L-SHADE with Alternative Population Size Reduction for Unconstrained Continuous Optimization

Christopher Renkavieski
Santa Catarina State University - UDESC
Joinville, SC, Brazil
chris.renka@gmail.com

Rafael Stubs Parpinelli
Santa Catarina State University - UDESC
Joinville, SC, Brazil
rafael.parpinelli@udesc.br

ABSTRACT

Differential Evolution (DE) is a powerful and versatile algorithm for numerical optimization, but one of its downsides is its number of parameters that need to be tuned. Multiple techniques have been proposed to self-adapt DE's parameters, with L-SHADE being one of the most well established in the literature. This work presents the A-SHADE algorithm, which modifies the population size reduction schema of L-SHADE, and also EB-A-SHADE, which applies a mutation strategy hybridization framework to A-SHADE. These algorithms are applied to the CEC2013 benchmark set with 100 dimensions, and it's shown that A-SHADE and EB-A-SHADE can achieve competitive results.

KEYWORDS

Evolutionary Computing, Differential Evolution, Self Adaptation

1 INTRODUCTION

Differential Evolution (DE) is an evolutionary algorithm designed for the optimization of continuous problems [7]. Although DE is a very simple and competitive algorithm, its performance is dependent on the control parameters used, and these are dependent on the problem being optimized.

It is possible to identify four parameters that need to be tuned in DE. These are: population size (NP), mutation scaling factor (F), crossover rate (CR) and the mutation strategy used. Since it is time consuming to find a good setting for these parameters to each problem that needs to be optimized, it is desirable to have an algorithm that self adapt their values. Multiple methods have already been proposed on the literature to self adapt DE's parameters, with SHADE [8] and L-SHADE [9] being examples of popular algorithms built for this purpose. SHADE adapts parameters F and CR, while L-SHADE adds a linear schema for population size reduction to SHADE, such that it also adapts the parameter NP.

In this paper, it is proposed a variation for the L-SHADE algorithm that uses an alternative method for the population size reduction, which is called A-SHADE. The main difference between the two population size reduction methods is that, in the one used by A-SHADE, the value of NP decreases faster at the beginning of the evolution, giving the algorithm more generations to con-verge while still keeping the exploration provided by the initial population size.

This paper also proposes the use of A-SHADE in conjunction with a hybridization framework for the mutation strategy, based on the algorithm EB-L-SHADE [3]. This framework uses a second mutation strategy alongside the one used by L-SHADE. The algorithm that uses it in conjunction with A-SHADE is called EB-A-SHADE.

The rest of this paper is organized as follows: section 2 explains the Differential Evolution algorithm, while section 3 shows some of the main self-adaptive variants of DE found in the literature, highlighting the algorithms SHADE, L-SHADE and EB-L-SHADE, which are the basis for this work. Section 4 explains the algorithms proposed in this work, namely A-SHADE and EB-A-SHADE. The experiments are explained in section 5, while the results obtained are shown and discussed in section 6. The author's final considerations are exposed in section 7.

2 DIFFERENTIAL EVOLUTION

Differential Evolution is an evolutionary algorithm in which the individuals are represented as real number vectors of the form $x_i = (x_1, ..., x_D)$, with $i = [1, NP]$ and D being the problem dimensionality. The initial population is distributed in the search space by a uniform random distribution.

After the creation of the initial population, the algorithm begins the evolutionary loop. During each iteration of the loop, each individual in the population – called a target individual – produces a trial individual. The steps for the creation of a trial individual are mutation and crossover.

During the mutation step, a mutant individual v_i is generated from the population according to a mutation strategy. One such strategy is DE/Rand/1/bin, defined as:

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{1}$$

where x_{r1}, x_{r2} and x_{r3} are distinct individuals randomly chosen from the population.

After the mutant individual is generated, it undergoes crossover alongside the target individual, producing the trial individual u_i. This operation is as follows:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } rand(j) \leq CR \text{ or } j = rand(i) \\ x_{i,j} & \text{if } rand(j) > CR \text{ and } j \neq rand(i) \end{cases} \quad (2)$$

where $rand(j)$ is a uniformly distributed real random number in the range [0,1], and $rand(i)$ is a uniformly distributed integer in the range [1,D] so that the trial individual inherits at least one gene from the mutant one.

Once the trial individual is generated comes the selection step: its fitness value is evaluated and, if it's better than it's target individual fitness, the trial will take its place in the next generation.

3 SELF-ADAPTIVE DIFFERENTIAL EVOLUTION

Multiple algorithms have been proposed in the literature to self adapt the parameters in DE [5]. Some of the most well known ones are presented in this section.

SaDE [6] generates F and CR values for each individual using different random distributions. It also uses two mutation strategies, and adapts the probability of which one will be used for each trial vector based on their success on previous generations.

Another popular algorithm is jDE [1], in which individuals can either inherit their values of F and CR from their target vectors, or have them regenerated following specific random distributions.

CoDE [10] has a set of predefined values for F and CR, as well as three different mutation strategies. Each target individual randomly selects a F and CR pair and generates three trial individuals, one for each mutation strategy. All three trial individuals are evaluated, and the best one is kept for the selection step.

In JADE [11], each individual's F and CR values are generated from the means μ_F and μ_{CR}, which are adapted during the evolution. This algorithm also uses a different mutation strategy and an external archive containing individuals that were removed from the population during previous generations, and can be selected in the mutation step.

