Probability Matching-based Adaptive Strategy Selection vs. Uniform Strategy Selection within Differential Evolution

An Empirical Comparison on the BBOB-2010 Noiseless Testbed

Álvaro Fialho
Microsoft Research – INRIA
Joint Centre
Parc Orsay Université
91893 Orsay, France
alvaro.fialho@inria.fr

Wenyan Gong
School of Computer Science
China University of Geosciences
Wuhan, 430074 P.R. China
cug11100304@yahoo.com.cn

Zhihua Cai
School of Computer Science
China University of Geosciences
Wuhan, 430074 P.R. China
zhcai@cug.edu.cn

ABSTRACT

Different strategies can be used for the generation of new candidate solutions on the Differential Evolution algorithm. However, the definition of which of them should be applied to the problem at hand is not trivial, besides being a sensitive choice with relation to the algorithm performance. In this paper, we use the BBOB-2010 noiseless benchmarking suite to further empirically validate the Probability Matching-based Adaptive Strategy Selection (PM-AdapSS-DE) [4], a method proposed to automatically select the mutation strategy to be applied, based on the relative fitness improvements recently achieved by the application of each of the available strategies on the current optimization process. It is compared with what would be a timeless (naïve) choice, the uniform strategy selection within the same sub-set of strategies.

Categories and Subject Descriptors

G.1.6 [Numerical Analysis]: Optimization—global optimization, unconstrained optimization; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords


1. INTRODUCTION

Differential Evolution (DE) is a simple yet powerful evolutionary algorithm, that uses the weighted difference between two or more candidate solutions to generate a new one. Proposed by Storn and Price [13], the DE algorithm was initially provided with a single mutation strategy for its offspring generation. Ten other strategies were lately suggested by the same authors [11], increasing the robustness of the algorithm with relation to many different application domains, such as data mining, pattern recognition, digital filter design, neural network training, etc.

But, although bringing advantages, such flexibility creates an extra difficulty to the user. Given an optimization problem, there are no definitive guidelines or rules-of-thumb to assist the user into the definition of which strategy should be used. Such critical choice is usually done following the intuition, or by means of statistics over an expensive set of experiments (off-line tuning).

The latter approach can be always used to find the best strategy; however, besides being computationally and timely expensive, it will very probably yield a sub-optimal choice. Intuitively, a subsequent use of strategies with different characteristics during the optimization process should achieve better performance, as the search tends to migrate from a global (early) exploration of the search space to a more focused, exploitation-like behavior.

This is the main motivation for the use of adaptive parameter control methods, that automatically selects the strategy that should be applied for the generation of each offspring while solving the problem, based on their recent performance on the current optimization process, what is referred to as Adaptive Strategy Selection.

In this paper, the BBOB-2010 noiseless benchmarking suite is used to further empirically validate a technique recently proposed to do so, the PM-AdapSS-DE [4]. For the sake of self-containedness, the description of the proposed adaptive method is resumed in Section 2, referring the reader to the original paper [4] for more a complete and detailed view. The rest of the paper is organized as follows. The settings used for the empirical comparison are presented in Section 3. The complete results are shown in Section 4, while Section 5 presents the timing complexity of each algorithm. Some final considerations conclude the paper in Section 6.

2. THE PM-ADAPSS-DE ALGORITHM

Inspired by some recent works in the Genetic Algorithms community (see, e.g., [14, 1]), Adaptive Strategy Selection aims at automatically selecting between the available (possibly ill-known) mutation strategies while solving the prob-
lem. Two components are needed to achieve such objective: the Credit Assignment scheme defines how to assess the performance of each strategy, measuring the impact of its applications on the progress of the current search/optimization process; while the Strategy (or Operator) Selection mechanism sets how the strategies are chosen, based on their known performance.

In the PM-AdapSS-DE algorithm, the relative fitness improvement \( \eta_i \) proposed in [9], was adopted to assess the performance of each strategy application, which can be formalized as follows:

\[
\eta_i = \frac{\delta}{c_f_i} \cdot |p_f_i - c_f_i|
\]

where \( i = 1, \cdots , NP \), being \( NP \) the population size, \( \delta \) the fitness of the best-so-far solution in the population, and \( p_f_i \) and \( c_f_i \), the fitness of the target parent and of its offspring, respectively. In case of no improvement (i.e., the offspring is worse than or equal to its target parent), a null reward is awarded. Finally, the credit assigned to each strategy is the absolute average value (referred to as \( \text{AvgAbs} \) in the original paper [4]) of the rewards received by them during a given period (a generation in this case).

This credit is used by the Probability Matching (PM) technique to update the known empirical quality of each strategy. Its impact on the update of this estimate is weighted by a user-defined parameter, the adaptation rate \( \alpha \in [0, 1] \) [3]. The probability of selection of each strategy is then defined proportionally to its known performance, possibly being lower bounded by another user-defined parameter, the minimal probability \( p_{\text{min}} \), which ensures that no strategy gets lost during the process [14].

