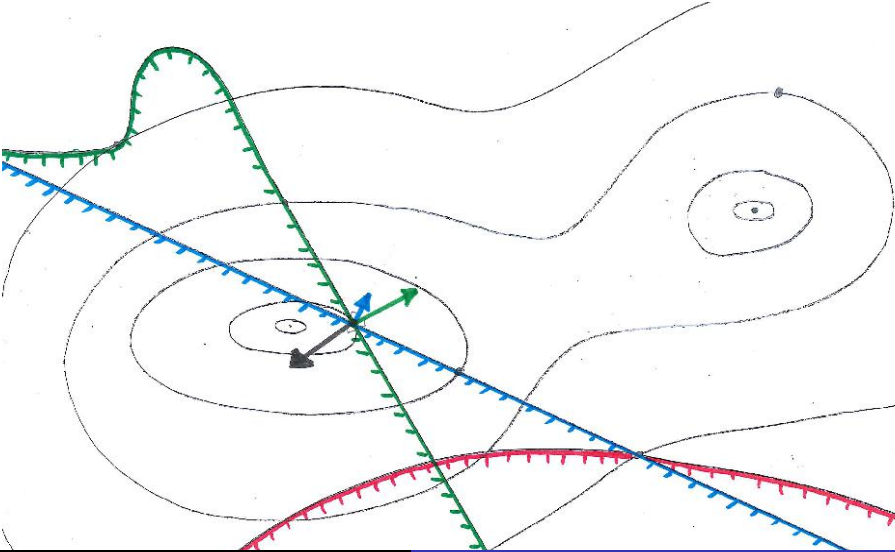
Exemple 2D avec 3 contraintes d'inégalité

minimum local (avec 1 contrainte active)



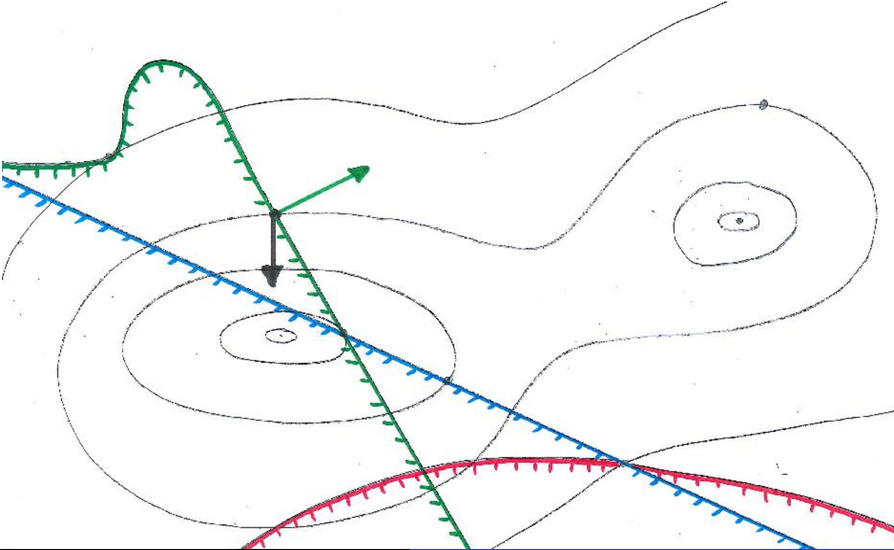
Exemple 2D avec 3 contraintes d'inégalité