SHADE and L-SHADE [8, 9] are also popular self-adaptive DE algorithms. Since these are the basis for the algorithms used in this work, they will be presented in more detail in the next section.

3.1 SHADE

SHADE was developed using JADE as a basis, therefore it shares multiple characteristics with it, including the mutation strategy and the way individuals generate their F and CR values. It's worth noting that there are two versions of this algorithm: the one presented in the original paper [8] and an updated one that is used as basis for L-SHADE [9]. Since this work is based on L-SHADE, the updated version will be explained in this section.

In place of the means μ_F and μ_{CR} from JADE, SHADE uses two lists, called M_F and M_{CR}, that store each H values. During each generation, every individual uniformly chooses an index r_i in the range [1,H], and generates their F and CR values as follows:

$$F_i = randc(M_{F,r_i}, 0.1) \quad (3)$$

$$CR_i = \begin{cases} 0 & \text{if } M_{CR,r_i} = \perp \\ randn(M_{CR,r_i}, 0.1) & \text{otherwise} \end{cases} \quad (4)$$

where $randc(M_{F,r_i}, 0.1)$ represents a random value generated with a Cauchy distribution, with mean M_{F,r_i} and variance 0.1, and $randn(M_{CR,r_i}, 0.1)$ is a random value generated with a Gaussian distribution, with mean M_{CR,r_i} and standard deviation 0.1.

At the end of each generation, the values with index k are updated in these lists, with k being initialized with value 1, and being incremented by 1 each time the lists are updated. If $k > H$, k is then set to 1. The values for $M_{F,k}$ and $M_{CR,k}$ are updated as follows:

$$M_{Q,k} = \begin{cases} mean_{WL}(S_Q) & \text{if } S_Q \neq \emptyset \\ M_{Q,k} & \text{otherwise} \end{cases} \quad (5)$$

where Q means both F and CR. S_F and S_{CR} (represented as S_Q in equation 5) are two lists that store, respectively, the values of F and CR of the individuals that generated a successful offspring in the current generation. The value $mean_{WL}(S_Q)$ is the weighted Lehmer mean over the contents of S_Q, and it is computed as follows:

$$mean_{WL}(S_Q) = \frac{\sum_{k=1}^{|S_Q|} w_k \cdot S_{Q,k}^2}{\sum_{k=1}^{|S_Q|} w_k \cdot S_{Q,k}} \quad (6)$$

$$w_k = \frac{\Delta f_k}{\sum_{j=1}^{|S_Q|} \Delta f_j} \quad (7)$$

and $\Delta f_k = |f(u_k) - f(x_k)|$ is the fitness difference between the trial individual u_k and the target individual x_k that was replaced by u_k for the next generation.

For the update of $M_{CR,k}$, its value will be set to $\perp$ if $max(S_{CR}) = 0$, and once $M_{CR,k} = \perp$, its value will not be changed for the rest of the evolution. If this happens, then only one gene of the target individual will be changed in the trial individual, and that is the gene with index j in equation 2.

SHADE also uses an external archive A, that is empty at the start of the evolution. Whenever an individual is replaced by a better trial vector in the population, it is sent to this archive instead of being lost. This archive has a maximum size of $NP \cdot r^{arc}$, and when its size exceeds this limit, at the end of a generation, individuals are randomly removed from it until it is again at maximum size.

The individuals stored in the archive can be used during the mutation step. The mutation strategy used by SHADE is:

$$v_i = x_i + F_i(x_{best}^p - x_i) + F_i(x_{r1} - x'_{r2}) \quad (8)$$

This strategy is called DE/current-to-$p$best/1/bin, and was first introduced by JADE [11]. In it, x_i is the target vector, F_i is its F value, and x_{best}^p is a randomly selected individual among the p% best in the population, with $p \in [0, 1]$. The individual x_{r1} is selected randomly from the population, while x'_{r2} is selected randomly from the union between the population and the external archive. The individuals x_{r1} and x'_{r2} must be different from each other, as well as from x_i.

The crossover operation in SHADE is done in the same way as in canonical DE, with the difference that the CR value is generated for each individual following equation 4, instead of being a global parameter.

3.2 L-SHADE

L-SHADE is an improved version of SHADE that adds a linear scheme for population size reduction to it [9]. Instead of a fixed value for the parameter NP, it has a initial size (NP_0) and a final size (NP_f). At the end of each generation g, the size for the next generation is computed as:

$$NP_{g+1} = \text{round}\left[\frac{NP_f - NP_0}{MAX\_NFE} \cdot NFE + NP_0\right] \tag{9}$$

where MAX_NFE is the maximum number of function evaluations allowed for the algorithm, and NFE is the current amount of function evaluations spent.

Figure 1 shows the population size evolution, in relation to the number of generations, for L-SHADE. The values for NP_0, NP_f and MAX_NFE used to generate the figure are presented in section 5.

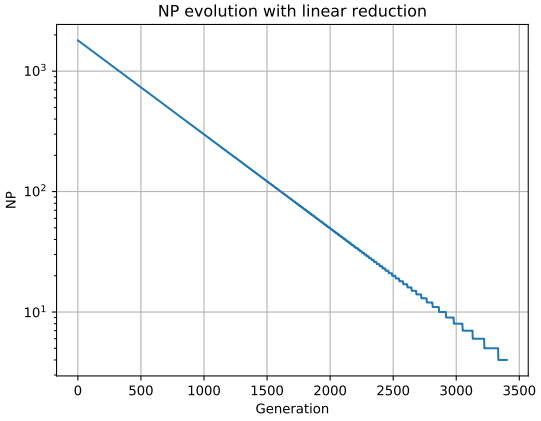

Figure 1: Population size in relation to generation number for the L-SHADE algorithm.

