

# Quelques méthodes classiques d'optimisation locale

# Présentation

- Recherche aléatoire Voisinage
- Amélioration itérative
- Méthode du polytope de Nelder-Mead

# Principe

Le principe d'une méthode d'optimisation locale est le suivant :

à partir d'une solution de départ  $x_0$ , considérée temporairement comme étant la valeur minimale  $x_{min}$ ,

on engendre par transformations élémentaires une suite finie de voisins.

# Recherche aléatoire

C'est la plus simple des méthodes stochastiques.

Répéter

Cette méthode consiste à tirer à chaque itération une solution au hasard.

La fonction objectif  $f$  est évaluée en ce point.

La nouvelle valeur est comparée à la précédente.

Si elle est meilleure que la précédente, alors

cette valeur est enregistrée, ainsi que la solution correspondante

Sinon on repart du point précédent

Recommencer le procédé, jusqu'à ce que les conditions d'arrêt soient atteintes.

# Pseudocode d'un algorithme de recherche aléatoire

$\mathbf{x} :=$  solution aléatoire

$f_{min} := f(\mathbf{x})$

$\mathbf{x}_{min} := \mathbf{x}$

REPETER

$\mathbf{x} :=$  solution aléatoire

SI  $f(\mathbf{x}) < f_{min}$

$f_{min} := f(\mathbf{x})$

$\mathbf{x}_{min} := \mathbf{x}$

FIN SI

JUSQU'A conditions d'arrêt satisfaites

# Amélioration itérative

- Cette méthode vise à déterminer une solution  $s(\mathbf{x})$  dans le voisinage de la solution courante  $\mathbf{x}$ , telle que  $f(s(\mathbf{x})) < f_{min}$  ( $f_{min}$  désigne la valeur minimale courante de  $f$ ).
- La méthode consiste à engendrer, à chaque itération, un  $N$ -échantillon, suivant un procédé aléatoire ou cyclique, ou suivant une loi de distribution uniforme, dans le voisinage de la solution courante  $\mathbf{x}$ .
- La fonction objectif  $f$  est évaluée en chaque point de l'échantillon, et la solution  $\mathbf{x}'$  correspond à la plus petite valeur de  $f$  obtenue,

$$f(\mathbf{x}') = f(s(\mathbf{x})) = \min_{1 \leq i \leq N} [f(s_i(\mathbf{x}))].$$

Cette nouvelle valeur  $f(\mathbf{x}')$  est comparée à la valeur minimale courante  $f_{min}$ . Si elle est meilleure, cette valeur est enregistrée, ainsi que la solution correspondante, et le processus continue. Sinon l'algorithme prend fin, on a atteint un minimum local

Après avoir atteint un minimum local, cette procédure peut repartir d'un autre point pris au hasard.

# Pseudocode de la méthode d'amélioration itérative

$\mathbf{x} :=$  solution aléatoire

$f_{min} := f(\mathbf{x})$

$\mathbf{x}_{min} := \mathbf{x}$

REPETER

engendrer un  $N$ -échantillon  $s_i(\mathbf{x}) \in$  voisinage  $S(\mathbf{x})$  et  
calculer  $f(s(\mathbf{x})) = \min_{1 \leq i \leq N} [f(s_i(\mathbf{x}))]$

SI  $f(s(\mathbf{x})) < f_{min}$

$f_{min} := f(s(\mathbf{x}))$

$\mathbf{x}_{min} := s(\mathbf{x})$

SINON

sortir de REPETER

FIN SI

FIN REPETER

# Méthode du polytope de Nelder-Mead

- C'est une méthode d'optimisation locale [Neld65] [Wood85] qui est fréquemment utilisée.
- Cette méthode déterministe est dite "directe" : elle tente de résoudre le problème en utilisant directement la valeur de la fonction objectif, sans faire appel à ses dérivées.
- Cette méthode est surtout appréciée pour sa robustesse, sa simplicité de programmation, sa faible consommation de mémoire (peu de variables) et son faible temps de calcul.
- Cet algorithme est robuste car il est très tolérant aux bruits dans les valeurs de la fonction objectif.
- En conséquence, la fonction n'a pas besoin d'être calculée exactement et il est possible d'avoir recours à une approximation de la valeur de la fonction.

