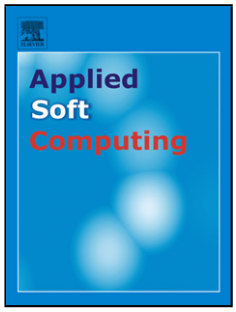


Title: A Switched Parameter Differential Evolution with Optional Blending Crossover for Scalable Numerical Optimization

Authors: Arka Ghosh, Swagatam Das, Sankha Subhra Mullick, Rammohan Mallipeddi, Asit K. Das

PII: S1568-4946(17)30125-4
DOI: http://dx.doi.org/doi:10.1016/j.asoc.2017.03.003
Reference: ASOC 4086

To appear in: *Applied Soft Computing*

Received date: 21-6-2016
Revised date: 27-2-2017
Accepted date: 2-3-2017

Please cite this article as: Arka Ghosh, Swagatam Das, Sankha Subhra Mullick, Rammohan Mallipeddi, Asit K.Das, A Switched Parameter Differential Evolution with Optional Blending Crossover for Scalable Numerical Optimization, Applied Soft Computing Journalhttp://dx.doi.org/10.1016/j.asoc.2017.03.003

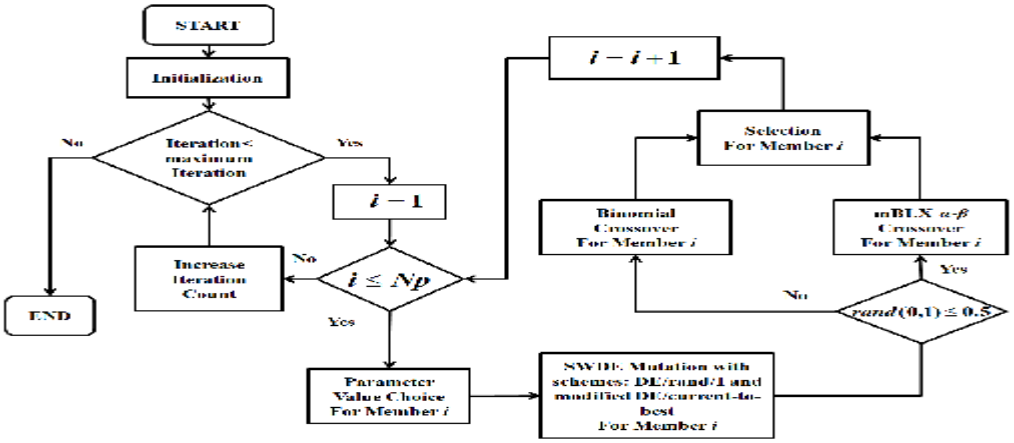
A Switched Parameter Differential Evolution with Optional Blending Crossover for Scalable Numerical Optimization

Arka Ghosh[1,3], Swagatam Das[1], Sankha Subhra Mullick[1], Rammohan Mallipeddi[2], and Asit K. Das[3]

[1] Indian Statistical Institute, 203 B. T. Road, Kolkata -700 108, India.
[2]College of IT Engineering, Kyungpook National University, Daegu -702 701, Republic of Korea.
[3]Dept. of Computer Science & Technology, IIEST, Shibpur, Howrah -711 103, India.

E-mails: arka_t@isical.ac.in, swagatam.das@isical.ac.in, sankha_r@isical.ac.in, mallipeddi.ram@gmail.com, akdas@cs.iiests.ac.in.

Graphical Abstract



Research Highlights:

- A simple yet very efficient variant of DE for scale-free optimization.
- Novel switching based adaptation of F and Cr parameters.
- A unit memory success-based mutation scheme used.
- A modified BLX-alpha-beta crossover scheme is integrated with DE.
- Extensive simulation results on CEC 2013 and CEC 2010 benchmark sets.

Abstract: Differential Evolution (DE) is currently one of the most competitive Evolutionary Algorithms (EAs) for optimization problems involving continuous parameters. This article presents three very simple modifications to the basic DE scheme such that its performance can be improved and made scalable for optimizing functions having a real-valued moderate-to-high number of variables (dimensions) while focusing on preservation of the simplicity offered by its algorithmic framework. Contrary to the common policies of coupling a complicated scheme for adaptation of the control parameters or introducing additional local search algorithms, we present here a (population member and generation specific) control parameter choosing strategy which uniformly and randomly switches the values of the mutational scale factor and crossover rate between two extremities of their feasible ranges. Furthermore, each population member is mutated either by using the DE/rand/1 scheme or a proposed version of the DE/current-to-best scheme. The mutation scheme for a population member is chosen based on its performance in the current generation. Hence if a mutation strategy successfully replaces a target vector by the corresponding trial vector, then it is reused by the population member of the same index in the following generation, else a switch of mutation method is executed. In the crossover phase, each member undergoes either the common binomial crossover or the BLX-α-β crossover modified for applying in DE, with equal probabilities. Our experiments using the benchmark optimization functions proposed for the IEEE Congress on Evolutionary Computation (CEC) 2013 competition on real parameter optimization and CEC 2010 competition on large-scale global optimization demonstrate that the basic DE optimizer when coupled with these elementary alterations and/or schemes can indeed provide a very competitive result against some of the most prominent state-of-the-art algorithms.

Keywords: Evolutionary computation; differential evolution; large-scale optimization; BLX crossover; parameter switching.

1.Introduction

Over the past few decades, a variety of evolutionary computing algorithms belonging to several families has been proposed for solving bound-constrained global optimization problems. Continuous parameter optimization problems form a significant subset of the global optimization problems involving real-valued decision variables within some feasible range. Such problems are ubiquitous through various streams of science and technology and a few significant examples include molecular distance geometry problems, optimal control problems, spacecraft trajectory optimization problems, economic load dispatch and so on (Das and Suganthan, 2011). Most of the modern metaheuristic algorithms including DE is notable for sustaining a considerably good performance when applied to problems with a small or moderate number of dimensions. However, if the number of dimensions of the search space increases (usually beyond 100 and more) these methods find it hard to accurately locate the global optimum without increasing the cost for Function Evaluations (FEs). This is an expected behavior which is primarily caused by the exponential increase of the search volume with dimensions. For example, suppose covering a real-valued search space say $[0,1]$ by placing 100 points. To achieve a coverage similarly dense in nature (in the form of distance between adjacent points), in the 10-dimensional space, i.e. $[0,1]^{10}$ would require $100^{10} = 10^{20}$ points. Unfortunately if we still want to maintain a coverage by only 100 points we will end up with a group of sparsely distributed (in fact isolated) points in a vast empty space. Moreover, a distance measure (for example Euclidean), which is otherwise useful in small dimensions, breaks down in higher dimensionalities and reduces the effectiveness of a search strategy for high or even moderate dimensional problems. Unfortunately many real world

