Using the empirical attainment function for analyzing single-objective black-box optimization algorithms

López-Ibáñez, M.; Vermetten, D.L.; Dreo, J.; Doerr, C.

Citation
López-Ibáñez, M., Vermetten, D. L., Dreo, J., & Doerr, C. (2025). Using the empirical attainment function for analyzing single-objective black-box optimization algorithms. *Ieee Transactions On Evolutionary Computation*. doi:10.1109/TEVC.2024.3462758

Version:	Accepted Manuscript
License:	Licensed under Article 25fa Copyright Act/Law (Amendment Taverne)
Downloaded from:	https://hdl.handle.net/1887/4246127

Note: To cite this publication please use the final published version (if applicable).

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

Using the Empirical Attainment Function for Analyzing Single-objective Black-box Optimization Algorithms

Manuel López-Ibáñez, *Senior Member, IEEE*, Diederick Vermetten, Johann Dreo, Carola Doerr

***Abstract*—A widely accepted way to assess the performance of iterative black-box optimizers is to analyze their empirical cumulative distribution function (ECDF) of pre-defined quality targets achieved not later than a given runtime. In this work, we consider an alternative approach, based on the empirical attainment function (EAF) and we show that the target-based ECDF is an approximation of the EAF. We argue that the EAF has several advantages over the target-based ECDF. In particular, it does not require defining a priori quality targets per function, captures performance differences more precisely, and enables the use of additional summary statistics that enrich the analysis. We also show that the average area over the convergence curves is a simpler-to-calculate, but equivalent, measure of anytime performance. To facilitate the accessibility of the EAF, we integrate a module to compute it into the IOHanalyzer platform. Finally, we illustrate the use of the EAF via synthetic examples and via the data available for the BBOB suite.**

***Index Terms*—Performance assessment, evolutionary computation, empirical attainment function, empirical cumulative distribution function**

I. Introduction

In the context of benchmarking sampling-based single-objective optimization algorithms, one of the most common ways to illustrate the performance on a single problem instance is to plot the objective value of a best-so-far solution as a function of the number of solutions that have been evaluated. There are different options for the aggregation of such *convergence curves* over different problem instances. In evolutionary computation, one of the most popular ways to aggregate performance data over several problems is based on the so-called *empirical cumulative distribution function* (ECDF) of the number of targets attained at a given runtime. This alternative was notably promoted by the COCO platform [1]. To define an ECDF curve, one fixes a set of target values and counts, as a function of the solutions evaluated, how many of these targets have been achieved by the algorithm. These numbers are then normalized across all runs of the algorithm on each problem instance and then across all problem instances. The area under the target-based ECDF curve (AUC) can be seen as a measure of the anytime performance of the algorithm [2].[1]

This *target-based ECDF* generalizes the concept of time-to-target [3] or runtime distribution of a Las Vegas algorithm for a decision problem [4] to multiple targets.

One may argue that the choice of the target values for the definition of the target-based ECDF is somewhat arbitrary. We therefore consider in this paper an alternative that directly works with the probabilities that an algorithm achieves a certain quality within a given budget of function evaluations. We approximate this probability by the observed performance data, in the form of an empirical attainment function (EAF) [5]. The EAF was originally proposed for the analysis and visualization of multi-objective stochastic optimizers [6, 7]. In this multi-objective optimization context, the EAF estimates the probability that a single run of an algorithm attains a certain vector of objective values within a fixed runtime.

Our contribution: In this paper we investigate the use of EAFs for the analysis of single-objective optimizers, previously considered in [8, Sec. 3.2.4], [9, Chap. 3], [10]. We first observe that the target-based ECDF is the average value of the EAF at the pre-defined targets. We also show that the target-based ECDF with an increasing number of well-spread targets converges in the limit to an EAF-based ECDF computed as the area under the EAF across the quality dimension divided by a scaling constant. As a result, we claim that the EAF captures more information than the target-based ECDF and therefore conclude that the EAF-based ECDF has all the benefits of the target-based approach [2] without the disadvantage of requiring an a priori set of targets.

We also show that the AUC of the EAF-based ECDF is equivalent to the AUC of the EAF. In fact, they both measure the expected area over the convergence curve (when minimizing quality) of a single run. Thus, any of these AUC values can be seen as a measure of anytime performance without requiring pre-defined targets.

The main drawback of the EAF is its higher computation and storage costs compared to the target-based ECDF. However, there are efficient algorithms for computing the EAF in this context and even more efficient approximations are possible. Therefore, we argue that employing the EAF in benchmarking single-objective black-box optimizers brings

M. López-Ibáñez is with Alliance Manchester Business School, University of Manchester, Manchester, M13 9PL, UK (Email: manuel.lopez-ibanez@manchester.ac.uk).

D. Vermetten is with Leiden Institute for Advanced Computer Science, Leiden, The Netherlands (Email: d.l.vermetten@liacs.leidenuniv.nl).

J. Dreo is with Pasteur Institute, Université Paris Cité, Bioinformatics and Biostatistics Hub, Department of Computational Biology, Systems Biology Group, Paris, France (Email: johann.dreo@pasteur.fr).

C. Doerr is with Sorbonne Université, CNRS, LIP6, Paris, France (Email: carola.doerr@lip6.fr).

[1]The COCO platform [1] minimizes area over the ECDF curve, which is equivalent to the maximum runtime minus the AUC, because the former approximates the geometric average runtime.

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

significant benefits with only a moderate overhead over the target-based ECDF.

To facilitate the use of EAFs for the analysis of single-objective black-box optimization algorithms, we provide with this work an integration of the `eaf` package [7] into the IOHanalyzer platform [11]. We illustrate the use of this module by analyzing performance data from the well-known Black-Box Optimization Benchmark (BBOB) collection [12]. In particular, we investigate the differences between the target-based ECDF and the EAF-based ECDF in dependence of the number of targets, and analyze how the relative ranking between algorithms may change when replacing the ECDF with the EAF as the ranking criterion.

Outline of the paper: The relationship between the target-based ECDF and the EAF as well as the equivalence of various AUC metrics is discussed in Section II. In Section III, we argue that the EAF and the EAF-based ECDF can be efficiently computed in this single-objective context and we discuss their integration into IOHanalyzer. In Section IV, we present the results from the analysis of the BBOB data. Section V discusses additional statistical measures derived from the EAF and its application to the analysis single-objective optimizers. We conclude in Section VI with a summary of our arguments and perspectives for the future.

II. Relationship between the ECDF and the EAF

We recall the key definitions of the target-based ECDF in Section II-A and those of the EAF in Section II-B. In Section II-C we discuss the relationship between the two notions, and in Section II-D we treat the area under the target-based ECDF and that of the EAF-based ECDF.

A. Target-based Empirical Cumulative Distribution Function

In the following, we consider the *minimization* of objective functions $f\colon \mathbb{R}^D \to \mathbb{R}$, $D \in \mathbb{N}^+$, by means of an optimization algorithm A. Each run of algorithm A on a problem instance f returns a trajectory $\{(x^{(j)}, f(x^{(j)})) \mid j = 1, \ldots, t_{\max}\}$, with $x^{(j)}$ denoting the j-th point evaluated by A, ties broken arbitrarily in case of parallel evaluations, and $t_{\max}$ denoting the maximal number of function evaluations (the "*budget*"). The algorithm A may be stochastic, such that different runs on the same problem instance may return different trajectories.

