Impact of boundary control methods on bound-constrained optimization benchmarking

Impact of Boundary Control Methods on Bound-constrained Optimization Benchmarking

Tomas Kadavy, Adam Viktorin, Anezka Kazikova, Michal Pluhacek, Member, IEEE, and Roman Senkerik, Member, IEEE

Faculty of Applied Informatics, Tomas Bata University in Zlin, nam. T. G. Masaryka 5555, Zlin 76001, Czech Republic {kadavy, aviktorin, kazikova, pluhacek, senkerik} @utb.cz

Abstract

Benchmarking various metaheuristics and their new enhancements, strategies, and adaptation mechanisms has become standard in computational intelligence research. Recently, many challenges and issues regarding fair comparisons and recommendations towards good practices for benchmarking of metaheuristic algorithms, have been identified. This paper is aimed at an important issues in metaheuristics design and benchmarking, which are boundary strategies or boundary control methods (BCM). This work aims to investigate whether the choice of a BCM could significantly influence the performance of competitive algorithms. The experiments encompass the top three performing algorithms from IEEE CEC competitions 2017 and 2020 with six different boundary control methods. We provide extensive statistical analysis and rankings resulting in conclusions and recommendations for metaheuristics researchers and possibly also for the future direction of benchmark definitions. We conclude that the BCM should be considered another vital metaheuristics input variable for unambiguous reproducibility of results in benchmarking and for a better understanding of population dynamics, since the BCM setting could impact the optimization method performance.

Index Terms—Evolutionary computation, computational intelligence, performance evaluation, benchmark testing, boundary control method, optimization.

I. Introduction

In recent decades, metaheuristic algorithms have become popular and frequently used tools for solving optimization tasks of various levels of complexity in both real and discrete domains. Ongoing research in metaheuristic [**1**] is undoubtedly focused on expanding the theories, hybridizations [**2**], hyperparameters tuning [**3**], implementing new learning and adaptive mechanisms, and of course, benchmarking [**4**].

Any newly introduced improvements or new strategies of metaheuristic algorithms typically prove their efficiency and robustness on a set of test problems with various characteristics. Popular official benchmarking testbeds are IEEE CEC benchmarks[1] [**5**] and COCO[2] platform [**6**] BBOB testbed. The results of benchmarking competitions impact the design, comparisons, and modifications direction of metaheuristic algorithms in the next few years.

Recently, challenges and questions have been raised regarding good practices for benchmarking of metaheuristic algorithms [**7**], and fair comparisons with more in-depth insights into statistics [**8**].

[1] *https://github.com/P-N-Suganthan*

[2] *https://coco.gforge.inria.fr/*

Best practices in benchmarking represent a significant open issue nowadays when metaheuristic algorithms are used for increasingly more complex optimization problems, and it is beneficial to improve the comprehensibility of the results and resulting recommendations. That is why researchers, independently and with the support of an IEEE professional organization, have recently set up a benchmarking network[3] and taskforce[4].

This paper aims at an important matter in metaheuristics design and benchmarking, which are the boundary control methods (BCM). We provide extensive analysis resulting in conclusions and recommendations for metaheuristics researchers and possibly also for the future direction of benchmark testbeds profiling.

Regardless of the origin of the optimization task, one of the common attributes is that the optimized parameters are subject to certain limits (bounds). The specified parameter bounds are often caused by real-world limitations or the test function(s) definition. Because of the inherent presence of randomness in metaheuristic algorithms, a trial solution could arise outside of the set parameter boundaries. Such a situation may represent a significant hurdle for solving a particular optimization task. The straightforward approach to handle box constraints lies in the checking of each newly generated solution if such a solution still meets the criteria of parameter bounds. In case that the newly created parameters are located outside the space of feasible solutions, specific corrections have to be made.

The motivation for this research is based on the findings in several of the authors' papers [**9**], [**10**], [**11**] (details are given later in this section), where the influence of BCMs on the performance of selected basic versions of metaheuristic algorithms was examined. Thus, a research question arose as to whether the choice of a BCM could significantly influence other competitive algorithms, especially the CEC competition winners. Furthermore, it was motivating to find out whether just changing the BCM can help achieve even better results for the top three performing algorithms from a given year of the competition, possibly changing their final order. The results presented here showed that the impact on the empirical performance of the top 3 algorithms from CEC17 is affected by choice of BCMs.

It is our intention for the boundary control methods to be considered as one of the hyperparameters of metaheuristics, which must not only be carefully optimized and selected but must also be included in the description of the algorithm for unambiguous reproducibility of benchmarking results and a better understanding of population dynamics of metaheuristics.

The problem of BCMs is covered in a number of research publications with different levels of problem coverage. For the Particle Swarm Optimization (PSO) algorithms, the experimental analysis of bound handling techniques written by Helwig, Branke, and Mostaghim in 2012 [**12**] compare several BCMs used for the PSO algorithm and concluded that such methods can have a major impact on the algorithm performance and may introduce a significant search bias. The comparison also took into account various aspects of PSO, such as the velocity of individual particles and the variables of the best position. Oldewage, Engelbrecht, and Cleghorn published a similar extensive study of BCMs aimed at PSO algorithm in 2018 [**13**]. The study concluded that the bestperforming method was hyperbolic (the method uses one of the main characteristics of PSO, the velocity vector), which is, however, limited only to PSO. Another, slightly less detailed, research done by Clerc in 2006 [**14**] represents the BCMs as "confinements" in PSO and describes and tests several of them on a limited data set. A similar study, but with a different set of compared methods was done in 2004 by Zhang et al. [**15**].

[3] *https://sites.google.com/view/benchmarking-network/home*

[4] *https://cmte.ieee.org/cis-benchmarking/*

Michalewicz and Koziel published the comprehensive study on parameter bounds, mixed with constrained numerical optimization for Genetic Algorithms (GA) in [**16**], [**17**]. The paper by Mostaghim et al. in 2006 [**18**] is focused on a multiobjective version of PSO, and although the main focus is devoted to objective function constraints, part of the work summarizes the BCMs of a previous work, which was focused primarily on objective function constraints. Four boundary handling techniques were also discussed in the tutorial paper for the Covariance Matrix Adaptation Evolution Strategy algorithm (CMA-ES) by Hansen [**19**]. However, the influence of these methods on the performance of CMA-ES was not part of the tutorial paper. A recent study [**20**] investigated the BCM for CMA-ES more thoroughly. The article by Ronkkonen et al. [**21**] concerning the DE algorithm describes a BCM where the trial solution is reflected from the bound by the amount of the violation. A similar technique is used in papers by Brest et al. [**22**], [**23**] by Price and Storn, in [**24**] by Guo et al., and in [**25**] by Zhang et al. More detailed work on a structural bias (mainly caused by boundary constraints) with detailed results and discussion was done by Caraffini et al. in [**26**]. A detailed study about BCM for DE can be found in the recent paper [**27**].

Obtained results and suggestions from the collected works imply that the boundary control methods might have a direct impact on the overall performance of a metaheuristic algorithm; however, many newly introduced algorithms, tutorials, or overviews omitted or neglected this fact. Just to mention a few: a tutorial for PSO [**28**], general paper for Evolutionary Algorithms (EA) [**29**], articles focused on the Firefly Algorithm (FA) [**30**], [**31**], publication on the Cuckoo Algorithm [**32**], or the paper that introduced new mechanics for Differential Evolution (DE) [**33**].

