Paper

A Comparative Study of PSO and CMA-ES Algorithms on Black-box Optimization Benchmarks

Paweł Szynkiewicz

Research and Academic Computer Network (NASK), Warsaw, Poland

https://doi.org/10.26636/jtit.2018.127418

Abstract—Numerous practical engineering applications can be formulated as non-convex, non-smooth, multi-modal and ill-conditioned optimization problems. Classical, deterministic algorithms require an enormous computational effort, which tends to fail as the problem size and its complexity increase, which is often the case. On the other hand, stochastic, biologically-inspired techniques, designed for global optimum calculation, frequently prove successful when applied to real life computational problems. While the area of bio-inspired algorithms (BIAs) is still relatively young, it is undergoing continuous, rapid development. Selection and tuning of the appropriate optimization solver for a particular task can be challenging and requires expert knowledge of the methods to be considered. Comparing the performance of viable candidates against a defined test bed environment can help in solving such dilemmas. This paper presents the benchmark results of two biologically inspired algorithms: covariance matrix adaptation evolution strategy (CMA-ES) and two variants of particle swarm optimization (PSO). COCO (COmparing Continuous Optimizers) – a platform for systematic and sound comparisons of real-parameter global optimization solvers was used to evaluate the performance of CMA-ES and PSO methods. Particular attention was paid to the efficiency and scalability of both techniques.

Keywords—benchmarking, black-box optimization, CMA-ES, global optimization, PSO, stochastic optimization.

1. Introduction

Many issues related to real-life problems require that the optimal solution be calculated. Traditionally, optimization problems are solved using deterministic solvers which normally assume that the objective function and the set of admissible solutions are convex and known in an analytical form. In practice, however, there are many problems for which an algebraic model is missing. Either, we lack the insight into the system to be optimized and the model is entirely unavailable, or its analytical form is too complicated and intractable to conventional optimization solvers. In the latter case the load of mathematical and practical knowledge fed into the optimization model usually results in numerous formulas the solution of which can be obtained only numerically. In general, problems for which algebraic conventional optimization solver models are unsuitable or entirely unavailable, are referred to as black-box problems. Thereby we assume that the black-box can be queried through a simulation, experimental measurements or the so-called *surrogate model* to provide crucial performance characteristics [1]–[5] for specified values of system inputs. Practical applications in network system design, cybersecurity, large scale systems modeling, optimization and control, etc. are discussed in [6]–[11].

Let us consider the following black-box optimization problem:

$$\min_{x\in\Re^{dim}} f(x)$$
$$\text{subject to: } x_i^{min} \le x_i \le x_i^{max}, \quad i = 1,\ldots,dim\ , \quad (1)$$

where f is the real-valued, dim-dimensional function. In general, f can be a non-convex, non-smooth, ill-conditioned or multi-modal. It is assumed that in the above problem the function values of the evaluated search points x are the only accessible information. The search points to be evaluated can be freely chosen. Hence, the search cost is equal to the number of function evaluations executed in order to reach the target solution.

This paper addresses issues associated with the black-box optimization benchmarking (1) for the comparison of two biologically-inspired global optimum calculation algorithms, i.e., the covariance matrix adaptation evolution strategy (CMA-ES) [12], [13] and particle swarm optimization (PSO) [14]. CMA-ES and PSO have already proved successful in solving various black-box problems. The aim of the research was to examine how well CMA-ES and PSO perform on both well- and ill-conditioned optimization problems, and how strongly the efficiency of both algorithms depends on the complexity of the problem and on the prescribed accuracy of the solution.

All experiments were conducted with the use of the black box optimization benchmarking (BBOB) test bed [15] that provides numerous testing problems with various characteristics, dimensions and degrees of complexity. These problems are divided into groups, each with specific characteristics of the objective function, i.e., separable, moderate, ill-conditioned, multi-modal and weakly structured multi-modal functions.

The remainder of this paper is organized as follows. A short survey of black-box techniques is presented in Section 2. A brief description of two bio-inspired algorithms, CMA-ES and PSO in their local and global versions, is presented in Section 3. The overview of implementation, organization and usage of the benchmark COCO and BBOB test bed platform is presented in Section 4. The experimental procedure and the performance measures are presented in Section 5. The results of performance evaluation of both optimization algorithms, conducted with the use of various benchmarks, are presented and discussed in Section 6. Finally, conclusions are drawn in Section 7.

2. Related Work

Many optimization techniques that could be employed for solving the black-box optimization problem (1) have been reported in literature. The overview of various techniques is presented in [2]–[3], [16]–[18]. The algorithms can be classified into the following groups: stochastic approximation techniques (gradient-based approaches), sample path optimization, response surface methodology, deterministic search methods, random search methods, heuristics and metaheuristics.

A stochastic approximation is the well-known gradient search method that is similar to the steepest descent gradient algorithm. The procedure requires a gradient estimation. A computer simulation is performed to obtain estimates of the gradient. It seems that the simulation outcome has to contain gradient evaluations. The advantages and disadvantages of this technique are discussed in detail in [5].

Stochastic gradient algorithms need a simulation run for every iteration in order to calculate the gradient value (K iterations require at least K experiments). In the sample path method [5], the original problem is converted into an approximated deterministic problem. The approximation of the objective function f in (1) is calculated based on simulations performed for a randomly generated set of independent observations. Then, standard optimization algorithms are applied to locate the optimal solution.

Response surface methodology [5], [16] is a sequential strategy based on local approximation $F(x, \alpha^{(k)})$ of the performance f in the neighborhood of x, where the parameters α are calculated using simulations performed every k-th iteration. Next, the minimal value of $F(x, \alpha^{(k)})$ is calculated. The process is repeated until the acceptable solution is found.

Standard deterministic search techniques [18], [19], i.e. algorithms developed by Hook and Jeeves, Rosenbrock or nonlinear simplex (as Nelder and Mead) or complex methods can be applied to solve non-differentiable simulation optimization problems.

As the capabilities of modern computers increase, we can observe a growing interest in the development of the global optimization methods. Global optimization algorithms are linked with the computation of a global solution of non-convex, non-smooth, ill-conditioned and multi-extreme problems. The greatest challenge in the process of designing such algorithms consists in deciding whether the local optimum is a global one in the absence of any local criteria. Over the past decades, numerous theoretical and computational contributions, whose results are described broadly in literature, helped solve such problems. Many of the developed and widely recommended techniques are based on random searches [5], [17], [18], [20]. Pure random search, multi-start local search and controlled random search methods belong to this category. Many algorithms utilize random search and are biologically- or heuristics-inspired by biology or physics. Genetic algorithms, evolutionary strategies, simulated annealing, swarm optimization are all of the heuristic nature [13], [14], [21]–[23]. Randomized search algorithms are regarded to be flexible, robust and less demanding in terms of problem properties than deterministic methods. Unfortunately, efficient stochastic algorithms require a large number of iterations and objective function evaluations, especially for high dimension problems. Therefore, the efficiency and scalability of solvers is often a key issue.