problems demand optimization of functions involving a large number of variables (even containing inter-variable dependence). A few typical examples of such problems are shape optimization (Foli *et al*., 2006; Sonoda *et al*., 2003) , high-dimensional waveform inversion (Wang and Gao, 2012), and large-scale economic load dispatch involving 140 units or more (Das and Suganthan, 2011). The recently increasing abundance of large-scale global optimization problems generated from the real world, requiring an economically attractive solution conserving both time and computational resource is encouraging the researchers to pay attention to the issue of designing scalable (whose performance will not be dependent on the dimensionality of the problem) heuristic optimization techniques. This results in the scientific community's conspicuous interest, visible from the spirited participations in the competitions on large-scale single-objective global optimization problems with bound constraints held under the IEEE CEC (Congress on Evolutionary Computation) in various years.

Differential Evolution (DE) (Storn and Price, 1995; Das and Suganthan, 2011; Das *et al*., 2016) currently stands out as a very attractive EA for optimization in continuous search spaces, mainly for of its simplicity, a small number of parameters to tune and notable performance. Several attempts over the past decades have been made to improve the scalability of DE, and the quest is very much alive. Such attempts are made following the major research directions such as success-history based parameter adaptation (like Self-adaptive DE (SaDE) (Qin *et al*., 2009), JADE (Zhang and Sanderson, 2009)), new mutation and crossover strategies (like Pro-DE (Epitropakis *et al*., 2011), MDE-pBX (Islam *et al*., 2012)), combining various offspring generation strategies (Composite DE (CoDE) (Wang *et al*., 2011), DE with Ensemble of Parameters and mutation Strategies (EPSDE) (Mallipeddi and Suganthan, 2010; 2011) etc.). For high-dimensional problems (more than 500 dimensions) DE has been coupled with methods like cooperative co-evolution, micro-population, and hybridization with local search techniques (Mahdavi *et al*., 2015). Appreciating its inherent simplicity, DE was used as the base optimizer by Yang *et al*. (2008) in their primary work on random grouping based Co-operative Co-evolutionary Evolutionary Algorithms (CCEAs). Later CCEA was used as a dimensional decomposition scheme by Zamuda *et al*. (2008) who alongside improved DE by log-normal self-adaptation of its control parameters. A cooperative micro-DE was proposed by Parsopoulos (2009), and SaDE was hybridized with Multi-Trajectory Search by Zhao *et al*. (2011), both targeting large-scale global optimization. Other notable approaches intending to improve the scalability of DE can be found in works like (Brest and Maučec, 2011; Wang *et al*., 2010; Weber *et al*., 2011; Wang *et al*., 2011).

From the above discussion, it is evident that DE went through significant modifications to cope with the growing complexity of the optimization problems. However, as a cost of the reported performance improvement, the simplicity of the basic framework of DE, which was the cause for the algorithm's primary attraction, had to be sacrificed. This observation motivated the present work. Thus, on the contrary to the existing approaches, we attempt to improve the scalability of DE by simple parameter control strategies and combining the slightly modified building blocks of DE (like mutation or crossover operations) without introducing any additional computational time and/or resource requirements (likely to emerge from the use external archives, proximity and rank based parent selection mainly for mutation, local search schemes, maintaining logs of successful individuals over multiple generations etc.).

We introduce three basic modifications in the conventional DE framework in the forms of parameter value selection, success-based switching of mutation schemes, and a random selection between two crossover strategies. Moreover, we suggest an improved version of the popular DE/current-to-best mutation and adopt the BLX – α-β crossover scheme (Picek and Jakobovic, 2014) from the literature of Genetic Algorithms (GAs) and modify it suitably to fit the structure of DE. The performance of DE is largely affected by two parameters, namely the scale factor F (controlling the mutation step size) and the crossover rate Cr. In our proposal, values of F and Cr are switched uniformly at random between their respective limiting values for each member of the population. In addition, each member of the population is mutated using either the explorative DE/rand/1 strategy or a modified version of the DE/current-to-best strategy (explained in detail in Section 4.2). The mutation strategy for a particular target vector is retained as similar to the last generation only if the strategy resulted in a successful trial vector (with better fitness as compared to the target vector) in the last generation for the same population index, otherwise, the mutation strategy is switched to the remaining option. Hence, the choice of the mutation strategy is dependent only on the history of the success in the last generation, which can easily be recorded with the help of a binary array having the length of the population size. Finally, in the crossover phase, each of the target-donor (generated by mutation) pairs is subjected either to a conventional binomial crossover or a BLX – α-β crossover, redefined and alternatively interpreted by us in the attempt of making it suitable for DE for the first time.

A challenge faced by DE while in solving the high-dimensional problems is to perform a coarse (explorative) search over the large space alongside sustaining the ability of refined (exploitative) searches over smaller basins of attraction. Switching the value of F between the two limits of its allowable range (here 0.5 and 2) provides the scope for fulfilling such a requirement. Similarly, by switching the Cr values of binomial crossover between 0 and 1, a balance can be struck between a coordinate-wise search (when 0) and a rotationally invariant search (when 1). The BLX – α-β crossover, on the other hand, can perform a search in the vicinity of the target and donor (somewhat mimicking a local search affording only the same asymptotic cost as binomial) as well as on the hyper-plane joining the two, by again switching the value of the associated parameters. Our experiments indicate that the simple parameter switching technique when coupled with the success-based mutation and random crossover strategy selection involving modified mutation and proposed crossover schemes can significantly improve the scalability and performance of DE. This conclusion is supported through a rigorous performance comparison of the proposed DE variant with that of some of the popular moderate and large-scale optimizers on the test-suites proposed for CEC 2013 competition on real parameter optimization (Liang *et al.* 2013), and CEC 2010 competitions on large-scale global optimization (Tang *et al.* 2009) . Unfortunately, it is quite hard (if not impossible) to justify analytically the suitability of the proposed changes made to DE. Hence we perform a variety of empirical studies based on the population spread and conservation of diversity to highlight the effectiveness of each of the suggested modifications.

