Comparison-based adaptive strategy selection with bandits in differential evolution
— **Source link** ⧉

Álvaro Fialho, Raymond Ros, Marc Schoenauer, Michèle Sebag

Institutions: Microsoft, French Institute for Research in Computer Science and Automation

Topics: Selection (genetic algorithm), Differential evolution, Test suite and Mutation (genetic algorithm)

Related papers:

- Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization

- JADE: Adaptive Differential Evolution With Optional External Archive

- Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems

- Analyzing bandit-based adaptive operator selection mechanisms

- Adaptive strategy selection in differential evolution

Comparison-based Adaptive Strategy Selection with Bandits in Differential Evolution

Álvaro Fialho, Raymond Ros, Marc Schoenauer, Michèle Sebag

HAL Id: inria-00493005
https://hal.inria.fr/inria-00493005v2

Submitted on 12 Sep 2010

Comparison-based Adaptive Strategy Selection with Bandits in Differential Evolution

Álvaro Fialho[1], Raymond Ros[2], Marc Schoenauer[1,2], and Michèle Sebag[1,2]

[1] Microsoft Research - INRIA Joint Centre, Orsay, France
[2] TAO team, INRIA Saclay - Île-de-France & LRI (UMR CNRS 8623), Orsay, France
FirstName.LastName@inria.fr

Abstract. Differential Evolution is a popular powerful optimization algorithm for continuous problems. Part of its efficiency comes from the availability of several mutation strategies that can (and must) be chosen in a problem-dependent way. However, such flexibility also makes DE difficult to be automatically used in a new context. *F-AUC-Bandit* is a comparison-based Adaptive Operator Selection method that has been proposed in the GA framework. It is used here for the on-line control of DE mutation strategy, thus preserving DE invariance w.r.t. monotonous transformations of the objective function. The approach is comparatively assessed on the BBOB test suite, demonstrating significant improvement on baseline and other *Adaptive Strategy Selection* approaches, while presenting a very low sensitivity to hyper-parameter setting.

1 Introduction

Differential Evolution (DE) uses the weighted difference between two or more parent solutions to generate offspring [19]. DE has been successfully applied to many real-world applications[17] thanks to its simplicity and high flexibility. This flexibility is mostly provided by the number of different mutation strategies that can be used for the offspring generation. However, this flexibility is also a limitation to its wide dissemination, as the user needs to choose the mutation strategy for every new problem — and the efficiency of the algorithm is highly sensitive to this choice.

Such choice is usually done by following the user's intuition, or by using an off-line tuning procedure aimed at identifying the best strategy for the problem at hand. Besides being computationally expensive, off-line tuning however generally delivers sub-optimal performances, as the appropriate strategy depends on the stage of the optimization process: exploration-like strategies should be more frequently used in the early stages while priority should be given to exploitation when approaching the optimum.

For this reason, the present paper focuses on on-line tuning, aimed at selecting the strategy for the next offspring generation on the basis of the current results of the strategies (*i.e.*, the quality of the recent offspring they generated). Such on-line selection, referred to as *Adaptive Strategy Selection* (*AdapSS*), is similar in spirit to *Adaptive Operator Selection* (*AOS*) in the GA framework. A

brief review of some *AdapSS/AOS* techniques is presented in Section 2, focusing on the ones used in the experiments and compared to the proposed approach.

Another important feature of DE (and many other bio-inspired algorithms) is that it is comparison-based, and thus invariant to monotonous transformations of the fitness function. Such invariance property significantly increases the robustness of the approach; in particular it implies that no fitness scaling is ever required when dealing with a new application. Unfortunately, this invariance is generally lost when adding mechanisms like *AOS*. *Fitness-based Area-Under-Curve - Bandit* (*F-AUC-Bandit*) is a recently introduced *AOS* that only uses the ranks of the most recent offspring to assess the strategy credit, combining an *Area Under the ROC Curve* (*AUC*) measure [3] and a Multi-Armed Bandit algorithm [9]. It is here ported to the *AdapSS* context, preserving the invariance properties of DE, while implementing on-line operator choice. For the sake of completeness, *F-AUC-Bandit* is briefly described in Section 3.