3. Description of Optimization Algorithms

3.1. CMA-ES

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [12] is an evolutionary algorithm based on Gaussian mutation and deterministic selection. Evolutionary strategies are stochastic search methods inspired by the principles of biological evolution typically using a multivariate normal mutation distribution. They operate on a set of search points. CMA-ES is considered to be one of the best choices against ill-conditioned, non-convex black-box optimization problems in the continuous domain. The general algorithm to solve the black-box problem (1) is to sample a numerous independent points from a given distribution P, evaluate these points based on their performance measures f and update the distribution parameters. All these steps are executed until the termination criterion is met. In the CMA-ES algorithm, P is a multivariate normal distribution that is a generalization of the one-dimensional (univariate) normal distribution to higher dimensions. Hence, a random vector is said to be n-variate normally distributed, if every linear combination of its n components has a univariate normal distribution. Normal distribution is a good candidate for randomized searches – its entropy for the mean values, variances and covariances given is the largest of all distributions in $\Re^n$ and the coordinate directions are not distinguished in any way. Therefore, in CMA-ES, a population of new search points (set of individuals) is generated by sampling a multivariate normal distribution. In every iteration k new individuals $x_i^k \in \Re^n$ are calculated as:

$$x_i^{k+1} = m^k + \sigma^k \times \mathcal{N}_i^k(0, C^k), \quad i = 1, \ldots, I\ , \qquad (2)$$

where m^k and C^k denote the approximated mean value and the $n \times n$ covariance matrix of the search distribution at iteration k, $\sigma^k > 0$ is the standard deviation – step-size at the k-th iteration, $\mathcal{N}_i^k(0, C^k)$ is a normal distribution with the mean 0 and I is a population size. Hence, the mutation is performed by a perturbation with a covariance matrix which is iteratively updated to guide the search towards areas with expected lower objective values.
After generation of the population of individuals, they are evaluated on f (1), sorted and transformed according to (2). Upon every iteration, all distribution parameters (m^k, C^k, σ^k) are updated.

3.2. PSO

The particle swarm optimization (PSO) algorithm [14] is a simple population-based stochastic optimization technique that emerged from the simulations of the behavior of bird flocks and fish schools. At each iteration a set of points (a swarm of particles) evolve their position in the search space with a velocity vector associated with each particle to find the global optimum. A new population of particles is generated from the previous swarm using randomly generated weights and parameters specific to the algorithm. Hence, both positions and velocities of the i-th particle $x_i^k \in \Re^n$ ($i = 1, \ldots, I$) are updated for every k iteration ($k = 1, \ldots, K$) according to the following rules:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \ , \tag{3}$$

where x_i^{k+1} and v_i^{k+1} denote the position and the velocity of the i-th particle at the $k+1$ iteration, respectively. The velocity is updated as:

$$v_i^{k+1} = wv_i^k + \varphi_1 U_1^k (x_{i_{opt}}^k - x_i^k) + \varphi_2 U_2^k (x_{i_{nopt}}^k - x_i^k) \ , \tag{4}$$

where w is the weighting parameter, φ_1 and φ_2 denote weights of global and local information, U_1^k and U_2^k are $n \times n$ diagonal matrices with elements randomly drawn from a uniform distribution $[0, 1]$, $x_{i_{opt}}^k$ is the best position of the i-th particle found so far and $x_{i_{nopt}}^k$ is the best position of all particles within the neighborhood of x_i^k or in the special case, within the swarm.
The first component in (4) makes a particle move in the previous direction, the second one makes a particle return to the best position calculated so far and the last component makes a particle follow the best position in the current iteration (within neighborhood or whole swarm). The main disadvantage of the PSO algorithm is its sensitivity to a velocity – if the velocity is too low the convergence speed is low, if it is too high, the algorithm falls into the local minimum.
In the research presented two versions of PSO with different concepts of $x_{i_{nopt}}^k$ in (4) were considered:

- Global-best PSO algorithm (PSO-glob) using a star topology. Every particle compares itself with the best-performing particle in the swarm.

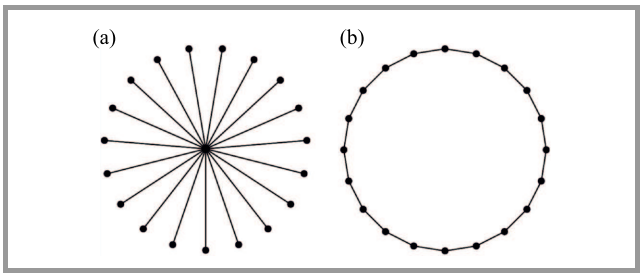

Fig. 1. PSO variants and topologies: (a) star topology (PSO-glob) and (b) ring topology (PSO-loc).

- Local-best PSO algorithm (PSO-loc) using a ring topology. Every particle compares itself only with its nearest-neighbors computed by applying the chosen distance metric.

4. Implementation Overview

The COmparing Continuous Optimizers (COCO) [24] benchmarking platform version 2.2.1 was used to evaluate the performance of PSO and CMA-ES methods. COCO is an integrated software environment that can be successfully used for systematic and sound comparisons of global optimization solvers. It provides a set of benchmark functions and tools for processing and visualizing results of calculations. The COCO source code is available at https://github.com/numbbo/coco. A detailed description of the experimental procedure for conducting the experiments using the COCO system is presented in [25]. The measures of performance assessment implemented in COCO are described in [26].
The COCO platform offers Python language support to provide the implementation of optimization solvers. Therefore, both CMA-ES and PSO algorithms implemented in Python were adopted and incorporated in COCO. The CMA-ES solver was taken from the *pycma* library [27], while the PSO solver was taken from the *pyswarms* library [28].
The benchmark optimization problems were taken from the black box optimization benchmarking (BBOB) [15] suite. It provides numerous testing problems with various characteristics, dimensions and complexities. All benchmark functions to be minimized are divided into groups, each with specific characteristics of the objective function, i.e., separable, moderate, ill-conditioned, multi-modal, weakly structured multi-modal.

5. Experimental Procedure and Performance Measures

Numerous experiments whose results are presented in this paper were conducted. The experimental procedure was executed within the COCO benchmarking platform, following the best practices for the assessment of performance

of optimization algorithms executed in a black-box scenario [25]. The performance of the algorithms was measured based on the number of run times, i.e. objective function evaluations, needed to reach one or several quality indicator target values. The details of the experimental setup are presented below.

5.1. Experimental Procedure

The goal of each experiment was to find the global solution of the problem (1) with the prescribed accuracy, i.e. the point $\{x \in \Re^{dim} : f(x) = f_{target}\}$ with f_{target} defined as:

$$f_{target} = f_{opt} + \varepsilon_{min} \ , \tag{5}$$

where $f(x)$ denotes the objective function, $f_{opt} = f(x_{opt})$, where x_{opt} is the solution of (1), and $\varepsilon_{min} = 10^{-8}$ is the assumed precision. As mentioned above, the performance assessment of each experiment was based on the number of the objective function evaluations. The evaluations conducted by the algorithm were recorded for each target $f^{\varepsilon}_{target} = f_{opt} + \varepsilon$ that was reached for various precisions ε. Precisions were chosen uniformly on the logarithmic scale from the interval $\varepsilon \in [10^{-8}, 10^{2}]$.

