

Cours Contrôle Avancé

Partie Optimisation

Ecole Centrale Paris

Continuous optimization with stochastic algorithms: Evolution Strategies and CMA-ES

Anne Auger, Dimo Brockhoff
firstname.lastname@inria.fr

<http://researchers.lille.inria.fr/~brockhof/advancedcontrol/>

Dans cette séance nous nous intéressons à l'optimisation numérique, i.e. l'optimisation de fonctions $f : \mathbb{R}^d \rightarrow \mathbb{R}$ par algorithmes appelés Stratégies d'Évolutions (ES).

I Introduction

Nous allons tester la convergence de plusieurs algorithmes sur des fonctions tests, en particulier la fonction sphere

$$f_{\text{sphere}}(x) = \sum_{i=1}^d x_i^2$$

et la fonction ellipsoïde

$$f_{\text{elli}}(x) = \sum_{i=1}^d (100^{\frac{i-1}{d-1}} x_i)^2 .$$

1. Quelle est la forme géométrique des lignes d'iso-valeurs (ou iso-fitness) des deux fonctions tests ci-dessus pour la dimension $d = 2$? Quels sont les optima des fonctions ? Quel est le conditionnement des fonctions ?

Nous rappelons que:

- ★ pour une fonction convexe quadratique $f(x) = \frac{1}{2}x^T Hx$ où H est une matrice symétrique définie positive, le conditionnement de la fonction est défini comme le conditionnement de la matrice H .
- ★ pour une matrice symétrique définie positive, le conditionnement est donné par le quotient entre la plus grande et la plus petite valeur propre de la matrice.

2. Utiliser le logiciel Matlab pour créer les fonctions sphere et ellipsoïde. On pourra créer deux fichiers `fsphere.m` and `felli.m` prenant en entrée un vecteur \mathbf{x} et retournant la valeur des fonctions évaluées en \mathbf{x} .

II Méthodes à pas adaptatif

L'algorithme (1+1)-ES est un des algorithmes évolutionnaires les plus simples pour l'optimisation numérique. Nous allons commencer par étudier un (1+1)-ES à pas constant dont le pseudo-code est donné par:

```

Initialize  $\mathbf{x} \in \mathbb{R}^d$  and  $\sigma > 0$ 
while not terminate
     $\mathbf{x}' = \mathbf{x} + \sigma \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
    if  $f(\mathbf{x}') \leq f(\mathbf{x})$ 
         $\mathbf{x} = \mathbf{x}'$ 

```

où $\mathcal{N}(\mathbf{0}, \mathbf{I})$ dénote un vecteur gaussien de moyenne $\mathbf{0}$ et matrice de covariance identité. Le *pas de mutation* σ correspond à l'écart-type selon chaque coordonnée de $\sigma \mathcal{N}(\mathbf{0}, \mathbf{I})$ et contrôle la taille du pas effectué à chaque itération.

1. Implémenter l'algorithme en Matlab. On pourra écrire une fonction qui prend en argument le vecteur \mathbf{x} initial, le pas de mutation, le nombre maximum d'évaluation de cette fonction et retourne un vecteur contenant les valeurs successives de la fonction à optimiser pour chaque parent \mathbf{x} considéré.
2. Utiliser l'algorithme pour minimiser la fonction sphère en dimension 5. On prendra comme point initial $\mathbf{x}^0 = (1, \dots, 1)$ [`x=ones(1,5)`] et pas initial $\sigma = 10^{-3}$ [`sigma=1e-3`] et comme critère d'arrêt un nombre maximum d'évaluation égal à 2×10^4 .
3. Tracer l'évolution des valeurs prises de la fonction à optimiser. On utilisera une échelle log pour l'axe des y (`semilogy`).
4. Expliquer les trois "phases" observées sur la figure.