The following quote comes from an article on experimental analysis of BCMs in Particle Swarm Optimization by Helwig et al. [**12**]:

"As was shown, the bound handling technique has a huge impact on the performance of PSO, especially if the number of dimensions of the search space is high, as this dramatically increases the probability of a particle leaving the feasible area."

Alongside the researched articles, the authors own previous work also confirms the described general conclusion. A study published in 2017 [**9**] concluded that a selection of used BCMs affects the performance of an algorithm. The study compared Clipping, Random, Periodic, and Soft methods on the generic version of PSO and the more advanced variant called Attractive and Repulsive PSO (ARPSO) [**34**]. A paper from 2018 [**10**] compared Clipping, Random, Reflection, and Periodic methods on the FireFly Algorithm (FA) and on a hybrid of FA and PSO, called FFPSO [**35**]. The experiments were performed on CEC 2017 benchmark set [**36**]. A recent study published in 2019 [**11**] examined the influence of BCM on SOMA All-To-One and All-To-All. The study compared Clipping, Random, Reflection, and Periodic methods on the CEC 2017 benchmark set and concluded that for both tested versions of SOMA, the BCMs Random and Periodic achieved better results than other two BCMs.

To conclude the introduction, the motivation behind the paper is to establish if the BCM can influence the algorithm performance from the competition results point of view. Thus, raise awareness about the need for careful selection of the BCM, similar to other hyperparameters of the metaheuristic algorithms. The presented results confirm that ill-selected BCM can negatively influence the algorithm's overall performance.

II. Boundary Control Methods

This section contains summarized list of commonly used boundary control methods (BCM). The list consists of strategies that can be applied to a generic metaheuristic algorithm; therefore, it is not a complete overview of all existing strategies. The selected BCMs are frequently used among the algorithms submitted for studied competitions (CEC17 and CEC20).

The following list contains mathematical representations of BCMs; hence, the recapitulation and meaning of the variables used in individual equations are included here. One individual solution $X = \{x_1, x_2, \cdots, x_D\}$ is a vector of real-valued parameters of length D, which stands for the dimensionality of the search space. Each parameter x_j (dimension) has defined bounds, which delimit the space of feasible solutions. The parameter bounds are defined as $X_J^L \leq x_j \leq X_J^H$, where X_J^L is the lower bound and x_J^H is the upper bound for j-th dimension.

1) *Clipping:* The first listed method is rather simple in principle, as well as quite easy to implement and probably often one of the first choices. Each individual solution x cannot cross the given boundaries in any dimension; the individuals are instead ''clipped" to the given parameter bounds. The equation describing the clipping method is given in (1).

$$x_{i,j}^{k+1} = \begin{cases} x_j^H & , \text{if } \left(x_{i,j}^k > x_j^H\right) \\ x_j^L & , \text{if } \left(x_{i,j}^k < x_j^L\right) \\ x_{i,j}^k & , \text{otherwise} \end{cases} \qquad (1)$$

TABLE III: Friedman ranks for CEC17. The values in each BCM column represent the Friedman rank in a particular row; the lower the value, the better rank of the algorithm. The p-values are accompanied by the symbol representing different significance levels: $*$ = 0.1, †= 0.05, $**$ = 0.01, $***$ = 0.001. The last column CD stands for Nemenyi Critical Difference -if two BCM ranks differ more than CD value, they are significantly different.

		Default	Clipping	Random	Periodic	Reflection	Halving	p-value	CD
EBOwithCMAR	10D	3.48E+00	3.67E+00	3.41E+00	3.12E+00	3.43E+00	3.88E+00	6.17E–01	1.40E+00
	30D	3.43E+00	3.64E+00	3.52E+00	3.47E+00	3.71E+00	3.24E+00	9.39E–01	
	50D	3.45E+00	3.55E+00	3.62E+00	3.38E+00	3.55E+00	3.45E+00	9.97E–01	
	100D	3.64E+00	4.09E+00	3.43E+00	4.16E+00	2.53E+00	3.16E+00	5.83E–03**	
	Mean	**3.50E+00**	**3.74E+00**	**3.50E+00**	**3.53E+00**	**3.31E+00**	**3.43E+00**		
jSO	10D	Halving	3.64E+00	2.81E+00	2.88E+00	2.62E+00	3.05E+00	4.72E–02†	1.13E+00
	30D		3.22E+00	2.84E+00	3.16E+00	2.97E+00	2.81E+00	7.73E–01	
	50D		3.31E+00	2.66E+00	3.38E+00	2.76E+00	2.90E+00	2.18E–01	
	100D		3.07E+00	2.83E+00	3.69E+00	2.31E+00	3.10E+00	1.52E–02†	
	Mean		**3.31E+00**	**2.78E+00**	**3.28E+00**	**2.66E+00**	**2.97E+00**		
LSHADE-cnEpSin	10D	Halving	3.83E+00	2.38E+00	2.17E+00	3.28E+00	3.34E+00	9.14E–07***	1.13E+00
	30D		3.48E+00	2.59E+00	2.69E+00	3.21E+00	3.03E+00	1.17E–01	
	50D		3.66E+00	2.62E+00	2.55E+00	3.10E+00	3.07E+00	3.40E–02†	
	100D		3.41E+00	2.62E+00	2.97E+00	2.97E+00	3.03E+00	3.94E–01	
	Mean		**3.60E+00**	**2.55E+00**	**2.59E+00**	**3.14E+00**	**3.12E+00**		

TABLE IV: Friedman ranks for CEC20. The values in each BCM column represent the Friedman rank in a particular row; the lower the value, the better rank of the algorithm. The p-values are accompanied by the symbol representing different significance levels: * = 0.1, †= 0.05, ** = 0.01, *** = 0.001. The last column CD stands for Nemenyi Critical Difference -if two BCM ranks differ more than CD value, they are significantly different.

		Default	Clipping	Random	Periodic	Reflection	Halving	p-value	CD
IMODE	5D	3.25E+00	4.05E+00	3.45E+00	3.75E+00	3.40E+00	3.10E+00	4.60E–01	2.38E+00
	10D	3.40E+00	3.20E+00	3.60E+00	3.70E+00	3.10E+00	4.00E+00	8.63E–01	
	15D	3.95E+00	3.65E+00	2.85E+00	3.35E+00	3.55E+00	3.65E+00	7.44E–01	
	20D	3.90E+00	3.60E+00	3.40E+00	3.90E+00	2.80E+00	3.40E+00	7.19E–01	
	Mean	**3.63E+00**	**3.62E+00**	**3.32E+00**	**3.68E+00**	**3.21E+00**	**3.54E+00**		
AGSK	5D	Halving	3.60E+00	3.10E+00	3.00E+00	3.15E+00	2.15E+00	9.27E–02*	1.93E+00
	10D		3.90E+00	2.30E+00	2.80E+00	2.70E+00	3.30E+00	1.45E–01	
	15D		3.45E+00	2.55E+00	3.25E+00	2.90E+00	2.85E+00	4.84E–01	
	20D		3.80E+00	2.45E+00	3.35E+00	2.55E+00	2.85E+00	1.27E–01	
	Mean		**3.69E+00**	**2.60E+00**	**3.10E+00**	**2.82E+00**	**2.79E+00**		
j2020	5D	Periodic	3.70E+00	2.70E+00	2.90E+00	3.00E+00	2.70E+00	5.11E–01	1.93E+00
	10D		3.70E+00	1.70E+00	3.30E+00	2.90E+00	3.40E+00	1.98E–02†	
	15D		3.80E+00	2.20E+00	3.45E+00	2.65E+00	2.90E+00	9.56E–02*	
	20D		3.70E+00	2.80E+00	3.35E+00	2.70E+00	2.45E+00	3.32E–01	
	Mean		**3.73E+00**	**2.35E+00**	**3.25E+00**	**2.81E+00**	**2.86E+00**		

The $X_{i,J}^{k+1}$ is the i-th individual in j-th dimension in $k + 1$ calculation step, and the pair X_J^H and X_J^L represents the parameter bounds, maximum and minimum respectively.