All benchmark problems, i.e. 24 noiseless unconstrained optimization problems concerned with the minimization of objective functions from the BBOB test suite $\Omega_f = \{f1, f2, \ldots, f24\}$ were taken into consideration. Problems were solved with the increasing number of decision variables (dimension), i.e. $dim \in \{2, 3, 5, 10, 20, 40\}$.

Every single setup, i.e. the problem of optimization of a given benchmark function and the dimension of the problem, was executed over $N_{trial} = 15$ independent trial runs. The performance was evaluated over all trials.

Three solvers were used and compared: CMA-ES, global-best PSO and local-best PSO. Additionally, the performance of the considered methods was confronted with that of the artificial algorithm labeled *best2009*. The scores of the *best2009* algorithm are based on the results reached by solvers submitted to the COCO 2009 competition. For each of the setups the best performing algorithm's results were incorporated.

The calculations were stopped after reaching the target value of the objective function f_{target} defined in (5) or depleting the budget Max_{iter} for the number of objective function evaluations equal to $3000 \cdot (dim + 2)$, where dim denoted the problem dimension for a given setup (trial).

5.2. Performance Measures

Two measures were used to evaluate the performance and to compare the efficiency of CMA-ES and two variants of PSO algorithms. The aim of the first measure – the average running time (aRT) – is to show the successful performance of the tested techniques. Average runtime is an estimation of the algorithm's expected runtime expressed in the number of objective function evaluations. The smaller the value, the better [25].

$$aRT = \frac{1}{N_{success}} \sum_{t=1}^{N_{trial}} fevals_t, \quad N_{success} \leq N_{trial} \ , \tag{6}$$

where aRT denotes the average runtime for N_{trial} relevant trials for a given setup. $fevals_t$ is the number of objective function evaluations performed in the t-th trial until the target value f_{target} was reached. $N_{success}$ is the number of successful trials, i.e., trials where f_{target} was reached.

The second measure – an empirical cumulative distribution function (ECDF) [29] – was used to display the proportion of problems solved within a given limit of objective function evaluations. In general, the purpose of the measure is to show and compare the speed of convergence of specific optimization algorithms [25].

Let us consider a subset of the available benchmarking functions $\omega_f \subseteq \Omega_f$ with a fixed number of dimensions dim. We define S as the set of bootstrap executions, where each element $s \in S$ is denoted by a triplet $s = (f_m, f^{\varepsilon}_{m,target}, t)$, $f_m \in \omega_f$. $f^{\varepsilon}_{m,target}$ is the target solution with the required precision ε. In this experiment, the solutions were calculated for 51 precisions chosen from $[10^{-8}, 10^{2}]$. t denotes a trial index, $t = 1, \ldots, N_{trial}$.

Every triplet s corresponds to the number of function f_m evaluations performed by the optimization algorithm to reach the target solution with $f^{\varepsilon}_{m,target}$. This number is denoted by R_s.

We calculate ECDF as follows:

$$\hat{F}_S(r) = \frac{|\{s \mid R_s \leq r\}|}{|S|}, \tag{7}$$

where the variable r denotes the number of function evaluations, $|S|$ is a number of all triplets from the set S. Hence, $\hat{F}_S(r)$ is the fraction of bootstrap executions for which the target value of the objective function was reached after R_s function evaluations, such that $R_s \leq r$.

To provide a compact assessment of all tested algorithms, taking into account both quality of the solution obtained and efficiency of the optimization solvers, ECDF was computed over subsets of multiple benchmark functions (ω_f). The functions were grouped based on their characteristics. The proposed groups are described in the Section 6.

6. PSO and CMA-ES Performance Evaluation

Multiple experiments for 24 benchmark functions given in [15], [30] were conducted. The results are presented and discussed in this section.

6.1. CMA-ES and PSO Configuration Setup

The following values of parameters typical of the tested methods were determined.

Table 1
Number of particles in PSO for various problem dimensions

Problem dimension (dim)	Swarm size
$dim \leq 10$	40
$10 < dim \leq 25$	60
$25 < dim \leq 40$	90

- The CMA-ES method was executed with the step size of $\sigma = 0.2$.
- The hyperparameters of PSO solvers were consistent across all tests. The following values were used: inertial coefficient $w = 0.9$, acceleration coefficients $c_1 = 0.5$, $c_2 = 0.3$. For the local-best variation, the Euclidean norm was used to measure the distance between neighbors. The number of nearest neighbors considered was equal to $k = 2$. The swarm size was adapted to the size and complexity of the problem. Hence, the number of particles was different for different tests (see Table 1).

6.2. Experimental Results

The global minima of 24 benchmark functions in the search space $[-5, 5]^{dim}$ were calculated with the use of CMA-ES and PSO methods. Test results were compared with the reference solutions of *best2009*.
All experiments were performed on a unified hardware platform: Intel Core i7-2640M CPU @ 2.80 GHz with 1 processor and 4 cores.
The results, i.e. efficiency of all tested algorithms tested for various test functions and problem dimensions are presented in Table 2 and Figs. 2, 3, 4. Different markers used in all figures correspond to different algorithms:

- a circle – CMA-ES results,
- a square – PSO global-best results,
- a triangle – PSO local-best results.

Table 2
The average running time divided by the best *aRT* obtained for *best2009*; 24 benchmark functions ($dim = 5$), $N_{trial} = 15$, $Max_{iter} = 21000$. "–" denotes that the target solution was not reached.

Function	CMA-ES			PSO-glob			PSO-loc		
	$\varepsilon = 10^0$	$\varepsilon = 10^{-2}$	$\varepsilon = 10^{-5}$	$\varepsilon = 10^0$	$\varepsilon = 10^{-2}$	$\varepsilon = 10^{-5}$	$\varepsilon = 10^0$	$\varepsilon = 10^{-2}$	$\varepsilon = 10^{-5}$
f1	20	44	86	83	274	668	846	4361	–
f2	18	23	28	239	443	524	–	–	–
f3	21	–	–	88	188	190	–	–	–
f4	–	–	–	–	–	–	–	–	–
f5	26	27	27	–	–	–	–	–	–
f6	3.4	3.5	2.3	41	35	26	1343	–	–
f7	3.1	2.4	2.5	60	–	–	–	–	–
f8	8.3	9.1	10	210	–	–	–	–	–
f9	3.9	5.7	6.2	131	40 5	–	710	–	–
f10	3.3	3.5	3.2	–	–	–	–	–	–
f11	6.6	1.7	1.5	–	–	–	–	–	–
f12	12	14	6	1079	–	–	1099	–	–
f13	7.7	7.3	2.1	451	–	–	–	–	–
f14	5.3	7.6	6.8	23	44	–	295	–	–
f15	7.9	–	–	32	–	–	–	–	–
f16	2.6	1.1	1.3	26	–	–	46	–	–
f17	18	2.6	24	25	14	–	503	–	–
f18	8.6	3	–	30	–	–	374	–	–
f19	741	0.4	1.2	9820	–	–	–	–	–
f20	19	2.9	2.7	18	–	–	75	–	–
f21	7.8	8.2	8.2	23	20	23	77	–	–
f22	23	17	16	65	89	142	92	292	–
f23	7.2	1.1	0.99	39	–	–	16	–	–
f24	–	–	–	–	–	–	–	–	–