We focus in this work on the quality of the *best-so-far solution*. To this end, we denote by $V(A, f, t, i) := \min\{f(x^{(j)}) \mid 1 \le j \le t\}$, the function value of the best among the first t evaluated solution candidates in run i.

Assuming that we have run algorithm A on f for r independent runs, the target-based ECDF for a finite set $Z \subseteq [z_{\min}, z_{\max}] \subset \mathbb{R}$ of target values at a given budget t is defined as the fraction of (run, target) pairs (i, z) for which the algorithm has found, within time t, a solution at least as good as z in the i-th run [2, 13]. Formally,

$$\widehat{F}_Z(t) = \frac{1}{r}\sum_{i=1}^{r}\frac{1}{|Z|}\sum_{z\in Z}\mathbb{1}\left(V(A,f,t,i)\le z\right) \quad , \tag{1}$$

where $\mathbb{1}(\mathcal{C})$ denotes the indicator function that is 1 when the condition $\mathcal{C}$ is satisfied and 0 otherwise.

Other definitions of ECDF may be found in the literature. For example, the target-based ECDF defined above is not the same as the classical multivariate ECDF [14].

B. Empirical Attainment Function

In the context of benchmarking the anytime performance of single-objective optimizers, given a pair (t, z) of budget and function value, respectively, we say that run i of algorithm A *attains* (t, z) if and only if algorithm A solving problem f obtains an objective function value not worse than z not later than t, that is, if and only if $V(A, f, t, i) \le z$. We can formulate the EAF in this context as the fraction of r runs of algorithm A on function f that attain an objective value z at budget t:

$$\widehat{\alpha}(t, z) = \frac{1}{r}\sum_{i=1}^{r}\mathbb{1}\left(V(A,f,t,i)\le z\right) \quad . \tag{2}$$

The above matches the original formulation of the EAF [5], if we consider that each run i produces a set $S_i \subset \mathbb{R}^2$ of points $(t, V(A, f, t, i))$, i.e., $S_i = \{(t, V(A, f, t, i)) \mid 1 \le t \le t_{\max}\}$. In this original formulation, a set $S_i \subset \mathbb{R}^d$ *attains* the point $y = (y_1, \ldots, y_d) \in \mathbb{R}^d$ iff there exists at least one point $s = (s_1, \ldots, s_d) \in S_i$ such that, for all $1 \le j \le d$, it holds that $s_j \le y_j$. In this case, the value of the EAF of a set of sets $\{S_1, \ldots, S_n\}$ at a given point $y \in \mathbb{R}^d$ is defined as the fraction of sets S_i, $i = 1, \ldots, n$, that attain that point y. See Figure 1 for an illustrative example of the EAF for three sets, corresponding to three runs of an algorithm.

C. Comparison of the target-based ECDF and the EAF

If we have a finite number of targets $|Z| \in \mathbb{N}^+$, the target-based ECDF value for a given budget t (1) equals the average EAF value in these targets for time t, as the following simple swap of summands shows.

$$\begin{aligned}
&\frac{1}{|Z|}\sum_{z\in Z}\widehat{\alpha}(t,z)\\
&= \frac{1}{|Z|}\sum_{z\in Z}\left(\frac{1}{r}\sum_{i=1}^{r}\mathbb{1}\left(V(A,f,t,i)\le z\right)\right)\\
&= \frac{1}{r}\sum_{i=1}^{r}\frac{1}{|Z|}\sum_{z\in Z}\mathbb{1}\left(V(A,f,t,i)\le z\right) = \widehat{F}_Z(t) \quad .
\end{aligned} \tag{3}$$

With the above equality at hand, it is easy to show that the target-based ECDF of a given t, evaluated for increasingly finer, well-spread partitions of $[z_{\min}, z_{\max}]$ converges to the following partial integral

$$F_{\widehat{\alpha}}(t) := \frac{1}{z_{\max} - z_{\min}}\int_{z_{\min}}^{z_{\max}}\widehat{\alpha}(t,z)\,\mathrm{d}z \tag{4}$$

of the EAF over the quality dimension z in this interval, evaluated for the same budget t. That is, the target-based ECDF with infinite number of well-spread targets is the area under the EAF at a fixed value of t divided by a scaling constant. Thus, the integral $F_{\widehat{\alpha}}(t)$ defines an *EAF-based ECDF* that serves the same purpose as the target-based ECDF and does not require predefined targets.

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

Formally, let $Z = \{z_1, \ldots, z_{|Z|}\} \subset [z_{\min}, z_{\max}]$ with $z_i < z_{i+1}$ for all $1 \leq i \leq |Z| - 1$, and let $\delta(Z) := \max\{z_1 - z_{\min},\ \max_{1 \leq i \leq |Z|-1}\{z_{i+1} - z_i\},\ z_{\max} - z_{|Z|}\}$ be the largest "gap" between any two neighboring target values in the set $Z \cup \{z_{\min}, z_{\max}\}$. Then, one can show that

$$F_{\widehat{\alpha}}(t) = \lim_{\delta(Z) \to 0} \widehat{F}_Z(t) \tag{5}$$

in the same way as the Riemann sum of a function converges to the integral of the function:

$$\begin{aligned} &\lim_{\delta(Z) \to 0} \widehat{F}_Z(t) \\ &= \frac{1}{r} \sum_{i=1}^{r} \frac{1}{z_{\max} - z_{\min}} \int_{z_{\min}}^{z_{\max}} \mathbb{1}\left(V(A, f, t, i) \leq z\right) \mathrm{d}z \\ &= \frac{1}{z_{\max} - z_{\min}} \int_{z_{\min}}^{z_{\max}} \frac{1}{r} \sum_{i=1}^{r} \mathbb{1}\left(V(A, f, t, i) \leq z\right) \mathrm{d}z \\ &= \frac{1}{z_{\max} - z_{\min}} \int_{z_{\min}}^{z_{\max}} \widehat{\alpha}(t, z)\, \mathrm{d}z \quad = F_{\widehat{\alpha}}(t) \ . \end{aligned} \tag{6}$$

Figure 2 shows the target-based ECDF and the EAF-based ECDF corresponding to the example runs shown in Figure 1. This example illustrates that the target-based ECDF with a small number of targets may either over-estimate or under-estimate the EAF-based ECDF, an observation that will be further discussed in Section IV-A.

D. Area under the EAF as a measure of anytime performance

The area under the curve (AUC) of the target-based ECDF has been used in the literature as a measure of the anytime performance of single-objective optimizers [2, 15]. As shown above, the target-based ECDF is an approximation to the EAF-based ECDF (4). Thus, the AUC of the EAF-based ECDF would be a more precise measure of anytime performance. The example in Fig. 2 already shows that the AUC values of the two curves are not equivalent.

The AUC of the EAF-based ECDF is given by:

$$\mathrm{AUC}\left(F_{\widehat{\alpha}}(t)\right) = \frac{1}{z_{\max} - z_{\min}} \int_{1}^{t_{\max}} \mathrm{d}t \int_{z_{\min}}^{z_{\max}} \widehat{\alpha}(t, z)\, \mathrm{d}z \ . \tag{7}$$

By definition, the double integral in the AUC of the EAF-based ECDF (7) is equivalent to the area under the EAF given by the following Lebesgue integral:

$$\mathrm{AUC}\left(\widehat{\alpha}(t, z)\right) = \int_{(1, -\infty)}^{(t_{\max}, z_{\max})} \widehat{\alpha}(t, z)\, \mathrm{d}t\, \mathrm{d}z \ , \tag{8}$$

which is equivalent to computing the well-known hypervolume metric in multi-objective optimization [16] of the three-dimensional points $\{(t, z, 1 - \widehat{\alpha}(t, z)) \mid t \in [1, t_{\max}], z \in [-\infty, z_{\max}]\}$ with reference point $(t_{\max}, z_{\max}, 1)$. This computation only requires $\Theta(n \log n)$ [17], where n is the number of points.

