Evaluating the Population Size Adaptation Mechanism for CMA-ES on the BBOB Noiseless Testbed

Kouhei Nishida[1] Youhei Akimoto[1]

[1]Shinshu University, Japan

Introduction: CMA-ES

- The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a stochastic search algorithm using the multivariate normal distribution.
  1. Generate candidate solutions $(x_i^{(t)})_{i=1,2,\dots,\lambda}$ from $\mathcal{N}(m^{(t)}, C^{(t)})$.
  2. Evaluate $f(x_i^{(t)})$ and sort them, $f(x_{1:\lambda}) < \cdots < f(x_{\lambda:\lambda})$.
  3. Update the distribution parameters $\theta^{(t)} = (m^{(t)}, C^{(t)})$ using the ranking of candidate solutions.
- The CMA-ES has the default value for all strategy parameters (such as the population size λ, the learning rate η_c).
- A larger population size than the default value improves its performance on following scenarios.
  1. Well-structured multimodal function
  2. Noisy function
- It can be easily very expensive to tune the population size in advance.

Introduction: Population Size Adaptation

- As a measure for the adaptation, we consider the randomness of the parameter update.
- To quantify the randomness of the parameter update, we introduce the evolution path in the parameter space.
- To keep the randomness of the parameter update in a certain level, the population size is adapted online.

Advantage of adapting the population size online:

- It doesn't require tuning of the population size in advance.
- On rugged function, it may accelerate the search by reducing the population size after converging in a basin of a local minimum.

Rank-μ update CMA-ES

- The rank-μ update CMA-ES, which is a component of the CMA-ES, repeats the following procedure.
  1. Generate candidate solutions $(x_i^{(t)})_{i=1,2,\ldots,\lambda}$ from $\mathcal{N}(m^{(t)}, C^{(t)})$.
  2. Evaluate $f(x_i^{(t)})$ and sort them, $f(x_{1:\lambda}) < \cdots < f(x_{\lambda:\lambda})$.
  3. Update the distribution parameters $\theta^{(t)} = (m^{(t)}, C^{(t)})$ using the ranking of candidate solutions.

$$\theta^{(t+1)} = \theta^{(t)} + \Delta\theta^{(t)}$$

$$\Delta m^{(t)} = \eta_m \sum_i^{\lambda} w_i (x_{i:\lambda}^{(t)} - m^{(t)}),$$

$$\Delta C^{(t)} = \eta_c \sum_i^{\lambda} w_i ((x_{i:\lambda}^{(t)} - m^{(t)})(x_{i:\lambda}^{(t)} - m^{(t)})^T - C^{(t)})$$

Population Size Adaptation: Measurement

To quantify the randomness of the parameter update, we introduce the evolution path in the space Θ of the distribution parameter $\theta = (m, C)$.

$$p^{(t+1)} = (1-\beta)p^{(t)} + \sqrt{\beta(2-\beta)}\Delta\theta^{(t)}$$

The evolution path accumulates the successive steps in the parameter space Θ.

(a) less tendency (b) strong tendency

Figure: An image of the evolution path

Population Size Adaptation: Measurement

- We measure the length of the evolution path based on the KL-divergence.

$$\|p\|_\theta^2 = p^T \mathcal{I}(\theta) p \approx KL(\theta \| \theta + p)$$

  The KL-divergence measures the defference between two probability distributions.

- We measure the randomness of the parameter update by the ratio between $\|p^{(t+1)}\|_\theta^2$ and its expected value $\gamma^{(t+1)} \approx \mathbb{E}[\|p^{(t+1)}\|_\theta^2]$ under a random function.

$$\gamma^{(t+1)} = (1-\beta)^2 \gamma^{(t)} + \beta(2-\beta) \sum_i^\lambda w_i^2 (d\eta_m^2 + \frac{d(d+1)}{2} \eta_c^2)$$

- Two important cases
  - a random function: $\frac{\|p\|_\theta^2}{\gamma} \approx 1$
  - too large λ: $\frac{\|p\|_\theta^2}{\gamma} \to \infty$