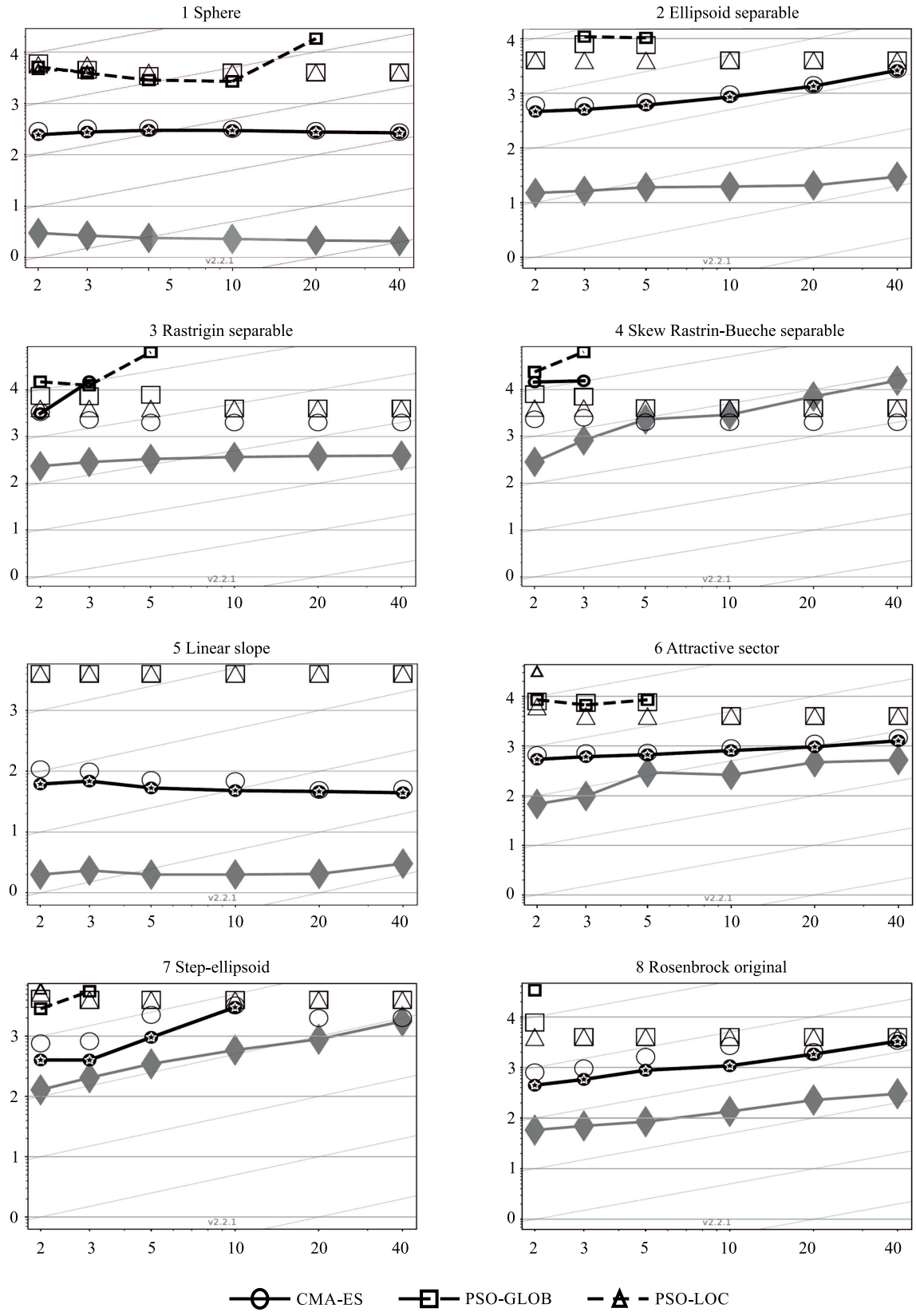

Fig. 2. PSO and CMA-ES efficiency. The average running time divided by dimension (aRT/dim); 24 benchmark functions, $\varepsilon = 10^{-8}$, number of trials $N_{trial} = 15$. Legend: ◯ – covariance matrix adaptation evolution strategy, □ – global particle swarm optimization, △ – local particle swarm optimization. Slanted grid lines indicate quadratic scaling with the dimension.