minimum local (et même global) avec 2 contraintes actives

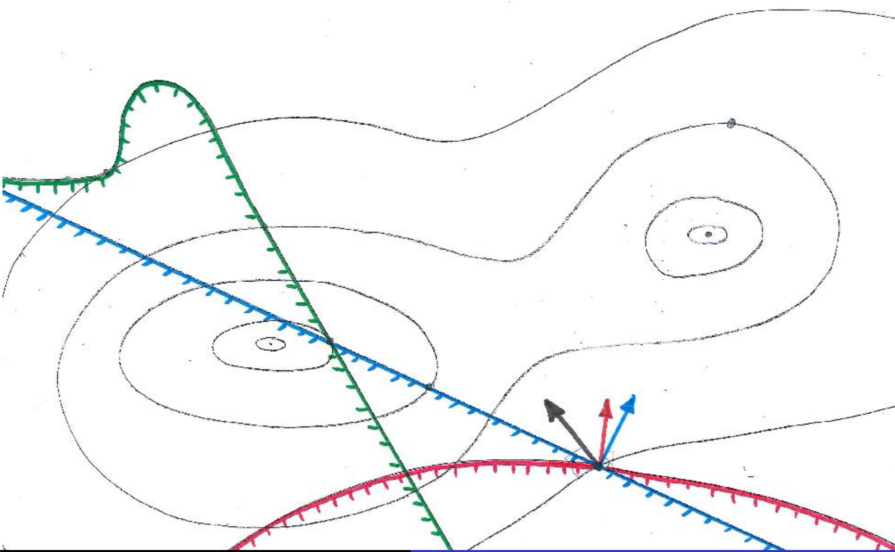


Exemple 2D avec 3 contraintes d'inégalité

Un point qui n'est pas minimum local (avec 1 contrainte active)



Un point qui n'est pas minimum local (avec 2 contraintes actives)



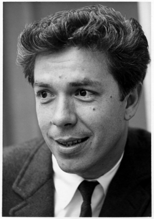
KKT : note historique

Théorème de Kuhn-Tucker

H. W. Kuhn and A. W. Tucker, Nonlinear programming, in (J. Neyman, ed.) Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, 1951, pp. 481–492.

Déjà dans :

W. **Karush**, Minima of Functions of Several Variables with Inequalities as Side Conditions, **Master's Thesis**, Department of Mathematics, University of Chicago, 1939.

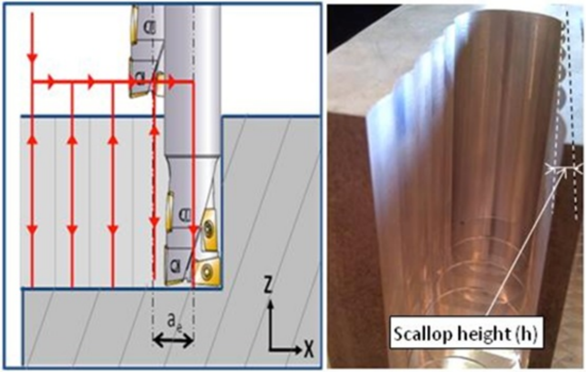

Dualité et analyse de sensibilité

- multiplicateurs de Lagrange $\lambda_i \longleftrightarrow$ variables duales
- Théorie de la dualité $\Longrightarrow$ algorithmes pratiques d'optimisation
- λ_i associé à une contrainte active $\longleftrightarrow$ information sur la valeur de la ressource associée

Exemple d'un problème d'optimisation sous contraintes

Tréflage en temps minimal de pièces aéronautiques

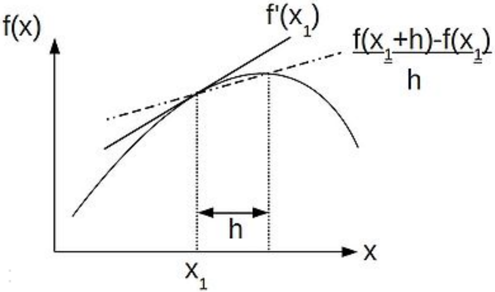
- Variables d'optimisation : trajet de l'outil, nombre de dents, vitesse de descente, avance latérale, etc.
- Contraintes : puissance, structure, précision, etc.
- Modèles mathématiques des fonctions objectif et contraintes (en termes des variables d'optimisation) avec dérivées



Sans dérivées ?

Problèmes du type boîte noire : **Que faire sans dérivées ?**

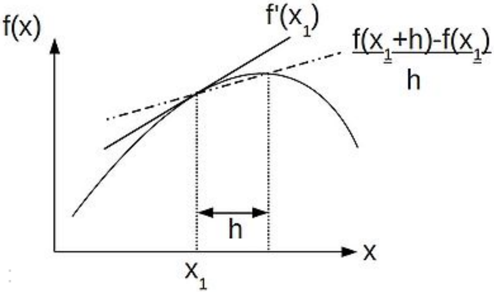
Demande $\Rightarrow$ logiciel avec approximations de $\nabla f(x)$ par différences



Désavantages :

1. problèmes numériques
    - erreurs d'annulation
    - division par petits nombres
2. nombre d'évaluations de f à chaque itération
   (chaque gradient remplacé par $n+1$ évaluations coûteuses !)