3.3 EB-L-SHADE

EB-L-SHADE adds to L-SHADE a second mutation strategy called DE/current-to-ord_pbest/1/bin [3]. This strategy works similarly to DE/current-to-*p*best/1/bin, in the way that first three random individuals are selected, being one from the p% best individuals, one from the population, and one from the union between the population and the external archive. Among these three individuals, the one with best fitness is called x_{ord_pbest}, the one with median fitness is called $x_{ord\_pmedian}$, and the worst one is called x_{ord_pworst}. The mutation then is done as follows:

$$v_i = x_i + F_i \cdot (x_{ord\_pbest} - x_i) + F_i \cdot (x_{ord\_pmedian} - x_{ord\_pworst}) \tag{10}$$

which is the same equation for DE/current-to-*p*best/1/bin presented in equation 8, but with the individuals ordered according the their fitness values.

It is used a memory M_{FCP}, with size H, to store a list of values between 0 and 1. Similar to the way they generate their F and CR values, each individual uses the value M_{FCP,r_i} to determine its probability to use one of the two mutation strategies.

At the end of each generation in which at least one individual is replaced, the value $M_{FCP,k}$ is updated. For that, first the amount of improvement that each strategy brought in the current generation is computed as:

$$\omega_{m1} = \sum_{j=1}^{n} f(x_i) - f(u_i) \tag{11}$$

which is the sum of the difference in fitness between all trial and target individuals that were selected for the next generation and were created using the strategy $m1$. The value ω_{m2} is computed in the same way as in equation 11, but using the trial individuals that were generated with the strategy $m2$.

The next step is to determine the improvement rate for strategy $m1$ as follows:

$$\Delta_{m1} = \min\left(0.8, \max\left(0.2, \frac{\omega_{m1}}{\omega_{m1} - \omega_{m2}}\right)\right) \tag{12}$$

in which 0.2 and 0.8 are the minimum and maximum probabilities that a strategy can have.

The value $M_{FCP,k}$ is then updated as:

$$M_{FCP,k} = (1 - c)M_{FCP,k} + c\Delta_{m1} \tag{13}$$

in which c is a learning rate that determines how much weight the previous value of $M_{FCP,k}$ has when computing the new value.

Since the sum of probabilities for $m1$ and $m2$ should equal to 1, the probability for an individual to choose $m1$ is M_{FCP,r_i}, while its probability to choose $m2$ is $1 - M_{FCP,r_i}$.

4 PROPOSED ALGORITHMS

This sections explain the proposed changes to L-SHADE, as well as the reasoning behind them.

4.1 A-SHADE

The first proposed algorithm is A-SHADE, which means L-SHADE with an alternative population size reduction method. In this algorithm, instead of reducing the value of NP linearly, it is done following the equation:

$$NP_i = NP_0\left(\frac{NP_f}{NP_0}\right)^{\frac{i}{MAX\_NFE}} \tag{14}$$

Equation 14 has an exponential behavior, meaning the value of NP decreases quickly during the first generations, and then slows down until it reaches the final value. This behavior can be seen in figure 2, which shows the population size evolution, in relation to the number of generations, for A-SHADE. The values for NP_0, NP_f and MAX_NFE used to generate the figure were the same used in the experiments, explained in section 5.

The reasoning behind this change is as follows: with the recommended values of $NP_0 = 18 \cdot D$ and $NP_f = 4$ for L-SHADE [9], the search begins with a large population, what allows a high quality exploration of the search space. It's not interesting, however, to keep this value high for too long, because as the population converges, less exploration happens, so that a large population will consume the available function evaluations faster but without offering a significant advantage. This slow decay of NP can be seen in figure 1.

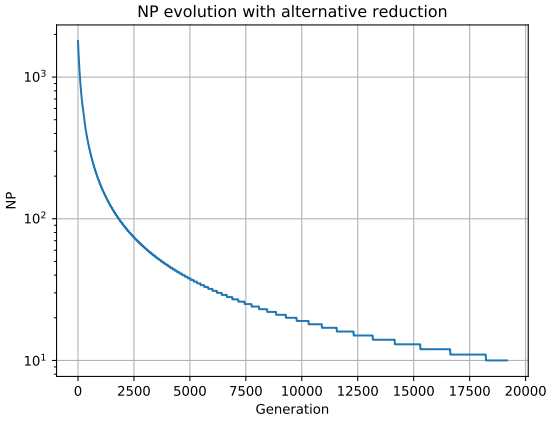

Figure 2: Population size in relation to generation number for the A-SHADE algorithm.

With A-SHADE, as figure 2 shows, the initial value of NP is kept high. This, as with L-SHADE, allows a global exploration in the beginning of the search. But differently than L-SHADE, the faster decay means the algorithm doesn't spend its function evaluations as fast, leaving more evaluations for the smaller population sizes. This allows for a better local search by the end of the evolution process.

With this change, it is expected that A-SHADE may be able to perform a more thorough local search for the optimal solutions, without losing the global search power of L-SHADE.

4.2 EB-A-SHADE

For EB-A-SHADE, the idea is to use the alternative method for population size reduction alongside the hybridization framework for mutation strategies presented in EB-L-SHADE.