The *F-AUC-Bandit AdapSS* approach is experimentally validated on the BBOB-2010 noiseless benchmarking suite [11]. Extensive comparative results are reported in Section 4, considering baseline strategies (uni-strategies and random selection) as well as three adaptive schemes: *Adaptive Pursuit* (*AP*) [20] and *Dynamic Multi-Armed Bandit* (*DMAB*) [4], both being here assessed for the first time in the continuous domain, and *PM-AdapSS-DE*, another technique recently proposed and analyzed in the context of *Adaptive Strategy Selection* within DE [10]. Finally, Section 5 concludes the paper, summarizing the presented results and pointing out possible directions for future work.

2   Adaptive Strategy Selection

Adaptive Strategy Selection performs on-line selection of the mutation strategy for the generation of each new offspring, based on the recent known performance of each of the available strategies. The *AdapSS* paradigm requires two ingredients: the *Credit Assignment* scheme assesses the performance of a given strategy, translating the fitness of the newly generated offspring into a numerical credit; the *Strategy Selection* method rules how to choose a given strategy among all available ones, based on their (continuously updated) credit. This paradigm closely parallels that of *Adaptive Operator Selection* in the Genetic Algorithms community [20, 4]. A brief review of some approaches previously proposed on both communities is presented in this section, without aiming at exhaustivity.

2.1   Credit Assignment

Different approaches for *Credit Assignment* have been proposed, differing mainly on three aspects: (i) how the impact of the strategy application should be measured; (ii) how to assign credit based on these impact assessments; and finally, (iii) to which strategy the credit should be assigned to.

The most common impact measure of a strategy application is the fitness improvement brought by the newly generated offspring, when compared to its

parent [7], to the current median [14] or to the best individual [5] in the population. In [10], a relative fitness improvement is used, taking into account the difference of the fitness of the offspring with that of its parent, and normalizing it by the ratio between its fitness and the best one in the current population.

These rewards are then transformed into a credit to be assigned to the strategy, thus updating its empirical quality estimate. This quality estimate is in turn used by the *Strategy Selection* for the selection of the next strategy. Such credit might be the instantaneous reward, *i.e.*, received after the last application; the average of the rewards received over a few recent applications; or the extreme (or maximum) reward recently received by the strategy [6]. The number of recent applications considered for the latter two is usually a user-defined parameter, referred to as W (size of the sliding window) in the following.

Finally, some authors [5, 14] have proposed to assign credit to the strategies that were used to generate the ancestors of the current individual, by means of a bucket brigade scheme. However, most works, including the present one, only assign the reward to the strategy used to generate the newborn offspring.

2.2 Strategy Selection

The *Strategy Selection* schemes select the next strategy between the available ones based on their known empirical quality, which is updated by the *Credit Assignment* mechanism after each application. The main difference between the proposed methods lies in how they use such empirical estimates to select the strategy to be applied. Two types of schemes are distinguished.

The probability-based methods *Probability Matching* (*PM*) and *Adaptive Pursuit* (*AP*) [20] calculate an application probability for each strategy, and use roulette wheel to select the next strategy. Both methods set a lower bound on the probabilities to preserve some exploration. *PM* sets each probability proportionally to the empirical quality of the strategy; *AP* implements a winner-takes-all scheme, quickly increasing the probability of the current best strategy.

The bandit-based methods *Multi-Armed Bandit* (*MAB*) and *Dynamic MAB* (*DMAB*) [4, 7] deterministically choose the strategy to be applied based on (a variant of) the *Upper Confidence Bound* (*UCB*) algorithm [1]:

$$\text{Select } \arg\max_i \left(\hat{q}_{i,t} + C \sqrt{\frac{2 \log \sum_k n_{k,t}}{n_{i,t}}} \right) \tag{1}$$

where $\hat{q}_{i,t}$ denotes the empirical quality of the i-th option (exploitation term), $n_{i,t}$ the number of times it has been selected so far (the right term corresponding to the exploration term), and C is a user-defined constant (hyper-parameter) controlling the balance between Exploration and Exploitation.