The coupling of such elements with the DE algorithm, referred to as the PM-AdapSS-DE, is shown in Algorithm 1, reproduced from [4]. The modified steps with respect to the classical DE algorithm are marked with a left arrow “\( \Leftarrow \)”. Summarizing, at each generation \( t \), for each target parent \( i \), a strategy \( S_I_i \) is selected based on the probability of each strategy. The offspring is generated with such strategy, and its impact (the relative fitness improvement \( \eta_i \)) is calculated and stored in the set \( S_{SI_i} \). Consequently, at the end of each generation, the assigned credit, quality, and probability of each strategy are updated.

3. EXPERIMENTAL SETTINGS

Given that the objective of the present work is to validate the ability of the proposed Adaptive Strategy Selection approach, a sub-set of 4 strategies was arbitrarily chosen, listed as follows.

1. “DE/rand/1”: \( v_i = x_r_1 + F \cdot (x_r_2 - x_r_3) \)
2. “DE/rand/2”: \( v_i = x_r_1 + F \cdot (x_r_2 - x_r_3) + F \cdot (x_r_4 - x_r_5) \)
3. “DE/rand-to-best/2”: \( v_i = x_r_1 + F \cdot (x_{\text{best}} - x_r_1) + F \cdot (x_r_2 - x_r_3) + F \cdot (x_r_4 - x_r_5) \)
4. “DE/current-to-rand/1”: \( v_i = x_r_1 + F \cdot (x_r_2 - x_r_3) + F \cdot (x_r_4 - x_r_5) \)

where \( x_r \) represents the current individual, \( x_{\text{best}} \) is the best individual in the current generation, \( r_1, r_2, r_3, r_4, r_5 \) are individuals randomly chosen from the population, being \( r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq i \). \( F \) is the mutation scaling factor.

The parameters of the PM technique were defined after a preliminary tuning phase. For the adaptation rate, the following values were tried: \( \alpha \in \{0.1, 0.3, 0.6, 0.9\} \); while for the minimal probability \( p_{\text{min}} \in \{0, 0.05, 0.1, 0.2\} \). Each of the sixteen configurations composed by the combination of such parameter values was tried once on all the functions and instances for dimensions 5 and 20 (a kind of representative set of all the analyzed dimensions, summing up to 720 instances), and the best configuration \( (p_{\text{min}} = 0; \alpha = 0.6) \) was found, according to the Friedman’s two-way Analysis of Variance by Ranks statistical test.

The user-defined parameters of the DE algorithm, namely the population size \( NP \) and the mutation scaling factor \( F \) were defined, respectively, to 10 \( \times D \), and 0.5, with \( D \) being the dimensionality of the problem. Differently from the original paper [4], \( CR = 1.0 \) was used here, in order to have a DE invariant with relation to rotation, and entirely dependent on the mutation strategy application [7].

The technique used as baseline for comparison represents what would be a possible (and costless) choice for a naive user, i.e., the uniform selection within the same sub-set of strategies. In the original paper [4], the PM-AdapSS-DE is

Algorithm 1 DE with probability matching-based adaptive strategy selection: PM-AdapSS-DE

1: Set \( CR = 1.0, F = 0.5 \) and \( NP = 10 \times D \)
2: Generate the initial population
3: Evaluate the fitness for each individual
4: Set the generation counter \( t = 1 \)
5: Set \( K = 4, p_{\text{min}} = 0, \) and \( \alpha = 0.6 \) \( \Leftarrow \)
6: For each strategy \( a \), set \( q_a(t) = 0 \) and \( p_a(t) = 1/K \)
7: While the halting criterion is not satisfied do
8: for \( i = 1 \) to \( NP \) do
9: Select the strategy \( S_I_i \) based on its probability \( \Leftarrow \)
10: Select uniformly randomly \( r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \) \( \Leftarrow \)
11: \( j_{\text{rand}} = \text{rndint}(1, D) \)
12: \( j_{\text{rand}} = 1 \) to \( D \) do
13: If \( \text{rndreal}[0, 1] < CR \) or \( j == j_{\text{rand}} \) then
14: If \( S_I_i == 1 \) then
15: \( u_{i,j} \) is generated by “DE/rand/1” strategy \( \Leftarrow \)
16: else if \( S_I_i == 2 \) then
17: \( u_{i,j} \) is generated by “DE/rand/2” strategy \( \Leftarrow \)
18: else if \( S_I_i == 3 \) then
19: \( u_{i,j} \) is generated by “DE/rand-to-best/2” strategy \( \Leftarrow \)
20: else if \( S_I_i == 4 \) then
21: \( u_{i,j} \) is generated by “DE/current-to-rand/1” \( \Leftarrow \)
22: end if
23: else
24: \( u_{i,j} = x_{i,j} \)
25: end if
26: end for
27: end for
28: for \( i = 1 \) to \( NP \) do
29: Evaluate the offspring \( u_i \)
30: if \( f(u_i) \) is better than or equal to \( f(x_i) \) then
31: Calculate \( \eta_i \) using Eqn. (1) \( \Leftarrow \)
32: Replace \( x_i \) with \( u_i \)
33: end if
34: Set \( \eta_i = 0 \)
35: end if
36: Set \( S_{SI_i} == \eta_i \)
37: end for
38: Calculate the credit \( r_a(t) \) for each strategy \( \Leftarrow \)
39: Update the quality \( q_a(t) \) for each strategy \( \Leftarrow \)
40: Update the probability \( p_a(t) \) for each strategy \( \Leftarrow \)
41: \( t = t + 1 \)
42: end while
also compared to a DE implementing each of the strategies alone, and to yet another adaptive scheme, known as SaDE [12] (with fixed CR and F in this case).