# Suite

- Contrairement aux autres méthodes qui démarrent à partir d'un point initial, la méthode de Nelder-Mead utilise un "polytope" de départ.
- Un polytope est une figure géométrique de  $(n+1)$  points,  $n$  étant la dimension du problème.
- Le polytope de départ est obtenu par le tirage aléatoire d'un point  $\mathbf{x}_1$  dans l'espace solution, les autres points  $\mathbf{x}_i$  sont choisis de manière à former une base, généralement une base orthogonale :

$$\mathbf{x}_i = \mathbf{x}_1 + \lambda \mathbf{e}_i,$$

Où :

- $\mathbf{e}_i, i=2, \dots, n+1$  sont des vecteurs unitaires linéairement indépendants,
- $\lambda$  est généralement une constante, adaptée à la caractéristique du problème (domaine de variation des différentes composantes).

On peut choisir des  $\lambda_i$  différents pour chaque vecteur de direction.

Mais généralement, pour définir une base orthonormée, on prend un seul  $\lambda_i$  égal à l'unité, et le produit scalaire entre deux vecteurs unitaires  $\mathbf{e}_i$  est nul.

Le polytope  $P_0$  défini par les sommets  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n+1}$ , est noté  $P_0 = \langle \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n+1} \rangle$ .

# Suite

A chaque itération de l'algorithme du polytope,  $n+1$  points sont utilisés pour déterminer un pas d'essai.

Les points  $\mathbf{x}_i$  sont ordonnés de manière à avoir  $f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \dots \leq f(\mathbf{x}_{n+1})$ .

Des points de test sont obtenus en utilisant de très simples opérations algébriques, qui se traduisent par des transformations géométriques élémentaires :

- réflexion,
- contraction,
- expansion,
- multicontraction appelée aussi rétrécissement),

Les points engendrés sont acceptés ou rejetés en fonction de la valeur de la fonction objectif.

# Suite

Le polytope se transforme, il s'étend, se contracte, à chaque mouvement.

Ainsi il s'adapte à l'allure de la fonction, jusqu'à ce qu'il s'approche de l'optimum.

Pour déterminer la transformation adéquate, la méthode utilise uniquement la valeur de la fonction objectif aux points considérés.

A chaque transformation, le plus mauvais point courant  $x_h$  du polytope est remplacé par le nouveau point déterminé.

Les conditions d'arrêt de l'algorithme dépendent de la différence de valeur de la fonction objectif entre le meilleur et le plus mauvais point ( différence est inférieure à un certain seuil

# Suite

Soit  $\bar{\mathbf{x}}$  le centre de gravité des points  $\mathbf{x}_i$ , avec  $i \neq n+1$ , formant le polytope  $P$  et nous écrivons  $[\mathbf{x}_i, \mathbf{x}_j]$  pour définir la distance de  $\mathbf{x}_i$  à  $\mathbf{x}_j$ .

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$$

Un exemple d'application de l'algorithme du polytope à une fonction à 3 variables est donné sur la figure 1.4.a, est centre de gravité du triangle formé par le triplet  $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ .

# Réflexion

Le point réfléchi  $\mathbf{x}_r$  est déterminé en calculant le symétrique, par rapport au centre de gravité, du plus mauvais point  $\mathbf{x}_{n+1}$  du polytope, et il est obtenu en utilisant la relation suivante :

$$\mathbf{x}_r = (1 + \alpha) \bar{\mathbf{x}} - \alpha \mathbf{x}_{n+1}$$

où  $\alpha$  est une constante positive, généralement égale à 1, appelée *constante de réflexion*.

Ainsi le point  $\mathbf{x}_r$  se trouve, sur la droite joignant  $\mathbf{x}_{n+1}$  et  $\bar{\mathbf{x}}$ , tel que  $[\mathbf{x}_r, \bar{\mathbf{x}}] = \alpha \cdot [\mathbf{x}_{n+1}, \bar{\mathbf{x}}]$

Le point réfléchi est accepté si  $f(\mathbf{x}_1) \leq f(\mathbf{x}_r) < f(\mathbf{x}_n)$ , et à l'itération suivante on réitère la réflexion avec le nouveau polytope défini par  $\langle \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \mathbf{x}_r \rangle$ .