Bandit algorithms have been proven to optimally solve the Exploration vs. Exploitation (EvE) dilemma, albeit in a stationary context. While an *AdapSS* method indeed faces an EvE dilemma (the algorithm should exploit as much as possible the current best mutation strategy, while maintaining some exploration

of the other strategies in case one of them becomes more efficient at a later stage of the optimization), it is a dynamic one: the performances of the strategies vary as the search advances. In order to better cope with these dynamics, the *DMAB* proposes the restart of the *MAB* process whenever a change in the reward distribution is detected by means of the Page-Hinkley statistical test [16].

3 Comparison-based Adaptive Strategy Selection: Fitness-based AUC Bandit

Although alleviating the user from the need of selecting which strategies should be applied to the problem at hand, and doing so in an on-line manner, each of the presented *Strategy Selection* methods involves some hyper-parameters that need to be tuned as well. Furthermore, the common use of fitness improvements as reward makes these hyper-parameters highly problem-dependent, as the range of fitness values varies widely from one problem to another – as well as in the course of an optimization run. A natural way to improve the *Strategy Selection* robustness w.r.t. fitness scaling is to preserve the comparison-based invariance property, that DE shares with many other bio-inspired optimization algorithms (ES, tournament-based GAs, PSO). For this reason, the paper focuses on the *Fitness-based Area-Under-Curve - Bandit* (*F-AUC-Bandit*), a fully comparison-based *AdapSS* recently proposed in the context of GAs [9].

The *Area Under the ROC Curve* (*AUC*) is a criterion originally used in Signal Processing and later adopted in Machine Learning to compare binary classifiers, with the property of being robust with respect to class imbalance [3]. The *Receiving Operator Curve* (*ROC*) depicts how the true positive rate varies with the false positive rate. This indicator is adapted to the comparison-based assessment of strategies/operators as follows. Let us consider the list of the offspring generated in a given time window, and let the list be ranked after the offspring fitness. The *Receiving Operator Curve* associated to a given strategy s is drawn by scanning the ordered list, starting from the origin: a vertical segment is drawn when the current offspring has been generated by s, a horizontal segment is drawn otherwise, and a diagonal one is drawn in case of ties (Fig. 1, reproduced from [9]). The credit associated to strategy s finally is the area under this curve.

While all rank positions have same weight in the above *AUC* calculation (as in Fig. 1 for the sake of clarity), *i.e.*, all horizontal and vertical segments have same length, it makes sense to give more weight to the top ranked offspring. Algorithmically, a decay factor D is used as follows. Let W denote the size of the time window storing the list of recently generated offspring, let r be a rank position, the length of the current fragment (its weight in the *AUC* calculation) is set to $D^r(W - r)$, with $D \in]0, 1]$. The smaller D, the faster the decay, *i.e.*, the more skewed the credit assignment is.

This *Credit Assignment* scheme is coupled with a bandit-based *Strategy Selection* using Equation (1), where $\hat{q}_{i,t}$ is the *AUC* credit and $n_{i,t}$ is the number of times the i-th strategy has been applied in the current window. Note that, in the original *MAB* algorithm for *AdapSS* [4], $\hat{q}$ is defined as the average over all
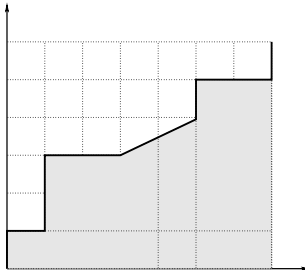
Fig. 1. Computing the AUC reward (reproduced from [9]) associated to strategy 1. Only two operators are considered; the list of the generated offspring is sorted by fitness value; replacing each offspring by the index of the generating strategy gives (1 2 1 1 2 2 [2 2 1] 1 2 2 1), where [2 2 1] stands for three offspring with same fitness values, resulting in the diagonal line between points (3 3) and (5 4).

recent credits assigned to the given strategy. However, the AUC indicator also provides an empirical statistics over the last W offspring, reflecting the up-to-date performance of the given strategy w.r.t. the others.; it is thus directly used as the exploitation term in the MAB formula.