This way, this algorithm works in the same way as EB-L-SHADE, explained in section 3.3. The exception to this is the NP reduction scheme, which is the same used in A-SHADE, presented in equation 14.

With this, it is expected that the improvements provided by EB-L-SHADE can help A-SHADE to achieve better results in the experiments.

5 EXPERIMENTS

The experiments were run using the CEC-2013 global optimization benchmark set, which is composed of 28 functions [2]. These are all unconstrained functions with dimensions bound in the range [-100,100] and a single global optimum. Their domain is also shifted, so that the global optimum is not at the origin of the coordinate system.

The algorithms evaluated were DE, SHADE, L-SHADE, EB-L-SHADE, A-SHADE and EB-A-SHADE. The experiments were done in 100 dimensions, with $10^4 \times D$ function evaluations available for each run. 30 independent runs were executed for each algorithm in each function.

At the end of each generation, the performance of the algorithms was evaluated with the error value of the best individual, as well as the mean population error. The error is defined as the difference between the objective value of the best individual found and the objective value of the global optimum.

Statistical analysis of the results were made using the Kruskal-Wallis test, followed by the post-hoc Dunn test, with a significance level of 95%.

Another output at the end of each generation is the population's genetic diversity. The diversity is computed as a variation of the moment of inertia method described by [4].

All codes were implemented in C++, and were run in a system equipped with an Intel Core i7-4770, 16 GB of RAM and a Linux OS, with distribution Ubuntu 18.04 LTS. The evaluation of the individuals was made in parallel, using OpenMP. All codes and results are available in https://github.com/ChrisRenka/TCC.

For DE, SHADE, L-SHADE and EB-L-SHADE, the parameters used in the experiments were those recommended by the authors in the literature. These parameters were:

- DE [7]: $NP = 100, F = 0.5, CR = 0.9$, mutation DE/rand/1/bin.
- SHADE [8]: $NP = 100$, $p = 0.10$, $H = 100$, $r^{arc} = 1$.
- L-SHADE [9]: $NP_i = 18 \cdot D$, $NP_f = 4$, $p = 0.11$, $H = 5$, $r^{arc} = 1.4$.
  - EB-L-SHADE [3]: $c = 0.2$.

The parameters for the proposed algorithms were kept the same as the ones used for L-SHADE and EB-L-SHADE, with the exception being the final population size NP_f. This value was changed from 4 to 10 to allow for a larger population pool in the last generations. The complete list of parameters used for the proposed algorithms are as follows:

- A-SHADE: $NP_i = 18 \cdot D$, $NP_f = 10$, $p = 0.11$, $H = 5$, $r^{arc} = 1.4$.
  - EB-A-SHADE: $c = 0.2$.

6 RESULTS

Table 1, in the last page, shows the results obtained in the experiments. Each line presents the mean and standard deviation of the error obtained by each algorithm for one of the functions in the benchmark set. In boldface are the best results obtained for each function, which were determined from the error values and the statistical tests.

The line "Number of times among best" shows, for each algorithm, in how many functions its results were among the best. The line B/S/W (Best/Same/Worst) is comparing each algorithm to EB-A-SHADE, and it is showing in how many functions EB-A-SHADE's results were best, equivalent or worst than the results obtained by each other algorithm.

Table 1 shows that the alternative population size reduction achieved competitive results in these experiments, being among the best algorithms for 17 functions with A-SHADE, and 19 with EB-A-SHADE. On the other hand, L-SHADE and EB-L-SHADE were among the best in 14 and 15 functions, respectively, while DE and SHADE were among the best in 4 and 7 functions. EB-A-SHADE achieved better results than EB-L-SHADE in 11 functions, equivalent in 10 and worst in 7, showing that its overall performance was superior.

One thing to notice was the relation between A-SHADE and EB-A-SHADE, which were equivalent in all 28 functions, even though EB-A-SHADE's results were among the best 19 times, against A-SHADE's 17 times. That happens because of functions 2 and 4, in which EB-A-SHADE was equivalent to the best algorithm, while A-SHADE was not. This shows that the mutation strategy hybridization used provided a small impact in the algorithm performance, such that, when compared in isolation, the statistical analysis could not find significant differences between them. When they are evaluated alongside other algorithms, though, these differences are highlighted, meaning that, overall, EB-A-SHADE's results are more consistent than those of A-SHADE.

Figures 3 and 4, in the next two pages, show the convergence and diversity plots of all six algorithms on functions 6, 8, 9, 12, 20 and 25, respectively. These functions were chosen because they give a good overview of the behaviors observed in the experiments. In both plots, the X axis represents the number of function evaluations, while the Y axis is the error for the plots on the left, and the genetic diversity for the plots on the right. Both plots show the values averaged over 30 runs.

From these plots, it can be seen that there's a strong difference in convergence behavior between DE, SHADE, both L-SHADE algorithms and both A-SHADE algorithms. The differences are less noticeable between L-SHADE and EB-L-SHADE, and between A-SHADE and EB-A-SHADE. These observed behaviors are discussed in the following paragraphs.

SHADE has a fast convergence, which usually results in it achieving worst results in relation to the other algorithms – the exceptions being function 8, in which it's convergence speed is similar to the others, and in function 6, in which it achieves good results even with the fast convergence. This can also be seen in the diversity plots, where SHADE's diversity decreases faster than the other algorithm's. From these observations, it can be concluded that SHADE tends to get stuck in a local optima.