Finally, a preliminary version of this article appeared in Das *et al*. (2015) where the values of the scale factor and crossover rate are respectively switched in a uniformly random way between $\{0.5, 2\}$ and $\{0,1\}$for each of the population members. Moreover, each target vector is mutated either by using the DE/rand/1 or by the DE/best/1mutation scheme following the success-based selection. This scheme is called SWDE_Success

and it exhibited very competitive performance on the high-dimensional functions compiled under the CEC 2008 and 2010 competitions on large-scale global optimization (Tang *et al.* 2007, Tang *et al.* 2009). The present paper is a significant extension of the conference article. The major differences with the conference article are in order:

1) Unlike (Das *et al*., 2015), which used the conventional binomial crossover, this work integrates a modified blending crossover (BLX – α-β) with DE to improve the balance between diversification and intensification of the searches performed.
2) Das *et al*. (2015) used a candidate pool of DE/best/1 and DE/rand/1 mutation strategies. However, this work uses a candidate pool consisting of DE/rand/1 and a modified version of the DE/current-to-best/1 mutation schemes to meet the needs for explorative search on rugged moderate-dimensional fitness landscapes.
3) The algorithmic framework of Das *et al*. (2015) was specifically suited to high-dimensional functions. However, this work attempts to develop a scale-free DE variant which can handle functions with dimensionality ranging from 30 to 2000. Unlike (Das *et al*., 2015), it undertakes extensive empirical studies and performance comparison on the CEC 2013 test-bed to highlight the efficacy and simplicity of the proposed algorithm.

The rest of the paper is organized as follows. In Section 2 we give a brief description of the canonical DE algorithm. In Section 3 we briefly discuss the existing popular variants of DE focusing on the parameter and strategy adaptation. Subsequently, in Section 4 we present the proposed variant of DE, separately detailing each phase of the introduced modifications. Next, we present our results and discussions highlighting the improvement shown by our work in Section 5. Finally, we conclude the paper in Section 6.

2. DE algorithm in canonical form- an Overview

DE starts with a set of candidate solutions for the problem at hand and iteratively attempts to improve them following evolutionary operators. DE algorithm in its classic form consists of the four basic steps – initialization, mutation, recombination or crossover, and selection, executing in the same order. Among these four steps, initialization is performed only once at the start of a DE run while the other three are repeated in every generation (say G). The generations are continued until a termination criterion is met, which in practice can be the exhaustion of a maximum number of Function Evaluations (FEs). DE uses only three control parameters and these are: the scale factor F (analogous to mutation step size), crossover rate Cr, and the population size Np. In what follows, we will explain the role of these parameters more elaborately.

2.1 Initialization

DE initializes the population with a set of Np vectors each of dimension D, where the optimization problem is D-dimensional. The values of each dimension of these primary vectors are randomly assigned a real number lying in the range of the corresponding dimension as mentioned in the problem description. Thus, each of these vectors can be treated as a candidate solution to the optimization problem. We use the following notation to identify the i^{th} solution of the present generation G as:

$\vec{X}_{i,G} = \left[x_{j,i,G}\right]_{1\times D} = \left[x_{1,i,G}, x_{2,i,G}, ...x_{D,i,G}\right]$. The decision space bounds are given by vectors $\vec{X}_{\max} = \left[x_{1,\max}, x_{2,\max}, ...x_{D,\max}\right]$and $\vec{X}_{min} = \left[x_{1,min}, x_{2,min}, ...x_{D,min}\right]$. Thus the j^{th} dimension of the i^{th} individual can be initialized as:

$$x_{j,i} = x_{j,\min} + rand_{i,j} \times (x_{j,\max} - x_{j,\min}), \tag{1}$$

where $rand_{i,j}$ is a uniformly distributed random number lying in the range [0, 1] and it is instantiated anew for each ordered pair (i, j).

2.2 Mutation

Each population member (say i) of the current generation G, is known as the *target vector* following the common DE terminology. For the i^{th} target, a vector called the base vector and denoted by $\vec{X}_{r_1,G}$ will be chosen (either randomly or following a specification) and is *differentially* mutated by the scaled difference of other randomly chosen vectors in the form $\left(\vec{X}_{r_2,G} - \vec{X}_{r_3,G}\right)$ to produce a mutant or *donor vector* $\vec{V}_{i,G}$. Here the indices r_1, r_2 and r_3 are sampled from $\{1,2,\ldots, Np\}$ and they are different from the running index i i.e. $(r_1, r_2, r_3 \in \{1,2,...Np\} \setminus \{i\})$, unless specified otherwise. Three commonly used DE mutation strategies (Das *et al*., 2016, Storn and Price, 1997) are listed below:

$$DE/rand/1: \vec{V}_{i,G} = \vec{X}_{r_1,G} + F\left(\vec{X}_{r_2,G} - \vec{X}_{r_3,G}\right), \tag{2a}$$

$$DE/best/1: \vec{V}_{i,G} = \vec{X}_{best,G} + F\left(\vec{X}_{r_2,G} - \vec{X}_{r_3,G}\right), \tag{2b}$$

$$DE/current\text{-}to\text{-}best: \vec{V}_{i,G} = \vec{X}_{current,G} + F_1(\vec{X}_{best,G} - \vec{X}_{current,G}) + F_2(\vec{X}_{r_1,G} - \vec{X}_{r_2,G}). \tag{2c}$$

The scaling factor F in (2a) and (2b) is a positive real value usually in (0.4, 2], and responsible for controlling the amplitude of the difference vectors, acting as a multiplicative factor. In (2c), the two scale factors F_1 and F_2 are often assigned the same value (i.e. $F_{1c} = F_2 = F$) to reduce the number of control parameters. $\vec{X}_{best,G}$ is the vector in the population at a generation G with the best fitness value (*i.e.* in the case of a minimization problem the one having the least fitness). The general convention followed for naming the various mutation strategies of DE is of the form DE/x/y, where DE stands for Differential Evolution, x represents a string denoting the vector to be perturbed (for example rand, best etc.) and y is the number of difference vectors considered for perturbation of x. The entire DE strategy can be referenced by extending the notation to DE/x/y/z, where z express the type of crossover (*exp*: exponential; *bin*: binomial etc.).