4 Experimental Results

This section reports on the evaluation of the *F-AUC-Bandit* approach coupled with standard DE, comparatively to single-strategies and other *AdapSS* schemes.

4.1 Experimental Setting

The goal of the experiments is to assess the comparative performances of the *AdapSS* schemes when coupled with standard *Differential Evolution* [19], the only difference regarding the strategy selection. DE is governed by three parameters NP, F and CR, respectively denoting the population size, the mutation scaling factor and the crossover rate. It must be emphasized that our goal is not to compete with state-of-the-art continuous optimizers; for this reason, no specific effort was put on tuning the DE parameters depending on the problem at hand. Population size NP is set to $10 \times d$, where d denotes the dimension of the search space; mutation scaling factor F is set to .5; crossover rate CR is set to 1, enforcing DE invariance w.r.t. rotation and stressing the impact of the mutation strategy. Along the same lines, four mutation strategies were chosen, retaining the same as in [10] for the sake of comparative evaluation:

1. "rand/1": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3})$
2. "rand/2": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5})$
3. "rand-to-best/2": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{best} - \mathbf{x}_{r_1}) + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5})$
4. "current-to-rand/1": $\mathbf{v}_i = \mathbf{x}_i + F \cdot (\mathbf{x}_{r_1} - \mathbf{x}_i) + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3})$

where $\mathbf{x}_i$ is the current (or target) individual, $\mathbf{x}_{best}$ is the current best one, and $\mathbf{x}_{r_1}, \mathbf{x}_{r_2}, \mathbf{x}_{r_3}, \mathbf{x}_{r_4}$ and $\mathbf{x}_{r_5}$ are individuals uniformly drawn in the population.

A first range of experiments considers DE using a single mutation strategy, and a uniform selection of the mutation strategy. A second range of experiments considers the adaptive schemes introduced in section 2. The *PM-AdapSS-DE* [10] uses the *Probability Matching* (*PM*) method coupled with the average relative fitness improvement gathered during the current generation. *Adaptive Pursuit* (*AP*) [20] and *Dynamic MAB* (*DMAB*) [4] use as credit assignment the extreme fitness value over a window of size W, as it was found to be the best one in earlier extensive experiments (albeit in a different context) [6,7].

For the sake of a fair empirical comparison, the parameters of the adaptive schemes, referred to as *hyper-parameters*, have been tuned using a racing technique; due to space limitations, the reader is referred to [10,20,4] for a detailed description of the hyper-parameters mentioned in the following.

The comparative validation thus shows the peak performance of each scheme, where the best hyper-parameter configuration has been determined by means of[3] *F-Race* [2]. The hyper-parameter tuning considers the performance of each hyper-parameter over all functions for a given dimension within the benchmark suite; the first elimination round happens after one run over all functions, and it goes on until achieving 10 runs or pruning all configurations but one.

Following this methodology, the hyper-parameters of *F-AUC-Bandit* are varied as follows. The scaling factor C is varied in $\{\{1,5\}.10^{\{-2 \le i \le 1\}}, 100\}$; the window size W in $\{50, 100, 500\}$; the decay factor D is set to .5, giving much more weight to the top-ranked rewards (although in a smoother way than the extreme value based reward mechanism [6]). The best configuration determined by the racing procedure over the benchmark suite, which will be used in all reported results, is: $C = .5$, $D = .5$, $W = 50$.