On note que  $\mathbf{x}_r$  n'est pas ordonné par rapport aux autres  $\mathbf{x}_i$ .

Si la valeur de la fonction objectif au point réfléchi est inférieure à celle au point  $\mathbf{x}_1$ , i.e.  $f(\mathbf{x}_r) < f(\mathbf{x}_1)$ , alors l'essai précédent a produit un "bon" point, et on continue dans cette direction par une expansion du pas.

# L'expansion

L'expansion représentée est déterminée par :

$$\mathbf{x}_e = \gamma \mathbf{x}_r + (1 - \gamma) \bar{\mathbf{x}}$$

La *constante d'expansion*  $\gamma$ , dont la valeur est supérieure à l'unité, désigne le

rapport de la distance  $[\mathbf{x}_e, \bar{\mathbf{x}}]$  à la distance  $[\mathbf{x}_r, \bar{\mathbf{x}}]$

Le point ainsi obtenu  $\mathbf{x}_e$  est accepté si  $f(\mathbf{x}_e) < f(\mathbf{x}_r)$ , sinon seul le point réfléchi  $\mathbf{x}_r$  est accepté, et le point  $\mathbf{x}_e$  est rejeté.

On ordonne le nouveau polytope ainsi obtenu (les points  $\mathbf{x}_i$ ), et on réitère le procédé.

Si le point réfléchi  $\mathbf{x}_r$  est moins bon que  $\mathbf{x}_n$ , i.e.  $f(\mathbf{x}_n) \leq f(\mathbf{x}_r)$ , alors un pas de contraction est à effectuer avec le meilleur point :  $\mathbf{x}_r$  ou  $\mathbf{x}_{n+1}$ .

# La contraction

Si le plus mauvais point  $\mathbf{x}_{n+1}$  est au moins aussi bon que le point réfléchi  $\mathbf{x}_r$ , i.e.  $f(\mathbf{x}_{n+1}) \leq f(\mathbf{x}_r)$ , alors la contraction interne est calculée par :

$$\mathbf{x}_c = \beta \mathbf{x}_{n+1} + (1 - \beta) \bar{\mathbf{x}} \quad ,$$

Sinon, la contraction, appelée contraction externe, est calculée comme suit :

$$\mathbf{x}_c' = \beta \mathbf{x}_r + (1 - \beta) \bar{\mathbf{x}} \quad .$$

La valeur du *coefficient de contraction*  $\beta$  se situe entre 0 et 1, et il désigne le rapport de la distance  $[\mathbf{x}_c, \bar{\mathbf{x}}]$  à la distance  $[\mathbf{x}_{n+1}, \bar{\mathbf{x}}]$

Le point ainsi obtenu  $\mathbf{x}_c$  est accepté si  $f(\mathbf{x}_c) \leq f(\mathbf{x}_{n+1})$ .

Pour chaque contraction rejetée, i.e. ( $f(\mathbf{x}_{n+1}) < f(\mathbf{x}_c)$ ), on remplace tous les  $\mathbf{x}_i$  du polytope par  $\frac{\mathbf{x}_j + \mathbf{x}_i}{2}$  on obtient ainsi le "rétrécissement" (ou "multi-contraction") du polytope et le processus redémarre.

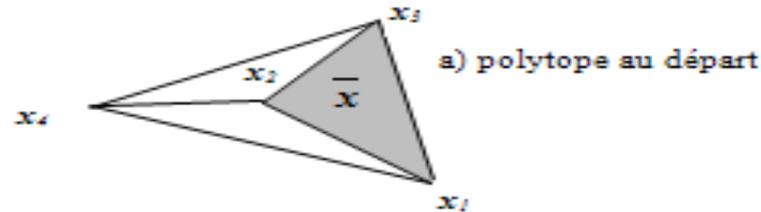
# Le critère d'arrêt

- Le critère d'arrêt est une mesure de déplacement du polytope d'une itération  $k$  à l'itération suivante  $(k+1)$ . L'algorithme s'arrête lorsque :

$$\frac{1}{n} \sum_{i=1}^n \left\| \mathbf{x}_i^k - \mathbf{x}_i^{k+1} \right\|^2 < \varepsilon$$