2.3 Crossover

In DE, the crossover step aims to combine the dimensional components of the target and the donor vectors into a single offspring called a *trial vector* $\vec{U}_{i,G} = \left[u_{j,i,G}\right]_{1\times D} = \left[u_{1,i,G}, u_{2,i,G}, ...u_{D,i,G}\right]$. DE usually employs one of the two classic crossover strategies, either the *exponential* (two-point modulo) or the *binomial* (uniform). Among the two, most of the literature prefer binomial crossover since it can successfully discard the inherent

representational bias in *n*-point crossover by simulating *D* random trials (Das and Suganthan, 2011). Moreover, analytical studies like (Zaharie, 2002; 2009) investigating the sensitivity of crossover with the changing value of *Np* support the higher efficiency of binomial crossover compared to the exponential. Keeping these theories and observations in mind we briefly describe here only the binomial crossover strategy which we will also use as a part of our proposed DE variant.

Binomial crossover requires a parameter called *crossover rate* (*Cr*), which is similar in nature to a probability and can lie in the range [0,1]. The process uniformly samples *D* independent random numbers between 0 and 1, each corresponding to a dimension, and compares with *Cr* to decide whether that dimension of the trial vector will be copied from the donor vector or the target vector. The method is outlined as:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, if\ rand_{i,j} < CR\ or\ j = j_r, \\ x_{j,i,G}\ otherwise, \end{cases} \quad (3)$$

where j_r is a randomly chosen index from {1, 2, …, *D*} ensuring the presence of at least one component of the donor vector in the generated trial.

2.4 Selection

DE uses a one-to-one comparison of fitness between the target and the corresponding trial to maintain a constant population size. The selection process for a minimization problem can be described as:

$$\vec{X}_{i,G+1} = \begin{cases} \vec{\mathrm{U}}_{\mathrm{i,G}}\ if\ f\left(\vec{\mathrm{U}}_{\mathrm{i,G}}\right) \le f\left(\vec{X}_{i,G}\right), \\ \vec{X}_{i,\mathrm{G}}\ otherwise, \end{cases} \quad (4)$$

where $f(.)$ is the objective function to be minimized. Therefore, if the trial vector yields an equal or lower value of the objective function, signifying to be a better solution, it replaces the target vector in the next generation; otherwise, no replacement of that target is made. The "≤" sign instead of "<" in (4) aids the DE population to navigate the flat portions of a fitness landscape and reduces the chance of the population becoming stagnated.

3. Strategy and Control Parameter Adaptation in DE - a Review

Research on and with DE has reached an impressive (and voluminous) state over the last few years and it is not possible to provide an overview of the multifaceted DE literature in a subsection like this. For an exhaustive account of various aspects of DE, please see the survey (Das *et al*., 2016). Here, we mainly confine ourselves to two important aspects of DE research: the strategy and parameter adaptation. Starting with the landmark paper on SaDE (Qin *et al*., 2009), there has been a growing trend to select the offspring generation strategies (combination of a mutation and a crossover strategy) from a pool (Spears, 1995) as well as adapting the control parameters *F* and *Cr* based on success histories of the generated trial vectors. A prominent work in this direction undertaken by Mallipeddi *et al*. (2011) propose a DE with Ensemble of Parameters and mutation Strategies (EPSDE). EPSDE limits the individual-specific strategy selection only to mutation, where a pool containing various mutation schemes, each designed to perform a different degree of

exploitation and exploration of the search space, is used. The possible values of F and Cr are confined to sets of discrete candidate values within certain ranges. At the time of initialization, not only the individuals of the population are created but also each of them is associated with a value of F and Cr, and a mutation scheme is randomly selected from the respective pools. If the generated trial is successful, i.e. it can replace the target vector after selection, then the strategy is kept the same for that individual in the following generation. However, if the trial fails to replace the target, then the algorithm picks either a new strategy or retains the old one with equal probability in the next generation. The mutation pool contains three distinct schemes, namely DE/best/2/bin, DE/rand/2/bin, and DE/current-to-rand/1. Mallipeddi (2013) proposed a variant of EPSDE by using another metaheuristic algorithm known as Harmony Search (HS) to optimize the evolution of the ensemble of F and Cr values.

Another variant of DE, adapting the mutation strategy was presented by Elsayed *et al*. (2011). The proposed method selects a mutation scheme from a pool of four allowable schemes, as the generations progress. The authors pointed that, in the different states of the progress of a DE run, use of mutation of different characteristics can be more beneficial, than a single mutation scheme utilized throughout. Thus, at a certain generation, a mutation strategy selected from the pool can only be used by the method if it is tested to be the most beneficial for progress. In Composite DE (CoDE) developed by Wang *et al*. (2011), three primary trial vectors are generated by using three different DE strategies, only the best (in term of fitness) among the three primary trials is considered for participating in selection. The mutation pool consists of three schemes (DE/rand/1/bin, DE/rand/2/bin and DE/current-to-rand/1), while the control parameters can have three popular set of values (F = 1.0, Cr = 0.1; F = 1.0, Cr = 0.9; F = 0.8, Cr = 0.2). The strategy for generating the primary trials is a random combination of mutation scheme and parameter values. Gong *et al*. (2011) proposed an adaptive DE algorithm where 4 mutation strategies (variants of DE/current-to-*p*best/1, DE/rand-to-*p*best/1 with and without an external archive (Zhang and Sanderson 2009) are used to create the pool. An individual-specific mutation strategy is selected from the pool following a controlled probability. The parameter adaptation method is similar to the one proposed in JADE algorithm (Zhang and Sanderson, 2009). Wu *et al.* (2015) have developed a DE variant called MPEDE, which realizes an adaptive ensemble of three mutation strategies (i.e. "current-to-*p*best/1" and "current-to-rand/1" and "rand/1") through the use of a multi-population based framework.