Along the same lines, the hyper-parameters of *PM-AdapSS-DE* are varied as follows: minimal probability $p_{min} \in \{0, .05, .1, .2\}$, adaptation rate $\alpha \in \{.1, .3, .6, .9\}$. Same values are tried for *AP*, with the additional learning rate β varied in $\{.1, .3, .6, .9\}$. The *DMAB* hyper-parameters are varied as follows: scaling factor $C \in \{\{1,5\}.10^{\{-2 \le i \le 1\}}, 100\}$, and change-detection threshold $\gamma \in \{Range(\mathcal{C}), 1000\}$. The window size W, involved in *AP* and *DMAB* is varied in $\{50, 100, 500\}$. Ultimately, the best *PM-AdapSS-DE* configuration is $p_{min} = 0$ and $\alpha = .6$; the best *AP* configuration is $p_{min} = .2$, $\alpha = .3$, $\beta = .3$, $W = 100$; the best *DMAB* configuration is $C = 100$, $\gamma = .1$, and $W = 50$.

Experiments are conducted using the BBOB-2010 noiseless testbed [12], including 24 single-objective functions from 5 different classes. They are performed following the default guidelines, 15 trials per function [11], with the maximum number of function evaluations being fixed at $10^5 \times d$. The BBOB-2010 experimental set-up uses as performance measurement the *Expected Running Time* (ERT), defined as follows: given a target function value, ERT is the empirical

---

[3] F-Race is an off-line tuning method, running all candidate configurations and stopping them as soon as it is shown to be statistically worse than the current best one at a given confidence level (95% in this case).

expected number of function evaluations for attaining a fitness value below the target, *i.e.*, the ratio of the number of function evaluations for reaching the target value over successful trials, plus the maximum number of evaluations for unsuccessful trials, divided by the number of successful trials. Only the results over the *separable*, *moderate* and *ill-conditioned* function classes are reported here, for none of the considered schemes was able to perform well on the *multi-modal* and *weak-structure* ones with the given budget. Due to space constraints, the presented results are restricted to dimension $d = 20$, referring the reader to [8] for a comprehensive presentation including dimension 5, all function classes, and exhaustive pair-wise statistical comparisons between *F-AUC-Bandit* and the other schemes. The results are summarized in Fig. 2, being complemented by Table 1.

4.2   Comparative results

F-AUC-Bandit is firstly compared with non-adaptive schemes referred to as DE1..DE4, and Uniform-DE, respectively using the single mutation strategy 1..4, and a uniformly selected mutation strategy. DE4 shows unable to solve any of the functions in dimension $d = 20$, and it is thus discarded in the following (although being occasionally used by the adaptive schemes). All other non-adaptive schemes achieve the target value on all trials for the *ill-conditioned* functions, and on 60% of the trials for the *separable* ones, failing on the multi-modal ones. For the *moderate* functions, both *F-AUC-Bandit* and $DE3$ are able to achieve 100% of success, while $DE1$ and $DE2$ respectively get 98% and 75% success.

Compared with $DE1$, *F-AUC-Bandit* shows to be around 3 times faster on the 3 analyzed function classes. $DE2$ is around 20 times slower than *F-AUC-Bandit* on around 65%, 50% and 80% of the trials, respectively, for the *separable*, *moderate* and *ill-conditioned* function classes. $DE3$ is the best one out of the single strategies, performing 10 times faster than $DE2$; overall, it is around 2 times slower than *F-AUC-Bandit*.

F-AUC-Bandit shows to be around 1.5 times faster than *Uniform-DE* in around 80% of the trials; the difference is statistically significant for most functions (referring the reader to the pair-wise statistical comparisons presented in [8]). This difference seems to be moderate, relatively to the price to pay for an adaptive scheme. It might thus be observed that the uniform strategy considers here a small number of strategies, most of which perform well: $DE1$ and $DE3$ perform quite well; although much slower, $DE2$ still reaches the target; the only inefficient strategy is $DE4$. In the general case however, the performance of the strategies is unknown; the performance of the above strategies was assessed through extensive experiments. The use of an *Adaptive Strategy Selection* scheme is thus relevant in the general case.