Population Size Adaptation: Adaptation

- If $\frac{\|p^{(t+1)}\|^2_{\theta^{(t)}}}{\gamma^{(t+1)}} < \alpha$, regarding the update as inaccurate, the population size is increased with

$$\lambda^{(t+1)} = \left\lfloor \lambda^{(t)} \exp\left(\beta\left(\alpha - \frac{\|p^{(t+1)}\|^2_{\theta^{(t)}}}{\gamma^{(t+1)}}\right)\right) \right\rfloor \vee \lambda^{(t)} + 1.$$

- If $\frac{\|p^{(t+1)}\|^2_{\theta^{(t)}}}{\gamma^{(t+1)}} > \alpha$, regarding the update as sufficiently accurate, the population size is decreased with

$$\lambda^{(t+1)} = \left\lfloor \lambda^{(t)} \exp\left(\beta\left(\alpha - \frac{\|p^{(t+1)}\|^2_{\theta^{(t)}}}{\gamma^{(t+1)}}\right)\right) \right\rfloor \vee \lambda_{\min}.$$

Algorithm Variant

We use the default setting for most of parameters. The modified parameters are the learning rate for the mean vector, c_m, and the threshold α to decide whether the parameter update is considered accurate or not.

PSAaLmC $\alpha = \sqrt{2}, c_m = 0.1$

PSAaLmD $\alpha = \sqrt{2}, c_m = 1/D$

PSAaSmC $\alpha = 1.1, c_m = 0.1$

PSAaSmD $\alpha = 1.1, c_m = 1/D$

- The greater α is, the greater the population size tends to be kept
- From our preliminaly study, we set $c_c = \sqrt{2/(D+1)}c_m$.

Restart Strategy

For each (re-)start of the algorithm, we initialize the mean vector $m \sim \mathcal{U}[-4, 4]^D$ and the covariance matrix $C = 2^2 I$. The maximum #f-call is set to $10^5 D$.

Termination conditions

tolf: median(fiqr_hist) < 10 - 12abs(median(fmin_hist))

- the objective function value differences are too small to sort them without being affected by numerical errors.

tolx: median(xiqr_hist) < 10 - 12min(abs(xmed_hist))

- the coordinate value differences are too small to update parameters without being affected by numerical errors.

maxcond: cond(C) > 10^{14}

- the matrix operations using C are not reliable due to numerical errors.

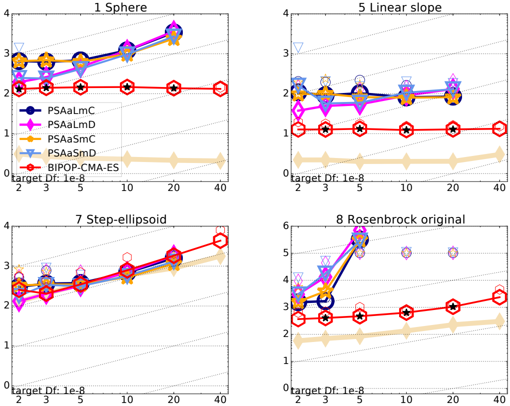
maxeval: #f-call $\geq 5 \times 10^4 D$ (for noiseless) or $10^5 D$ (for noisy)

BIPOP-CMA-ES

BIPOP restart strategy: A restart strategy with two budgets of function evaluations.

- one is for incremental population size.
  - to tackle well-structured multimodal functions or noisy functions
- the other is for relatively small population size and a relatively small step-size.
  - to tackle weakly-structured multimodal functions

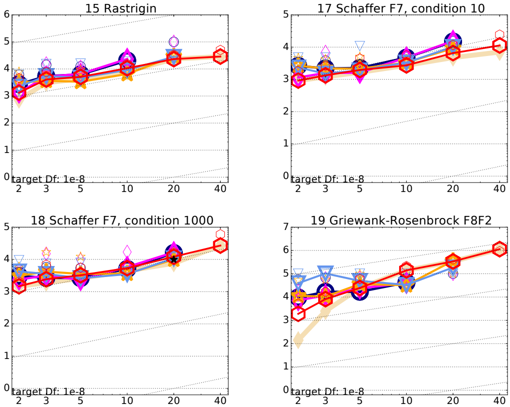
Noiseless: Unimodal Function



The aRT is higher for most of the unimodal functions than the best 2009 portfolio due to lack of the step-size adaptation.