The only differences between (7) and (8) are the normalization constant $1/(z_{\max} - z_{\min})$ and the presence of finite $z_{\min}$ in the integral. If we wish to define $z_{\min}$ in (8), e.g., for normalization purposes, we only have to clip $V(A, f, t, i)$ so that its minimum is never lower than $z_{\min}$. Evidently, if

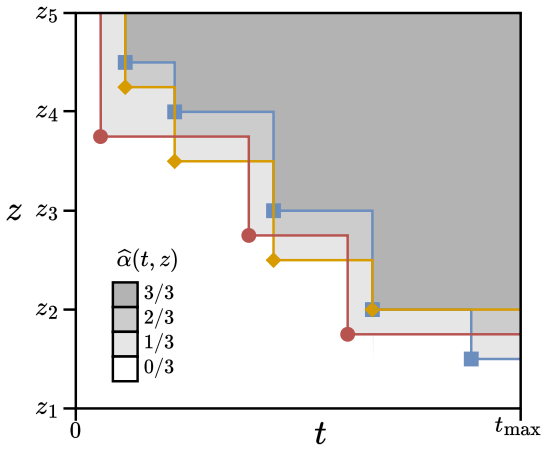

Figure 1. Visualization of the EAF corresponding to three runs (red circles, orange diamonds, and blue squares) of an hypothetical single-objective optimizer. For each run, we record as a point, the runtime t (x-axis) at which a new best-so-far objective function value z (y-axis) was found (assuming minimization). The value of the EAF $\widehat{\alpha}$ at point (t, z) is given by the shade of gray specified in the legend. We mark 5 quality targets in the y-axis $\{z_1, z_2, \ldots, z_5\}$ for computing the target-based ECDF in Figure 2.

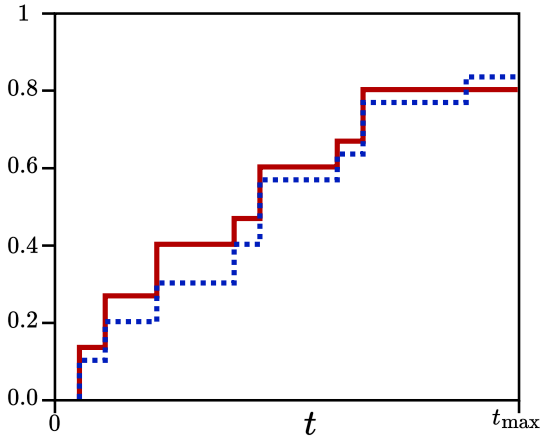

Figure 2. Target-based ECDF (solid red line) and EAF-based ECDF (blue dashed line) corresponding to the runs shown in Figure 1. The target-based ECDF uses the quality targets $\{z_1, z_2, \ldots, z_5\}$ defined in Figure 1. For the EAF-based ECDF, we only need $z_{\min} = z_1$ and $z_{\max} = z_5$.

$V(A, f, t, i)$ is never lower than $z_{\min}$—e.g., because $z_{\min}$ is the optimal value of f or better—then $z_{\min}$ has no effect in the AUC values, except for the normalization constant. With this in mind, the presence of a finite $z_{\min}$ is not necessary for calculating the AUC of the EAF (8), thus we use $-\infty$ for generality.

Another measure of anytime performance proposed in the literature [18] is the area over the convergence curve (AOCC), $S_i = \{(t, V(A, f, t, i)) \mid 1 \leq t \leq t_{\max}\}$, which can be calculated by summing up the areas of the rectangles formed by the points $(t, V(A, f, t, i))$ and $(t + 1, z_{\max})$ as follows:

$$\mathrm{AOCC}(S_i) = \sum_{t=1}^{t_{\max}-1} \max\{0, z_{\max} - V(A, f, t, i)\} \ . \tag{9}$$

We can see that Eq. (8) is equivalent to the mean AOCC

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

value of the multiple runs if we expand the former as follows:

$$\begin{aligned} &\int_1^{t_{\max}} \mathrm{d}t \int_{-\infty}^{z_{\max}} \widehat{\alpha}(t,z)\,\mathrm{d}z \\ &= \int_1^{t_{\max}} \mathrm{d}t \int_{-\infty}^{z_{\max}} \frac{1}{r}\sum_{i=1}^{r} \mathbb{1}\left(V(A,f,t,i) \le z\right)\mathrm{d}z \\ &= \frac{1}{r}\sum_{i=1}^{r} \int_1^{t_{\max}} \mathrm{d}t \int_{-\infty}^{z_{\max}} \mathbb{1}\left(V(A,f,t,i) \le z\right)\mathrm{d}z \ . \end{aligned} \tag{10}$$

For a fixed $V(A,f,t,i)$, the indicator function $\mathbb{1}\left(V(A,f,t,i) \le z\right)$ is 1 from $z = V(A,f,t,i)$ up to ∞, and 0 otherwise. Thus, its integral over z bounded by $(-\infty, z_{\max}]$ is $z_{\max} - V(A,f,t,i)$ when $z_{\max} \ge V(A,f,t,i)$ and zero otherwise. Therefore,

$$\begin{aligned} &\frac{1}{r}\sum_{i=1}^{r} \int_1^{t_{\max}} \mathrm{d}t \int_{-\infty}^{z_{\max}} \mathbb{1}\left(V(A,f,t,i) \le z\right)\mathrm{d}z \\ &= \frac{1}{r}\sum_{i=1}^{r} \sum_{t=1}^{t_{\max}-1} \max\{0, z_{\max} - V(A,f,t,i)\} \\ &= \frac{1}{r}\sum_{i=1}^{r} \mathrm{AOCC}(S_i) \ . \end{aligned} \tag{11}$$

In other words, not only the AUC of the EAF-based ECDF is equivalent to the AUC of the EAF (multiplied by a normalization constant), but also both are equivalent to the mean AOCC of the runs used to compute the EAF. A more general proof of the relationship between the expected value of a generalized hypervolume indicator (i.e., where the AOCC is multiplied by a weight function) and the first-order attainment function (i.e., the EAF when the number of runs goes to infinity) is provided by [19]. The simpler proof in Eq. (10)–(11) does not consider the weighted variant of the hypervolume indicator, but it applies to a finite number of empirical samples.

There are situations in which we are only interested in the AUC value, for example, for benchmarking purposes or when using the AUC as the metric that guides the tuning of parameters. In those situations, instead of storing the data for all runs to compute the target-based ECDF or the EAF and their AUC value, it is much more convenient to compute the AOCC of each run as soon as it finishes and only store the AOCC values. Moreover, recomputing the mean AOCC after performing additional runs is trivial, whereas updating the EAF or the ECDF is less so. The use of the mean AOCC to guide the tuning of parameters was already demonstrated in the literature [18], but the connection with the AUC of the target-based ECDF was not known at the time.