2) Random: If a trial solution violates the boundary in any dimension, the position for this individual in a particular dimension is reinitialized inside the lower and upper bounds (with a pseudo-random number generator using uniform distribution - U). Again, this technique is simple to implement, as the equation (2) shows.

$$x_{i,j}^{k+1} = \begin{cases} U\left(x_j^L, x_j^H\right) & \text{, if } \left(x_{i,j}^k > x_j^H \text{ OR } x_{i,j}^k < x_j^L\right) \\ x_{i,j}^k & \text{, otherwise} \end{cases} \quad (2)$$

3) Reflection: As the name suggests, the reflection method reflects the individual back to the feasible space of solution if it tries to violate the defined borders. This technique resembles the reflection characteristic of a mirror. The correction of a position of an individual in the violated dimension is computed as (3).

$$x_{i,j}^{k+1} = \begin{cases} x_j^H - \left(x_{i,j}^k - x_j^H\right) & \text{, if } \left(x_{i,j}^k > x_j^H\right) \\ x_j^L + \left(x_j^L - x_{i,j}^k\right) & \text{, if } \left(x_{i,j}^k < x_j^L\right) \\ x_{i,j}^k & \text{, otherwise} \end{cases} \quad (3)$$

4) Periodic: This possible solution to prevent the infeasibility takes advantages of an infinite space of solution (infinite copies of the optimized hyper-space). This method involves only mapping the individual back to the space of available solutions using the modulo function.

$$x_{i,j}^{k+1} = x_j^L + \left(\left(x_{i,j}^k - x_j^H\right) \text{ MOD } \left(x_j^H - x_j^L\right)\right) \quad (4)$$

5) Halving the Distance: The principle of this method is to halve the distance between the original position and the crossed bound. The implementation is slightly more complicated than previous techniques because the algorithm must keep track of the starting position of an individual $X_{i,J}^{k-1}$.

$$x_{i,j}^{k+1} = \begin{cases} x_{i,j}^{k-1} + \left(x_j^H - x_{i,j}^{k-1}\right)/2 & ,\text{if } \left(x_{i,j}^k > x_j^H\right) \\ x_j^L + \left(x_{i,j}^{k-1} - x_j^L\right)/2 & ,\text{if } \left(x_{i,j}^k < x_j^L\right) \\ x_{i,j}^k & ,\text{otherwise} \end{cases} \quad (5)$$

TABLE I: CEC17 - Algorithm overview

Algorithm	BCM	Mentioned
EBOwithCMAR [41]	Halving, Clipping	No
LSHADE-cnEpSin [42]	Halving	No
jSO [43]	Halving	Partially
DES [44]	Penalization	Yes
DYYPO [45]	–	No
IDEbestNsize [46]	Reflection	Partially
LSHADE_SPACMA [47]	–	No
MM_OED [48]	–	No
MOS-CEC2013 [49]	–	No
MOS-SOCO2011 [49]	–	No
PPSO [50]	Random bounce	Yes
RB-IPOP-CMA-ES [51]	–	No
TLBO-FL [52]	Clipping	Yes

III. Experiment setup

Since 2005 [**37**], a new benchmark set for single-objective optimization for continuous problem domain as a special session in IEEE Congress on Evolutionary Computation (CEC) is announced regularly. The composition of included test functions is periodically updated over the years. The series of CEC benchmark, therefore, represents a substantial pool of the most suitable test functions. Recent CEC benchmark test suites [**5**] encompass four groups of test functions: unimodal, multimodal, hybrid, and composition functions. An advantageous feature of the CEC benchmark is the fact that all incorporated test functions are defined with equal and static (same values across all dimensions) search range for all parameters. The original implementation also supports a shift of the global optimum and rotation of each function.

One exception among the CEC benchmarks is the CEC19 [**38**] , which consists of 10 test functions, each of a different search range of parameters.

This paper is focused on the three top ranking participants of two recent benchmark competitions: CEC17 [**36**], and CEC20 [**39**]. The goals for both testbeds are to:

- determine which BCM was used by the three winning algorithms,
- examine if there is a better choice of a BCM for a particular algorithm,
- if the algorithms used a different BCM, could it have changed the final order?

The description of each benchmark is summarized in the following subsections alongside the descriptions of the three top ranking algorithms of each benchmark.

A. CEC17

The testbed CEC17 published in 2016 [**36**] encompasses 30 test functions for dimension sizes of 10, 30, 50, and 100. The following subsections briefly describe the top three performing algorithms according to the official results [**40**]. **Table I** contains a list of all participants, including the used boundary control method, and if the BCM was mentioned in the accompanying paper.

1) EBOwithCMAR: An algorithm originally proposed for the CEC17 benchmark and successfully obtained the first position among 11 competitors. The hybrid algorithm is based on the Effective Butterfly Optimizer (EBO) and Covariance Matrix Adapted Retreat phase (CMAR), which improves the local search capability of EBO. The paper [**41**] does not specify any used BCM; however the analysis of the code of the algorithm showed that EBOwithCMAR uses two BCMs, Halving for EBO and Clipping for CMAR.

TABLE II: CEC20 - Algorithm overview

Algorithm	BCM	Mentioned
IMODE [56]	Random, Halving	No
AGSK [57]	Halving	No
j2020 [58]	Periodic	Yes
Cssin [59]	–	No
MP-EEH [60]	Clamping	Yes
RASP-SHADE [61]	Halving	Yes
DISH-XX [62]	Halving	No
jDE100e [63]	–	No
OLSHADE [64]	Clipping, Halving	Yes
mpmL-SHADE [65]	Halving	Yes
SOMA-CL [66]	Random	No

2) jSO: The jSO [**43**] represents an improved variant of the iL-SHADE algorithm [**53**] and ranked in second place. The improvement lies predominantly in the new version of the mutation strategy. The jSO uses Halving BCM, which is referred in the paper as a “repeat mechanism” without any detailed description or citation.

3) LSHADE-cnEpSin: The third algorithm used in this study represents extension to the LSHADE-EpSin [**54**], which was ranked as the joint winner in competition IEEE CEC 2016. The enhancement lies in the ensemble of sinusoidal approaches and covariance matrix learning for the crossover operator. The LSHADE-cnEpSin [**42**] ranked third in the CEC17 competition and uses Halving BCM, which is unfortunately not mentioned in the paper by the authors.

B. CEC20

The CEC20 benchmark [39] introduced in 2019 includes 10 test functions for dimension sizes of 5, 10, 15, and 20. Again, the following subsections briefly describe the top three performing algorithms according to the official results [**55**], and Table II contains a list of all participants, what boundary control method they used, and if the BCM was mentioned in the related paper.

1) IMODE: This Differential Evolution (DE) based algorithm ranked as the winner in the CEC20 competition. IMODE [56] benefits from multiple differential evolution operators, with more emphasis placed on the best-performing operator. The algorithm employ two BCMs: Clipping and Halving, selected randomly each time it is used. Unfortunately, BCMs are not mentioned in the paper.