Problèmes du type boîte noire : **Que faire sans dérivées ?**
Demande ⇒ logiciel avec approximations de $\nabla f(x)$ par différences

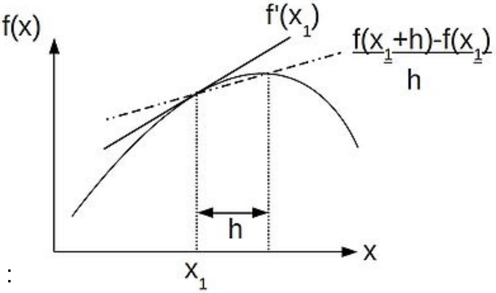


Désavantages :

1. problèmes numériques
   - erreurs d'annulation
   - division par petits nombres
2. nombre d'évaluations de f à chaque itération
   (chaque gradient remplacé par $n+1$ évaluations coûteuses !)

Sans dérivées ?

Problèmes du type boîte noire : **Que faire sans dérivées ?**
Demande $\Rightarrow$ logiciel avec approximations de $\nabla f(x)$ par différences

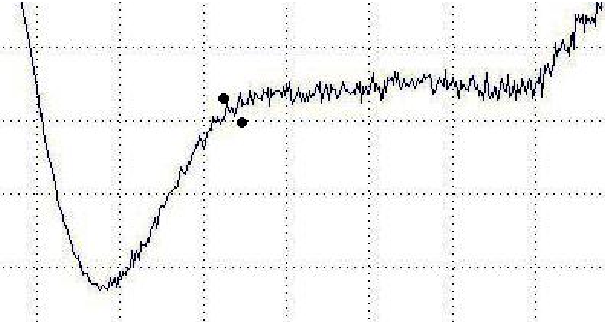


Désavantages :

1. problèmes numériques
   - erreurs d'annulation
   - division par petits nombres
2. nombre d'évaluations de f à chaque itération
   (chaque gradient remplacé par $n+1$ évaluations coûteuses !)

Sans dérivées ? (suite)

Autre inconvénient : cas bruité

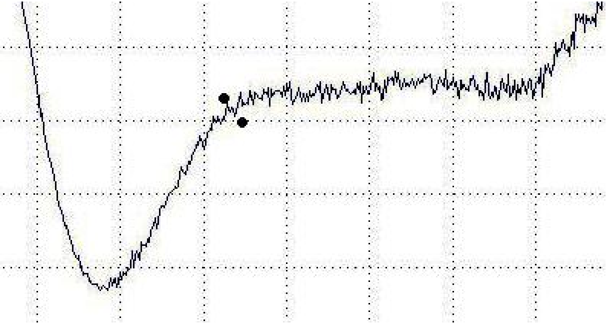


$\Longrightarrow$ information fausse !

∇f = Information locale !

$\Longrightarrow$ méthodes d'optimisation sans dérivées ?

Autre inconvénient : cas bruité



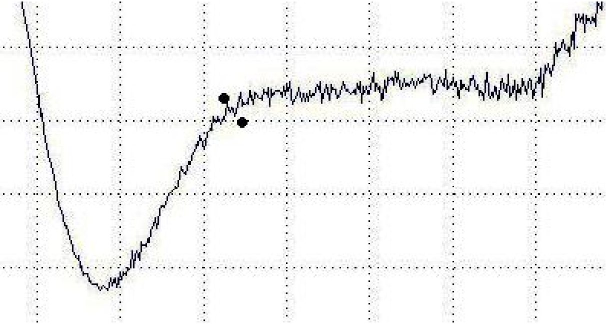
$\Longrightarrow$ information fausse !

∇f = Information locale !

$\Longrightarrow$ méthodes d'optimisation sans dérivées ?

Sans dérivées ? (suite)

Autre inconvénient : cas bruité



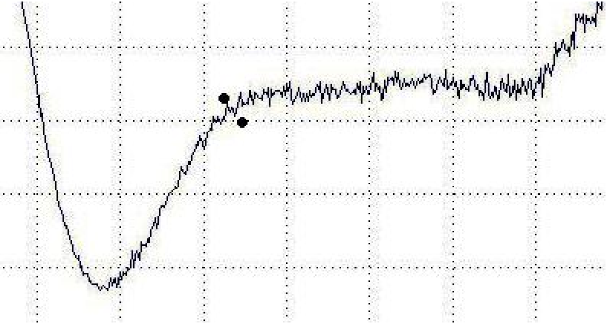
$\Longrightarrow$ information fausse !