L-SHADE and EB-L-SHADE have a tendency to show an intense convergence by the end of the evolution, so that they are still converging when it ends. That's an undesirable behavior, because it shows that these algorithms are not achieving the best results that they could. In the plots presented, this behavior can be observed in the error of functions 8, 9, 12 and 20, and it can also be observed in the diversity plots of functions 6, 8, 9, 20 and 25. In function 20, these algorithms still achieved the best results, even with this behavior.

A-SHADE and EB-A-SHADE, on the other hand, showed a better convergence behavior, which is faster than L-SHADE's, to the point that it stabilizes before the evolution ends, but is not fast enough to be considered premature. This shows that the alternative NP size reduction keeps the exploratory power of the linear reduction, while giving more time to the population intensify it's results. This way, A-SHADE's and EB-A-SHADE's convergence speed lies between that of SHADE and L-SHADE.

Analysing the diversity evolution shown in the figures, A-SHADE and EB-A-SHADE follow L-SHADE and EB-L-SHADE until approximately 200.000 function evaluations in most functions. The exception is function 8, in which this behavior lasts until approximately 500.000 evaluations. After this period, the diversity starts to decay faster in the A-SHADE algorithms, in comparison to what is observed with L-SHADE. This shows that the proposed algorithms switch to local search faster than the original ones, but without compromising the initial global search. This observation reinforces the reasoning behind the alternative NP reduction, presented in section 4.1.

When it comes to the difference between the algorithms with and without mutation strategy hybridization, they behave mostly in the same way in the plots. Small differences between them can be seen in the plots for functions 8, 20 and 25. For function 8, EB-A-SHADE achieved a smaller error than A-SHADE, while for function 25, this can be observed for both EB algorithms. In function 20, the algorithms without hybridization performed better, going against the overall behavior of better results for the EB algorithms. These observations reinforce what was observed in table 1, showing that the hybridization increases the consistency of the results obtained by the algorithms.

7 CONCLUSIONS

In this paper, two algorithms were presented: A-SHADE, which is a variation of L-SHADE [9] using an alternative method for population size reduction, and EB-A-SHADE, which applies to A-SHADE a framework for mutation strategy hybridization [3]. These algorithms were applied to the CEC2013 global optimization benchmark set in 100 dimensions, and their results were evaluated and compared to those of DE, SHADE, L-SHADE and EB-L-SHADE.

The results showed that A-SHADE and EB-A-SHADE outperformed L-SHADE and EB-L-SHADE in this experimental setting. It was shown that the alternative population size reduction method achieves a more stable convergence than the linear population size reduction, giving the algorithm enough time to stabilize its population's fitness before the end of the evolution process.

It was also shown that EB-A-SHADE achieved better results overall than A-SHADE, even though, when only the two were compared, they showed no significant statistical difference. From this, it can be concluded that it's possible to improve the performance of L-SHADE and A-SHADE through mutation strategy hybridization, but the hybridization method and strategies used can still be improved.

ACKNOWLEDGMENTS

The authors would like to thank UDESC and CNPq for their support to this research.

REFERENCES

[1] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer. 2006. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Transactions on Evolutionary Computation* 10, 6 (Dec 2006), 646–657. https://doi.org/10.1109/TEVC.2006.872133

[2] J. J. Liang, B. Y. Qu, P. N. Suganthan, and Alfredo G. Hernández-Díaz. 2013. *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization.* Technical Report 201212. Zhengzhou University and Nanyang Technological University, Cancún, Mexico.

[3] Ali W. Mohamed, Anas A. Hadi, and Kamal M. Jambi. 2019. Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization. *Swarm and Evolutionary Computation* (Nov 2019). https://doi.org/10.1016/j.swevo.2018.10.006

[4] Ronald W. Morrison and Kenneth A. De Jong. 2001. Measurement of Population Diversity. *Artificial Evolution* (Oct 2001), 31–41. https://doi.org/10.1007/3-540-46033-0_3