The experiments were performed following the BBOB guidelines [5], with the maximum number of evaluations being fixed at $10^5 \times D$. The mentioned parameter values were used on all the experiments, for all dimensions, thus crafting effort is equal to zero.

4. RESULTS

Results from experiments according to [5] on the benchmark functions given in [2, 6] are presented in Figures 1, 2 and 3 and in Table 1. The expected running time (ERT), used in the figures and table, depends on a given target function value, $f_t = f_{\text{opt}} + \Delta f$, and is computed over all relevant trials as the number of function evaluations executed during each trial while the best function value did not reach $f_t$, summed over all trials and divided by the number of trials that actually reached $f_t$ [5, 10].

**Statistical significance** is tested with the rank-sum test for a given target function value, $\Delta f_t$ (10$^{-8}$ in Figure 1) using, for each trial, either the number of needed function evaluations to reach $\Delta f_t$ (inverted and multiplied by $-1$), or, if the target was not reached, the best $\Delta f$-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration.

5. CPU TIMING EXPERIMENTS

For the timing experiments, both algorithms were run on a 8 and restarted until at least 30 seconds (according to [5]).

The experiments have been conducted with an Intel Xeon E5345 processor (2.33 GHz) running Linux 2.6.31.12. For the PM-AdapSS-DE, the results were 1.6, 1.6, 1.6, 1.8, 2.1 and 2.9 $\times 10^{-6}$ seconds per function evaluation, for the dimensions 2, 3, 5, 10, 20, 40 respectively. For the baseline technique, the Uniform-DE, the results were 3.3, 3.5 4.0, 5.5, 8.3 and 16 $\times 10^{-7}$ seconds per function evaluation, for the dimensions 2, 3, 5, 10, 20, 40 respectively. The same C++ implementation (gcc version 4.4.1) was used for both, with the only difference being the portions of code that refer to the strategy selection, thus the timing difference shows exactly the price to be paid for using this adaptive scheme.

6. FINAL CONSIDERATIONS

This work presented a more extensive empirical validation of the PM-AdapSS-DE, recently proposed in [4]. This method provides Adaptive Strategy Selection capabilities to the DE algorithm, by means of the Probability Matching strategy selection scheme, that selects the operators according to the relative fitness improvements brought by their recent applications.

The objective of this work was not to compete with the state-of-the-art continuous optimizers, but rather to analyze the advantages brought by the adaptive method when compared to the naive (uniform) choice. Although in the multi-modal and weak-structure group of functions it was less clear, improvements were achieved over the baseline method in most of the functions, especially for the larger dimensions, being statistically equivalent otherwise. A possible explanation for the fewer successes concerning the smaller dimensions might lie in the fact that those experiments were too short to allow the adaptive scheme to learn and show its skills. The ill-conditioned functions were the ones in which the advantages of the adaptive scheme was better shown.

There is still a lot of space for further improvements, especially concerning the strategy selection mechanism. Other schemes, such as the Adaptive Pursuit [14] and the Bandit-based approaches [1] should be analyzed. Another path that could also be explored lies in the automatic adaptation of other DE parameters, the crossover rate CR and the mutation scaling factor F, as proposed for the SaDE [12] scheme. Besides, especially concerning the multi-modal functions, the diversity could also be considered somehow by the credit assignment scheme, as proposed in [8].