∇f = Information locale !

$\Longrightarrow$ méthodes d'optimisation sans dérivées ?

Sans dérivées ? (suite)

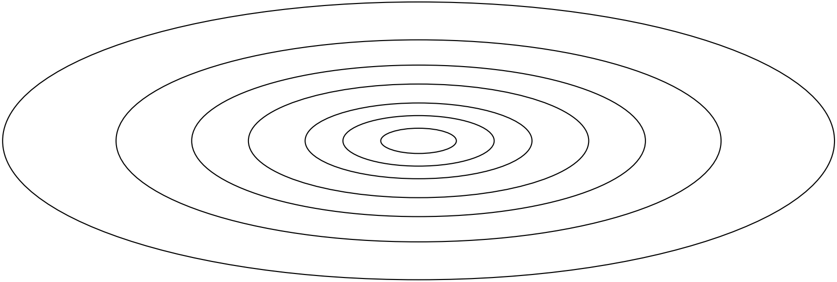
Autre inconvénient : cas bruité



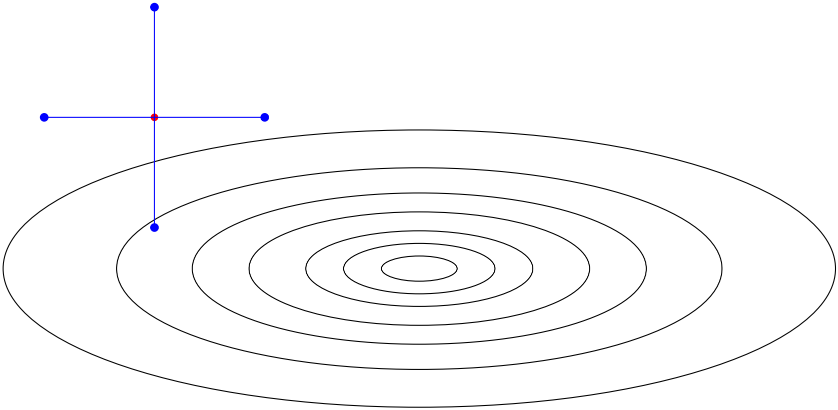
$\Longrightarrow$ information fausse !

∇f = Information locale !
$\Longrightarrow$ méthodes d'optimisation sans dérivées ?