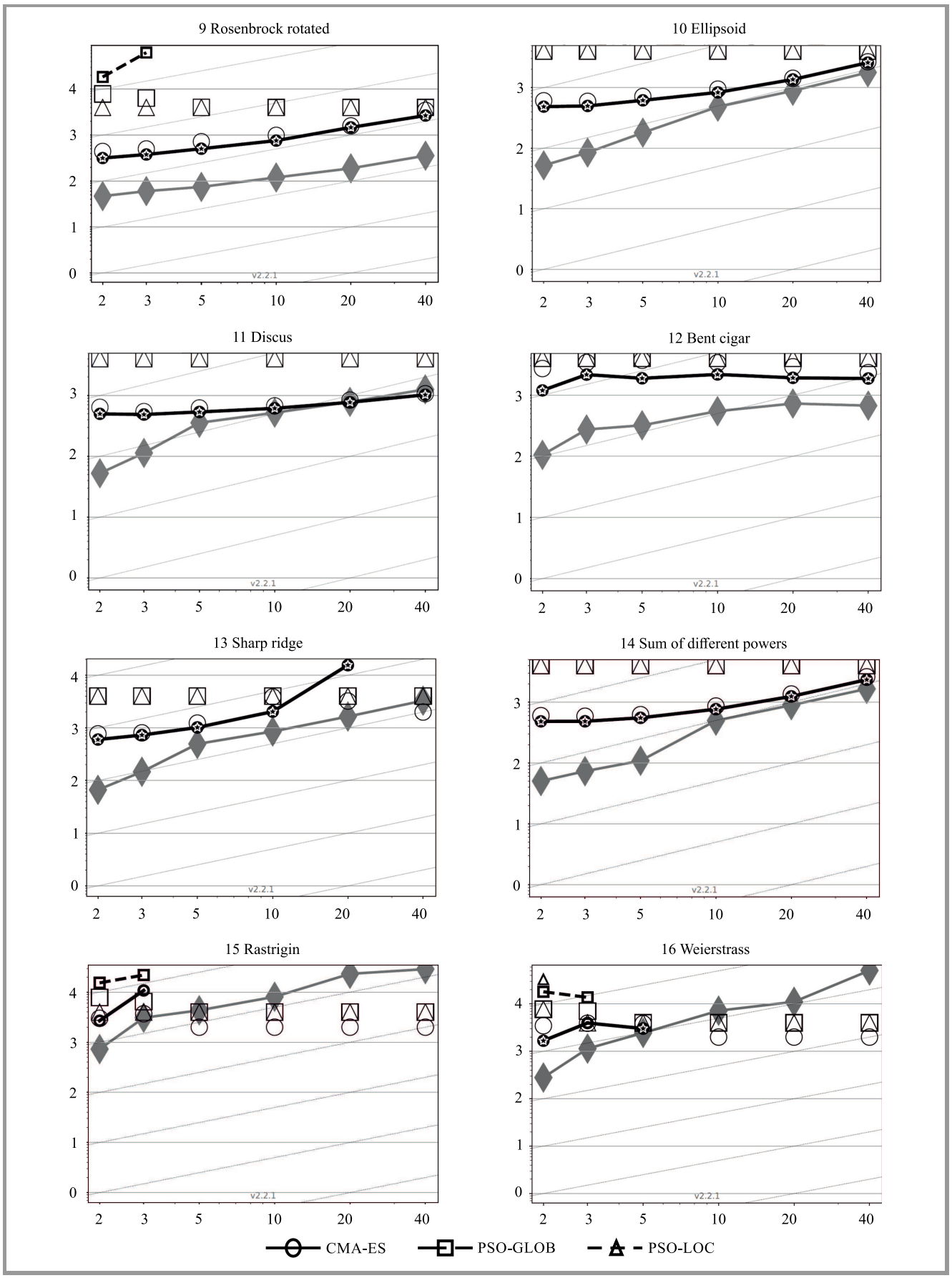

Fig. 2 – continued.

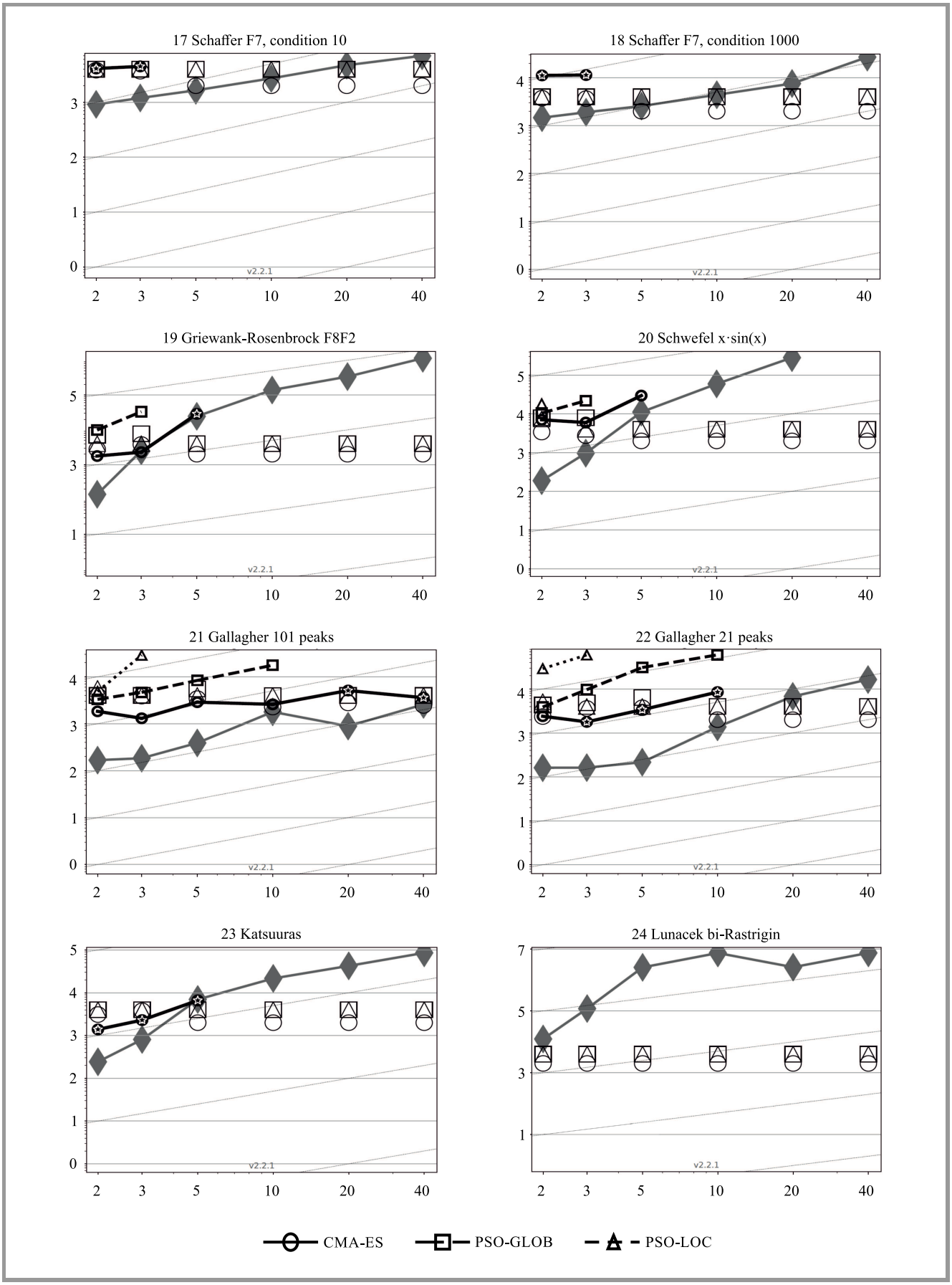

Fig. 2 – continued.

The table shows the average running time *aRT* (6) divided by the respective best *aRT* obtained for *best2009*, each for one of 24 benchmark functions and the problem dimension of $dim = 5$. The plots in Fig. 2 show the average running time *aRT* (6) as $\log_{10}$ value divided by the problem dimension, i.e. $\log(aRT)/dim$, each for one of 24 benchmark functions and various problem dimensions. The experiments were conducted for the following number of decision variables $dim \in \{2, 3, 5, 10, 20, 40\}$ for each function and the prescribed accuracy of $\varepsilon = 10^{-8}$. All tests were executed $N_{trial} = 15$ times. The meaning of the additional symbols in all 24 plots are as follows:

- light symbols (circle, square, triangle) give the maximum number of function evaluations from the largest trial divided by the problem dimension,
- black stars indicate a statistically better result compared to all other algorithms.

The results obtained, as shown in Table 2 and Fig. 2, indicate that PSO was unable to reach the target function value f_{target} for the problems with $dim > 3$ for most of the benchmark functions. The best results were recorded for the *Sphere* function, Fig. 2-1 – global-best PSO (PSO-glob) was able to find the solution for a 20-dimensional problem. In this case, the number of function evaluations decreased until the number of dimensions reached 10, which suggest that a better choice of the algorithm's hyperparameters could be made. For the *Gallagher* functions, Fig. 2-21, we can observe that *aRT* grows on a nearly quadratic basis until the number of dimensions is equal to 10. Local-best PSO (PSO-loc) performed very poorly, reaching f_{target} only for *Sphere* and *Gallagher* functions with a small number of dimensions.