Elsayed *et al*. (2013) proposed an adaptive control parameter selection scheme for DE. Two sets of allowable values for the parameters are defined by the authors, and the trial vector is generated by a modified DE/rand/1/bin strategy (the mutation is performed by randomly selecting two vectors from the population for generating the difference vector, while the base is randomly chosen from the top 10% to 40% individuals (ranked according to their performances), a scheme similar to the one introduced in JADE (Zhang and Sanderson, 2009)). A counter is initialized with zero for each set of combinations of F and Cr values. A trial vector is generated by randomly picking a set from the allowable combinations, and if the trial gets successfully selected for the next generation, then the corresponding counter to the chosen combination is increased by one. After each predefined interval of generations, only a half of the combinations are retained (corresponding to the better counter values indicating top performers), and their corresponding counters are

reset to zero, discarding the remaining sets of *F* and *Cr* values. If the number of generations exceeds a limit, then a restart is performed initializing the iteration count and the counters associated with the sets of combination of the control parameters. Gong and Cai (2012) developed a simple probability-matching-based adaptive parameter selection scheme for DE.

Sarker *et al*. (2014) improved over the work of Elsayed *et al*. (2013) such that the best combination of all the three control parameters of DE (*Np*, *F*, and *Cr*) can be identified and adapted dynamically. After every periodic interval of generations, the algorithm decreases the value of *Np*. Only the better solutions are retained in the reduced population, while an achieve is used to store the others. When the value of *Np* cannot be reduced further, the average fitness improvements achieved by the populations having different *Np* values are calculated, and the *Np* value which exhibits the best fitness improvement over the interval is selected as the ideal population size. The population is then filled with the required number of solutions chosen from the archive based on their fitness value. The mutation scheme and the adaptation strategies for *F* and *Cr* follow the work of Elsayed *et al*. (2013).

The popular DE variant JADE (Zhang and Sanderson, 2009) is further improved by Tanabe and Fukunaga (2013) in the form of Success History based DE (SHADE). JADE samples the value of *F* and *Cr* from parameterized probability distributions, and the adaptation process updates the parameters of the distributions rather than directly altering the value of the DE control parameters. Only those *F* and *Cr*, which generate successful trial vectors in a generation, are allowed to modify the parameters of the corresponding distributions. In SHADE, the sampling of *F* and *Cr* values from gradually adapted probability distributions is replaced by direct sampling of the parameter space close to one of the stored pairs in historical memory archives. These archives are denoted by M_{Cr} and $M_{F,}$ respectively and they contain the set of better performing *Cr* and *F* values (only over a pre-defined number of earlier generations). In the IEEE CEC competition on real parameter single-objective optimization (Liang *et al*., 2013), SHADE ranked 3rd among the 21 commentating algorithms and first among all the DE variants. Tanabe and Fukunaga (2014) further improve the SHADE algorithm by introducing a linear reduction of *Np* and called this variant L-SHADE. A non-linear sinusoidal model for the adaptation of *F* and *Cr* is proposed recently by Draa *et. al.* (2015), who presented six variants of DE for the purpose.

Motivated by SaDE, Zhao *et al*. (2016) developed a self-adaptive DE variant which keeps up diversity by initializing the population with symmetric Latin hypercube design. In this algorithm, the trial vector generation strategy is adaptively selected from the strategy candidate pool to match different stages of the evolution according to their previous successful experience. The authors employ two parameterized probability distributions (Cauchy and Gaussian) to update the control parameters *Cr* and *F* during the evolutionary search. Recently Pranav and Jeyakumar (2015) critically reviewed a number of adaptation strategies for *F* and *Cr* in DE and provided insightful discussions on these strategies.

4. The Proposed Method

The proposed DE variant introduces modifications in three stages of DE - the parameter control, the mutation, and finally the crossover. Hereafter, we will call the complete algorithm as SWDE_Success_mBLX (Switched parameter DE with Success-based mutation and modified BLX crossover). In this section, we start with a high-level description of the whole algorithm. Then we provide a detailed description of the different components of the algorithm.

4.1 SWDE_Success_mBLX - A Bird's Eye View

Fig. 1 shows the complete flowchart of the SWDE_Success_mBLX algorithm, which offers a three-stage modification of DE pertaining to the parameter control, mutation, and crossover. It does not require any complex parameter tuning, Euclidean distance calculation or probability matrix manipulation in any stage. Since for each member, the objective function is evaluated only once per iteration, the maximum numbers of iteration allowed (used as the conventional stopping criterion) is equal to the maximum number of FEs allowed divided by *Np*.

As per Fig. 1, the flow of the algorithm at first checks for whether the maximum number of iterations has been exhausted. If not, for each individual of the population, it selects the parameters F and Cr following the switching strategy as described in Section 4.2. Then each member undergoes a success-based modified mutation scheme as discussed in Section 4.3. Next, each member is subjected to either binomial or a modified BLX crossover. Finally selection is performed for each member in an asynchronous way. This means new offspring individuals can enter into the population and take part in modifying population members at subsequent indices.

4.2 The Switching-based Parameter Control

The execution of DE is controlled by three basic parameters, namely the scale factor *F*, the crossover rate *Cr*, and the size of the population *Np*. Among these three control parameters mainly *F* and *Cr* are important for their ability to guide the explorative and exploitative searching capability of DE in different functional landscapes. The allowable range for the values of these parameters is already established through a collection of studies (see for example, Section III of (Das and Suganthan, 2011) and the references therein). For *Cr* being a variable similar to a probability, the value should lie in [0, 1]. A lower limit of *F* was studied by Zaharie (2002), who also claimed that if a sufficiently small value of *F* is taken, the population can converge (prematurely) even in the absence of any selection pressure. Detailing further, Ronkkonen *et al*. (2005) stated that a typical value in the range [0.4, 0.95] can be a good choice for *F*, especially focussing on $F = 0.9$. They also commented on the relation of the value of *Cr* and the nature of the objective function, stating *Cr* should lie in (0, 0.2) for a separable optimization problem, while in (0.9, 1) for functions containing variable inter-dependence. From these studies, the importance of the two control parameters in optimizing a problem using DE can be established. However, none of them can describe a specific guidance beyond defining a good general range for a black-box large-scale optimization problem. As discussed in Section 2.2, numerous variants of DE were published aiming to tackle the issue but the overhead hindered the computational complexity and methodical simplicity of DE. Here we target to address the similar problem by varying only

the value of *F* and *Cr* such that a balance can be maintained between the explorative (important for large search space) and exploitative (necessary to locate the global optimum with accuracy) searches performed by DE.

