Evolving the Structure of Evolution Strategies

SSCI 2016

Sander van Rijn, Hao Wang, Matthijs van Leeuwen, Thomas Bäck .

Universiteit Leiden
The Netherlands

Introduction

Over the past decades, many different optimizers have been proposed

No single method will be better for all optimization problems

Introduction

Over the past decades, many different optimizers have been proposed

No single method will be better for all optimization problems

Evolution Strategies (ES) know many structural variations, few combinations have been tested

For a specific target function (class), which is best?

Introduction — Research Questions

1. Can we define a modular and extensible CMA-ES framework?
2. How to determine an efficient ES structure, within budget?
3. Are there novel variations that outperform known variants?

Problem Description

We examine the set of real-valued minimization problems in D dimensions:

$$\mathbb{F} = \{f : \mathbb{R}^D \to \mathbb{R}\}$$

The goal of an optimization method is to find $\vec{x}_{\text{opt}} \in \mathbb{R}^D$ such that

$$\forall \vec{x} \in \mathbb{R}^D : f(\vec{x}_{\text{opt}}) \leq f(\vec{x})$$

Approach — Defining the Search-Space

We consider **11** modules

Each module has **2** or **3** options

The framework can create
$\mathbf{2^9 \cdot 3^2} = 4\,608$ ES-structures

Selected CMA-ES Modules
Active Update
Elitism
Mirrored Sampling
Orthogonal Sampling
Sequential Selection
Threshold Convergence
TPA
Pairwise Selection
Recombination Weights
Quasi-Gaussian Sampling
Increasing Population

Approach — Defining the Search-Space

Each module has corresponding dependencies and parameters

- Modules were chosen to require minimal dependency checking
- Default values from literature are used for all parameters

Note: This means we do not tune any parameters!

Approach — Algorithm

Algorithm 1 Modular CMA-ES Framework

options $\leftarrow$ which modules are active
init-params $\leftarrow$ initial/default parameter values
while not terminate **do** // *Local restart loop*
  params $\leftarrow$ Initialize(init-params)
  $\boldsymbol{t} \leftarrow \mathbf{0}$
  $\bar{\boldsymbol{x}} \leftarrow$ randomly generated individual
  while not terminate local **do** // *ES execution loop*
    $\vec{\boldsymbol{x}} \leftarrow$ Mutate($\bar{\boldsymbol{x}}$, *options*) // *Sampler, Threshold*
    $\vec{\boldsymbol{f}} \leftarrow$ Evaluate($\vec{\boldsymbol{x}}$, *options*) // *Sequential*
    $\boldsymbol{P}^{(t+1)} \leftarrow$ Select($\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{f}}$, *options*) // *Elitism, Pairwise*
    $\bar{\boldsymbol{x}} \leftarrow$ Recombine($\boldsymbol{P}^{(t+1)}$, *options*) // *Weights*
    UpdateParams(params, *options*) // *Active, TPA*
    $\boldsymbol{t} \leftarrow \boldsymbol{t} + \mathbf{1}$
  end while
  AdaptParams(init-params) // *(B)IPOP*
end while

Approach — How to search?

Brute Force enumeration?

Time required goes up exponentially with number of modules

Instead we create a Genetic Algorithm (GA) that can ...

Approach — How to search?

Brute Force enumeration?

Time required goes up exponentially with number of modules

Instead we create a Genetic Algorithm (GA) that can . . .

Evolve the Structure of Evolution Strategies

Experimental Setup

- $\mathbf{(1, 12)}$ mutation-only self-adaptive GA
- Brute force search as “ground truth”

Parameter	Value
Implemented in	Python
Runs per ES	32
ES evaluation budget	10^3D
GA evaluation budget	20λ
Dimensionalities	2, 3, 5, 10, 20
BBOB Functions	Noiseless F1–F24