III. Computation of the EAF

A. Efficiency of computation

In the formulation of the EAF discussed here for benchmarking single-objective black-box optimization algorithms (2), points are pairs (t,z) of runtime and objective function value. The EAF of an algorithm at this point is the probability that the algorithm has found, within the first t function evaluations, a solution that is at least as good as z. In this two-dimensional context, the computation of the EAF requires $\Theta(n \log n + rn)$ time/steps [20], where r is the number of runs and $n \le rt_{\max}$ is the total number of points, i.e., objective value improvements recorded across all runs.

The number of points required to fully represent the EAF is $\Theta(rn)$ [20]. Reducing n by merging points representing very quick or very small improvements in succession is an easy way to speed-up the computation of the EAF and reduce its size. Therefore, we argue that computing or storing the EAF is feasible for a reasonable number of runs and points.

The computation of the EAF-based ECDF (4) may seem more complicated than the target-based ECDF (1). However, given that $\widehat{\alpha}(t,z)$ for a fixed t is a step function monotonically increasing with respect to z, the integral $F_{\widehat{\alpha}}(t)$ can be seen as the Lebesgue integral over z for each t, which is easy to calculate as a sum of rectangles. Moreover, we need $z_{\min}$ to show the exact equivalence between the target-based and EAF-based ECDFs, however, the computation of the EAF does not require defining $z_{\min}$.

B. Integration into IOHanalyzer

For practical usage of the EAF with various types of benchmarking data, we have integrated several ways of using EAF-based analysis into IOHanalyzer [11], which is a part of the IOHprofiler benchmarking platform. For this purpose, the computations of the EAF are calculated by version 2.4.1 of the `eaf` R package [7].[2] Through this integration, several ways of computing, visualizing and comparing EAF-based statistics, are available directly in the web-interface of IOHanalyzer (and the corresponding R package). Benchmarking data from platforms such as COCO [1], Nevergrad [21], and IOHexperimenter [22] can also be analyzed using these methods.

To speed up computations when creating visualizations, we make use of a subsampling approach when calculating the EAF, ECDF, and AUC values. This subsampling is done on the runtime-axis, where we select a set of 50 log-spaced runtimes at which we calculate the fraction of targets hit for the target-based ECDF or at which we take the function values to pass to the EAF calculation. Given this subsampling, the computation of the EAF and the EAF-based ECDF took just 432 seconds in a single CPU for all datasets considered in the next section (1295 datasets in total), which is a reasonable time for the analysis of an experiment of this scale.

IV. Use in Practice

We illustrate the use of the EAF for benchmarking single-objective black-box optimizers by analyzing data from two commonly used optimization algorithms: BFGS [23–26] (from the `scipy` package [27]) and CMA-ES [28] (from the `pyCMA` package [29]). Both these algorithms are run on the 24 functions from the BBOB suite. All functions allow specifying the dimension of the decision space D. Each algorithm is run 15 times on each function with different random seeds, each time using a different instance (i.e., rotation and translation) of the function. Each individual run is stopped after $t_{\max} = 10^4 \cdot D$ function evaluations.

[2] https://mlopez-ibanez.github.io/eaf/

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

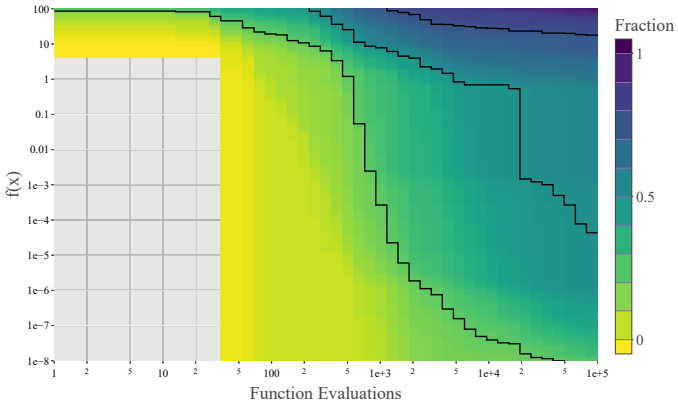

Figure 3. EAF of BFGS over the 24 10-dimensional BBOB functions. The color of a point gives the fraction of runs that reach a given value of $f(x)$ not later than a given number of function evaluations. The black lines delimit the regions with a value (from bottom to top) ≤ 0.25, ≤ 0.5 and ≤ 0.75.

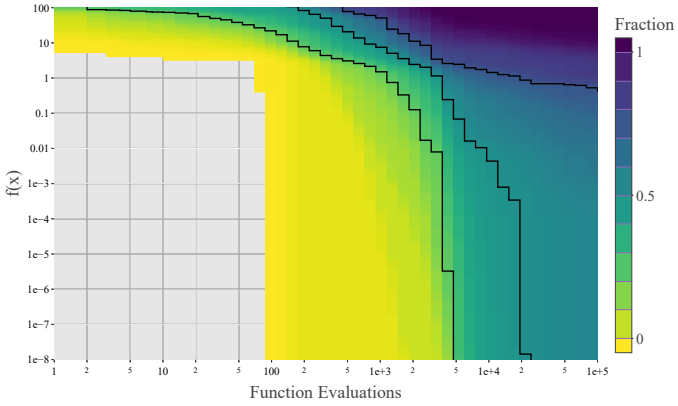

Figure 4. EAF of CMA-ES over the 24 10-dimensional BBOB functions. See the caption of Figure 3 for more details.

For computing the EAF-based ECDF (4) and its AUC on the BBOB functions, we follow the convention set by COCO [1] and define $z_{\min} = 10^{-8}$ and $z_{\max} = 10^{2}$. To match the logarithmic scale of the targets used by COCO, we use a logarithmic scaling of the interval $[z_{\min}, z_{\max}]$ (effectively taking the log-precision as our function values throughout any EAF-based ECDF calculations). Figures 3 and 4 show the resulting EAF of the BFGS and CMA-ES algorithms, respectively, aggregated over all 24 BBOB functions in dimension 10. Darker colors indicate a higher probability of attaining a certain objective function value within the number of function evaluations given by the x-axis. From these figures, we see a clear trend in the optimization behavior, where BFGS converges faster at the beginning of the search (darker colors on the lower left of the colored region), but quickly stagnates (small contrasts going from left to right). We highlight specific levels of the EAFs $(0.25, 0.5, 0.75)$ as black lines, which suggest that the results of CMA-ES are more robust because the 0.5 and 0.75 lines of CMA-ES almost completely dominate the corresponding ones of BFGS.

In addition to the individual EAF plots, we can also perform a comparative analysis by plotting the differences between two EAFs. This is illustrated in Figure 5, where the differences between CMA-ES and BFGS on a single BBOB function are

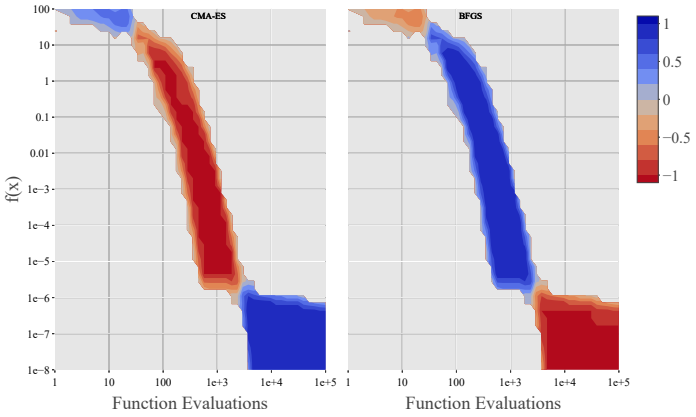

Figure 5. Differences in the EAFs of CMA-ES and BFGS on BBOB function 14, in 10 dimensions. Blue colors indicate regions where the algorithm in the plot title outperforms the other algorithm.

shown. In this example, BFGS outperforms CMA-ES between 50 and 1e+3.5 (approx. 3162) evaluations, whereas CMA-ES wins after that, managing to converge to $f(x) = 10^{-8}$ more reliably. This analysis can additionally be extended to highlight the difference between an algorithm and a portfolio, by taking the upper envelope of the portfolio's EAF. This portfolio-based comparison is illustrated in Figure 6, where a set of four algorithms are compared on benchmark function F11. The selected portfolio consists of CMA-ES, BFGS, Differential Evolution [30] (DE) and Cobyla [31], and their performance data is taken from the COCO data repository. From this figure, we see that in three stages of the search, Cobyla, BFGS, and CMA-ES, respectively, outperform every other considered algorithm, whereas there is no pair (t, z) with $t > 10$ in which the EAF of DE dominates the maximum of the EAF values of the other three algorithms.