The last series of experiments deals with the previously proposed *AdapSS* schemes, *PM-AdapSS-DE* [10], *AP* [20] and *DMAB* [4], the two latter schemes being fed by extreme rewards [6]. Compared with *AP*, *F-AUC-Bandit* is around 1.5 times faster on around 90% of the trials on the three function classes. It also shows to be around 3 times faster than *DMAB* on half of the trials, being at least around 1.5 times faster on all trials. *PM-AdapSS-DE* shows to be the
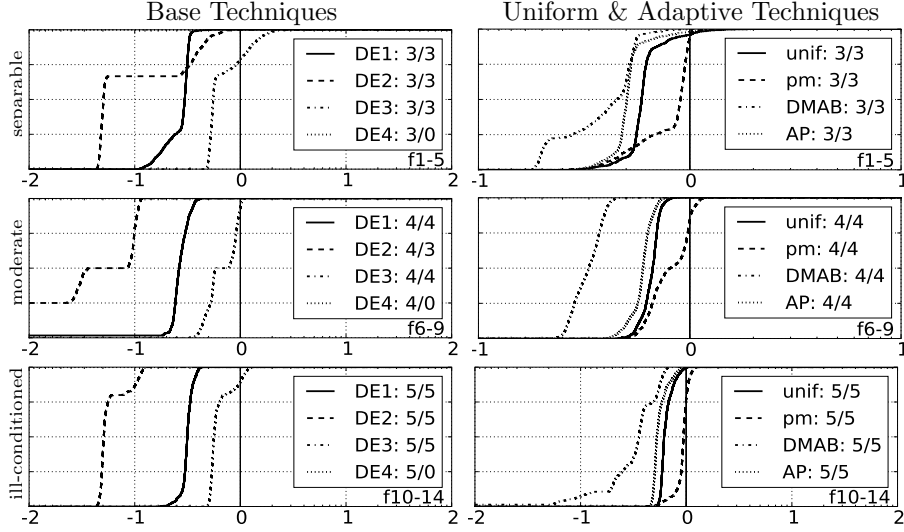
Base Techniques | Uniform & Adaptive Techniques

Fig. 2. Empirical cumulative distribution function of the speed-up ratios in dimension $d = 20$ for the *F-AUC-Bandit* compared with the base techniques (left) and with the uniform and adaptive ones (right). The speed-up ratios are the pairwise ratios of the number of function evaluations for *F-AUC-Bandit* to surpass the target function value 10^{-8} over the one of the baseline techniques over all trials of one functions. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being > 0 or < 1 (for this reason, the lines for DE4 are not visible, as they coincide with the axes). The legends also indicate the number of functions that were solved in at least one trial (*F-AUC-Bandit* first).

Table 1. Median ERT speed-up in dimension $d = 20$ (inter-quartile range in brackets) for a given budget of FEvals. For a given test function, the ERT speed-up is computed as the ratio of the ERT of the algorithm considered (row) over the ERT of *F-AUC-Bandit* (median and inter-quartile range given in first row) for the smallest function value attained by it after a budget of 10, 10^3, 10^5 times the dimension function evaluations or 10^{-8} if it was smaller. The best three values are in bold. The probability of success for reaching the precision 10^{-8} is given in the rightmost column.

(a) Base Techniques

F	budg.	200	20,000	2M	p_s
separable	FAUC	**43**(71)	**16e3**(14e3)	**51e3**(42e4)	.6
	DE1	1 (.84)	**3.3** (.97)	**3.2** (2.1)	.6
	DE2	**1** (4.3)	20 (6.2)	21 (∞)	.6
	DE3	**.86** (.14)	**1.7** (.56)	**1.9** (2.6)	.6
	DE4	1 (.39)	∞	∞	0
moderate	FAUC	**110** (16)	**16e3** (6e3)	**12e4** (5e4)	1
	DE1	1.7 (1.1)	**3.5** (.58)	4 (1.1)	.98
	DE2	3.9 (3.8)	28 (3.4)	22 (∞)	.75
	DE3	**.92** (.48)	**1.9** (.31)	**1.5** (.92)	1
	DE4	**1.1** (.79)	1442 (1e3)	∞	0
ill-condit.	FAUC	67 (125)	**19e3** (314)	**52e3** (52e3)	1
	DE1	**.87** (.55)	**3.2** (.2)	**3.2** (.23)	1
	DE2	1.2 (2.7)	20 (1.9)	20 (3.4)	1
	DE3	**.89** (.61)	**1.9** (.15)	**1.8** (.35)	1
	DE4	**.69** (.53)	∞	∞	0