Results — GA Convergence

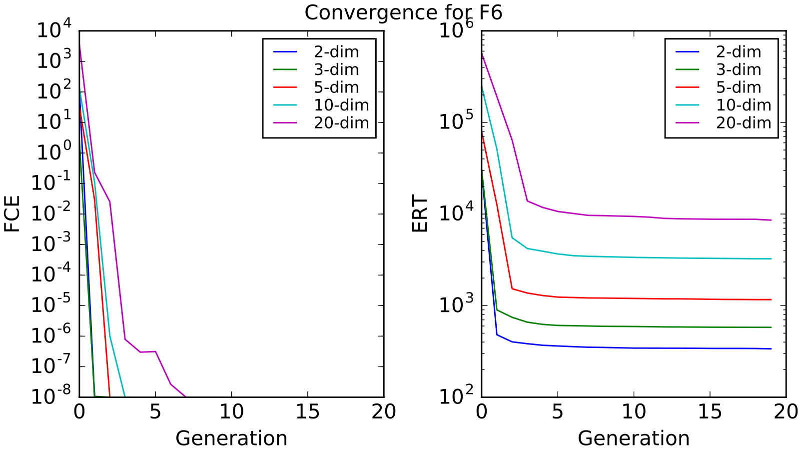

Figure : *Attractive Sector Function*

Results — GA Convergence

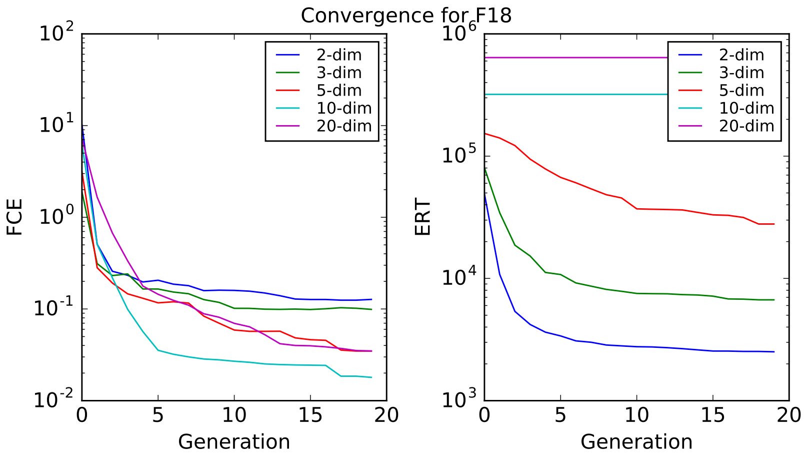

Figure : *Schaffers F7 Function*

Results — Module Frequency

Module name	% chosen by GA
Active Update	27.0
Elitism	44.3
Mirrored Sampling	58.4
Orthogonal Sampling	54.1
Sequential Selection	34.8
Threshold Convergence	22.6
TPA	31.1
Pairwise Selection	21.3
Recombination Weights	17.9
Sobol/Halton	85.6
IPOP/BIPOP	78.8

Results — Performance Improvement

F-id	D	Best common variant	Improvement
F1	2	Active IPOP $(\mu + \lambda)$	1.78
F1	3	Active IPOP $(\mu + \lambda)$	2.01
F1	5	Active IPOP $(\mu + \lambda)$	1.81
F1	10	$(\mu + \lambda)$	2.25
F1	20	Active IPOP $(\mu + \lambda)$	2.37
F6	2	$(\mu + \lambda)$	1.36
F6	3	$(\mu + \lambda)$	1.40
F6	5	$(\mu + \lambda)$	1.38
F6	10	Mirrored Pairwise	1.31
F6	20	Mirrored Pairwise	1.15
F18	2	Active BIPOP $(\mu + \lambda)$	3.50
F18	3	Mirrored Pairwise	4.68
F18	5	BIPOP	8.59
F18	10	BIPOP	7.70
F18	20	BIPOP	6.27

Conclusions

1. *Can we define a modular and extensible CMA-ES framework?*

   Yes, this is a viable and extensible method

2. *How to determine an efficient ES structure, within budget?*

   ES-structures can successfully be evolved by a GA

3. *Are there novel variations that outperform known variants?*

   GA-found combinations generally outperform common variants

Future Work

- Include parameters
- Extend the framework with more modules
- Analyze the impact of individual modules
- Test on real-world problem classes

Thanks for your attention