A. *Sensitivity of the ECDF wrt Target Values*

In this subsection, we study the sensitivity of the target-based ECDF to the number of pre-defined target values. That is, we empirically evaluate the convergence speed of the limit in (4). Following COCO, we use a logarithmic scale by selecting equally spaced targets that are powers of 10 between 10^{-8} and 10^{2}. In addition to the default of COCO with 51 targets, we also consider 5, 10, and 25 targets. For each of these values and each algorithm under comparison, we compute the target-based ECDF (1) and its difference in value to the EAF-based ECDF (4). To increase the amount of algorithms being compared, we now make use of the data from the BBOB repository that matches the setup from the previous section: 1 run on each of 15 different instances. For the 10-dimensional functions, this leads to a set of 211 algorithms.[3]

In Figure 7, we illustrate the impact of changing the number of targets for the BFGS and CMA-ES. As can be seen from this figure, the ECDF based on a limited number of targets consistently overestimates the EAF-based ECDF.

[3]For readers familiar with the COCO platform, we note that we do not apply the bootstrapping approach suggested in [1] to simulate restarts. Instead, we compute the EAF from the available trajectories.

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

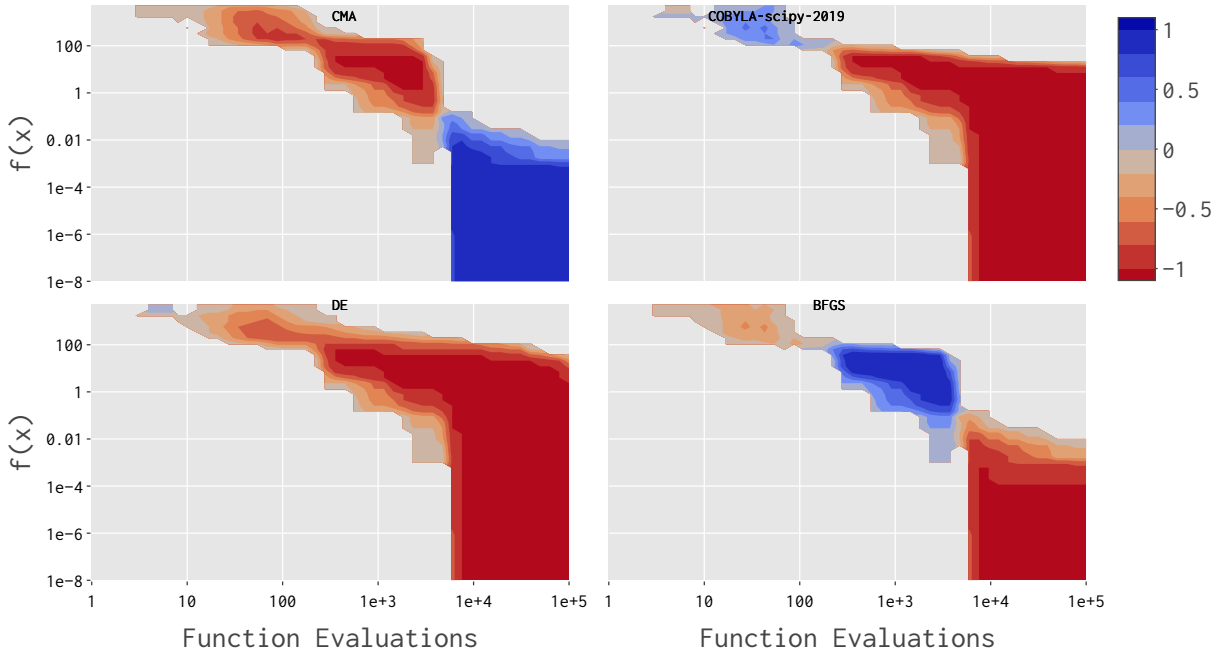

Figure 6. Differences in the EAFs of a portfolio of 4 algorithms on BBOB function 11, in 10 dimensions. Blue colors indicate regions where the algorithm in the plot title outperforms the other algorithms.

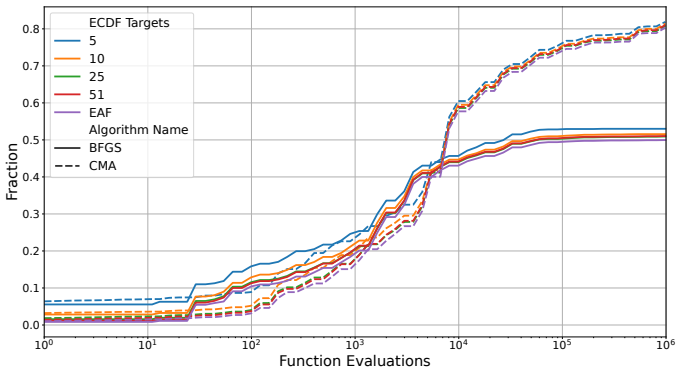

Figure 7. ECDF as calculated using the EAF (4) and the target-based approach (1), with different numbers of targets, for two different algorithms, BFGS and CMA-ES. Data is aggregated over all 24 10-dimensional BBOB functions.

Figure 8 shows the mean and standard deviation of these differences over all 211 algorithms and for the different numbers of targets. As expected, the differences decrease rapidly with increasing number of targets. For the data in this experiment and with 51 targets, differences are larger for higher runtime. In addition, we clearly see that the overestimation of the target-based ECDF was not an artifact of the selected algorithms in Figure 7, but a consistent phenomenon. The differences between the target-based and EAF-based ECDF values are a combination of two distinct effects. The first and more important effect is inherent in the approximation induced by the choice of targets: any objective value attained between two successive targets will result in the same value of the target-based ECDF, whereas any objective value improvement increases the value of the EAF-based ECDF. For a single run, the size of the differences caused by this effect is at most $1/|Z|$, where $|Z|$ is the number of targets.

