Algorithm Configuration Data Mining for CMA Evolution Strategies

Sander van Rijn, Hao Wang, Bas van Stein and Thomas Bäck .

Universiteit Leiden
The Netherlands

Introduction

- EA popularity $\rightarrow$ many variants
- Focus: CMA-ES
- Few combinations are tested

Modular CMA-ES Framework

Selected CMA-ES Modules
Active Update
Elitism
Mirrored Sampling
Orthogonal Sampling
Sequential Selection
Threshold Convergence
Two-Point step-size Adaptation (TPA)
Pairwise Selection
Recombination Weights
Quasi-Gaussian Sampling
Increasing Population

S. van Rijn, H. Wang, M. van Leeuwen and T. Bäck, "Evolving the structure of Evolution Strategies," 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8. DOI: 10.1109/SSCI.2016.7850138

Modular CMA-ES Framework

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

$\rightarrow$

```
[0,0,0,0,0,0,0,0,0,0,0]
[0,0,0,0,0,0,0,0,0,0,1]
[0,0,0,0,0,0,0,0,0,0,2]
          ...
[1,1,1,1,1,1,1,1,1,2,0]
[1,1,1,1,1,1,1,1,1,2,1]
[1,1,1,1,1,1,1,1,1,2,2]
```

$$\mathbf{2^9} \times \mathbf{3^2} = 4\,608$$

S. van Rijn, H. Wang, M. van Leeuwen and T. Bäck, "Evolving the structure of Evolution Strategies," 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8. DOI: 10.1109/SSCI.2016.7850138

Modular CMA-ES Framework

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

→

[0,0,0,0,0,0,0,0,0,0,0]
[0,0,0,0,0,0,0,0,0,0,1]
[0,0,0,0,0,0,0,0,0,0,2]
...
[1,1,1,1,1,1,1,1,1,2,0]
[1,1,1,1,1,1,1,1,1,2,1]
[1,1,1,1,1,1,1,1,1,2,2]

→

$$\mathbf{2^9} \times \mathbf{3^2} = 4\,608$$

S. van Rijn, H. Wang, M. van Leeuwen and T. Bäck, "Evolving the structure of Evolution Strategies," 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8. DOI: 10.1109/SSCI.2016.7850138

Modular CMA-ES Framework

Algorithm 1 Modular CMA-ES Framework

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

S. van Rijn, H. Wang, M. van Leeuwen and T. Bäck, "Evolving the structure of Evolution Strategies," 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8. DOI: 10.1109/SSCI.2016.7850138

Algorithm Quality

Problem: How to determine quality of a configuration $\mathbf{c}$?

- FCE: arbirtrary values
- ERT: only defined on $\text{FCE}(\mathbf{c}) < \text{FCE}_{\text{target}}$

Algorithm Quality

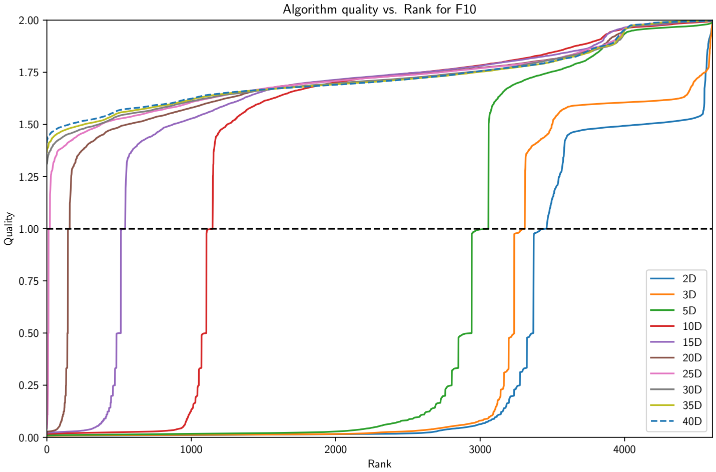
Problem: How to determine quality of a configuration $\mathbf{c}$?