2) AGSK: The AGSK is the second-best performing algorithm in the CEC20 competition. AGSK [**57**] is the enhanced version of the Gaining Sharing Knowledge-based algorithm (GSK) [**67**], which uses adaptive settings to its control parameters. The algorithm utilizes Halving BCM; however, this is not specified by the authors.

3) j2020: The algorithm [58] ranked third place in the competition and it is based on the two self-adaptive DE algorithms jDE [**68**], and jDE100 [69]. The used BCM is Periodic, which is mentioned in the paper.

The source codes of the three top ranking algorithms of both CEC competitions were provided by the organizer of the competitions via GitHub[5].

It was found out that algorithms implemented in the programming platform Matlab and belonging mainly to the DE family of algorithms used the same, or similar, library for BCM, which contains the implementation of the Halving BCM. This probably encourages the researchers to use it, as it is already prepared.

IV. Results

This section presents the results of both experiments performed for benchmarks CEC17 and CEC20. Each test scenario used a different number of independent runs as defined by the used benchmark. CEC 17 testbed defines 51 independent runs, while CEC20 testbed requires 30 independent runs.

This section is divided into three subsections. Each subsection describes one used methodology: Friedman rank test, CEC scoring system, and selection of the best performing BCM variant for the algorithms.

A. Friedman rank test

As a first step, each algorithm was tested and evaluated while using different BCMs. The evaluation was performed by Friedman rank test [**70**] and the results are presented in **Table III** for CEC17 and Table IV for CEC20. The values in cells are the rankings for each algorithm for particular dimension size. The columns indicate the tested BCM. If the algorithm used different BCM than the selected (Clipping, Random, Periodic, Reflection, Halving), the column Default was used. Otherwise, this column states the name of the used BCM of the algorithm. The last column contains p-values of the Friedman rank test. The tested significance levels are 0.1, 0.05, 0.01, and 0.001. Each level corresponds to a certain symbol: *, †, **, and *** respectively. Therefore, a symbol represents the significance level of the result. The last row of each algorithm also contains the mean rank (given in bold) across the dimension sizes for a particular BCM. The last column CD stands for Nemenyi Critical Difference - if the difference between a pair of BCMs' ranks is higher than CD value, they are significantly different. For example, in Table III, the Periodic BCM in LSHADE-cnEpSin in 10D is significantly different (better according to rank) from Clipping BCM. But the same Periodic BCM is not significantly different from Random BCM.

Table III contains higher number of results with statistically significant difference than **Table IV**. The likely reasons are that: the CEC20 benchmark contains only 10 test functions, lower dimensionality might cause a lower number of BCM use (see [**12**] - low dimensionality leads to a lower probability of creation of an infeasible trial solution), and the top ranking algorithms in the competition are robust and similar in the performance.

B. CEC scoring systém

The second step was to use the scoring system employed by the CEC competitions. The motivation for this test was to determine if a change of BCM may cause a change in the order of the algorithms. The CEC scoring system is in detail provided in the technical reports accompanying the CEC benchmarks [**36**], [**39**].

The algorithms are sorted by the final score value, and the higher this Score is, the better the performance of the algorithm. This Score is a sum of two partial scores, Score 1 and Score 2. Score 2 is based on the weighted rank values, and Score 1 is computed from the normalized error values for CEC20 or mean (not normalized) error values for CEC17. Score 1 is computed using equations (6) and (7).

$$SE = 0.1 \cdot \sum_{i=1}^{29} ef_{10D} + 0.2 \cdot \sum_{i=1}^{29} ef_{30D} + 0.3 \cdot \sum_{i=1}^{29} ef_{50D} + 0.4 \cdot \sum_{i=1}^{29} ef_{100D} \quad (6)$$

$$Score1 = \left(1 - \frac{SE - SE_{min}}{SE}\right) \cdot 50 \quad (7)$$

Where ef is the mean error value for certain dimension size and SE_{min} is the minimal sum of errors among all algorithms. The Score 2 is then computed based on equations equations (8) and (9).

$$SR = 0.1 \cdot \sum_{i=1}^{29} rank_{10D} + 0.2 \cdot \sum_{i=1}^{29} rank_{30D} + 0.3 \cdot \sum_{i=1}^{29} rank_{50D} + 0.4 \cdot \sum_{i=1}^{29} rank_{100D} \quad (8)$$

$$Score2 = \left(1 - \frac{SR - SR_{min}}{SR}\right) \cdot 50 \quad (9)$$

The final score is then defined as (10).

$$Score = Score1 + Score2 \qquad (10)$$

The equations (6) - (10) describes the computation of the score for CEC17. The CEC20 score is computed by the same equations but uses a different dimension sizes.

Tables V(a) - **V**(f) contain the Score and rank of BCMs used for a particular algorithm. The default BCM is in bold. From the given results, not once did the default BCM ranked as the best performing variant; therefore, potentially better results for the algorithm may be achieved using the BCM with the highest Score. The parentheses under the score display percentual contribution of each dimensional setting to the score. The significant disproportion in values of Score 1 for dimension size 50 for CEC17 was caused by the last test function f30, Since this disproportion is observed across all tested algorithms and their BCMs, the obtained results are still comparable, however further investigation is needed to find the cause of such behavior.

Friedman ranks suggest that higher dimension size has a more significant impact on the final score, as can be seen in **Table V**.

TABLE V: Score and rank of BCMs used for a particular algorithm. The rank is based on the final Score, which is a sum of partial scores 1 and 2. The default BCM of the algorithm is in bold. The parentheses under the score display percentual contribution of each dimensional setting to the score (CEC17 - {10D, 30D, 50D, 100D}, CEC20 - {5D, 10D, 15D, 20D}).

(a) CEC17 – EBOwithCMAR

Rank	BCM	Score 1	Score 2	Score
1	Reflection	4.81E+01	5.00E+01	9.81E+01
		(0.4, 1.1, 92.6, 5.9)	(10.8, 23.4, 33.7, 32.)	
2	Random	4.98E+01	4.52E+01	9.49E+01
		(0.4, 1.1, 92.1, 6.3)	(9.7, 20.1, 31., 39.2)	
3	Halving	4.71E+01	4.75E+01	9.46E+01
		(0.4, 1.1, 92.7, 5.9)	(11.6, 19.5, 31., 37.9)	
4	Periodic	5.00E+01	4.30E+01	9.30E+01
		(0.4, 1.2, 92., 6.4)	(8.5, 18.8, 27.5, 45.2)	
5	**Default**	**4.78E+01**	**4.49E+01**	**9.27E+01**
		(0.4, 1.1, 92.4, 6.)	**(9.9, 19.5, 29.4, 41.3)**	
6	Clipping	4.62E+01	4.17E+01	8.79E+01
		(4.8, 1., 88.4, 5.8)	(9.7, 19.2, 28.1, 43.1)	