A secondary effect is due to the difference between $z_{\max}$ and the first target. If $z_{\max}$ equals the first target, then the ECDF value when attaining $z_{\max}$ differs between the two approaches. For simplicity, let us consider a single run that exactly attains $z_{\max}$ at some t. In the target-based ECDF, this run has a value of $1/|Z|$, while this same run has value 0 in the EAF-based ECDF, at time t. This initial difference causes the target-based ECDF to consistently overestimate the EAF-based ECDF, as seen in Figure 8. We could remove this initial difference by increasing $z_{\max}$ relative to the first target by a factor of $(z_{\max} - z_{\min})/(|Z| - 1)$ before computing the EAF. However, this adjustment would actually cause under-estimation due to the first effect mentioned above. Moreover, a run that attains an objective value strictly between the adjusted $z_{\max}$ and the first target will result in a positive value of the EAF-based ECDF, but a value of 0 of the target-based ECDF, thus also leading to under-estimation. In other words, if $z_{\max}$ is larger than the first target, then the target-based ECDF may initially under-estimate the EAF-based ECDF. In summary, due to the unavoidable first effect, adjusting $z_{\max}$ to remove the secondary effect does not eliminate the differences, it just shifts them from consistent over-estimation to primarily under-estimation. In addition, applying this adjustment would require computing an EAF-based ECDF for each set of targets, which greatly complicates the comparison between target-based and EAF-based ECDFs. Thus, we have decided to define $z_{\max}$ as the first target in all our experiments.

In Figure 9, we can see that the differences observed earlier also translate into differences in AUC. We can again observe that the differences between the AUCs of the target-based ECDF and EAF-based ECDF decrease quickly with the

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

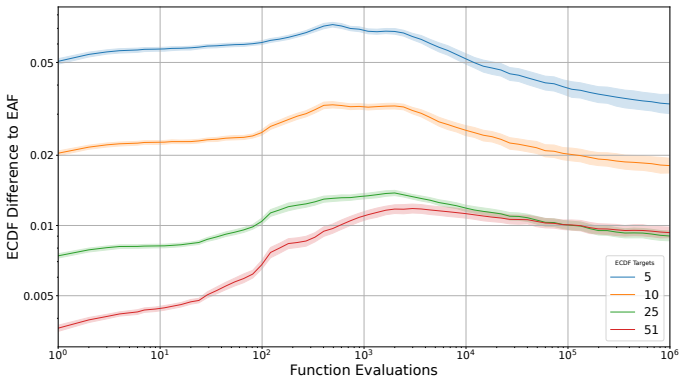

Figure 8. Mean and standard deviation of differences in target-based ECDFs with different numbers of targets relative to the EAF-based ECDF. Each individual ECDF value is an aggregation of the 24 10-dimensional BBOB problems for one algorithm. These are aggregated over all 211 processed algorithms from COCO's repository. Both axes are in logarithmic scale.

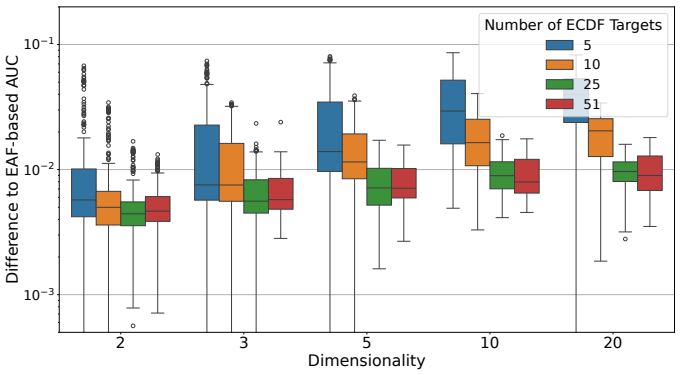

Figure 9. Distribution of differences between the AUC of the EAF-based ECDF and the AUC of the target-based ECDF, for different number of targets. Each data point is the mean AUC difference for a single algorithm averaged over 15 runs on the 24 BBOB problems of the dimension given in the x-axis and the box plots summarize the data distribution of all 211 processed algorithms from the COCO repository

number of targets. At the same time, the effect of the number of targets is stronger for problems in higher dimensions. We can also observe that going from 25 to 51 targets does not help much in reducing the differences below 1%. The relatively large variances of the boxplots indicate that the low variance observed in Figure 8 does not translate into consistent differences across algorithms. In other words, the estimation error of the target-based ECDF is larger for some algorithms.

B. Changes in ranking when using ECDF vs EAF

Finally, we may question whether these small differences matter at all when ranking algorithms according to the AUC values. To answer this question, we rank all 211 algorithms according to different AUC values, i.e., of the EAF-based ECDF as well as the target-based ECDFs with 5, 10, 25 and 51 targets. We highlight the rank-differences between the 51 target and EAF setting in Figure 10. We additionally calculate the rank differences between two rankings by summing the absolute difference between ranks and dividing by two, because each rank change contributes twice when taking the absolute value. In the case of the 2-dimensional BBOB problems, the rank differences between the target-based ECDFs and the EAF-based ECDFs are: 298 (5 targets), 174 (10), 118 (25) and 101 (51). In the case of the 10-dimensional BBOB problems, the rank differences between the target-based ECDFs and the EAF-based ECDFs are: 163 (5 targets), 81 (10), 49 (25), and 65 (51). These numbers show that, although the differences in ranks quickly go down with increasing number of targets, a high number of rank differences remain even with 51 targets. Moreover, the last result shows that increasing the number of targets does not always guarantee a more precise AUC value, since the precision depends on the particular targets not only on their number.

V. EAF-based Statistics

Several summary statistics can be computed starting from the EAF. In particular, the level sets of the EAF are quantile-like functions that can be represented as convergence curves. For example, the boundary region in the time versus objective value space where the EAF has a value less than or equal to 0.5 is a synthetic convergence curve that represents the median convergence curve (50% percentile), such that any combination of time and objective function value on or above this curve will be attained in 50% of the runs of an algorithm, but not necessarily all of them simultaneously in any single run. Similarly, the 100% percentile is the boundary that is attained by all runs of an algorithm and represents the worst-case performance, whereas the lowest available percentile represents the best-case performance. Examples of such percentiles are shown in Figures 3 and 4.

Similarly, considering the level sets of the EAF as quantile-like functions, Binois et al. [32] suggest the Vorob'ev expectation as a useful definition of the expected value of a random set [33, chap. 2]. In a nutshell, the Vorob'ev expectation is the quantile of the EAF with the closest AOCC value to the expected value of the AOCC of a single run of the algorithm. In other words, the Vorob'ev expectation is the synthetic convergence curve that corresponds to a particular quantile of the EAF whose AOCC value matches the mean AOCC of the actual runs of an algorithm. From a benchmarking perspective, the Vorob'ev expectation is the "mean" convergence curve and it can be used to summarize a large number of runs for visualization and comparison purposes. In addition, dispersion statistics may be computed as the probability of a single run deviating from this mean convergence curve [32].

In addition to the above, the second-order EAF [34] measures the probability of attaining an objective function value z not later than t if the algorithm has already obtained an objective value z' not later than t'. From a benchmarking perspective, such information is useful to diagnose convergence issues, particularly, for algorithms that switch behavior after a number of steps or evaluations.