- FCE: arbirtrary values
- ERT: only defined on $\mathrm{FCE}(\mathbf{c}) < \mathrm{FCE}_{\mathrm{target}}$

Solution: scaled combination ERT $\times$ FCE $\rightarrow [\mathbf{0}, \mathbf{2}]$

$$q(\mathbf{c}) = \begin{cases} \dfrac{\mathrm{ERT}(\mathbf{c})}{\mathrm{ERT}_{\max}} & \text{if } \mathrm{ERT}(\mathbf{c}) \text{ exists} \\ 1 + \dfrac{\log(\mathrm{FCE}(\mathbf{c})/\mathrm{FCE}_{\mathrm{target}})}{\log(\mathrm{FCE}_{\max}/\mathrm{FCE}_{\mathrm{target}})} & \text{otherwise,} \end{cases}$$

Example



Algorithm quality vs. Rank for F10
Quality
Rank
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
1000
2000
3000
4000
2D
3D
5D
10D
15D
20D
25D
30D
35D
40D

Random Forest Regression

Forest predicts $\boldsymbol{q}$ per experiment

Forest of 250 trees

Mean feature importance[1] over all experiments

[1] A measure of how *pure* the split according to a feature is

Random Forest Regression

Forest predicts $\boldsymbol{q}$ per experiment

Forest of 250 trees

Mean feature importance[1] over all experiments

Module	Importance
Active	0.04
Elitism	0.06
Mirrored	0.04
Orthogonal	0.05
Sequential	**0.16**
Threshold	**0.31**
TPA	0.03
Pairwise	0.02
Weights	0.02
Base-Sampler	**0.20**
(B)IPOP	0.07

[1] A measure of how *pure* the split according to a feature is

Compare $\mathbf{x}_{\text{on}}$ and $\mathbf{x}_{\text{off}}$:

$$\boldsymbol{I}_{\boldsymbol{x}} = \bar{\boldsymbol{q}}(\mathrm{C}^{\boldsymbol{x}}_{\text{off}}) - \bar{\boldsymbol{q}}(\mathrm{C}^{\boldsymbol{x}}_{\text{on}})$$

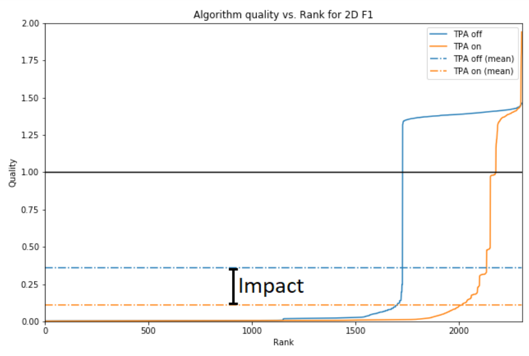
$\bar{\boldsymbol{q}}(\mathrm{C})$: mean $\boldsymbol{q}$ of set C

Impact

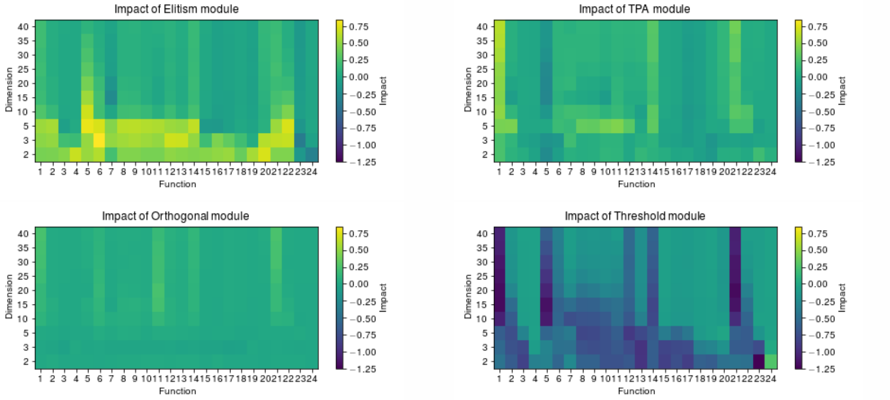
Compare $\mathbf{x}_{\text{on}}$ and $\mathbf{x}_{\text{off}}$:

$$\boldsymbol{I_x} = \bar{\boldsymbol{q}}(\mathrm{C}^{\boldsymbol{x}}_{\text{off}}) - \bar{\boldsymbol{q}}(\mathrm{C}^{\boldsymbol{x}}_{\text{on}})$$