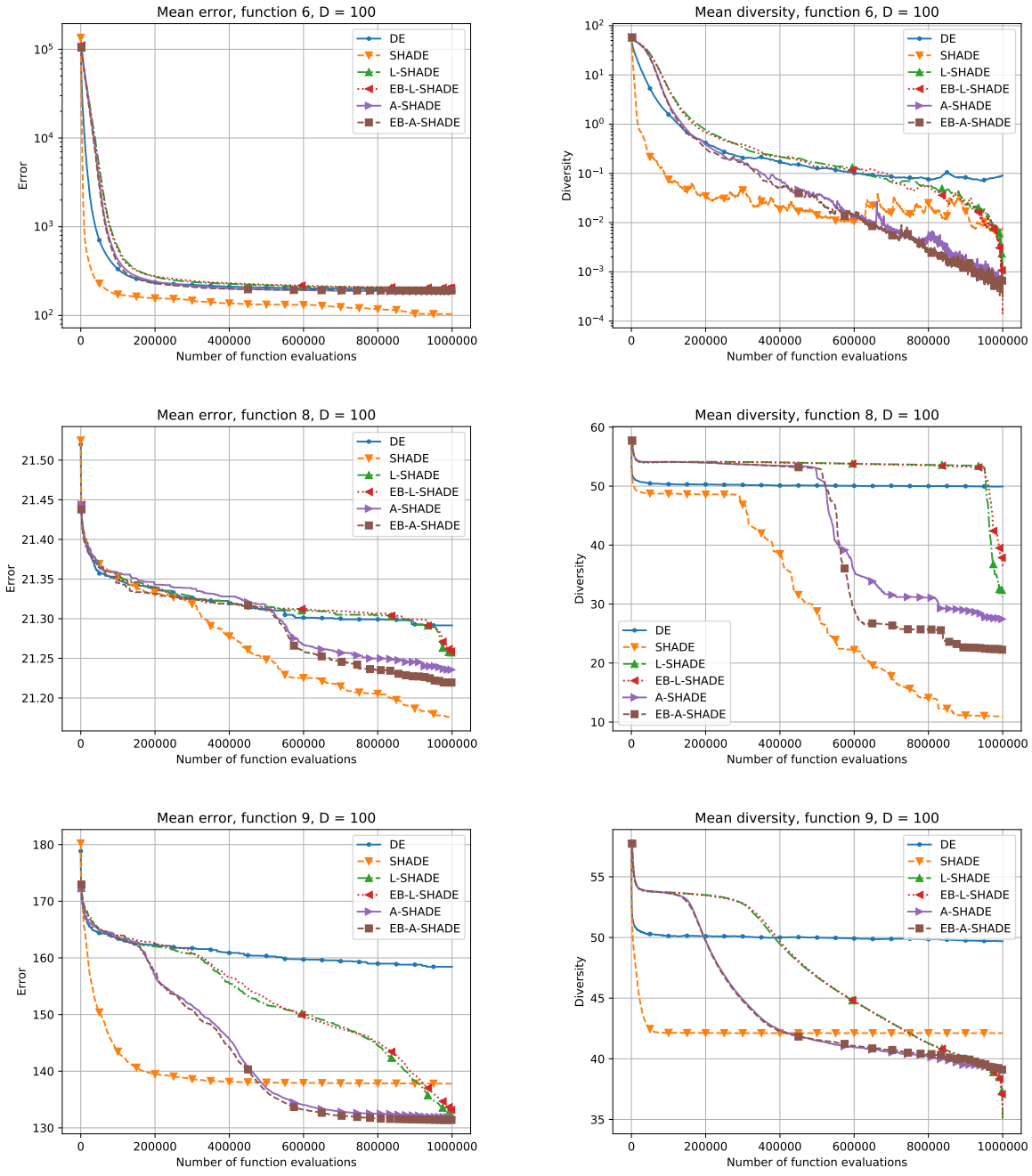

Figure 3: Error convergence and diversity plots for functions number 6, 8 and 9.

[5] Rafael Parpinelli, Guilherme Felippe Plichoski, Renan Silva, and Pedro Narloch. 2019. A review of techniques for online control of parameters in swarm intelligence and evolutionary computation algorithms. *International Journal of Bio-Inspired Computation* 13 (Jan 2019), 1–17. https://doi.org/10.1504/IJBIC.2019.097731

[6] A. K. Qin, V. L. Huang, and P. N. Suganthan. 2009. Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization. *IEEE Transactions on Evolutionary Computation* 13, 2 (April 2009), 398–417. https://doi.org/10.1109/TEVC.2008.927706

[7] Rainer Storn and Kenneth Price. 1997. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* 11, 4 (Dec 1997), 341–359. https://doi.org/10.1023/A:1008202821328

[8] R. Tanabe and A. Fukunaga. 2013. Success-history based parameter adaptation for Differential Evolution. *2013 IEEE Congress on Evolutionary Computation* (June 2013), 71–78. https://doi.org/10.1109/CEC.2013.6557555