The scaling factor *F* acts as a multiplicative factor altering the amplitude of the difference vector (or assigning it a weight) before adding it to a base vector. Thus, the amplitude of the donor vector is inherently guided by the value of *F*. A large value of *F* is likely to create a donor vector far away from the target, thus, favoring exploration (Zaharie, 2002; Das *et al*., 2016). This explorative search property is desirable mostly in the case of high-dimensional problems as the large search space can be covered using the technique. Unfortunately, the large displacement of the donor from the vicinity of the target due to exploration cannot be beneficial for either the convergence of the population or the fine-tuned search required to locate the global optimum with accuracy. Hence, a small value of F is needed to support the exploitation of the search space to achieve convergence and pinpoint the optimum.

In our proposal, for each of the target vectors, we switch the value of *F* between 0.5 and 2 in a uniformly randomized way. The point to note here is the use of a rather unusually large value of $F = 2$, which is not previously used by DE researchers. However, the experiments reported in (Das *et al*., 2015) revealed that the performance is largely improved when *F* is switched between 0.5 and 2 rather than the common limits of the value of *F* i.e. {0.5,1} especially for the problems with large number of dimensions. This behavior can be explained as follows; when *F* becomes 2, a large weighting of the difference vector helps to throw the donor vector to a distant part of the large search space enhancing exploration of untouched areas. However, too large a value may lose a proper control over the length of the jump and the placement of the donor, thus $F > 2$ may not be profitable, as our empirical study suggested. On the other hand, when *F* takes a value of 0.5, the vector lies close to the base vector, as the amplitude of the difference is scaled down to half, performing an exploitative search over the already explored region. This is graphically described in Fig. 2(a) showing the two possible locations of the donor vector for two different values of *F*, given a base (denoted by $\vec{X}_{base}$) and a difference vector on the two-dimensional search space. In the following Fig. 2(b) the DE mutation scheme is described as an affine combination of the base vector $\vec{X}_{base}$ and the difference vector. It also illustrates how a donor vector positions itself parallel to the difference vector, only varying in length with the change of *F*.

For each target-donor pair, the *Cr* value used in the binomial crossover is also switched between 0 and 1. This indicates, in the proposed variant of DE, the trial vector is formed by either directly accepting the entire donor (for $Cr = 1$) or differing from the target in only one dimension determined by j_r (for $Cr = 0$), which is copied from the donor. While the former situation conserves the property of rotational invariance, the latter induces movement of the solutions along any one of the coordinate axes (thus, performing an axis parallel local search).

As we have mentioned earlier in Section 2.2, several attempts have been made to randomly sample and tune the value of control parameters, both using parametric (for example using a Cauchy or Gaussian distribution

for sampling, see e.g. (Qin *et al*., 2009; Zhang and Sanderson, 2009)) and non-parametric (sampling from a uniform distribution like (Brest *et al*., 2006)) approaches, with or without the support of an external archive. However, such switching between only two extreme values is a novelty of our algorithm. We name this variant of DE as Switching DE or SWDE.

4.3 Mutation

Each target vector can be mutated by any common or novel mutation strategies in SWDE. However, use of a pool of mutation schemes with different characteristics is always a better strategy as it can increase the diversity of the population (Qin *et al*., 2009). To reduce the computational efforts, we suggest only two schemes, namely DE/rand/1 and an improved version of DE/current-to-best to create a simple mutation pool. It is an empirically observed fact that DE/rand/1 (defined in eqn. (2a)) can induce a higher degree of randomization (since a random base vector is perturbed to generate the donor vector) and thus, explorative power (Mezura-Montes *et al*., 2006). Hence this can be beneficial for solving multimodal problems with distributed global basins of attractions. On the other hand, DE/current-to-best strategy (shown in eqn. (2c)) adds a scaled difference of two random vectors, with a point lying on the line joining the target (current) and the best vectors, thus, generating a mutated recombinant. Hence this scheme is more exploitative or greedy in nature and suitable for unimodal functions with narrow basins of attraction. In order to achieve a finer balance between exploration and exploitation, we further modify DE/current-to-best/1 scheme in the following way. An empirical investigation of this mutation scheme reveals that the values of F_1 and F_2 in eqn. (2c) can be fine-tuned to obtain better performance. Here, F_2 can be replaced by the scale factor F and follow the parameter switching process described above, while F_1 can be a random value from the interval (0,1). From our empirical simulation results, we also noted that adding the scaled difference vector with a point lying on the extended line joining the best and the up-scaled target is more beneficial. Considering these facts we modify the DE/current-to-best/1 mutation scheme, originally shown in eqn. (2c) , as follows:

$$\vec{V}_{i,G} = \vec{X}_{best,G} + randi(0,1)(\vec{X}_{best,G} - rand[1,1.5].\vec{X}_{current,G}) + F.(\vec{X}_{r_1,G} - \vec{X}_{r_r,G}), \qquad (5)$$

where $randi(0,1)$ takes a value of either 0 or 1 with equal probabilities and F is the usual scale factor of DE. The justification of the parameter choices of the proposed mutation can be given by observing the search regions explored by the original DE/current-to-best/1 and its proposed variant (given in eqn. (5)) as shown in Fig. 3(a) and 3(b) respectively . Note that in the conventional DE/current-to-best/1 scheme, the intermediate point on the $(\vec{X}_{best,G} - \vec{X}_{current,G})$ vector in Fig. 3(a) is E and this acts as the base point for perturbation with the difference vector. The circle is the search region if we take F = 0.8. However, the proposed mutation scheme can operate using one of the two possible scenarios:

1. $randi(0,1)$ chooses either 0 or 1 uniform at random, when it is 0, the mutant can be a point like B in Fig. 3(b), which is near the $\vec{X}_{best}$ point, thus, helping in the local search. This is essentially the DE/best/1 scheme and the corresponding search region is shown by the green circle in Fig. 3(b).

2. $rand[1,1.5]$ chooses a continuous value from the closed interval [1, 1.5]. This interval is chosen on the basis of empirical means. If it is 1 then the mutation scheme creates point like A_2 and if it is 1.5 then it creates points similar to A_1. Such points contribute to performing a large radius search, indicated by the blue circle in Fig. 3(b).
3.