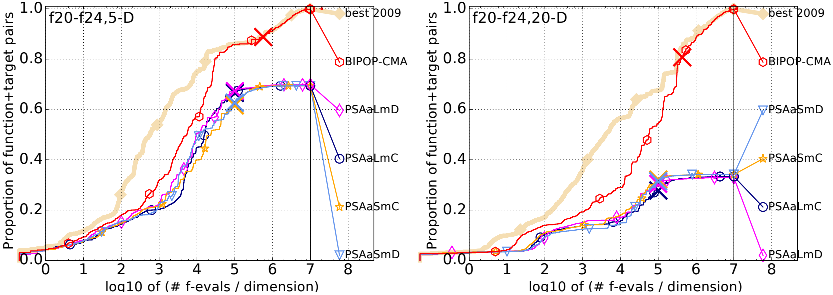
On Step-ellipsoid function, where the step-size adaptaiton is less important, our algorithm performs well.

Noiseless: Well-structured Multimodal Function



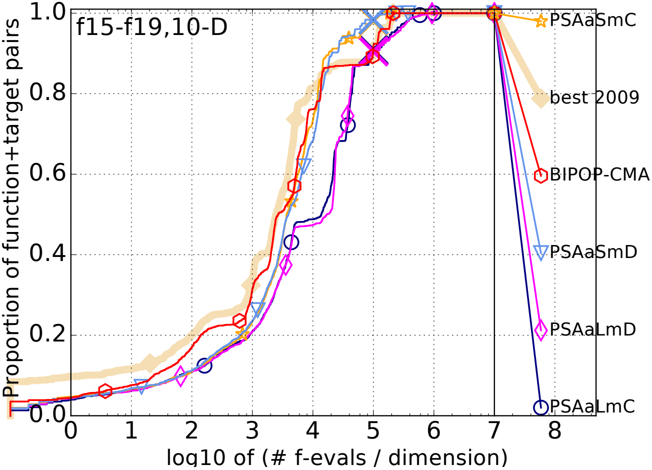
The performance of the tested algorithms is similar to the performance of the BIPOP-CMA-ES without the step-size adaptation.
Especially on Griewank-Rosenbrock, the tested algorithm is partly better than the best 2009 portfolio.

Noiseless: Weakly-structured Multimodal Function



The BIPOP-CMA-ES performs better than the tested algorithm because the tested algorithms doesn't have the mechanism to tackle weakly-structure.

Noiseless: Comparing the variants

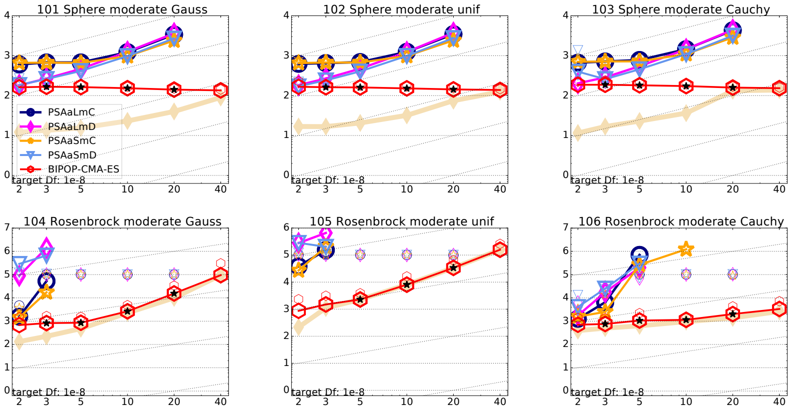


Variants with $\alpha = 1.1$ are better than ones with $\alpha = \sqrt{2}$.

Noiseless Summary

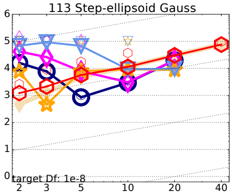
- On Well-structured multimodal function, the tested algorithm performs well without the step-size adaptaiton.
- For lack of the step-size adaptation, the aRT is higher for most of the unimodal functions and the than the best 2009 portfolio.
- When the step-size is less important, the tested algorithm performs well.
- Variants with $\alpha = 1.1$ tends to be better than ones with $\alpha = \sqrt{2}$

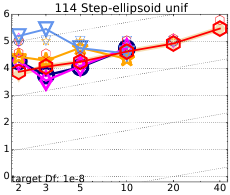
Noisy: Unimodal Function

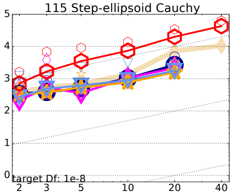


On sphere function, the algorithm is slower than the BIPOP-CMA-ES for lack of the step-size adaptation.
The failure on the Rosenbrock functions is mainly due to the same reason.