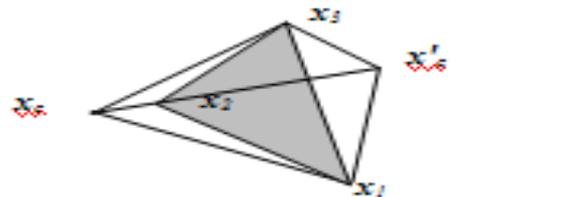
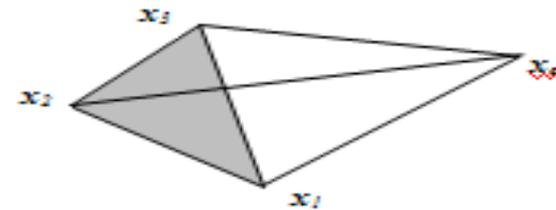
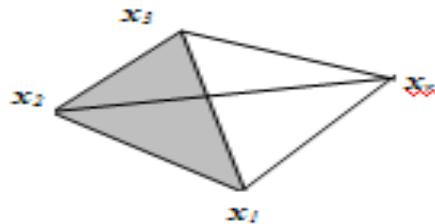
- où  $\mathbf{x}^{k+1}$  est le sommet remplaçant le sommet  $\mathbf{x}^k$  à l'itération  $(k+1)$ , et  $\varepsilon$  est un nombre réel positif donné.

# Les mouvements possibles dans la méthode du polytope de Nelder-Mead dans $\mathbb{R}^3$



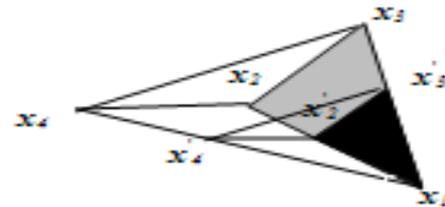
$x_1$  : représente le sommet où la valeur de la fonction objectif est la plus basse  
 $x_4$  : représente le sommet où la valeur de la fonction objectif est la plus élevée

Soit  $\bar{x}$  le centre de gravité des points  $x_i$  :  $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$



$$x_{4e} = \beta x_{4e+1} + (1 - \beta) \bar{x}$$

$$x'_{4e} = \beta x_{4e} + (1 - \beta) \bar{x}$$



# Pseudocode de la méthode du polytope de Nelder-Mead

*POLYT* tableaux de taille  $n+1$

*POLYT* := initialiser polytope

$\alpha, \beta, \gamma$  := initialiser

*SIMPL* := ordonner polytope pour obtenir  $P_0 = \langle x_1, x_2, \dots, x_{n+1} \rangle$

REPETER

$$\bar{x} := \frac{1}{n} \sum_{i=1}^n x_i$$

*réflexion* ( $x_r = (1 + \alpha)\bar{x} - \alpha x_{n+1}$ )

$x^k := x_r$

SI *réflexion acceptée* ( $f(x_r) < f(x_n)$ ) alors

SI ( $f(x_r) < f(x_1)$ )

*expansion* ( $x_e = \gamma x_r + (1 - \gamma)\bar{x}$ )

SI *expansion acceptée* ( $f(x_e) < f(x_n)$ ) alors  $x^k := x_e$

SINON  $x^t := x_{n+1}$  (pour contraction interne)

SI ( $f(x_r) \leq f(x^t)$ ) alors  $x^t := x_r$  (pour contraction externe)

*contraction* ( $x_c = \beta x^t + (1 - \beta)\bar{x}$ )

SI *contraction acceptée* ( $f(x_c) \leq f(x_n)$ ) alors  $x^k := x_c$

SINON *rétrécissement du simplexe*

$$x_j := \frac{(x_j + x_1)}{2} \quad \text{pour } j = 2, \dots, n$$

$$x^k := \frac{(x_{n+1} + x_1)}{2}$$

Trier  $f(x_1), f(x_2), \dots, f(x^k)$  pour obtenir  $P_k = \langle x_1, x_2, \dots, x_{n+1} \rangle$

JUSQU'A conditions d'arrêt satisfaites

# Algorithme hybride