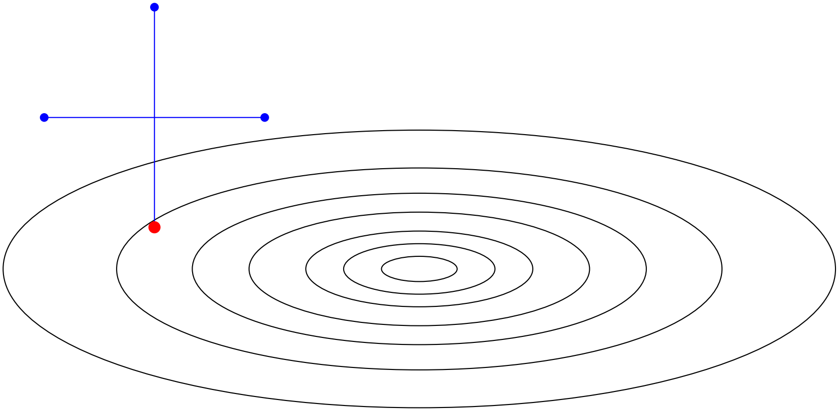
Méthodes directes : *Coordinate search*



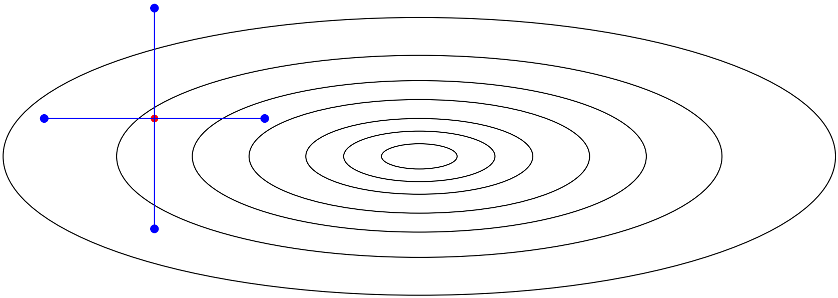
Méthodes directes : *Coordinate search*



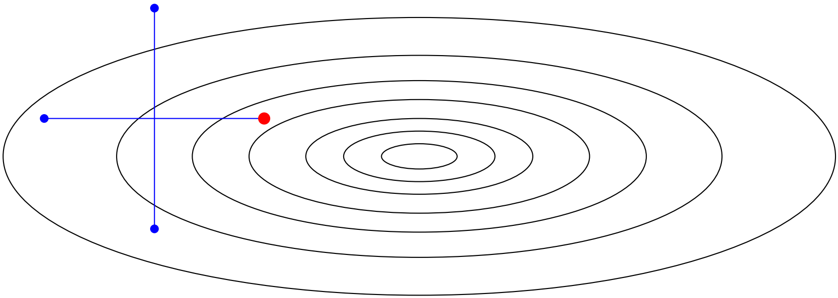
Méthodes directes : *Coordinate search*



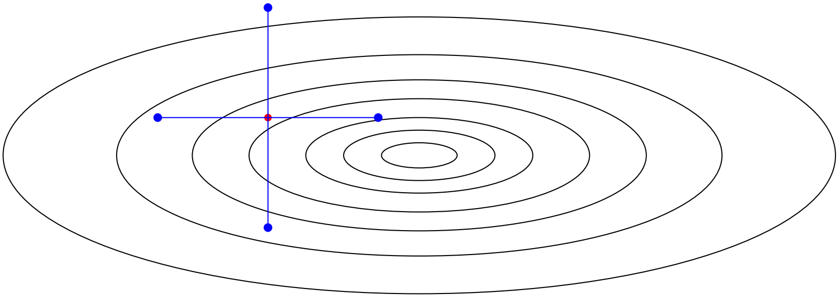
Méthodes directes : *Coordinate search*



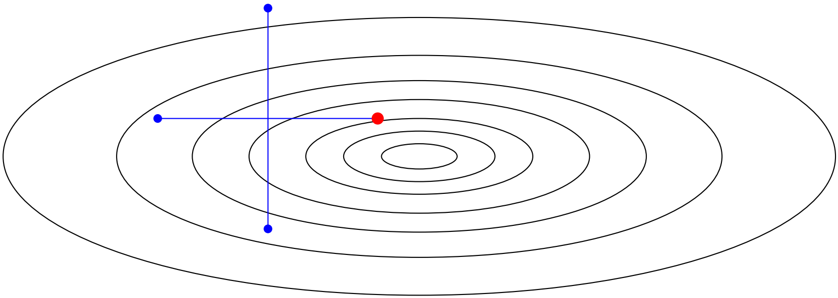
Méthodes directes : *Coordinate search*



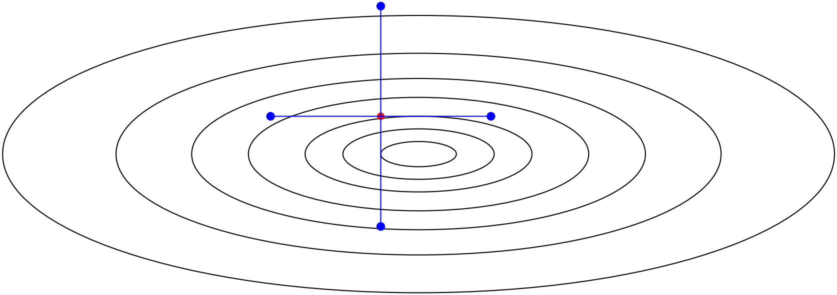
Méthodes directes : *Coordinate search*



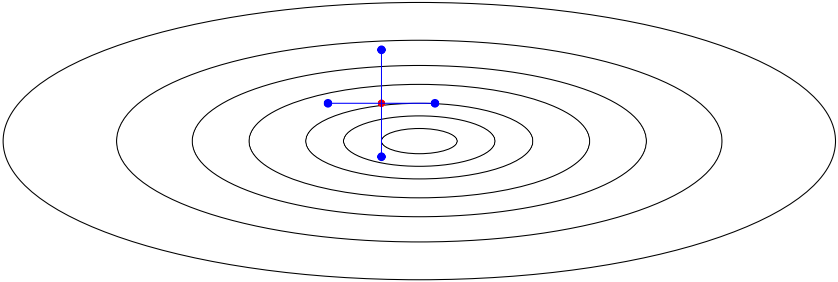
Méthodes directes : *Coordinate search*



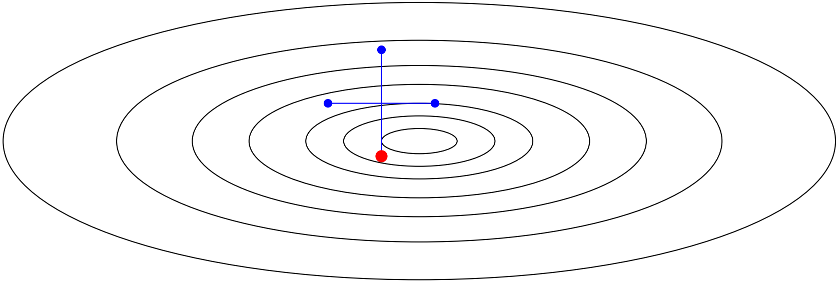
Méthodes directes : *Coordinate search*



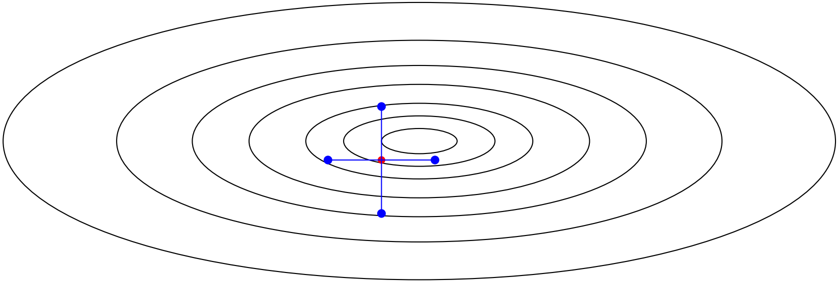
Méthodes directes : *Coordinate search*

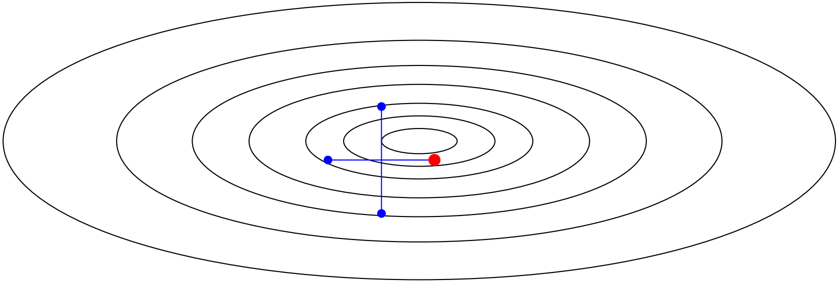


Méthodes directes : *Coordinate search*

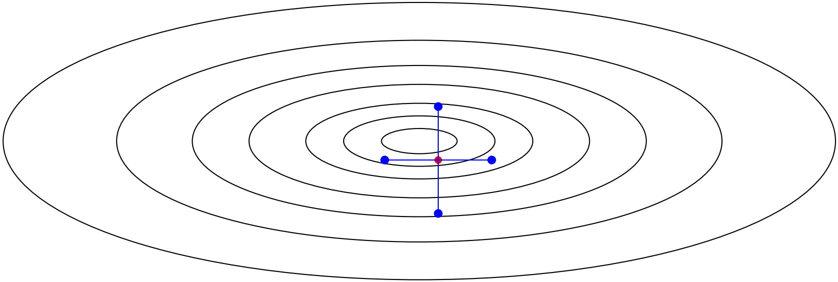


Méthodes directes : *Coordinate search*

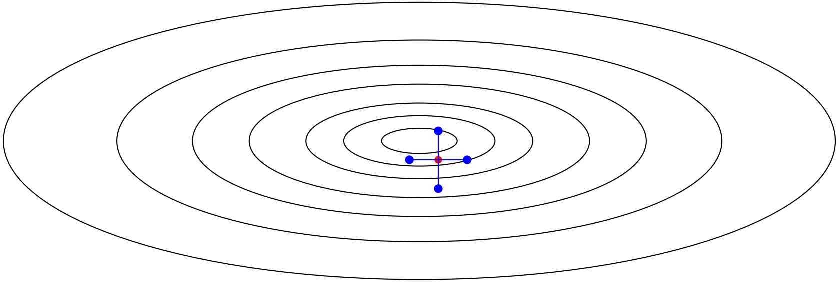




Méthodes directes : *Coordinate search*



Méthodes directes : *Coordinate search*



Méthodes directes : histoire

- Hooke-Jeeves (*coordinate search*) 1961
- Nelder-Mead (ou *non-linear simplex* ou du polytope mouvant) 1965
- Depuis 1990 : méthodes mathématiques (avec preuve de convergence) :
  - *Pattern Search* (Torczon)
  - *Derivative Free Opimization* (Conn et al.)