As for CMA-ES, the results were much better. The method succeeded in finding the target solution of 11 functions from the benchmark set, even for $dim = 40$. In other cases the results of CMA-ES were similar to PSO, e.g. they failed completely for *Lunacek bi-Rastrigin*, Fig. 2-24. In some cases like the *Elipsoid* function, Fig. 2-10, or *Discus*, Fig. 2-11, the *aRT* was similar to the referential solutions from *best2009* (for bigger dimensions). However, in general, CMA-ES did considerably better then both PSO versions.

The goal of the second series of experiments was to test the statistical significance of results. All benchmark functions (f1–f24) listed in Fig. 2 were divided into 6 groups with different characteristics:

1. Separable functions (f1–f5) – optimal value of a given coordinate of the decision variable does not depend on the choice of the remaining coordinates.
2. Moderate functions (f6–f9) – moderate dimension of the decision variable vector.
3. Ill-conditioned functions (f10–f14) – different variables, or different directions in search space, show a largely different sensitivity in their contribution to the objective function value.
4. Multi-modal functions (f15–f19) – multiple minima and maxima.
5. Weakly structured multi-modal functions (f20–f24) – many solutions with similar values of the performance measure.
6. All functions.

Multiple experiments for all functions, as well as PSO and CMA-ES optimization methods were performed. The number of trials $N_{trial} = 15$. Calculations were conducted for 51 values of f_{target} with various precisions $\varepsilon \in [10^{-8}, 10^{2}]$. Rank-sum test for a given target f_{target} using, for each trial, either the number of needed function evaluations needed to reach f_{target} (inverted and multiplied by -1), or, if the target was not reached, the best precision – ε-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration. Problems with two dimensions were tested: $dim = 5$ (Fig. 3) and $dim = 20$ (Fig. 4). Both figures present the cumulative distribution of the measure $\hat{F}_S$ (7). The results obtained for PSO and CMA-ES are compared with the reference solutions from *best2009* (shown as a thick line with diamond markers).

Figures depicting empirical cumulative distribution functions (ECDFs) confirm that overall CMA-ES performs better then both versions of the PSO method. For dimensions smaller then 5 (Fig. 3), the differences in the optimization of separable functions are not so significant, PSO-glob performs rather well, compared to CMA-ES. On the other hand, in the case of ill-conditioned functions, CMA-ES hugely outranks PSO, with its performance nearly matching the reference solution from *best2009*. A difference is also noticeable for the larger dimensions, Fig. 4, especially in the case of ill-conditioned and moderate functions. However, for the more demanding, multi-modal problems, the results of all tested algorithms fall short compared to the reference solution. PSO-loc clearly stands out as the most ineffective of all tested methods.

As the final observation, it is worth mentioning that in solving a given black-box problem, the choice of the proper optimization algorithm and proper tuning of its parameters are of crucial significance. In the presented experiments, the *best2009* solutions outclass, in all cases, the results calculated by CMA-ES and PSO. This is to be expected, as *best2009* consists of the solutions of multiple algorithms, each adjusted for a given set of problems.

6.3. Possible Improvements of CMA-ES

CMA-ES is a population-based stochastic technique. The population size plays a big factor in the algorithm's efficiency, depending on the use case. With a default (small) population size, CMA-ES is a robust and fast method intended mainly for local search optimization. By increasing the size of the population, the algorithm can be successfully employed for more global search problems. Taking into account both of those principles, a modified version of this

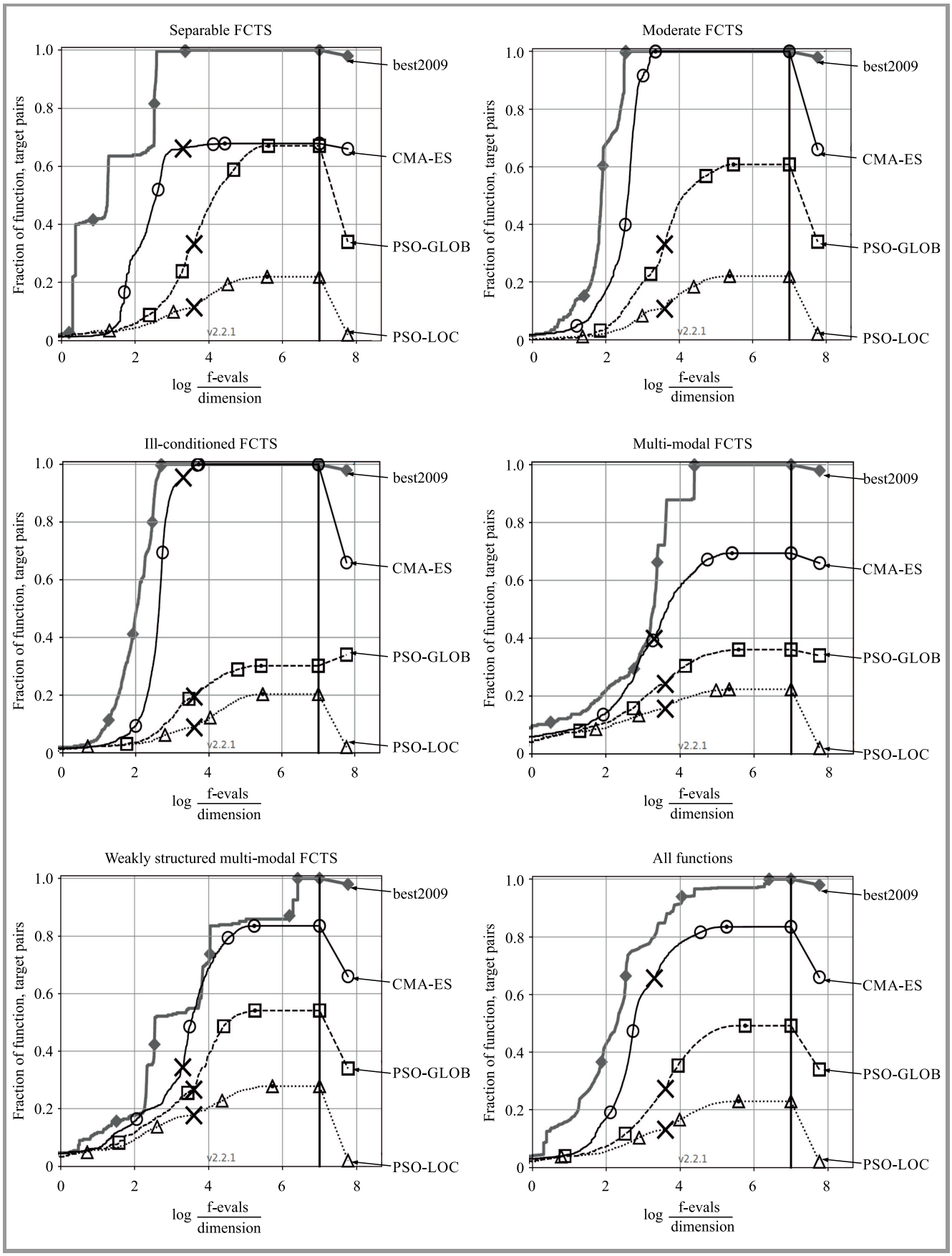

Fig. 3. Bootstrapped empirical cumulative distribution of the objective function evaluations divided by dimension ($fevals/dim$), $dim = 5$, $\varepsilon \in [10^{-8}, 10^{2}]$.