Noisy: Unimodal Function

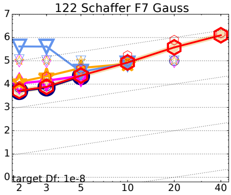


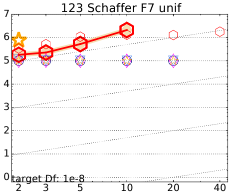


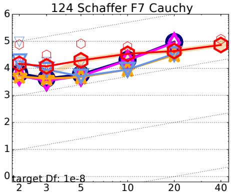


On step-ellipsoid function, where the step-size adaptation is less important, the algorithm performs well.

Noisy: Well-structured Multimodal Function

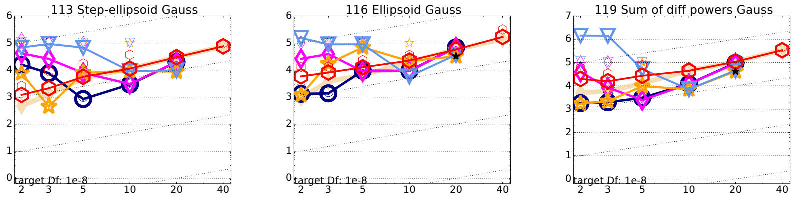






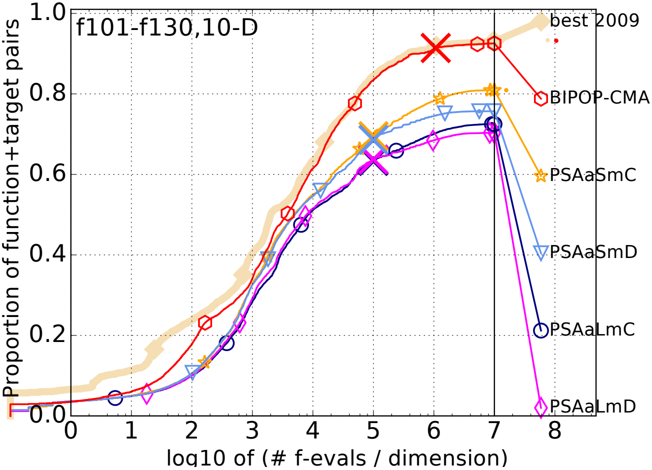
On schaffer function, the performance of the tested algorithm is similarly to the best 2009 portfolio, and partly better than it.

Noisy: Compairing the variants



The algorithms using $c_m = 1/D$ sometimes get worse in low dimension because the learning rate is too large.

Noisy: Compairing the variants



Variants with $\alpha = 1.1$ are better than ones with $\alpha = \sqrt{2}$.

Noisy Summary

- On Well-structured multimodal function, the tested algorithm performance is similarly to the best 2009.
- For lack of the step-size adaptation, the convargence speed scales worse on Sphere function and the aRT is higher for most of the unimodal functions than the best 2009 portfolio.
- variants with $\alpha = 1.1$ tends to be better than ones with $\alpha = \sqrt{2}$
- $c_m = 1/D$ is too large at low dimension.

Conclusion

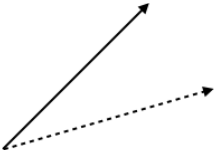
Summary

- On well-structured multimodal function, the tested algorithm performance is similarly to the best 2009.
- For lack of the step-size adaptation, the aRT is higher for most of the unimodal function and the weakly-structured function than the best 2009 portfolio.
- On noisy function, $c_m = 1/D$ is too large at low dimension.

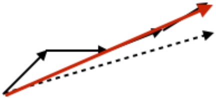
Future Work

- We incorporate the rank-one adaptation and the step-size adaptation.

Using the small learning rate works as averaging the mean vector in successive iteration.



(a) with a larger learning rate



(b) with a smaller learning rate