7. REFERENCES


Figure 1: ERT ratio of PM-AdapSS-DE divided by Uniform-DE versus $\log_{10}(\Delta f)$ for $f_1$–$f_{24}$ in 2, 3, 5, 10, 20, 40-D. Ratios < $10^0$ indicate an advantage of PM-AdapSS-DE, smaller values are always better. The line gets dashed when for any algorithm the ERT exceeds thrice the median of the trial-wise overall number of f-evaluations for the same algorithm on this function. Symbols indicate the best achieved $\Delta f$-value of one algorithm (ERT gets undefined to the right). The dashed line continues as the fraction of successful trials of the other algorithm, where 0 means 0% and the y-axis limits mean 100%, values below zero for PM-AdapSS-DE. The line ends when no algorithm reaches $\Delta f$ anymore. The number of successful trials is given, only if it was in $\{1 \ldots 9\}$ for PM-AdapSS-DE (1st number) and non-zero for Uniform-DE (2nd number). Results are significant with $p = 0.05$ for one star and $p = 10^{-**}$ otherwise, with Bonferroni correction within each figure.
Figure 2. Expected running time (ERT in log10 of number of function evaluations) of PM-AdapSS-DE versus Uniform-DE for 46 target values $\Delta \in [10^{-8}, 10]$ in each dimension for functions $f_1$–$f_{24}$. Markers on the upper or right edge indicate that the target value was never reached by PM-AdapSS-DE or Uniform-DE respectively.

Markers represent dimension: 2: $\times$, 3: $\triangledown$, 5: $\star$, 10: $\circ$, 20: $\Diamond$, 40: $\blacklozenge$. 

- 1 Sphere
- 2 Ellipsoid separable
- 3 Rastrigin separable
- 4 Skew Rastrigin-Bueche
- 5 Linear slope
- 6 Attractive sector
- 7 Step-ellipsoid
- 8 Rosenbrock original
- 9 Rosenbrock rotated
- 10 Ellipsoid
- 11 Discus
- 12 Bent cigar
- 13 Sharp ridge
- 14 Sum of diff. powers
- 15 Rastrigin
- 16 Weierstrass
- 17 Schaffer F7, cond.10
- 18 Schaffer F7, cond.1000
- 19 Griewank-Rosenbrock
- 20 Schwefel $x \sin(x)$
- 21 Gallagher 101 peaks
- 22 Gallagher 21 peaks
- 23 Katsuuras
- 24 Lunacek bi-Rastrigin
- 17 Schaffer F7, cond.10
- 18 Schaffer F7, cond.1000
- 14 Sum of diff. powers
- 10 Ellipsoid
- 6 Attractive sector
- 2 Ellipsoid separable
- 15 Rastrigin
- 11 Discus
- 7 Step-ellipsoid
- 3 Rastrigin separable
- 20 Schwefel $x \sin(x)$
- 16 Weierstrass
- 12 Bent cigar
- 8 Rosenbrock original
- 4 Skew Rastrigin-Bueche
Figure 3: Empirical cumulative distributions (ECDF) of run lengths and speed-up ratios in 5-D (left) and 20-D (right). Left sub-columns: ECDF of the number of function evaluations divided by dimension $D$ ($\text{FEvals}/D$) to reach a target value $f_{opt} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{-1, -4, -8\}$ is given by the first value in the legend, for PM-AdapSS-DE (solid) and Uniform-DE (dashed). Light beige lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right sub-columns: ECDF of FEval ratios of PM-AdapSS-DE divided by Uniform-DE, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being $> 0$ or $< 1$. The legends indicate the number of functions that were solved in at least one trial (PM-AdapSS-DE first).
<table>
<thead>
<tr>
<th>5-D</th>
<th>20-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Delta f )</td>
<td>( \Delta f )</td>
</tr>
<tr>
<td>( f_1 )</td>
<td>11</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>8.6</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>20.1</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>83.6</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>20</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>29</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>20</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>1000</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>1.2e4</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>1500</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>5e-1</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>1000</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>1.2e4</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>20</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>1000</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>1.2e4</td>
</tr>
<tr>
<td>( f_7 )</td>
<td>1500</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_8 )</td>
<td>20</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>1000</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>1.2e4</td>
</tr>
<tr>
<td>( f_9 )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{10} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{11} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{12} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{13} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{14} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{15} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{16} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{17} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{18} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{19} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
<tr>
<td>( f_{20} )</td>
<td>1000</td>
</tr>
<tr>
<td>0: Uniform</td>
<td>43</td>
</tr>
<tr>
<td>1: AdapSS</td>
<td>43</td>
</tr>
</tbody>
</table>

Table 1: Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009 (given in the respective first row) for different \( \Delta f \) values for functions \( f_1 - f_{20} \). The median number of conducted function evaluations is additionally given in italics, if ERT(10^-7) = \( \infty \). #succ is the number of trials that reached the final target \( f_{opt} \) = 10^{-8}. 0: Uniform is Uniform-DE and 1: AdapSS is PM-AdapSS-DE. Bold entries are statistically significantly better compared to the other algorithm, with \( p = 0.05 \) or \( p = 10^{-3} \) where \( k > 1 \) is the number following the ⋆ symbol, with Bonferroni correction of 48.