(b) Uniform & Adaptive Techniques

	budg.	200	20,000	2M	p_s
	FAUC	**43**(71)	**16e3**(14e3)	**51e3**(42e4)	.6
	UNIF	1 (.28)	**1.6** (.16)	**1.6** (.63)	.6
	PM	**1** (.42)	**1.1** (.25)	**1.1** (.88)	.6
	AP	**.82** (.53)	1.9 (.22)	1.9 (.5)	.6
	DMAB	1 (.42)	7.7 (6 .8)	3.5 (∞)	.6
	FAUC	**110** (16)	**16e3** (6e3)	**12e4** (5e4)	1
	UNIF	**.89** (.35)	**1.7** (.23)	**1.5** (.12)	1
	PM	**1.1** (1.1)	**1.1** (.06)	**1.3** (.49)	1
	AP	1.1 (.45)	1.9 (.21)	1.6 (.22)	1
	DMAB	1.1 (5.6)	10 (7)	3.1 (.76)	1
	FAUC	67 (125)	**19e3** (314)	**52e3** (52e3)	1
	UNIF	**.86** (.54)	**1.7** (.08)	**1.7** (.24)	1
	PM	1 (.44)	**1.1** (.03)	**1.1** (.11)	1
	AP	**9** (64)	1.9 (.14)	1.9 (.32)	1
	DMAB	1.3 (2)	6.1 (8)	3.9 (2.6)	.99

best out of the three other adaptive schemes, which is attributed to the use of a relative instead of a raw reward. *F-AUC-Bandit* is around 1.5 times faster than *PM-AdapSS-DE* on around 25% of the trials on the *separable*, and 40% for the *moderate* functions, with an even smaller performance gain on the *ill-conditioned* ones, although still being faster on around 75% of the functions.

This performance improvement of the *F-AUC-Bandit* w.r.t. the others is attributed mostly to: (i) the use of a comparison-based *Credit Assignment*, which is robust to all the very different situations tackled within this benchmark suite, while efficiently following the changes in the qualities of the strategies (the reduction of the AUC for one operator, by definition, results in the augmentation of the AUC for one of the others); and to (ii) the use of a bandit-based *Strategy Selection*, which has already shown to be very efficient in the GA context [6, 7].

5   Conclusion and Perspectives

F-AUC-Bandit, an *Adaptive Operator Selection* scheme that has been recently proposed within the GA framework [9], is fully comparison-based. Thus, when used within a comparison-based algorithm, the invariance w.r.t monotonous transformations of the fitness is preserved. *F-AUC-Bandit* has been used here to select between mutation strategies within a *Differential Evolution* (DE) algorithm. Such combination has been assessed on a set of single-objective continuous problems, defined in the BBOB-2010 [11] noiseless benchmark: *F-AUC-Bandit*+ DE was empirically compared with naive DE schemes, as well as with 3 other adaptive schemes from the literature, *PM-AdapSS-DE* [10], *AP* [20], and *DMAB* [4]. In terms of *expected running time* to achieve a given function target value, *F-AUC-Bandit*+DE was found to outperform the other techniques.

Furthermore, *F-AUC-Bandit* was found to be very robust w.r.t. the tuning of its hyper-parameter – the same setting was found by the racing procedure for all dimensions and all function classes. However, the robustness of such tuning needs to be further assessed, as was done in the GA framework [9].

