$(\mu/\mu, \lambda) - ES$ with Search Path in C++

Florence Carton[1], Alvaro Correia[1], Gabirel Quéré[1] and Antonin Raffin[1]

I. Introduction

The main goal of our work was to implement the algorithm $(\mu/\mu, \lambda) - ES$ with Search Path presented in the paper [1]. To test and compare our algorithm with other implementations, we used the COCO plateform [2].

In the next section, we introduce the algorithm as it is written in the corresponding paper, then we explain how we implemented it and which results we obtained.

II. The algorithm

The algorithm we implemented is the following :

Algorithm 1 The $(\mu/\mu, \lambda)$ - ES with Search Path

given $n \in \mathbb{N}$, $\lambda \in \mathbb{N}$, $\mu \approx \lambda/4 \in \mathbb{N}$, $c_\sigma \approx \sqrt{\mu/(n+\mu)}$, $d \approx 1 + \sqrt{\mu/n}$, $d_i \approx 3n$
initialize $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{\sigma} \in \mathbb{R}^n_+$, $\boldsymbol{s_\sigma} = \mathbf{0}$
while not happy **do**
  for $k \in \{1, ..., \lambda\}$ **do**
    $\boldsymbol{z_k} = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ iid for each k
    $\boldsymbol{x_k} = \boldsymbol{x} + \boldsymbol{\sigma} \circ \boldsymbol{z_k}$
  end for
  $\mathcal{P} \leftarrow sel\_\mu\_best(\{\boldsymbol{x_k}, \boldsymbol{z_k}, f(\boldsymbol{x_k}) | 1 \leq k \leq \lambda\})$ recombination and parent update
  $\boldsymbol{s_\sigma} \leftarrow (1 - c_\sigma)\boldsymbol{s_\sigma} + \sqrt{c_\sigma(2 - c_\sigma)} \frac{\sqrt{\mu}}{\mu} \sum_{\boldsymbol{z_k} \in \mathcal{P}} \boldsymbol{z_k}$
  $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma} \circ exp^{1/d_i}(\frac{|\boldsymbol{s_\sigma}|}{\mathbb{E}|\mathcal{N}(0,1)|} - \mathbf{1}) \times exp^{c_\sigma/d}(\frac{||\boldsymbol{s_\sigma}||}{\mathbb{E}||\mathcal{N}(\mathbf{0},\mathbf{1})||} - 1)$
  $\boldsymbol{x} = \sum_{x_k \in \mathcal{P}} x_k$
end while

In this algorithm, f in the function we want to minimize, $\boldsymbol{x}$ its parameter, and n the dimension of $\boldsymbol{x}$. λ is the number of offsprings and μ is the number of parents. Therefore the selection function will select the μ best in the offspring population. The x_k are the offsprings, s_σ is the search path, z_k are the mutation steps and σ the mutation vectors.

III. Our implementation

A. Implementation of the algorithm

We implemented this algorithm in C++, and in this section we present how we implemented it and which choices we made.

Regarding the initialisation, the vector $\boldsymbol{x}$ is initialized randomly with a uniform distribution and every element of the vector $\boldsymbol{\sigma}$ is initialized to 0.1.

In the while loop, two stop criterion can be found in our implementation. The first one is the budget, i.e. the number of iterations, and the second one is the 'happy' criterion : the programme will stop if there is no change between two iterations bigger that 10^{-8} .

B. Tuning the hyperparameters

IV. Analysis of the results

To analyse the results, we launched our algorithm on the COCO plateform : [2]. The COCO (COmparing Continuous Optimizers) plateform provides benchmarks for comparison between optimizing algorithm, and visualizing tools to plot the data.

V. Conclusion

VI. Bibliography

References

[1] Hansen, N., D.V. Arnold and A. Auger (2015). Evolution Strategies. In Janusz Kacprzyk and Witold Pedrycz (Eds.): *Handbook of Computational Intelligence*, Springer, Chapter 44, pp.871-898
[2] https://github.com/numbbo/coco

[1] École Nationale Supérieure de Techniques Avancées (ENSTA-ParisTech), Paris, France