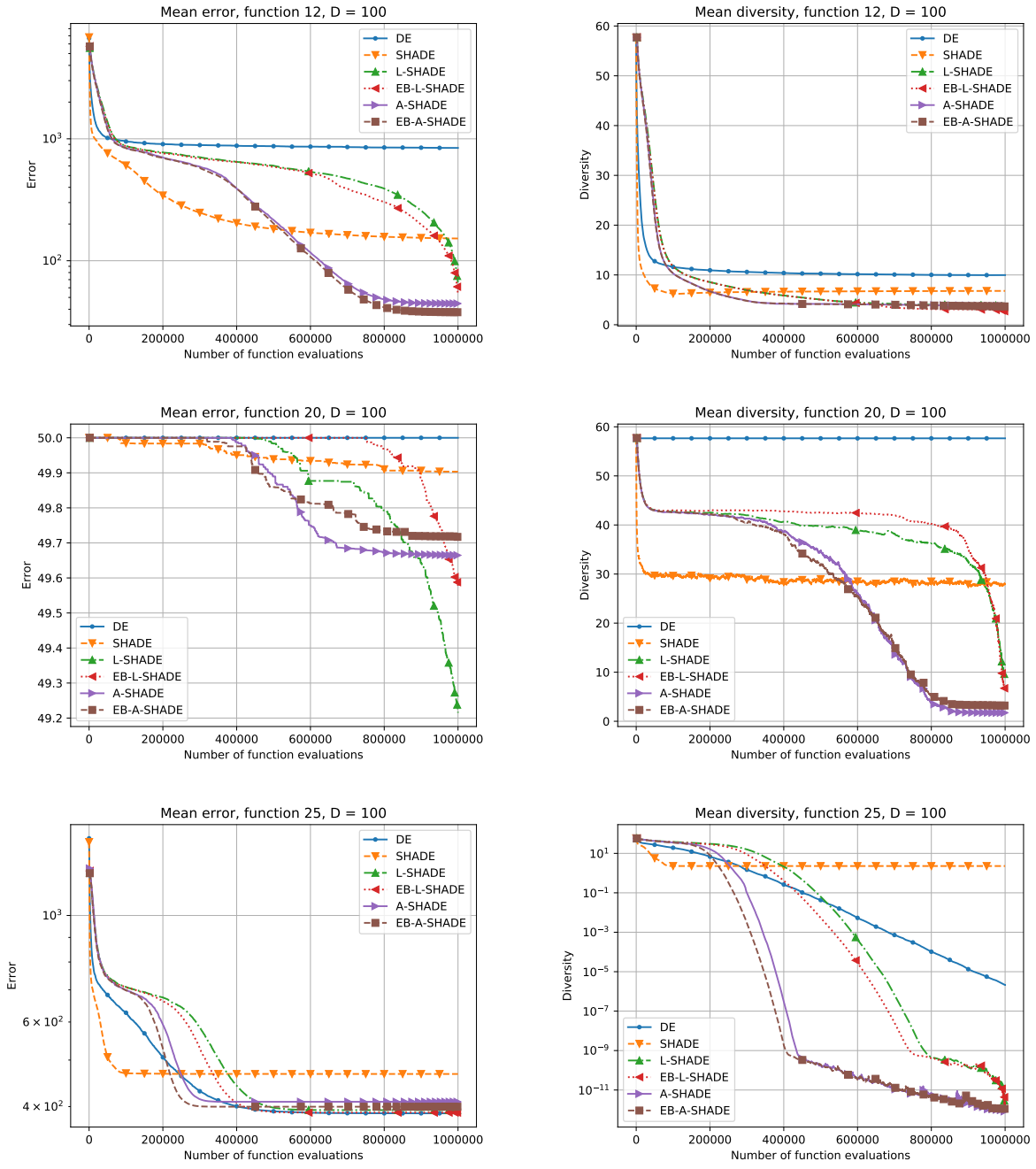

Figure 4: Error convergence and diversity plots for functions number 12, 20 and 25.

[9] R. Tanabe and A. S. Fukunaga. 2014. Improving the search performance of SHADE using linear population size reduction. *2014 IEEE Congress on Evolutionary Computation (CEC)* (July 2014), 1658–1665. https://doi.org/10.1109/CEC.2014.6900380

[10] Y. Wang, Z. Cai, and Q. Zhang. 2011. Differential Evolution With Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Transactions on Evolutionary Computation* 15, 1 (Feb 2011), 55–66. https://doi.org/10.1109/TEVC.2010.2087271

[11] J. Zhang and A. C. Sanderson. 2009. JADE: Adaptive Differential Evolution With Optional External Archive. *IEEE Transactions on Evolutionary Computation* 13, 5 (Oct 2009), 945–958. https://doi.org/10.1109/TEVC.2009.2014613

Table 1: Mean and standard deviation obtained for each algorithm in each function.