Nevertheless, the main goal of this work has been reached – validate the *F-AUC-Bandit* approach in a different context than the one it had been designed for originally. However, though much improved over the results of all naive strategies used within DE, the best results of the *F-AUC-Bandit* +DE algorithm remains below those of state-of-the-art optimizers [13]. But the DE algorithms to which *F-AUC-Bandit* has been applied here only use the basic DE techniques, and several improvements have been recently proposed, *e.g.*, adding adaptive parameter control for F and CR [18] . The applicability of *F-AUC-Bandit* in DE framework opens the path for fruitful research using the numerous recent DE variants.

Another further work is to address the multi-modality issue: all tested algorithms fail on multi-modal functions (40% of the separable class – see Table 1). On-going work is concerned with preserving the comparison-based property in the framework of the rewards proposed in [15] to tackle multi-modality.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multi-armed bandit problem. Machine Learning 47(2-3), 235–256 (2002)
2. Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K.: A racing algorithm for configuring metaheuristics. In: W. B. Langdon et al. (ed.) Proc. GECCO. pp. 11–18. Morgan Kaufmann (2002)
3. Bradley, A.P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30, 1145–1159 (1997)
4. Da Costa, L., Fialho, A., Schoenauer, M., Sebag, M.: Adaptive operator selection with dynamic multi-armed bandits. In: M. Keijzer et al. (ed.) Proc. GECCO. pp. 913–920. ACM (2008)
5. Davis, L.: Adapting operator probabilities in genetic algorithms. In: Schaffer, J.D. (ed.) Proc. ICGA. pp. 61–69. Morgan Kaufmann (1989)
6. Fialho, A., Da Costa, L., Schoenauer, M., Sebag, M.: Extreme value based adaptive operator selection. In: G. Rudolph et al. (ed.) Proc. PPSN X. LNCS, vol. 5199, pp. 175–184. Springer (2008)
7. Fialho, A., Schoenauer, M., Sebag, M.: Analysis of adaptive operator selection techniques on the royal road and long k-path problems. In: G. Raidl et al. (ed.) Proc. GECCO. pp. 779–786. ACM (2009)
8. Fialho, A., Ros, R.: Adaptive strategy selection within differential evolution on the BBOB-2010 noiseless benchmark. Research Report RR-7259, INRIA (2010)
9. Fialho, A., Schoenauer, M., Sebag, M.: Toward comparison-based adaptive operator selection. In: J. Branke et al. (ed.) Proc. GECCO. ACM Press (2010)
10. Gong, W., Fialho, A., Cai, Z.: Adaptive strategy selection in differential evolution. In: J. Branke et al. (ed.) Proc. GECCO. ACM Press (2010)
11. Hansen, N., Auger, A., Finck, S., Ros, R.: Real-parameter black-box optimization benchmarking 2010: Experimental setup. Tech. Rep. RR-7215, INRIA (2010)
12. Hansen, N., Finck, S., Ros, R., Auger, A.: Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Tech. Rep. RR-6829, INRIA (2009)
13. Hansen, N., Auger, A., Ros, R., Finck, S., Pošík, P.: Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In: GECCO Workshop Proceedings. ACM Press (2010)
14. Julstrom, B.: What have you done for me lately? Adapting operator probabilities in a steady-state genetic algorithm. In: L. J. Eshelman et al. (ed.) Proc. ICGA. pp. 81–87. Morgan Kaufmann (1995)
15. Maturana, J., Lardeux, F., Saubion, F.: Autonomous operator management for evolutionary algorithms. Journal of Heuristics (2010)
16. Page, E.: Continuous inspection schemes. Biometrika 41, 100–115 (1954)
17. Price, K., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization. Springer-Verlag, Berlin (2005)
18. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transactions on Evolutionary Computation 13(2), 398–417 (2009)
19. Storn, R., Price, K.: Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4) (1997)
20. Thierens, D.: Adaptive strategies for operator allocation. In: Lobo, F.G. et al. (ed.) Parameter Setting in Evolutionary Algorithms, pp. 77–90. Springer (2007)