Pour accélérer la convergence, nous allons implémenter un algorithme à pas adaptatif, i.e. σ n'est plus fixé une fois pour toute au début de l'optimisation. La méthode pour adapter le pas s'appelle la règle des 1/5. Le pseudo-code du (1+1)-ES avec règle des 1/5 est donné par:

```

Initialize  $\mathbf{x} \in \mathbb{R}^d$  and  $\sigma > 0$ 
while not terminate
     $\mathbf{x}' = \mathbf{x} + \sigma \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
    if  $f(\mathbf{x}') \leq f(\mathbf{x})$ 
         $\mathbf{x} = \mathbf{x}'$ 
         $\sigma = 1.5 \sigma$ 
    else
         $\sigma = (1.5)^{-1/4} \sigma$ 

```

5. Coder l'algorithme (1+1)-ES avec règle des 1/5 et tester l'algorithme sur les fonctions $f_{\text{sphere}}(x)$ en dimension 5 ($d = 5$) en utilisant $\mathbf{x}^0 = (1, \dots, 1)$, $\sigma_0 = 10^{-3}$ et comme critère d'arrêt un nombre maximal d'évaluations égal à 6×10^2 . Tracer la valeur de la fonction objectif du parent à chaque iteration en fonction du nombre d'évaluations de la fonction objectif. On utilisera une échelle logarithmique pour l'axe des ordonnées. Comparer au graphique obtenu à la question 3. Tracer également l'évolution du pas σ en fonction du nombre d'évaluations de la fonction objectif.
6. Utiliser l'algorithme pour minimiser la fonction f_{elli} en dimension $d = 5$. Tracer l'évolution de la fonction objectif en fonction du nombre d'évaluations. Pourquoi le (1+1)-ES avec la règle des 1/5 a besoin de beaucoup plus lent sur f_{elli} que sur la fonction sphère ?
7. Même question avec la fonction : $f_{\text{Rosenbrock}}(x) = \sum_{i=1}^{d-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$.
8. Nous considérons maintenant les fonctions, $g(f_{\text{sphere}})$ et $g(f_{\text{elli}})$ où $g : \mathbb{R} \rightarrow \mathbb{R}, y \mapsto \ln(1 + y^2)$. Modifier le code utilisé en 5. et 6., de façon à sauvegarder à chaque itération la distance entre \mathbf{x} et l'optimum. Tracer l'évolution de la distance à l'optimum en fonction du nombre d'évaluations de la fonction objectif sur les fonctions f_{sphere} et $g(f_{\text{sphere}})$ ainsi que sur les fonctions f_{elli} et $g(f_{\text{elli}})$. Qu'observez-vous ? Expliquer.

III Covariance-Matrix-Adaptation-ES (CMA-ES)

Récupérer le code MATLAB de l'algorithme CMA-ES (`cmaes.m`)¹.

1. Utiliser l'algorithme CMA-ES pour minimiser la fonction f_{elli} en dimension 5 (`cmaes('felli',x0,sigma0)`). Interpréter les différentes courbes qui apparaissent à l'écran. Quelle est la relation entre les valeurs propres de la matrice de covariance et les valeurs propres de la matrice hessienne des fonctions ?
2. Comparer la vitesse de convergence de CMA-ES et du (1+1)-ES avec règle des 1/5 sur f_{elli} . Expliquer cette différence.
3. La fonction $f_{\text{ellirot}}(x)$ est définie par $f_{\text{ellirot}}(x) = f_{\text{elli}}(Px)$ où P est une matrice de rotation (tirée uniformément dans l'ensemble des matrices de rotation). Faire tourner l'algorithme CMA-ES sur f_{ellirot} et f_{elli} (`cmaes('felli',x0,sigma0)` et `cmaes('fellirot',x0,sigma0)`). Interpréter les différences obtenues au niveau de la courbe en bas à droite de l'écran.
4. La taille de population par défaut dans CMA-ES est égale à $4 + \text{floor}(3 * \log(d))$. Réaliser 5 runs de CMA-ES avec la taille de la population par défaut pour minimiser la fonction multimodale Rastrigin en dimension 10 (`cmaes('rastrigin',rand(10,1),10)`). Mesurer la probabilité de succès. Réaliser la même expérience en multipliant la taille de population par défaut par 2, 4, 8, 16, 32. Qu'observez vous? Expliquer.

- $f_{\text{rastrigin}}(x) = 10d + \sum_{i=1}^d (x_i^2 - 10 \cos(2\pi x_i))$

¹Le code se trouve en libre accès en divers langage sur la page de Nikolaus Hansen <http://www.lri.fr/~hansen/>