(b) CEC17 – jSO

Rank	BCM	Score 1	Score 2	Score
1	Reflection	5.00E+01	5.00E+01	10.0E+01
		(0.4, 1.1, 90.5, 7.9)	(10.1, 22.8, 31.7, 35.4)	
2	Random	5.00E+01	4.69E+01	9.69E+01
		(0.4, 1.2, 89.9, 8.5)	(10.1, 20.5, 28.7, 40.7)	
3	**Halving**	**4.87E+01**	**4.38E+01**	**9.25E+01**
		(0.4, 1.1, 90.5, 8.)	**(10.2, 18.9, 29.2, 41.7)**	
4	Clipping	4.80E+01	4.04E+01	8.83E+01
		(2.7, 1.1, 88.7, 7.5)	(11.3, 20., 30.8, 38.)	
5	Periodic	4.98E+01	3.82E+01	8.80E+01
		(0.4, 1.2, 89.8, 8.6)	(8.4, 18.5, 29.7, 43.3)	

(c) CEC17 – LSHADE-cnEpSin

Rank	BCM	Score 1	Score 2	Score
1	Random	5.00E+01	5.00E+01	1.00E+02
		(0.5, 1.2, 91.8, 6.5)	(9.2, 20., 30.4, 40.5)	
2	Periodic	4.92E+01	4.78E+01	9.71E+01
		(0.5, 1.2, 91.5, 6.8)	(8., 19.9, 28.3, 43.8)	
3	Reflection	4.60E+01	4.20E+01	8.79E+01
		(0.5, 1.1, 92.5, 5.9)	(10.6, 20.8, 30.2, 38.4)	
4	**Halving**	**4.44E+01**	**4.21E+01**	**8.65E+01**
		(3.7, 1.1, 89.7, 5.5)	**(10.9, 19.7, 29.9, 39.5)**	
5	Clipping	4.43E+01	3.66E+01	8.08E+01
		(0.4, 1.1, 92.7, 5.8)	(10.8, 19.7, 31., 38.6)	

(d) CEC20 – IMODE

Rank	BCM	Score 1	Score 2	Score
1	Reflection	5.00E+01	5.00E+01	1.00E+02
		(1.4, 10.9, 34.5, 53.2)	(10.8, 19.7, 33.9, 35.6)	
2	Halving	4.95E+01	4.41E+01	9.36E+01
		(2.4, 11.1, 32.4, 54.1)	(8.7, 22.4, 30.7, 38.1)	
3	Random	4.47E+01	4.79E+01	9.27E+01
		(4.4, 10.3, 32.7, 52.6)	(10.5, 22., 26.1, 41.5)	
4	Periodic	4.35E+01	4.27E+01	8.63E+01
		(5.5, 7.9, 39.5, 47.)	(10.2, 20.1, 27.3, 42.4)	
5	**Default**	**4.37E+01**	**4.19E+01**	**8.57E+01**
		(4.3, 15.9, 29.9, 49.8)	**(8.7, 18.1, 31.6, 41.6)**	
6	Clipping	4.14E+01	4.39E+01	8.53E+01
		(4.1, 8.9, 27.4, 59.6)	(11.3, 17.9, 30.6, 40.2)	

(e) CEC20 – AGSK

Rank	BCM	Score 1	Score 2	Score
1	Reflection	4.90E+01	4.58E+01	9.48E+01
		(0.4, 5., 29.2, 65.4)	(11.5, 19.7, 31.7, 37.2)	
2	**Halving**	**5.00E+01**	**4.38E+01**	**9.38E+01**
		(0.1, 5., 29.6, 65.3)	**(7.5, 23., 29.8, 39.7)**	
3	Random	4.36E+01	5.00E+01	9.36E+01
		(0.4, 4.7, 38., 56.9)	(12.3, 18.3, 30.4, 39.)	
4	Periodic	4.60E+01	3.96E+01	8.56E+01
		(0., 4.2, 31.9, 63.9)	(9.4, 17.6, 30.7, 42.2)	
5	Clipping	4.67E+01	3.40E+01	8.07E+01
		(0., 4.7, 30.1, 65.2)	(9.7, 21.1, 28., 41.1)	

(f) CEC20 – j2020

Rank	BCM	Score 1	Score 2	Score
1	Halving	5.00E+01	4.27E+01	9.27E+01
		(0.5, 0.5, 42.1, 56.8)	(9.6, 24.3, 31.1, 35.)	
2	Random	4.19E+01	5.00E+01	9.19E+01
		(0., 5., 35.2, 59.7)	(11.3, 14.2, 27.6, 46.9)	
3	Reflection	4.35E+01	4.34E+01	8.69E+01
		(0., 5.6, 43.1, 51.4)	(10.9, 21.1, 28.9, 39.2)	
4	**Periodic**	**3.62E+01**	**3.59E+01**	**7.22E+01**
		(0., 4.3, 33.1, 62.6)	**(8.7, 19.8, 31.1, 40.3)**	
5	Clipping	3.68E+01	3.20E+01	6.88E+01
		(0., 6., 14., 80.)	(9.9, 19.8, 30.6, 39.7)	

TABLE VI: CEC17 - Score - Default BCM

Rank	Algorithm	Score 1	Score 2	Score
1	EBOwithCMAR	5.00E+01	5.00E+01	1.00E+02
2	jSO	4.97E+01	4.33E+01	9.30E+01
3	LSHADE-cnEpSin	4.68E+01	4.47E+01	9.15E+01
4	LSHADE_SPACMA	4.64E+01	4.47E+01	9.11E+01
5	DES	4.59E+01	4.12E+01	8.71E+01
6	MM_OED	4.60E+01	3.62E+01	8.22E+01
7	IDEbestNsize	2.98E+01	2.63E+01	5.61E+01
8	MOS-CEC2013	1.89E+01	1.74E+01	3.63E+01
9	RB-IPOP-CMA-ES	3.79E+00	3.21E+01	3.59E+01
10	MOS-SOCO2011	1.11E+01	1.92E+01	3.03E+01
11	PPSO	3.93E+00	1.73E+01	2.12E+01
12	DYYPO	5.93E–01	1.71E+01	1.77E+01
13	TLBO-FL	2.87E–02	1.64E+01	1.64E+01

TABLE VII: CEC17 - Score – EBOwithCMAR

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	EBOwithCMAR	5.00E+01	5.00E+01	1.00E+02
2 (2)	jSO	4.96E+01	4.55E+01	9.51E+01
3 (4)	LSHADE_SPACMA	4.66E+01	4.83E+01	9.49E+01
4 (3)	LSHADE-cnEpSin	4.78E+01	4.25E+01	9.03E+01
5 (5)	DES	4.61E+01	4.23E+01	8.84E+01
6 (6)	MM_OED	4.62E+01	3.74E+01	8.35E+01
7 (7)	IDEbestNsize	3.00E+01	2.69E+01	5.69E+01
8 (9)	RB-IPOP-CMA-ES	3.81E+00	3.32E+01	3.70E+01
9 (8)	MOS-CEC2013	1.90E+01	1.78E+01	3.68E+01
10 (10)	MOS-SOCO2011	1.11E+01	1.97E+01	3.08E+01
11 (11)	PPSO	3.94E+00	1.77E+01	2.17E+01
12 (12)	DYYPO	5.96E–01	1.76E+01	1.82E+01
13 (13)	TLBO-FL	2.89E–02	1.68E+01	1.68E+01

However, the higher impact of higher dimension sizes is also implicit due to the weighting of dimension parts of the score computation in (6) and (8).

C. Selection of the BCM

The third and the last step was to implement the best performing BCM variant for the algorithms and check if the final order of the competition will be different. **Table VI** and **Table X** contain the Score and rank if the algorithms used their default BCMs. Unfortunately, the complete results of all competitors are available for the CEC17 benchmark only; therefore the **Table X** encompasses only the three top ranking algorithms of the CEC20 competition. The best BCM for each algorithm was selected according to the ranks in **Tables V**a -**V**f.