Statistical measures and visualization techniques [35] that can be applied to the target-based ECDF are also applicable to the EAF-based ECDF. Some techniques may be directly applicable to the EAF. For example, it is possible to compute bootstrapped confidence bands [35] around the EAF that estimate the uncertainty. However, applying other techniques

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

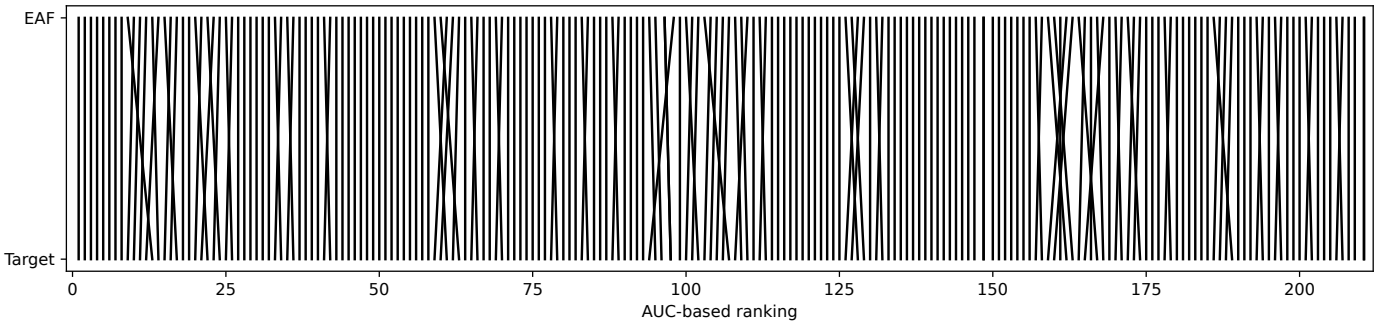

Figure 10. Ranking of 211 algorithms based on AUC (aggregated over the 24 10-dimensional BBOB problems) when using the EAF and the target-based ECDF with 51 targets. Lines crossing denote discrepancies in ranking. The sum of absolute rank differences (divided by two because each rank change contributes twice when taking the absolute value) is 65.

directly to the EAF may require additional research effort to handle the challenge of samples being sets of (runtime, quality) points rather than scalar numerical values.

VI. Conclusions

In this paper, we have argued that the EAF, and statistics derived from it, have significant benefits for the analysis and benchmarking of single-objective optimizers. In particular, we have shown how the widely-used target-based ECDF is just an approximation to the EAF-based ECDF and that the former eventually converges to the latter with increasing number of well-spread target values. We obtain a similar conclusion when considering the AUC of the target-based ECDF, which has been proposed as a measure of anytime performance. In this case, not only we argue that the AUC of the target-based ECDF is just an approximation of the AUC of the EAF-based ECDF, which is actually equal to the AUC of the EAF itself; but also we show that the latter is equivalent to the mean area over (or under in the case of maximization) the convergence curves (AOCC) of the individual runs. Therefore, we conclude that recording the mean AOCC value is sufficient for measuring anytime performance in situations where we are only interested in the AUC values, e.g., for benchmarking purposes or when guiding the tuning of parameters [18].

In addition to the theoretical benefits of replacing the target-based ECDF with the EAF, we also demonstrate its practical use by means of integrating the eaf package into the IOHanalyzer platform. In particular, we argue that the visualization of EAF differences provides clear information about differences in anytime performance between optimizers. We also show that the theoretical differences between the target-based ECDF and the EAF-based ECDF are measurable in real benchmarking data. Moreover, although the differences in value are small for the choice of targets recommended by the COCO platform, they still produce rank differences when comparing optimizers. Finally, the application of the EAF for analyzing single-objective optimizers opens the door to various EAF-based statistics that have not been explored in the benchmarking literature.

We have focused here on single-objective black-box algorithms for continuous optimization problems because that is the context where the target-based ECDF has gained popularity. However, our conclusions apply to benchmarking anytime algorithms in general, including gradient-based optimizers and optimizers for combinatorial optimization problems. Our conclusions also naturally extend to benchmarking multi-objective algorithms according to their anytime performance [36, 37], where quality targets are given as vectors of objective values. However, the visualization [38] of the EAF in three dimensions (i.e., 2 objectives plus runtime) and its computation in more than three dimensions [20] are still challenging.

References

[1] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tušar, and D. Brockhoff, "COCO: A platform for comparing continuous optimizers in a black-box setting," *Optimization Methods and Software*, vol. 36, no. 1, pp. 1–31, 2020.

[2] N. Hansen, A. Auger, D. Brockhoff, and T. Tušar, "Anytime performance assessment in blackbox optimization benchmarking," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 6, pp. 1293–1305, Dec. 2022.

[3] T. A. Feo, M. G. C. Resende, and S. H. Smith, "A greedy randomized adaptive search procedure for maximum independent set," *Operations Research*, vol. 42, pp. 860–878, Oct. 1994.

[4] H. H. Hoos and T. Stützle, "Evaluating Las Vegas algorithms — pitfalls and remedies," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, G. F. Cooper and S. Moral, Eds. Morgan Kaufmann Publishers, San Francisco, CA, 1998, pp. 238–245.

[5] V. Grunert da Fonseca, C. M. Fonseca, and A. O. Hall, "Inferential performance assessment of stochastic optimisers and the attainment function," in *Evolutionary Multi-criterion Optimization, EMO 2001*, ser. Lecture Notes in Computer Science, E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, Eds. Berlin/ Heidelberg: Springer, 2001, vol. 1993, pp. 213–225.

[6] V. Grunert da Fonseca and C. M. Fonseca, "The attainment-function approach to stochastic multiobjective optimizer assessment and comparison," in *Experimental Methods for the Analysis of Optimization Algorithms*, T. Bartz-Beielstein, M. Chiarandini, L. Paquete, and M. Preuss, Eds. Berlin, Germany: Springer, 2010, pp. 103–130.

[7] M. López-Ibáñez, L. Paquete, and T. Stützle, "Exploratory analysis of stochastic local search algorithms in biobjective optimization," in *Experimental Methods for the Analysis of Optimization Algorithms*, T. Bartz-Beielstein, M. Chiarandini, L. Paquete, and M. Preuss, Eds. Berlin, Germany: Springer, 2010, pp. 209–222.

[8] J. Dréo, "Adaptation de la métaheuristique des colonies de fourmis pour l'optimisation difficile en variables continues: Application en génie biologique et médical," Ph.D. dissertation, Université Paris XII Val de Marne, Dec. 2003. [Online]. Available: https://tel.archives-ouvertes.fr/tel-00093143

[9] M. Chiarandini, "Stochastic local search methods for highly constrained combinatorial optimisation problems," Ph.D. dissertation, FB Informatik, TU Darmstadt, Germany, 2005.

[10] J. Dréo, "Using performance fronts for parameter setting of stochastic metaheuristics," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO Companion 2009*, F. Rothlauf, Ed. New York, NY: ACM Press, 2009, pp. 2197–2200.

[11] H. Wang, D. Vermetten, F. Ye, C. Doerr, and T. Bäck, "IOHanalyzer: Detailed performance analyses for iterative optimization heuristics," *ACM Transactions on Evolutionary Learning and Optimization*, vol. 2, no. 1, pp. 3:1–3:29, 2022.

[12] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," INRIA, France, Tech. Rep. RR-6829, 2009, updated February 2010.

[13] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík, "Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO Companion 2010*, M. Pelikan and J. Branke, Eds. New York, NY: ACM Press, 2010, pp. 1689–1696.

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3462758

[14] V. Grunert da Fonseca and C. M. Fonseca, "A link between the multivariate cumulative distribution function and the hitting function for random closed sets," *Statistics & Probability Letters*, vol. 57, no. 2, pp. 179–182, 2002.

[15] F. Ye, C. Doerr, H. Wang, and T. Bäck, "Automated configuration of genetic algorithms by tuning for anytime performance," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 6, pp. 1526–1538, 2022.

[16] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms - A comparative case study," in *Parallel Problem Solving from Nature – PPSN V*, ser. Lecture Notes in Computer Science, A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds. Heidelberg, Germany: Springer, 1998, vol. 1498, pp. 292–301.