  - *Mesh Adaptive Direct Search* (Audet & Dennis)
  - NEWUOA, BOBYQA and LINCOA (Powell)

Cf. Livre : Conn, Scheinberg, Vicente. *Introduction to derivative-free optimization*, SIAM, 2009.

(Ne sont pas des méthodes d'optimisation globale !)

Méthodes directes : histoire

- Hooke-Jeeves (*coordinate search*) 1961
- Nelder-Mead (ou *non-linear simplex* ou du polytope mouvant) 1965
- Depuis 1990 : méthodes mathématiques (avec preuve de convergence) :
  - *Pattern Search* (Torczon)
  - *Derivative Free Opimization* (Conn et al.)
  - *Mesh Adaptive Direct Search* (Audet & Dennis)
  - NEWUOA, BOBYQA and LINCOA (Powell)

Cf. Livre : Conn, Scheinberg, Vicente. *Introduction to derivative-free optimization*, SIAM, 2009.

(Ne sont pas des méthodes d'optimisation globale !)

Exemple d'application : Équilibre des phases en génie chimique (Edinburgh Petroleum Services)

- Données : mélange de composants chimiques
- Variables d'optimisation :
  - se sépare en combien de phases (mélanges distincts) ? (discret)
  - de quel type de phases (liquide, vapeur) ? (discret)
  - en quelles proportions ? composition de chaque phase ? (continue)
- Objectif : minimiser une fonction thermodynamique
- Contraintes : lois de conservation chimiques

Seul un optimum global donnera une information sur le réservoir pétrolier