TABLE VIII: CEC17 - Score – jSO

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	EBOwithCMAR	4.89E+01	5.00E+01	9.89E+01
2 (2)	jSO	5.00E+01	4.76E+01	9.76E+01
3 (4)	LSHADE_SPACMA	4.60E+01	4.87E+01	9.47E+01
4 (3)	LSHADE-cnEpSin	4.71E+01	4.35E+01	9.06E+01
5 (5)	DES	4.55E+01	4.33E+01	8.88E+01
6 (6)	MM_OED	4.55E+01	3.81E+01	8.36E+01
7 (7)	IDEbestNsize	2.96E+01	2.76E+01	5.72E+01
8 (9)	RB-IPOP-CMA-ES	3.76E+00	3.38E+01	3.76E+01
9 (8)	MOS-CEC2013	1.88E+01	1.82E+01	3.70E+01
10 (10)	MOS-SOCO2011	1.10E+01	2.01E+01	3.11E+01
11 (11)	PPSO	3.89E+00	1.81E+01	2.20E+01
12 (12)	DYYPO	5.88E–01	1.79E+01	1.85E+01
13 (13)	TLBO-FL	2.84E–02	1.71E+01	1.72E+01

TABLE IX: CEC17 - Score - LSHADE-cnEpSin

Rank	Algorithm	Score 1	Score 2	Score
1 (3)	LSHADE-cnEpSin	5.00E+01	5.00E+01	1.00E+02
2 (1)	EBOwithCMAR	4.73E+01	4.92E+01	9.66E+01
3 (2)	jSO	4.73E+01	4.62E+01	9.35E+01
4 (4)	LSHADE_SPACMA	4.45E+01	4.81E+01	9.26E+01
5 (5)	DES	4.40E+01	4.34E+01	8.74E+01
6 (6)	MM_OED	4.40E+01	3.79E+01	8.19E+01
7 (7)	IDEbestNsize	2.86E+01	2.75E+01	5.61E+01
8 (9)	RB-IPOP-CMA-ES	3.63E+00	3.37E+01	3.73E+01
9 (8)	MOS-CEC2013	1.81E+01	1.83E+01	3.64E+01
10 (10)	MOS-SOCO2011	1.06E+01	2.02E+01	3.09E+01
11 (11)	PPSO	3.76E+00	1.82E+01	2.20E+01
12 (12)	DYYPO	5.68E–01	1.80E+01	1.86E+01
13 (13)	TLBO-FL	2.75E–02	1.72E+01	1.73E+01

TABLE X: CEC20 - Score - Default BCM

Rank	Algorithm	Score 1	Score 2	Score
1	j2020	5.00E+01	4.56E+01	9.56E+01
2	IMODE	2.14E+01	5.00E+01	7.14E+01
3	AGSK	2.31E+01	4.56E+01	6.88E+01

TABLE XI: CEC20 - Score – IMODE

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	j2020	5.00E+01	4.51E+01	9.51E+01
2 (2)	IMODE	2.34E+01	5.00E+01	7.34E+01
3 (3)	AGSK	2.30E+01	4.47E+01	6.77E+01

TABLE XII: CEC20 - Score – AGSK

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	j2020	5.00E+01	4.52E+01	9.52E+01
2 (2)	IMODE	2.14E+01	5.00E+01	7.14E+01
3 (3)	AGSK	2.25E+01	4.43E+01	6.68E+01

TABLE XIII: CEC20 - Score - j2020

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	j2020	5.00E+01	4.36E+01	9.36E+01
2 (2)	IMODE	2.86E+01	5.00E+01	7.86E+01
3 (3)	AGSK	3.06E+01	4.45E+01	7.51E+01

TABLE XIV: CEC17 – Score

Rank	Algorithm	Score 1	Score 2	Score
1 (3)	LSHADE-cnEpSin	5.00E+01	5.00E+01	1.00E+02
2 (1)	EBOwithCMAR	4.77E+01	5.00E+01	9.76E+01
3 (2)	jSO	4.84E+01	4.58E+01	9.42E+01
4 (4)	LSHADE_SPACMA	4.45E+01	4.78E+01	9.23E+01
5 (5)	DES	4.40E+01	4.30E+01	8.70E+01
6 (6)	MM_OED	4.40E+01	3.81E+01	8.21E+01
7 (7)	IDEbestNsize	2.86E+01	2.76E+01	5.62E+01
8 (9)	RB-IPOP-CMA-ES	3.63E+00	3.37E+01	3.73E+01
9 (8)	MOS-CEC2013	1.81E+01	1.82E+01	3.64E+01
10 (10)	MOS-SOCO2011	1.06E+01	2.02E+01	3.08E+01
11 (11)	PPSO	3.76E+00	1.82E+01	2.20E+01
12 (12)	DYYPO	5.68E–01	1.80E+01	1.86E+01
13 (13)	TLBO-FL	2.75E–02	1.72E+01	1.73E+01

TABLE XV: CEC20 – Score

Rank	Algorithm	Score 1	Score 2	Score
1 (1)	j2020	5.00E+01	4.19E+01	9.19E+01
2 (2)	IMODE	3.14E+01	5.00E+01	8.14E+01
3 (3)	AGSK	2.96E+01	4.53E+01	7.49E+01

For the CEC17, **tables VII**, **VIII**, and **IX** represent the situations when only one algorithm selects its best variant of the BCM. If the rank is changed against **Table VI**, the original rank is shown in parentheses. The most noticeable difference is in Table IX, where the LSHADE-cnEpSin obtained the first rank. **Table XIV** then contains the ranks accomplished if all three algorithms had used the best performing variant of the BCM, and again, the LSHADE-cnEpSin would have achieved the first position.

For the CEC20, the process is the same as for CEC17. The results are presented in **Table X** - **XV** and no change in the algorithms order was observed.

V. Conclusion

While the boundary control methods (BCM) are often an overlooked part of the experiment design in metaheuristics benchmarking, our work aims to highlight the importance of understanding the boundary control method as necessary input for results reproducibility.

Further, we bring to the attention the possibility of performance improvement by the use of alternate boundary control methods, as presented in the results of CEC17 benchmark participants, where:

- The LSHADE-cnEpSin algorithm would have won the CEC17 competition, if it did employ the random boundary control method.
- According to the scores as defined by the CEC17 benchmark, none of the three tested algorithms achieved best results with the original BCM. In other words, it was possible to improve the results of any of these algorithms by using a different BCM.
- Only five of the 12 participants reported on the employed BCM in the papers (albeit in two cases only partially, see **Table I**). Moreover, the results of the two best performing algorithms would be irreproducible without a detailed study of the source code.

The results of the second experiment, using the CEC20 benchmark seem to be less conclusive, as only three scenarios show statistically significant difference in performance (see **Table IV**). However, the scoring (**Tables V**(d), **V**(e) and **V**(f)) seems to indicate not only a difference in performance, but a possible benefit of using other than default BCM.

While the seemingly lower impact of BCM selection in the case of CEC20 might be related to the lower dimensionality of the problems (in accordance with the findings of Helwig et al. in [12]), more data and evaluation will be needed to confirm this in the future.

The authors of this paper would like to express their regret that their own proposed algorithms (DISH-XX and SOMA-CL) are among the algorithms with non-specified BCM at the CEC20 competition. Due to this unfortunate error, the number of algorithms with specified BCM in the CEC20 competition was five out of 11.