We again use a simple strategy for selecting one of the two mutation methods available, based on the success of the currently assigned strategy in generating a better trial. At the time of initialization, we generate a zero-filled binary array of size *Np* and assign a mutation strategy randomly to each of the targets. At the end of a generation, if the target is replaced by the generated trial then the corresponding cell in the array is set to 1, else reset to 0. In the next generation if a target has a 1 in the corresponding cell of the binary array (indicating the previous mutation scheme's success history), then the mutation strategy used in the last generation for that index will be retained again, otherwise (previous mutation scheme was unsuccessful) the remaining strategy will be used for generating trial.

When the mutation pool and the strategy selection mechanism described above is combined with the parameter control scheme discussed in Section 4.1, we call the resulting algorithm as SWDE_Success (SWitching DE with Success-based selection of mutation scheme). This algorithm uses the same binomial crossover scheme. One of the most important characteristics of SWDE_Success is the absence of any parameter tuning (no initial value is required for *F* and *Cr*, only knowing the boundaries of their range is enough, also no parameters associated with any probability distribution is used), except for the population size *Np*, which due to having the least ability to affect the optimization is usually kept a constant. Moreover, the mutation pool is flexible enough to change and/or increase and the strategy selection only requires a binary array, rather than any external archive or complicated record keeping strategy.

4.3 Crossover

SWDE_Sucess is further assisted with the addition of probability-based selection of a crossover operator, from a pool of two crossover strategies, namely BLX-α-β (Picek *et al*., 2013; Picek and Jakobovic, 2014) and binomial crossover. These two schemes are invoked with equal probabilities while generating the trial or final offspring. BLX-α-β is a modification of the BLX-α (blend) crossover scheme proposed for real-coded GAs by Eshelman and Schaffer (1993). The BLX-α-β operator creates a new offspring by selecting a random value from the interval between the two components of the parent solutions. The interval is increased in direction of the solution with better fitness by the factor α, and in the direction of the solution with worse fitness by the factor β (Picek and Jakobovic, 2014). The values of α and β are conventionally fixed to 0.75 and 0.25 respectively.

The common binomial crossover creates a new offspring at one corner of the simplex formed by $\vec{X}_{i,G}$ and $\vec{V}_{i,G}$. However, such a search cannot aid the crossover technique to further explore the region between and beyond the dimensional bounds established by the set $\{x_{j,i,G}, v_{j,i,G}\}$. While arithmetic crossover (Das and

Suganthan, 2011) can perform a search on the line between $x_{j,i,G}$ and $v_{j,i,G}$ by taking a convex combination of the two, any region beyond $x_{j,i,G}$ or $v_{j,i,G}$ is still not explored through the process. A good strategy for generating the trial should not only consider the vertices of the simplex (in our case where Cr = 0 or 1 only one of the D+1 vertices will be explored) formed by the donor and the target, but should also exploit the region in the close proximity of the two as potential locality for finding a better trial. In real coded GA, the BLX-α-β crossover can perform such a search but that requires an extra fitness evaluation for the donor vector, and thus, is not beneficial for a direct application in DE.

The crossover defined by us follows a similar philosophy of BLX-α-β but avoids the extra fitness evaluation. Also, the revised strategy uses independent α and β values, both of which can be randomly selected from a set of predefined values controlling the extent of the search region. The independence of α and β is necessary for DE as without any fitness evaluation no conclusions can be drawn about the quality of the target and the donor. Thus, both should be given an opportunity to explore for a better trial vector. In proposed recombination scheme the parameters α and β are chosen uniformly at random among the members of the set {0.11, 0.5, 0.99}. Such a set of possible values contains a low value to facilitate the fine-tuned search, a high value to perform a search over a larger area, and a value in between to maintain a balance. A random choice from these values will retain the diversity alongside the capability to perform a fine tune search necessary for convergence. Let us define $c_{\max} = \max(x_{j,i,G}, v_{j,i,G})$ and $c_{\min} = \min(x_{j,i,G}, v_{j,i,G})$, and $I = c_{\max} - c_{\min}$, respectively the two extremities of $x_{j,i,G}$ and $v_{j,i,G}$, and the region between them. The proposed crossover performs a search over the region of $\alpha.I$ and $\beta.I$ respectively surrounding c_{max} and c_{min}. Whether the search will be performed on I or on one of the extended region ($\alpha.I$ and $\beta.I$) is randomly decided for c_{max} and c_{min} with equal probabilities. The final trial dimension will be a random selection between c_{lb} and c_{ub}, where they are defined as follows:

$$c_{lb} = \begin{cases} c_{\min} & if \quad rand(0,1) \le 0.5, \\ c_{\min} + \beta.I & Otherwise. \end{cases} \tag{6a}$$

$$c_{ub} = \begin{cases} c_{\max} & if \quad rand(0,1) \le 0.5, \\ c_{\max} + \alpha.I & Otherwise. \end{cases} \tag{6b}$$

The concept is illustrated in Fig. 4, where the green lines indicate the extended regions, while the black line is the region between the target and the donor dimension.

The proposed crossover scheme (hereafter called mBLX-α-β) is integrated with SWDE_Success alongside the binomial crossover such that the new algorithm can enjoy the benefit of two different search strategies realized through the crossover schemes. These two schemes are also coupled in a switched manner, where each of them is given equal chance. For each member in the population, a number is drawn from a uniform distribution at random. If this number is less than or equal to 0.5, binomial recombination is applied; else the mBLX-α-β scheme is used as the crossover method. The final algorithm, thus designed, is termed as SWDE_Success_mBLX.

5. Results and Discussion

To test the effectiveness of the proposed SWDE_Success_mBLX, we use a suite consisting of benchmark minimization problems proposed for the IEEE CEC (Congress on Evolutionary Computation) competition on real parameter optimization. The benchmark problems are designed to be of diverse nature and complexity, helping to validate the performance of the evolutionary optimizers in a variety of scenarios.

We consider two test suites from the CEC competitions: the CEC 2013 test-suite (Liang *et al*., 2013) for moderate scale minimization problems (with D = 30, 50) , and the CEC 2010 test-suite (Tang *et al*., 2009) for large-scale global optimization problems (with D = 1000). Following the common protocol, we use the mean and standard deviation of the error value (taken over a total of 50 runs) to quantify the performance of an algorithm. The value of *Np* has been taken as 100, following convention, for all algorithms, unless otherwise specified in the corresponding literature. In this section, we first give a brief discussion on the experimental procedure, following which we describe the CEC 2013 benchmark, illustrate the results of the comparison, and present a discussion on evaluating the performance of SWDE_Success_mBLX. We also highlight how different modifications suggested for SWDE_Success_mBLX harmoniously boost up the overall performance of the algorithm. Finally, we show how the proposed variant successfully retains a considerable population diversity compared to DE/best/1/bin, such that the premature convergence can be avoided.