[17] N. Beume, C. M. Fonseca, M. López-Ibáñez, L. Paquete, and J. Vahrenhold, "On the complexity of computing the hypervolume indicator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1075–1082, 2009.

[18] M. López-Ibáñez and T. Stützle, "Automatically improving the anytime behaviour of optimisation algorithms," *European Journal of Operational Research*, vol. 235, no. 3, pp. 569–582, 2014.

[19] V. Grunert da Fonseca and C. M. Fonseca, "The relationship between the covered fraction, completeness and hypervolume indicators," in *Artificial Evolution: 10th International Conference, Evolution Artificielle, EA, 2011*, ser. Lecture Notes in Computer Science, J.-K. Hao, P. Legrand, P. Collet, N. Monmarché, E. Lutton, and M. Schoenauer, Eds. Heidelberg, Germany: Springer, 2012, vol. 7401.

[20] C. M. Fonseca, A. P. Guerreiro, M. López-Ibáñez, and L. Paquete, "On the computation of the empirical attainment function," in *Evolutionary Multi-criterion Optimization, EMO 2011*, ser. Lecture Notes in Computer Science, R. H. C. Takahashi, K. Deb, E. F. Wanner, and S. Greco, Eds. Berlin/Heidelberg: Springer, 2011, vol. 6576, pp. 106–120.

[21] J. Rapin and O. Teytaud, "Nevergrad: A gradient-free optimization platform," https://github.com/FacebookResearch/Nevergrad, 2018.

[22] J. de Nobel, F. Ye, D. Vermetten, H. Wang, C. Doerr, and T. Bäck, "IOHexperimenter: Benchmarking platform for iterative optimization heuristics," *Evolutionary Computation*, pp. 1–6, 2024.

[23] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms: 2. the new algorithm," *IMA Journal of Applied Mathematics*, vol. 6, no. 3, pp. 222–231, Sep. 1970.

[24] R. Fletcher, "A new approach to variable metric algorithms," *The Computer Journal*, vol. 13, no. 3, pp. 317–322, Sep. 1970.

[25] D. Goldfarb, "A family of variable-metric methods derived by variational means," *Mathematics of Computation*, vol. 24, no. 109, pp. 23–26, 1970.

[26] D. F. Shanno, "Conditioning of quasi-newton methods for function minimization," *Mathematics of Computation*, vol. 24, no. 111, pp. 647–656, 1970.

[27] P. Virtanen *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[28] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC'96)*, T. Bäck, T. Fukuda, and Z. Michalewicz, Eds. Piscataway, NJ: IEEE Press, 1996, pp. 312–317.

[29] N. Hansen, Y. Akimoto, and P. Baudis, "CMA-ES/pycma on Github," Zenodo, Feb. 2019.

[30] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[31] M. J. D. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in Optimization and Numerical Analysis*. Springer, 1994, pp. 51–67.

[32] M. Binois, D. Ginsbourger, and O. Roustant, "Quantifying uncertainty on Pareto fronts with Gaussian process conditional simulations," *European Journal of Operational Research*, vol. 243, no. 2, pp. 386–394, 2015.

[33] I. Molchanov, *Theory of Random Sets*. Springer, 2005.

[34] C. M. Fonseca, V. Grunert da Fonseca, and L. Paquete, "Exploring the performance of stochastic multiobjective optimisers with the second-order attainment function," in *Evolutionary Multi-criterion Optimization, EMO 2005*, ser. Lecture Notes in Computer Science, C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Eds. Heidelberg, Germany: Springer, 2005, vol. 3410, pp. 250–264.

[35] E. Arza, J. Ceberio, E. Irurozki, and A. Pérez, "Comparing two samples through stochastic dominance: A graphical approach," *Journal of Computational and Graphical Statistics*, pp. 1–38, Jun. 2022.

[36] A. Radulescu, M. López-Ibáñez, and T. Stützle, "Automatically improving the anytime behaviour of multiobjective evolutionary algorithms," in *Evolutionary Multi-criterion Optimization, EMO 2013*, ser. Lecture Notes in Computer Science, R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, and J. Shaw, Eds. Heidelberg, Germany: Springer, 2013, vol. 7811, pp. 825–840.

[37] D. Brockhoff, T. Tušar, D. Tušar, T. Wagner, N. Hansen, and A. Auger, "Biobjective performance assessment with the COCO platform," *Arxiv preprint arXiv:1605.01746*, 2016.

[38] T. Tušar and B. Filipič, "Visualizing 4D approximation sets of multiobjective optimizers with prosections," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2011*, N. Krasnogor and P. L. Lanzi, Eds. New York, NY: ACM Press, 2011, pp. 737–744.

Manuel López-Ibáñez received his Master in Computer Science from the University of Granada, Spain in 2004 and his Ph.D. degree from Edinburgh Napier University, Edinburgh, U.K., in 2009. He is Full Professor at the Alliance Manchester Business School, University of Manchester, U.K. He is a member of the IEEE Task Force on Automated Algorithm Design, Configuration and Selection, an elected member of the ACM SIGEVO Executive Board and the Business Committee of the Genetic and Evolutionary Computation Conference (GECCO), Editor-in-Chief of ACM Transactions on Evolutionary Learning and Optimization, Associate Editor of the Evolutionary Computation journal and Editorial Board Member of the Artificial Intelligence Journal. Prof. López-Ibáñez has published more than 100 papers in international peer-reviewed journals and conferences on topics that include stochastic local search, black-box optimization, empirical reproducibility, multi-objective and interactive optimization algorithms for continuous and combinatorial problems, and the automatic configuration and design of optimization algorithms.

Diederick Vermetten is a PhD candidate at the Natural Computing group at LIACS, Leiden University. He is part of the core development team of the IOHprofiler project. His research interests include benchmarking of optimization heuristics, dynamic algorithm selection and configuration as well as hyperparameter optimization.

Johann Dreo is a senior research engineer at Institut Pasteur, working on decision support systems for helping bioinformatics against cancer. He received masters degrees in ecology in 1999, from Rennes 1 University, in applied mathematics in 2000, from Sorbonne University, and a Ph.D. in biomedical engineering from Paris-Sud University in 2003. He worked for several years as a researcher in Thales' industrial research center, on automated optimization solvers design. He is a member of the IEEE task force on Automated Algorithm Design, Configuration & Selection, and the maintainer of the ParadisEO framework since 2010. His main interests are in hyperparameter optimization and explainable machine learning for computational biomedicine.

Carola Doerr (formerly Winzen) received the Diploma degree in mathematics from Kiel University, Kiel, Germany, in 2007, the Dr.-Ing. degree from the Max Planck Institute for Informatics and Saarland University, Saarbrücken, Germany, in 2011, and the Habilitation degree (HDR) from Sorbonne Université, Paris, France, in 2020. She is a CNRS Research Director of the LIP6 Computer Science Department, Sorbonne Université. From 2007 until 2012, she was a Business Consultant with McKinsey & Company (on educational leave from 2010 onward). Her main research activities are in the analysis of black-box optimization algorithms. Dr. Doerr is an Associate Editor of IEEE Transactions on Evolutionary Computation, ACM Transactions on Evolutionary Learning and Optimization, and Evolutionary Computation. She is a founding and a coordinating Member of the Benchmarking Network.

Authorized licensed use limited to: Universiteit Leiden. Downloaded on January 30,2025 at 12:36:32 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.