The examined algorithms and their results are available at A.I.Lab GitHub page [6].

To conclude, according to our findings, many competition entries lack the information about used BCM, leading to poor reproducibility of the results. Further, the designers of competitive metaheuristics should take advantage of the possible performance improvements, by moving their attention towards BCMs.

The importance of future research of BCMs lies in their universal applicability and impact on the whole field of metaheuristic optimizers in bound-constrained scenarios.

[6] *https://github.com/TBU-AILab/ResourceFiles_TEVC2021*

References

[1] W. Wong and C. I. Ming, “A review on metaheuristic algorithms: recent trends, benchmarking and applications,” in 2019 7th International Conference on Smart Computing & Communications (ICSCC). IEEE, 2019, pp. 1-5.

[2] G. R. Raidl, “A unified view on hybrid metaheuristics,” in International workshop on hybrid metaheuristics. Springer, 2006, pp. 1-12.

[3] A. Kazikova, M. Pluhacek, and R. Senkerik, “Why tuning the control parameters of metaheuristic algorithms is so important for fair comparison?” MENDEL, vol. 26, no. 2, pp. 9-16, Dec. 2020. [Online]. Available: **http://ib-b2b.test.infv.eu/index.php/mendel/article/view/120**

[4] —, “How does the number of objective function evaluations impact our understanding of metaheuristics behavior?” IEEE Access, vol. 9, pp. 44032-44048, 2021.

[5] A. Wagdy, A. A. Hadi, A. K. Mohamed, P. Agrawal, A. Kumar, and P. N. Suganthan, “Problem definitions and evaluation criteria for the cec 2021 special session and competition on single objective bound constrained numerical optimization.” Technical Report, Nanyang Technological University, Singapore, 2020.

[6] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tusar, and D. Brockhoff, “Coco: A platform for comparing continuous optimizers in a black-box setting,” Optimization Methods and Software, vol. 36, no. 1, pp. 114144, 2021.

[7] T. Bartz-Beielstein, C. Doerr, J. Bossek, S. Chandrasekaran, T. Eftimov, A. Fischbach, P. Kerschke, M. Lopez-Ibanez, K. M. Malan, J. H. Moore, B. Naujoks, P. Orzechowski, V. Volz, M. Wagner, and T. Weise, “Benchmarking in optimization: Best practice and open issues,” 2020.

[8] A. LaTorre, D. Molina, E. Osaba, J. Poyatos, J. Del Ser, and F. Herrera, “A prescription of methodological guidelines for comparing bio-inspired optimization algorithms,” Swarm and Evolutionary Computation, p. 100973, 2021.

[9] T. Kadavy, M. Pluhacek, A. Viktorin, and R. Senkerik, “Comparing strategies for search space boundaries violation in pso,” in International Conference on Artificial Intelligence and Soft Computing. Springer, 2017, pp. 655-664.

[10] —, “Boundary strategies for firefly algorithm analysed using cec’17 benchmark.” in ECMS, 2018, pp. 170-175.

[11] T. Kadavy, M. Pluhacek, R. Senkerik, and A. Viktorin, “Boundary strategies for self-organizing migrating algorithm analyzed using cec’17 benchmark,” in Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing. Springer, 2019, pp. 58-69.

[12] S. Helwig, J. Branke, and S. Mostaghim, “Experimental analysis of bound handling techniques in particle swarm optimization,” IEEE Transactions on Evolutionary computation, vol. 17, no. 2, pp. 259-271, 2012.

[13] E. T. Oldewage, A. P. Engelbrecht, and C. W. Cleghorn, “Boundary constraint handling techniques for particle swarm optimization in high dimensional problem spaces,” in International Conference on Swarm Intelligence. Springer, 2018, pp. 333-341.

[14] M. Clerc, “Confinements and biases in particle swarm optimisation,” 2006.

[15] W.-J. Zhang, X.-F. Xie, and D.-C. Bi, “Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space,” in Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753), vol. 2. IEEE, 2004, pp. 2307-2311.

[16] Z. Michalewicz and M. Schoenauer, “Evolutionary algorithms for constrained parameter optimization problems,” Evolutionary computation, vol. 4, no. 1, pp. 1-32, 1996.

[17] S. Koziel and Z. Michalewicz, “Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization,” Evolutionary computation, vol. 7, no. 1, pp. 19-44, 1999.

[18] M. Sanaz, H. Werner, and W. Anja, “Linear multi-objective particle swarm optimization,” in Stigmergic Optimization. Springer, 2006, pp. 209-238.

[19] N. Hansen, “The cma evolution strategy: A tutorial,” arXiv preprint arXiv:1604.00772, 2016.

[20] J. de Nobel, D. Vermetten, H. Wang, C. Doerr, and T. Back, Tuning as a Means of Assessing the Benefits of New Ideas in Interplay with Existing Algorithmic Modules. New York, NY, USA: Association for Computing Machinery, 2021, p. 1375-1384. [Online]. Available: **https://doi.org/10.1145/3449726.3463167**

[21] J. Ronkkonen, S. Kukkonen, and K. V. Price, “Real-parameter optimization with differential evolution,” in 2005 IEEE congress on evolutionary computation, vol. 1. IEEE, 2005, pp. 506-513.

[22] J. Brest, V. Zumer, and M. S. Maucec, “Self-adaptive differential evolution algorithm in constrained real-parameter optimization,” in 2006 IEEE international conference on evolutionary computation. IEEE, 2006, pp. 215-222.

[23] K. Price, R. M. Storn, and J. A. Lampinen, Differential evolution: a practical approach to global optimization. Springer Science & Business Media, 2006.

[24] S.-M. Guo, J. S.-H. Tsai, C.-C. Yang, and P.-H. Hsu, “A self-optimization approach for l-shade incorporated with eigenvector-based crossover and successful-parent-selecting framework on cec 2015 benchmark set,” in 2015 IEEE congress on evolutionary computation (CEC). IEEE, 2015, pp. 1003-1010.

[25] J. Zhang and A. C. Sanderson, “Jade: adaptive differential evolution with optional external archive,” IEEE Transactions on evolutionary computation, vol. 13, no. 5, pp. 945-958, 2009.

[26] F. Caraffini, A. V. Kononova, and D. Corne, “Infeasibility and structural bias in differential evolution,” Information Sciences, vol. 496, pp. 161179, 2019.

[27] R. Boks, A. V. Kononova, and H. Wang, Quantifying the Impact of Boundary Constraint Handling Methods on Differential Evolution. New York, NY, USA: Association for Computing Machinery, 2021, p. 1199-1207. [Online]. Available: **https://doi.org/10.1145/3449726.3463214**

[28] F. Marini and B. Walczak, “Particle swarm optimization (pso). a tutorial,” Chemometrics and Intelligent Laboratory Systems, vol. 149, pp. 153-165, 2015.

[29] A. E. Eiben, J. E. Smith et al., Introduction to evolutionary computing. Springer, 2003.

[30] X.-S. Yang, “Firefly algorithm, stochastic test functions and design optimisation,” International journal of bio-inspired computation, vol. 2, no. 2, pp. 78-84, 2010.

[31] —— , “Firefly algorithms for multimodal optimization,” in International symposium on stochastic algorithms. Springer, 2009, pp. 169-178.

[32] X.-S. Yang and S. Deb, “Engineering optimisation by cuckoo search,” International Journal of Mathematical Modelling and Numerical Optimisation, vol. 1, no. 4, pp. 330-343, 2010.