$\bar{\boldsymbol{q}}(\mathrm{C})$: mean $\boldsymbol{q}$ of set C



Impact per Module

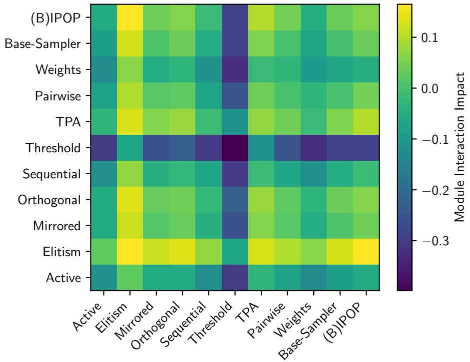


Impact: Module Interaction

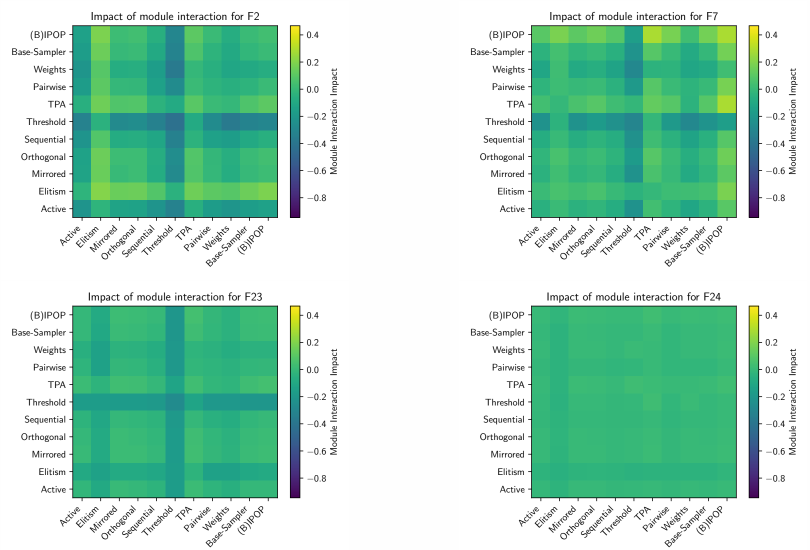
Compare $\mathbf{x}_{\text{on}} \wedge \mathbf{y}_{\text{on}}$ and $\neg(\mathbf{x}_{\text{on}} \wedge \mathbf{y}_{\text{on}})$

Impact: Module Interaction

Compare $\mathbf{x}_{\text{on}} \wedge \mathbf{y}_{\text{on}}$ and $\neg(\mathbf{x}_{\text{on}} \wedge \mathbf{y}_{\text{on}})$



Impact: Module Interaction



Module Progression

Configuration ranking:

```
#1: [0 0 1 1 0 1 1 0 0 2 0]
#2: [0 0 1 1 0 1 1 0 0 2 2]
#3: [0 0 1 1 0 1 1 0 0 2 1]
#4: [0 0 1 1 0 1 1 1 0 2 2]
...
#4608: ...
```

Which modules are active

- in the best configuration?

Module Progression

Configuration ranking:

```
#1: [0 0 1 1 0 1 1 0 0 2 0]
#2: [0 0 1 1 0 1 1 0 0 2 2]
#3: [0 0 1 1 0 1 1 0 0 2 1]
#4: [0 0 1 1 0 1 1 1 0 2 2]
...
#4608: ...
```

Which modules are active

- in the best configuration?
- in the 10 best configurations?

Module Progression

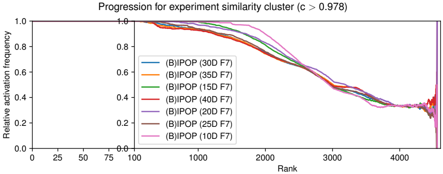
Configuration ranking:

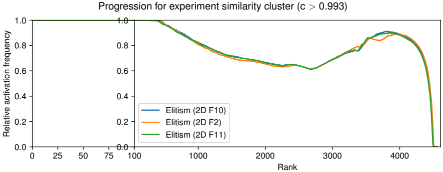
```
#1: [0 0 1 1 0 1 1 0 0 2 0]
#2: [0 0 1 1 0 1 1 0 0 2 2]
#3: [0 0 1 1 0 1 1 0 0 2 1]
#4: [0 0 1 1 0 1 1 1 0 2 2]
...
#4608: ...
```

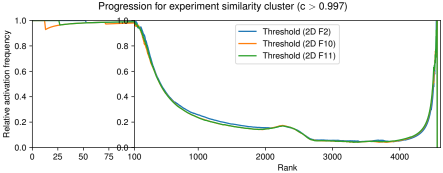
Which modules are active

- in the best configuration?
- in the 10 best configurations?
- in the 100 best configurations?
- ...