Function	DE	SHADE	L-SHADE	EB-L-SHADE	A-SHADE	EB-A-SHADE
f_1	**3.18e-13** ± **1.11e-13**	4.55e-13 ± 0.00e+0	**2.27e-13** ± **0.00e+0**	**2.27e-13** ± **0.00e+0**	**2.27e-13** ± **0.00e+0**	**2.27e-13** ± **0.00e+0**
f_2	5.56e+6 ± 1.40e+6	**1.19e+5** ± **4.28e+4**	**1.47e+5** ± **4.87e+4**	**1.43e+5** ± **4.10e+4**	1.53e+5 ± 3.13e+4	**1.36e+5** ± **3.05e+4**
f_3	2.15e+7 ± 1.66e+7	3.62e+7 ± 2.32e+7	**3.33e+6** ± **2.91e+6**	**3.32e+6** ± **3.40e+6**	1.40e+7 ± 1.27e+7	1.19e+7 ± 9.27e+6
f_4	4.80e+4 ± 7.96e+3	3.08e-4 ± 2.51e-4	1.57e-4 ± 1.34e-4	**1.46e-5** ± **8.24e-6**	8.59e-5 ± 6.08e-5	**4.37e-5** ± **2.74e-5**
f_5	**4.21e-13** ± **8.88e-14**	5.46e-13 ± 0.00e+0	**3.90e-13** ± **8.14e-14**	**3.41e-13** ± **0.00e+0**	6.40e-13 ± 9.04e-14	5.57e-13 ± 8.98e-14
f_6	1.93e+2 ± 2.67e+1	**1.04e+2** ± **5.46e+1**	1.98e+2 ± 2.75e+1	2.05e+2 ± 2.82e+1	1.86e+2 ± 3.77e+1	1.91e+2 ± 2.88e+1
f_7	1.89e+1 ± 6.95e+0	5.61e+1 ± 1.17e+1	**7.62e+0** ± **2.41e+0**	**7.07e+0** ± **1.54e+0**	1.18e+1 ± 2.67e+0	1.16e+1 ± 4.24e+0
f_8	2.13e+1 ± 2.08e-2	**2.12e+1** ± **9.66e-2**	2.13e+1 ± 4.36e-2	2.13e+1 ± 4.81e-2	**2.12e+1** ± **4.67e-2**	**2.12e+1** ± **5.54e-2**
f_9	1.58e+2 ± 2.25e+0	1.38e+2 ± 2.68e+0	**1.32e+2** ± **2.90e+0**	**1.33e+2** ± **2.40e+0**	**1.32e+2** ± **2.79e+0**	**1.31e+2** ± **3.21e+0**
f_{10}	5.58e-02 ± 2.93e-2	2.85e-2 ± 1.68e-2	**1.58e-2** ± **1.19e-2**	**1.29e-2** ± **1.28e-2**	**1.28e-2** ± **7.84e-3**	**1.99e-2** ± **1.36e-2**
f_{11}	7.74e+1 ± 1.83e+1	**1.71e-13** ± **0.00e+0**	1.21e-3 ± 6.64e-4	1.48e-3 ± 7.96e-4	**1.89e-13** ± **0.00e+0**	**1.71e-13** ± **0.00e+0**
f_{12}	8.42e+2 ± 2.29e+1	1.52e+2 ± 1.74e+1	6.54e+1 ± 9.00e+0	5.27e+1 ± 1.01e+1	**4.46e+1** ± **4.59e+0**	**3.78e+1** ± **5.27e+0**
f_{13}	8.35e+2 ± 2.13e+1	4.01e+2 ± 5.31e+1	1.50e+2 ± 1.84e+1	1.36e+2 ± 1.75e+1	**1.30e+2** ± **1.57e+1**	**1.09e+2** ± **1.94e+1**
f_{14}	2.31e+4 ± 2.12e+3	**2.73e-2** ± **1.01e-2**	7.52e+1 ± 1.05e+1	9.20e+1 ± 1.56e+1	6.39e-2 ± 1.68e-2	6.20e-2 ± 1.40e-2
f_{15}	3.04e+4 ± 4.57 e+2	1.39e+4 ± 6.08e+2	1.56e+4 ± 6.68e+2	1.56e+4 ± 4.68e+2	**1.24e+4** ± **5.50e+2**	**1.24e+4** ± **5.43e+2**
f_{16}	3.93e+0 ± 2.18e-1	1.76e+0 ± 1.78e-1	1.89e+0 ± 1.60e-1	1.89e+0 ± 1.39e-1	**1.47e+0** ± **2.11e-1**	**1.54e+0** ± **1.47e-1**
f_{17}	6.66e+2 ± 5.57e+1	**1.02e+2** ± **0.00e+0**	1.03e+2 ± 3.36e-1	1.03e+2 ± 3.11e-1	**1.02e+2** ± **0.00e+0**	**1.02e+2** ± **0.00e+0**
f_{18}	9.21e+2 ± 2.54e+1	2.92e+2 ± 1.89e+1	2.80e+2 ± 1.46e+1	2.79e+2 ± 1.44e+1	**1.52e+2** ± **4.54e+0**	**1.50e+2** ± **2.70e+0**
f_{19}	6.74e+1 ± 3.79e+0	7.26e+0 ± 1.14e+0	7.31e+0 ± 2.77e-1	7.39e+0 ± 2.62e-1	**4.38e+0** ± **1.83e-1**	**4.34e+0** ± **2.14e-1**
f_{20}	5.00e+1 ± 0.00e+0	5.00e+1 ± 0.00e+0	**4.97e+1** ± **4.11e-1**	**4.97e+1** ± **3.92e-1**	**5.00e+1** ± **0.00e+0**	**4.97e+1** ± **4.76e-1**
f_{21}	**3.73e+2** ± **4.42e+1**	4.00e+2 ± 0.00e+0	**3.47e+2** ± **4.99e+1**	**3.47e+2** ± **4.99e+1**	**3.60e+2** ± **4.90e+1**	**3.73e+2** ± **4.42e+1**
f_{22}	2.16e+4 ± 2.60e+3	**1.78e+1** ± **3.20e+0**	1.05e+2 ± 1.74e+1	1.15e+2 ± 2.42e+1	**1.82e+1** ± **5.00e-1**	**1.84e+1** ± **7.12e-1**
f_{23}	3.06e+4 ± 4.89e+2	1.66e+4 ± 1.10e+3	1.48e+4 ± 7.01e+2	1.50e+4 ± 6.40e+2	**1.21e+4** ± **7.78e+2**	**1.23e+4** ± **7.69e+2**
f_{24}	2.55e+2 ± 1.31e+1	3.17e+2 ± 1.74e+1	**2.35e+2** ± **6.51e+0**	**2.32e+2** ± **6.19e+0**	2.45e+2 ± 5.98e+0	2.42e+2 ± 5.32e+0
f_{25}	**3.87e+2** ± **9.39e+0**	4.47e+2 ± 1.85e+1	**3.93e+2** ± **1.01e+1**	**3.89e+2** ± **1.21e+1**	4.09e+2 ± 1.32e+1	4.00e+2 ± 9.44e+0
f_{26}	3.69e+2 ± 1.26e+1	4.43e+2 ± 1.69e+1	**3.43e+2** ± **6.50e+0**	**3.38e+2** ± **5.38e+0**	3.56e+2 ± 5.63e+0	3.50e+2 ± 7.41e+0
f_{27}	1.07e+3 ± 1.94e+2	1.86e+3 ± 2.08e+2	**6.91e+2** ± **9.84e+1**	**6.38e+2** ± **7.71e+1**	8.36e+2 ± 9.32e+1	8.02e+2 ± 8.36e+1
f_{28}	3.41e+3 ± 1.04e+3	3.37e+3 ± 1.01e+3	**2.51e+3** ± **1.97e+1**	**2.51e+3** ± **1.61e+1**	**2.54e+3** ± **2.09e+1**	**3.09e+3** ± **9.30e+2**
Number of times among best	4	7	14	15	17	19
B/S/W	19/7/2	17/9/2	11/12/5	11/10/7	0/28/0	–