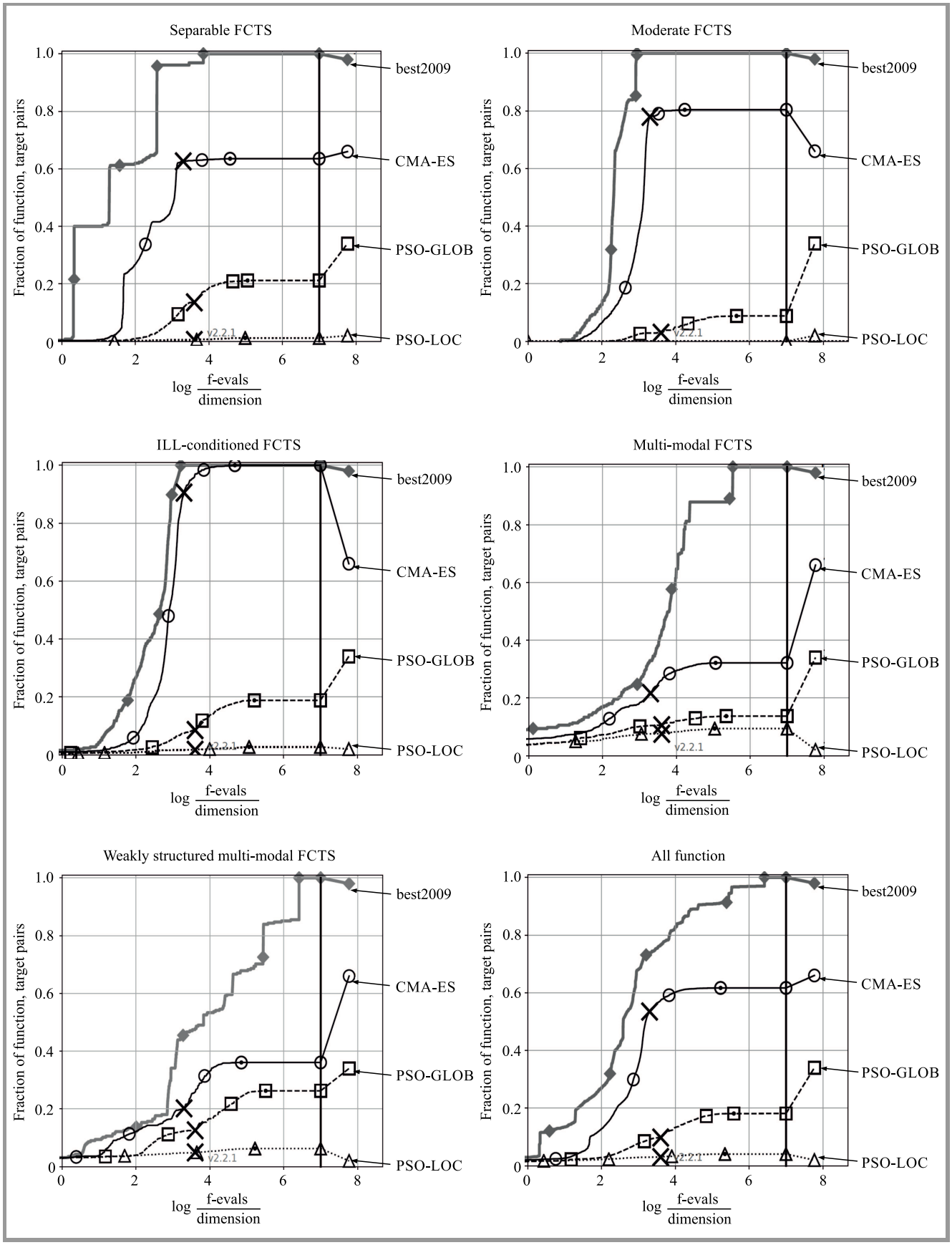

Fig. 4. Bootstrapped empirical cumulative distribution of the objective function evaluations divided by dimension (*fevals*/*dim*), $dim = 20$, $\varepsilon \in [10^{-8}, 10^{2}]$.

method, i.e. BI-POP CMA-ES was developed. It applies two interlaced multistart regimes altering the size of the population. One of them increases the population size by the factor of two and starts with a higher initial step, while the other decreases the population size and uses smaller steps equal to σ in 2. The promising results of BI-POP CMA-ES in global optimization are reported in literature, i.e. [31].

7. Summary and Conclusion

The paper provides a short report on the efficiency and availability of two biologically-inspired CMA-ES and PSO methods that are designed to tackle non-convex and ill-conditioned black-box optimization problems.

The worst performance was observed for the local version of the PSO method. The global version was better in all of the cases, while CMA-ES outranked both PSO methods.

This should not come as a surprise, as various modifications of the CMA-ES algorithm are currently considered to be state-of-the-art in the field of black-box optimization. The bare CMA-ES algorithm performs well, although the numerous experiments confirmed that the reference algorithm outclasses both PSO and the classic variant of CMA-ES.

The final conclusion is that PSO techniques are very sensitive to hyperparameters of the algorithms and tuning of these parameters is a challenging task. A better choice of the algorithm's hyperparameters adapted to each function and dimension can seriously influence the final result. Since the CMA-ES method does not require tedious parameter tuning, the choice of the strategy to be adopted while setting the internal parameters is not left to the user. Therefore, it is much more convenient then algorithms such as PSO.

With the standard version of CMA-ES, there is room for improvement. Modification of the original algorithm and its adaptation to the optimization problem to be solved can lead to better performance overall, making it a more reliable and versatile method. Therefore, we plan to compare the performance of the original CMA-ES and BI-POP CMA-ES mentioned in Subsection 6.3 in the future.

Acknowledgments

The work has been performed as part of the CYBERSECIDENT/369195/I/NCBR/2017 project, co-financed by the National Centre for Research and Development, under the CyberSecIdent Program.

References

[1] M. J. Asher, B. F. W. Croke, A. J. Jakeman, and L. J. M. Peeter, "A review of surrogate models and their application to groundwater modeling", *Water Resources Res.*, vol. 51, no. 8, pp. 5957–5973, 2013 (doi: 10.1002/2015WR016967).