Exemple d'application : Équilibre des phases en génie chimique (Edinburgh Petroleum Services)

- Données : mélange de composants chimiques
- Variables d'optimisation :
  - se sépare en combien de phases (mélanges distincts) ? (discret)
  - de quel type de phases (liquide, vapeur) ? (discret)
  - en quelles proportions ? composition de chaque phase ? (continue)
- Objectif : minimiser une fonction thermodynamique
- Contraintes : lois de conservation chimiques

Seul un optimum global donnera une information sur le réservoir pétrolier

Méthodes d'optimisation globale

Optimisation globale : problèmes très difficiles

- **Méthodes déterministes**
  - **exploitant** une **structure** particulière du problème (ex : analyse par intervalles, B&B, méthodes de décomposition, etc.)
    $\longrightarrow$ l'idéal
  - **heuristiques** (ex : Nelder & Mead, pattern search, DFO)
    $\longrightarrow$ lorsqu'on n'a pas le choix
- **Méthodes stochastiques** $\longrightarrow$ cas désespérés !
  - **à deux phases** (ex : *clustering*)
  - **méta-heuristiques** : $\longrightarrow$ tâcher d'y intégrer connaissance propre au domaine d'application
    - algorithmes évolutionnaires (ex : génétiques, fourmi, essaims particulaires etc.)
    - recuit simulé
    - recherche avec tabous
  - **surfaces de réponse** (ex : krigeage)