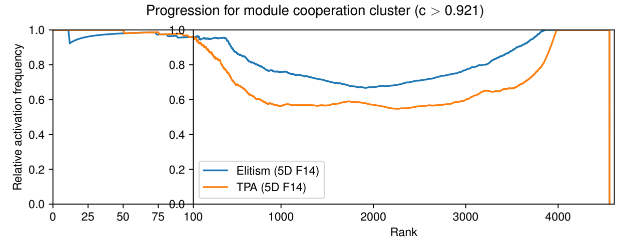
Progression Correlation

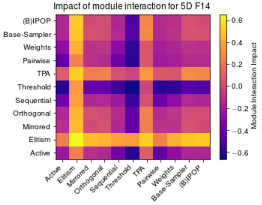


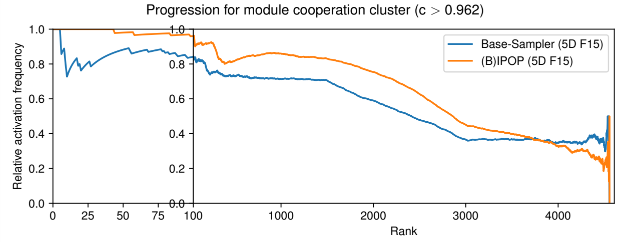


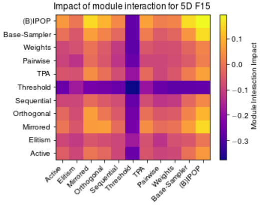


Progression Correlation









Summary & Outlook

Summary

- We can successfully identify useful options
- Similar landscapes show similar impact/progression behavior

Summary & Outlook

Summary

- We can successfully identify useful options
- Similar landscapes show similar impact/progression behavior

Outlook

- Combine with landscape features
- Include parameters tuning
- Expand to include more modules

Summary & Outlook

Summary

- We can successfully identify useful options
- Similar landscapes show similar impact/progression behavior

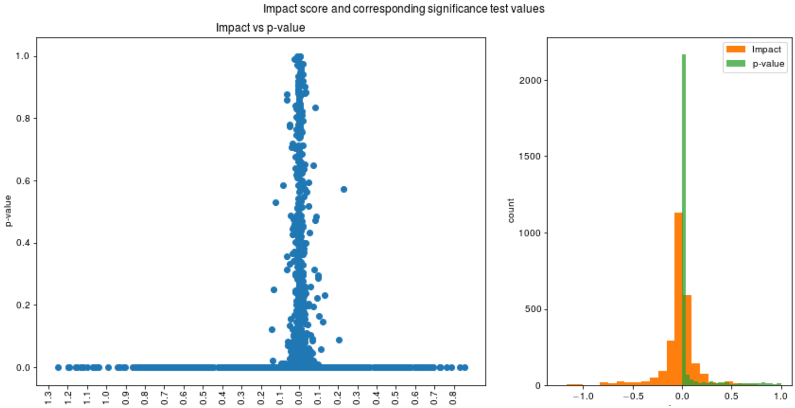
Outlook

- Combine with landscape features
- Include parameters tuning
- Expand to include more modules

Code on Github

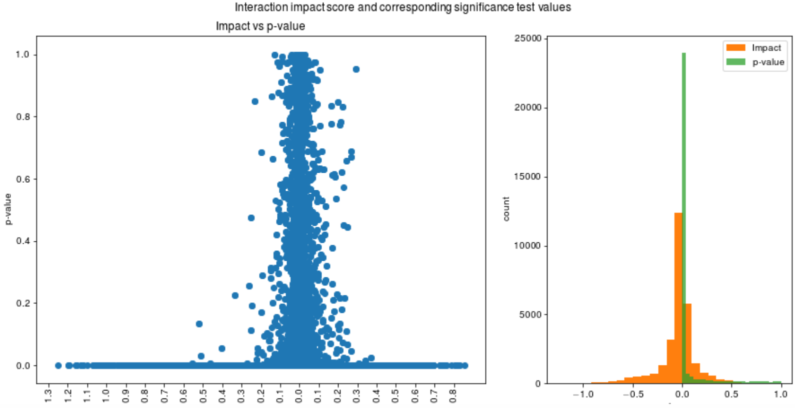
S. van Rijn, "module_analysis.ipynb", hosted at:
`https://github.com/Energya/cma-es-configuration-data-mining`

Appendix: Module Impact p-values



All I_x outside $[-\mathbf{0.144}, \mathbf{0.229}]$ have $p < \mathbf{0.01}$ (442 / 2 640 values)

Appendix: Interaction Impact p-values



All $\boldsymbol{I_x}$ outside $[\mathbf{-0.551, 0.371}]$ have $\boldsymbol{p} < \mathbf{0.01}$ (1 383 / 29 040 values)