[2] T. Bartz-Beielstein, "A survey of model-based methods for global optimization", in *Proc. of 7th Int. Conf. on Bioinspired Optimiz. Methods and their Applications BIOMA 2016*, Bled, Slovenia, 2016, pp. 3–20, 2016.

[3] A. T. Nguyen *et al.*, "A review on simulation-based optimization methods applied to building performance analysis", *Applied Energy*, vol. 113, pp. 1043–1058, 2014 (doi: 10.1016/j.apenergy.2013.08.061).

[4] E. Niewiadomska-Szynkiewicz and J. Błaszczyk, "Simulation-based optimization methods applied to large scale water systems control", in *Proc. of 16th IEEE Int. Conf. on Scalable Comput. and Commun. ScalCom2016*, Toulouse, France, 2016, pp. 649–656 (doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld. 2016.0108).

[5] J. C. Spall, *Introduction to Stochastic Search and Optimization*. Wiley, 2003 (doi: 10.1002/0471722138, ISBN 9780471330523).

[6] S. Gil, A. Kott, and A. L. Barabási, "A genetic epidemiology approach to cyber-security", *Scientific Rep.*, vol. 4, pp. 1–7, 2014 (doi: 10.1038/srep05659).

[7] G. Kumar, K. Kumar, and M. Sachdeva, "The use of artificial intelligence-based techniques for intrusion detection: a review", *Artif. Intell. Rev.*, vol. 34, no. 4, pp. 369–387, 2010 (doi: 10.1007/s10462-010-9179-5).

[8] E. Niewiadomska-Szynkiewicz, A. Sikora, and J. Kołodziej, "Modeling mobility in cooperative ad hoc networks", *Mobile Netw. and Appl.*, vol. 18, no. 5, pp. 610–621, 2013 (doi: 10.1007/s11036-013-0450-2).

[9] P. Szynkiewicz, "A novel GPU-enabled simulator for large scale spiking neural networks", *J. of Telecommun. and Inform. Technology*, no. 2, pp. 34–42, 2016.

[10] M. Marks, E. Niewiadomska-Szynkiewicz, and J. Kołodziej, "High performance wireless sensor network localisation system", *Int. J. of Ad Hoc and Ubiquitous Comput.*, vol. 17, no. 2–3, pp. 122–133, 2014 (doi: 10.1504/IJAHUC.2014.065776).

[11] P. Szynkiewicz and A. Kozakiewicz, "Design and evaluation of a system for network threat signatures generation", *J. of Comput. Sci.*, vol. 22, pp. 187–197, 2017 (doi: 10.1016/j.jocs.2017.05.006).

[12] N. Hansen, "The CMA evolution strategy: A tutorial", Tech. Rep., INRIA, 2016, arXiv: 1604.00772.

[13] J. A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, Eds., *Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms*. Berlin: Springer, 2006 (ISBN 9783540324942).

[14] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", in *Proc. of the 6th Int. Symp. on Micro Machine and Human Sci. MHS'95*, Nagoya, Japan, 1995, pp. 39–43 (doi: 10.1109/MHS.1995.494215).

[15] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions", Tech. Rep. RR-6829, INRIA, 2009 [Online]. Available: https://hal.inria.fr/inria-00362633/document

[16] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: a review of algorithms and applications", *Annals of Operations Res.*, vol. 240, no. 1, pp. 351–380, 2016 (doi: 10.1007/s10479-015-2019-x).

[17] J. April, F. Glover, J. P. Kelley, and M. Laguna, "Practical introduction to simulation optimization", in *Proc. of the 2003 Conf. on Winter Simul. WSC 2003*, New Orleans, LA, USA, 2003, pp. 71–78 (doi: 10.1109/WSC.2003.1261410).

[18] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: a review of algorithms and comparison of software implementations", *J. of Global Optimiz.*, vol. 56, no. 3, pp. 1247–1293, 2013 (doi: 10.1007/s10898-012-9951-y).

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C. The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1992 (ISBN 9780521431088).

[20] M. M. Ali, A. Törn, and S. Vittanen, "A numerical comparison of some modified controlled random search algorithms", *J. of Global Optimiz.*, vol. 11, no. 4, pp. 377–385, 1997 (doi: 10.1023/A:1008236920512).

[21] A. Dekkers and E. Aarts, "Global optimization and simulated annealing", *Mathem. Programming*, vol. 50, no 3, pp. 367–393, 1999 (doi: 10.1007/BF01594945).

[22] J. Kołodziej, *Evolutionary Hierarchical Multi-Criteria Metaheuristics for Scheduling in Large-Scale Grid Systems*. *Studies in Computational Intelligence*, vol. 419. Springer, 2012 (ISBN 9783642289705).

[23] Z. Michalewicz and D. B. Fogel, *How to Solve It: Modern Heuristics*. Berlin Heidelberg: Springer, 2002 (ISBN 3540660615).

[24] N. Hansen, A. Auger, O. Mersmann, T. Tušar, and D. Brockhoff, "COCO: A platform for comparing continuous optimizers in a black-box setting", *ArXiv e-prints*, 2016, arXiv: 1603.08785.

[25] N. Hansen, A. Auger, D. Brockhoff, D. Tušar, and T. Tušar, "COCO: Performance assessment", *ArXiv e-prints*, 2016, arXiv: 1605.03560.

[26] N. Hansen, T. Tušar, O. Mersmann, A. Auger, and D. Brockhoff. "COCO: The experimental procedure", *ArXiv e-prints*, 2016, arXiv: 1603.08776.

[27] N. Hansen, "Python implementation of CMA-ES", 2016 [Online]. Available: https://github.com/CMA-ES/pycma

[28] L. J. V. Miranda, "PySwarms: a research toolkit for particle swarm optimization in Python", *The J. of Open Source Softw.*, vol. 3, no. 21, pp. 1–2, 2018 (doi: 10.21105/joss.00433).

[29] A. W. van der Vaart, *Asymptotic Statistics*. Cambridge University Press, 2000 (doi: 10.1017/CBO9780511802256, ISBN 9780511802256).

[30] S. Finck, N. Hansen, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions", Tech. Rep. 20, Research Center PPE, 2009.

[31] H. Beyer and B. Sendhoff, "Simplify your covariance matrix adaptation evolution strategy", *IEEE Trans. on Evolut. Comput.*, vol. 21, no. 5, pp. 746–759, 2017 (doi: 10.1109/TEVC.2017.2680320).

Paweł Szynkiewicz received his M.Sc. in Computer Science from the Warsaw University of Technology, Poland, in 2015. Currently he is a Ph.D. student at the Systems Research Institute, Polish Academy of Science. Between 2015–2016 he was employed at Comarch, and between 2016–2017 at BrightSolutions IT. He has been working for NASK since 2017. His research interest focuses on software technologies, computer networks security, global optimization, machine learning, neural networks and evolutionary algorithms.

https://orcid.org/0000-0001-8872-5505
E-mail: pawel.szynkiewicz@nask.pl
Research and Academic Computer Network (NASK)
Kolska 12
01-045 Warsaw, Poland