Méthodes d'optimisation globale

Optimisation globale : problèmes très difficiles

- **Méthodes déterministes**
  - **exploitant** une **structure** particulière du problème (ex : analyse par intervalles, B&B, méthodes de décomposition, etc.)
    $\longrightarrow$ l'idéal
  - **heuristiques** (ex : Nelder & Mead, pattern search, DFO)
    $\longrightarrow$ lorsqu'on n'a pas le choix
- **Méthodes stochastiques** $\longrightarrow$ cas désespérés !
  - **à deux phases** (ex : *clustering*)
  - **méta-heuristiques** : $\longrightarrow$ tâcher d'y intégrer connaissance propre au domaine d'application
    - algorithmes évolutionnaires (ex : génétiques, fourmi, essaims particulaires etc.)
    - recuit simulé
    - recherche avec tabous
  - **surfaces de réponse** (ex : krigeage)

Recuit simulé (*simulated annealing*)

Réf en métallurgie : lent refroidissement d'un solide en une configuration d'énergie minimale

On se donne :

$T : \mathbb{N} \rightarrow]0, +\infty[$ décroissante

$x_T(t) =: s$: la solution courante

$V(s) \subseteq S - \{s\}$: un voisinage de s

$p_{ss'}$: la proba de choisir s' dans $V(s)$

Recuit simulé (*simulated annealing*)

Réf en métallurgie : lent refroidissement d'un solide en une configuration d'énergie minimale

On se donne :

$T : \mathbb{N} \to]0, +\infty[$ décroissante

$x_T(t) =: s$: la solution courante

$V(s) \subseteq S - \{s\}$: un voisinage de s

$p_{ss'}$: la proba de choisir s' dans $V(s)$

Recuit simulé (*simulated annealing*) (suite)

Algorithme (itération t) :

1. Choisir $s' \in V(s)$ aléatoirement
2.
   - Si $f(s') \leq f(s)$, $x_T(t+1) =: s'$ (meilleur)
   - sinon, $x_T(t+1) =: \begin{cases} s' & \text{avec proba } e^{\frac{f(s)-f(s')}{T(t)}} \\ s & \text{sinon} \end{cases}$

Recuit simulé (*simulated annealing*) (suite)

Algorithme (itération t) :

1. Choisir $s' \in V(s)$ aléatoirement
2. - Si $f(s') \leq f(s)$, $x_T(t+1) =: s'$ (meilleur)
   - sinon, $x_T(t+1) =: \begin{cases} s' & \text{avec proba } e^{\frac{f(s)-f(s')}{T(t)}} \\ s & \text{sinon} \end{cases}$

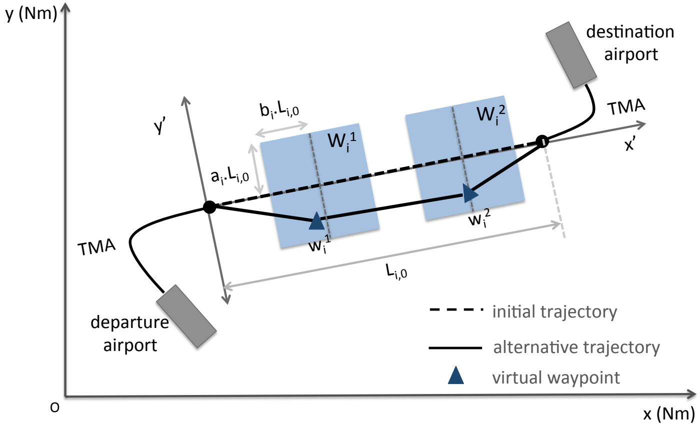
Recuit simulé (*simulated annealing*) (suite)

Algorithme (itération t) :

1. Choisir $s' \in V(s)$ aléatoirement
2.
   - Si $f(s') \leq f(s)$, $x_T(t+1) =: s'$ (meilleur)
   - sinon, $x_T(t+1) =: \begin{cases} s' & \text{avec proba } e^{\frac{f(s)-f(s')}{T(t)}} \\ s & \text{sinon} \end{cases}$

Recuit simulé (*simulated annealing*)

Exemple d'application (et de voisinage) : représentation d'une trajectoire et d'une trajectoire candidate voisine.



Conclusions

- Exemples d'applications $\Rightarrow$ omniprésence de l'optimisation
- Importance de la modélisation mathématique
- Utilité des dérivées
- Si on sait évaluer, on peut optimiser (avec ou sans dérivées)