MÉTHODES D'OPTIMISATION STOCHASTIQUES

Xavier Olive

avec l'aide de Nathalie Bartoli, Gaspard Berthelin, Alexandre Gondran, Olivier Poitou et Rémy Priem

- appréhender les situations dans lesquelles les méthodes stochastiques (*métaheuristiques*) sont pertinentes ;
- comprendre les grandes familles d'algorithmes d'optimisation stochastique ;
- manipuler (parfois coder) trois algorithmes parmi les plus utilisés et les essayer sur des problèmes « jouets ».

Évaluation Chaque binôme devra étudier une méthode d'optimisation non couverte par le cours et la présenter devant le groupe lors de la dernière séance (le 3 décembre).

DÉROULÉ DES SÉANCES

- 7 novembre :
  Recuit simulé – *Simulated annealing*
- 12 novembre :
  Algorithmes génétiques – *Genetic algorithms*
- 20 novembre :
  CMA-ES
- 3 décembre *Présentations*

BRAINSTORMING

- Ant colony optimisation
- Artificial bee colony
- Genetic algorithms (advanced version)
- Harmony search
- Particle swarm optimisation
- Shuffled frog leaping
- Stochastic (constraint based) local search
- Tabu search
- Cross-entropy method ?

- Ant colony optimisation
- Artificial bee colony
- Genetic algorithms (advanced version)
- Harmony search
- Particle swarm optimisation
- Shuffled frog leaping
- Stochastic (constraint based) local search
- Tabu search
- Cross-entropy method ?

Consigne : Présenter chaque méthode de manière intuitive. Une démonstration sur un problème jouet sera valorisée (codée par vos soins *ou* récupérée sur le net)

INTRODUCTION

$$\min_x f(x)$$

Notations :

- x est un vecteur de *variables*, ou d'inconnues ;
- f est une *fonction objectif*, ou fonction d'évaluation.

La *modélisation* d'un problème consiste à identifier des variables et une fonction objectif.

$$\min_x f(x) \text{ en respectant } c(x)$$

Notations :

- c est une fonction *contrainte*, à valeurs dans $\{\top, \bot\}$.

Dans le cas d'une optimisation non contrainte, on a

$$\forall x : c(x)$$

Les variables x peuvent être à valeurs dans :

- un ensemble *continu*, comme $\mathbb{R}$;
- un ensemble *discret*, comme $\mathbb{Z}$;
- un ensemble *fini*, comme $\{0, 1\}$.

Les variables d'un problème peuvent prendre leurs valeurs sur des ensembles de nature différente.

- Les méthodes de résolution adaptées à certains problèmes garantissent de trouver le minimum global s'il existe.
- Pour d'autres types de problèmes, il est difficile de savoir qu'on a trouvé un minimum global : les algorithmes convergent vers des minima locaux.

- Les méthodes de résolution peuvent toujours converger de la même manière : on parle de méthode déterministe.
- Si deux exécutions du programme convergent différemment, on parle alors de méthode stochastique.
  Attention à la graine de votre programme !

- Programmation non linéaire (gradient, gradient conjugué, BGFS, etc.) ;
- Programmation linéaire (simplexe ; Dantzig, 1947) ;
- Programmation linéaire en nombres entiers (MILP) ;
- Programmation par contraintes (CSP)

- Programmation non linéaire (gradient, gradient conjugué, BGFS, etc.) ;
- Programmation linéaire (simplexe ; Dantzig, 1947) ;
- Programmation linéaire en nombres entiers (MILP) ;
- Programmation par contraintes (CSP)

- Pour MILP et CSP, la taille du problème et/ou des domaines peut être un facteur fortement limitant.

C'est la solution de secours quand les méthodes classiques sont inefficaces : on obtient une solution de bonne qualité en un temps raisonnable.

- On cherche *au hasard* x_{k+1} dans le voisinage de x_k, en combinant aspects d'**optimisation** et d'**exploration**.
- Métaphores de comportements observés dans la nature.

- Métaheuristique = **méthode stochastique générique** à adapter à chaque problème.

Ces méthodes sont efficaces quand :

- la fonction d'évaluation est de type boîte noire ;
- le problème se formalise bien en MILP/CSP mais le domaine est trop grand (cf. complexité) ;
- la fonction d'évaluation est bruitée ($\rightarrow$ CMA-ES) ;

Deux grandes classes de méthodes :

- les **méthodes par trajectoire**: on manipule un élément à la fois dans l'espace des solutions pour tenter de construire une trajectoire qui converge vers un optimum.
  - recherche locale, recuit simulé, etc.
- les **méthodes par population**: on manipule plusieurs éléments à la fois ; les meilleurs éléments guident la génération de la population à l'itération suivante.
  - algorithmes génétiques, essaims particulaires, etc.

RECUIT SIMULÉ

Méthode par trajectoire inspirée de la métallurgie

- On choisit au hasard une solution initiale à **énergie** E_0 et une **température initiale** élevée T_0;
- À chaque itération, on choisit un voisin de l'état précédent, qui correspond à une variation d'énergie ΔE:
  - si $\Delta E < 0$, la modification est appliquée ;
  - sinon, on l'accepte avec une probabilité $e^{-\frac{\Delta E}{T}}$.
- La température suit une loi décroissante.

L'algorithme étant facile, nous allons le coder tout en nous assurant d'avoir bien compris où se situe la vraie difficulté :

- le choix des paramètres ;
- la définition d'un voisinage.

Nous allons le développer sur une fonction 2D générée aléatoirement, puis le tester sur un plus gros problème.

ALGORITHMES GÉNÉTIQUES

Méthode par population, l'évolution selon Darwin

- On choisit au hasard une population[1] initiale de taille fixée ;
- **Sélection**: à chaque itération, on choisit des éléments parmi les mieux évalués ;
- **Croisement**: à chaque itération, on hybride les éléments sélectionnés pour générer la génération suivante ;
- **Mutation**: à chaque itération, une partie de la population est légèrement modifiée pour la génération suivante ;

1. On parle ici de *chromosome*.

Aujourd'hui encore, nous allons coder cette méthode pour :

- essayer différentes stratégies de sélection, croisement et mutation sur un problème jouet ;
- trouver des tailles de population, taux de croisement et de sélection efficaces pour notre problème.

```
initialize population
foreach iteration :
    evaluate population
    append the k best elements into your new population
    for a1, a2 in selection(population) :
        b1, b2 = cross(a1, a2)
        ”sometimes” mutate b1 and/or b2
        append b1, b2 to your new population
```

- **Tournoi :**
  Tirer deux éléments au hasard, garder le meilleur des deux.

- **Roulette russe :**
  Tirer un élément au hasard (loi uniforme), avec une probabilité proportionnelle à l'évaluation de chaque élément.

- **Croisement en un point :**
  On découpe les chromosomes en un point tiré au hasard.
- **Croisement en deux points :**
  On découpe les chromosomes en deux points tirés au hasard.
- **Croisement uniforme :**
  On tire au sort pour chaque gène si on échange ou pas.

CMA-ES