[33] R. Mallipeddi, P. N. Suganthan, Q.-K. Pan, and M. F. Tasgetiren, “Differential evolution algorithm with ensemble of parameters and mutation strategies,” Applied soft computing, vol. 11, no. 2, pp. 16791696, 2011.

[34] J. Riget and J. S. Vesterstrpm, “A diversity-guided particle swarm optimizer-the arpso,” Dept. Comput. Sci., Univ. of Aarhus, Aarhus, Denmark, Tech. Rep, vol. 2, p. 2002, 2002.

[35] P. Kora and K. S. R. Krishna, “Hybrid firefly and particle swarm optimization algorithm for the detection of bundle branch block,” International Journal of the Cardiovascular Academy, vol. 2, no. 1, pp. 44-48, 2016.

[36] N. Awad, M. Ali, J. Liang, B. Qu, and P. Suganthan, “Problem definitions and evaluation criteria for the cec 2017 special sessionand competition on single objective real-parameter numerical optimization. nanyang technologial university, jordan university of science and technology and zhengzhou university, singapore and zhenzhou,” Nanyang Technological University, Jordan University of Science and Technology and Zhengzhou University, Singapore and Zhenzhou, China, Tech. Rep, vol. 201611, 2016.

[37] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, “Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization,” KanGAL report, vol. 2005005, no. 2005, p. 2005, 2005.

[38] K. Price, N. Awad, M. Ali, and P. Suganthan, “Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization,” in Technical Report. Nanyang Technological University, 2018.

[39] C. T. Yue, K. V. Price, P. N. Suganthan, J. J. Liang, M. Z. Ali, B. Y. Qu, N. H. Awad, and P. P. Biswas, “Problem definitions and evaluation criteria for the cec 2020 special session and competition on single objective bound constrained numerical optimization,” Technical Report 201911, 2019.

[40] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, “Cec 2017 special session on single objective numerical optimization single bound constrained real-parameter numerical optimization,” Jul 2019. [Online]. Available: https://github.com/P-N-Suganthan/CEC2017-BoundContrained/blob/master/Bound-Constrained-Comparisons.pdf

[41] A. Kumar, R. K. Misra, and D. Singh, “Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase,” in 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 1835-1842.

[42] N. H. Awad, M. Z. Ali, and P. N. Suganthan, “Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving cec2017 benchmark problems,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 372-379.

[43] J. Brest, M. S. Maucec, and B. Boskovic, “Single objective real-parameter optimization: Algorithm jso,” in 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 1311-1318.

[44] D. Jagodzinski and J. Arabas, “A differential evolution strategy,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 1872-1876.

[45] D. Maharana, R. Kommadath, and P. Kotecha, “Dynamic yin-yang pair optimization and its performance on single objective real parameter problems of cec 2017,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 2390-2396.

[46] P. Bujok and J. Tvrdik, “Enhanced individual-dependent differential evolution with population size adaptation,” in 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 1358-1365.

[47] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi, “Lshade with semi-parameter adaptation hybrid with cma-es for solving cec 2017 benchmark problems,” in 2017 IEEE Congress on evolutionary computation (CEC). IEEE, 2017, pp. 145-152.

[48] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, “Multimethod based orthogonal experimental design algorithm for solving cec2017 competition problems,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 1350-1357.

[49] A. LaTorre and J.-M. Pena, “A comparison of three large-scale global optimizers on the cec 2017 single objective real parameter numerical optimization benchmark,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 1063-1070.

[50] A. Tangherloni, L. Rundo, and M. S. Nobile, “Proactive particles in swarm optimization: A settings-free algorithm for real-parameter single objective optimization problems,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 1940-1947.

[51] R. Biedrzycki, “A version of ipop-cma-es algorithm with midpoint for cec 2017 single objective bound constrained problems,” in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017, pp. 14891494.

[52] R. Kommadath and P. Kotecha, “Teaching learning based optimization with focused learning and its performance on cec2017 functions,” in 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 2397-2403.

[53] R. Tanabe and A. S. Fukunaga, “Improving the search performance of shade using linear population size reduction,” in 2014 IEEE congress on evolutionary computation (CEC). IEEE, 2014, pp. 1658-1665.

[54] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, “An ensemble sinusoidal parameter adaptation incorporated with l-shade for solving cec2014 benchmark problems,” in 2016 IEEE congress on evolutionary computation (CEC). IEEE, 2016, pp. 2958-2965.

[55] C. T. Yue, K. V. Price, P. N. Suganthan, J. J. Liang, M. Z. Ali, B. Y. Qu, N. H. Awad, and P. P. Biswas, “Competition on single objective bound constrained numerical optimization,” Sep 2020. [Online]. Available: https://github.com/P-N-Suganthan/2020-Bound-Constrained-Opt-Benchmark/blob/master/CEC2020 BCC Results Analysis_R_Aug.16.pdf

[56] K. M. Sallam, S. M. Elsayed, R. K. Chakrabortty, and M. J. Ryan, “Improved multi-operator differential evolution algorithm for solving unconstrained problems,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-8.

[57] A. W. Mohamed, A. A. Hadi, A. K. Mohamed, and N. H. Awad, “Evaluating the performance of adaptive gainingsharing knowledge based algorithm on cec 2020 benchmark problems,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 18.

[58] J. Brest, M. S. Maucec, and B. Boskovic, “Differential evolution algorithm for single objective bound-constrained optimization: Algorithm j2020,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-8.

[59] R. Salgotra, U. Singh, S. Saha, and A. H. Gandomi, “Improving cuckoo search: Incorporating changes for cec 2017 and cec 2020 benchmark problems,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-7.

[60] A. Bolufe-Rohler and S. Chen, “A multi-population exploration-only exploitation-only hybrid on cec-2020 single objective bound constrained problems,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-8.

[61] V. Stanovov, S. Akhmedova, and E. Semenkin, “Ranked archive differential evolution with selective pressure for cec 2020 numerical optimization,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-7.

[62] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, and A. Zamuda, “Dish-xx solving cec2020 single objective bound constrained numerical optimization benchmark,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-8.

[63] P Bujok, P Kolenovsky, and V. Janisch, “Eigenvector crossover in jde100 algorithm,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-6.

[64] P. P. Biswas and P. N. Suganthan, “Large initial population and neighborhood search incorporated in lshade to solve cec2020 benchmark problems,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-7.

[65] Y.-C. Jou, S.-Y. Wang, J.-F. Yeh, and T.-C. Chiang, “Multi-population modified l-shade for single objective bound constrained optimization,” in 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2020, pp. 1-8.

[66] T. Kadavy, M. Pluhacek, A. Viktorin, and R. Senkerik, “Self-organizing migrating algorithm with clustering-aided migration,” in Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, ser. GECCO ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1441-1447. [Online]. Available: https://doi.org/10.1145/3377929.3398129

[67] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, “Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm,” International Journal of Machine Learning and Cybernetics, vol. 11, no. 7, pp. 1501-1529, 2020.

[68] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,” IEEE transactions on evolutionary computation, vol. 10, no. 6, pp. 646-657, 2006.

[69] J. Brest, M. S. Maucec, and B. Boskovic, “The 100-digit challenge: Algorithm jde100,” in 2019 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2019, pp. 19-26.

[70] M. Friedman, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” Journal of the american statistical association, vol. 32, no. 200, pp. 675-701, 1937.