5.1 Simulation Procedure

Following the standard procedure, we report the mean and standard deviation of the error value over 51 independent runs, for any competitor algorithm, where the error is calculated as the absolute difference between the actual value of the global optimum and the best-of-the-run value of the objective function approximating the optimum by an optimizer. Throughout the experiments, the value of the population size *Np* is kept fixed as 100 for SWDE_Success_mBLX, while all the other parameter values for this method have already been discussed in the previous section. The parameters for the other peer algorithms are set following the corresponding literature. The maximum number of FEs allowed for any algorithm is set to $10000\times D$ for CEC 2013 benchmark functions (Liang *et al*., 2013) and to $3000\times D$ for CEC 2010 optimization problems (Tang *et al*., 2010), following the norms described in the respective manuals. Uniform random initialization within the specified search space is employed for the population or any other required variables unless otherwise stated.

A non-parametric statistical test called the Wilcoxon's rank sum test for independent samples (Derrac et al., 2011) is conducted at 5% significance level to validate whether the results obtained by the best performing algorithm differs from the rest of its competitors in a statistically significant way. In the tabulated results, we also include the statistical test output summarizing in the following manner. The test confirms if the final error produced by an algorithm is statistically and significantly different from that of the best performing algorithm on a certain optimization problem. Thus if the performance of an algorithm is differing from the best result with a p value equal or less than 0.05 (ignoring the chance of the difference of two results being only apparent with a 95% certainty) then the mean error of the former is marked with a † symbol, otherwise the two performances can be considered equivalent (the difference is not statistically significant) and we mark the mean of the algorithm with the sign ≈. The best results are marked in boldface throughout all the result tables.

We also present a ranking of the individual algorithms, based on their mean error for a function. Furthermore, for each class (based on the dimension) of problems, we calculate an average rank to compare the performance of the competing optimizers. Moreover, a Win/Tie/Loss analysis is included in the result tables, where a win signifies that the algorithm is the sole best performer, for the problem of concern. If more than one method achieves the best mean error, then instead of win we increase the tie count for all of them, a loss is declared for every other case.

All simulations were executed on a workstation having Intel Xeon processor E5-2630 V3 running at 2.40GHz speed and with 32 GB of RAM. The operating system used is Windows 10 and all the codes were implemented with MATLAB 2015b.

5.2 Comparison on Moderate Scale Benchmark Problems

We compare SWDE_Success_mBLX with 8 state-of-the-art algorithms on the moderate-scale benchmark problems from the CEC 2013 test suite.

5.2.1 IEEE CEC 2013 Test Suite

IEEE CEC 2013 test-suite (Liang *et al*., 2013) contains 30 and 50-dimensional benchmark minimization problems. This suite consists of a total of 28 problems, among which 5 are unimodal and the rest of the 23 are multimodal functions (8 of them are composite in nature, and thus harder to optimize), of various complexity levels (in terms of symmetry, differentiability, linkage of the variables, properties of local optima etc.). All these functions have a search boundary of $[-100,100]^D$. A brief summary of the benchmark problems is provided in Table 1, describing the function number (used for referencing hereafter), the function name and the notable characteristics of the problem.

5.2.2 Compared Algorithms

The proposed SWDE_Success_mBLX is compared with 4 conventional DE variants: DE/best/1/bin, DE/rand/1/bin, DE/current-to-best/bin, DE/rand-to-best/bin, and 4 state-of-the-art DE variants: JADE (Zhang and Sanderson, 2009), jDE (Brest *et al*., 2006), SaDE (Qin *et al*., 2009), and SHADE (Tanabe and Fukunaga, 2013). We also compare against the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen *et al.*, 1995) owing to the competitive performance of the latter in continuous function optimization. The DE variants are carefully chosen such that they have a well-recorded performance on the moderate and small scale optimization problems.

We undertake extensive empirical comparison among proposed method and other six state-of-art CMA-ES variants namely, IPOP-CMA-ES (Hansen & Kern, 2004), IPOP-aCMA-ES (Hansen & Ros, 2010), BIPOP-CMA-ES (Hansen, 2009), BIPOP-aCMA-ES (Loshchilov *et al*., 2012), NIPOP-aCMA-ES (Loshchilov *et al*., 2012) and NBIPOP-aCMA-ES (Loshchilov *et al*., 2012). Control parameters for all these algorithmic variants of CME-ES are set as per corresponding literature. Maximum number of FEs is limited to $10000\times D$

for all the algorithms compared, following the standard procedure of the CEC 2013 competition on real parameter optimizers.

5.2.3 Results and Discussions on 30*D* CEC 2013 Benchmark Suite

We summarize the mean error of the 9 algorithms (8 existing and the one proposed) over the CEC 2013 benchmark minimization problems in Tables 2A-2D. A closer observation of the results reveals that among the 28 functions SWDE_Success_mBLX performed better than others in 22 cases. However, SWDE_Success_mBLX performed statistically significantly better than all other peer algorithms on 19 functions, and on the remaining 3, the results are equivalent to one or more of the competitors. Among the other methods, jDE performed better in only 4 cases, but among the 4, for function F14 the result is marginally better though not statistically significantly different from that of the proposed algorithm. SaDE achieved the best result alongside SWDE_Success_mBLX in two cases, and in the third case, though SaDE achieved the minimum error, SWDE_Success_mBLX performed statistically equivalent with it, as confirmed by the Wilcoxon's rank-sum test. SHADE remained the closest competitor to SWDE_Success_mBLX and it became the best optimizer on 6 functions.

We note that among all the 28 functions, the performance of SWDE_Success_mBLX appeared statistically worse than SHADE in only two functions and these are F3 and F4, both of which are ill-conditioned with condition number of the order of 10^6. Among these, F3 is the rotated bent cigar function with a smooth but narrow ridge and F4 is the rotated discus function with smooth local irregularities and one sensitive direction. The success of an optimizer on such ill-conditioned and non-separable functions largely depends on the proper invocation of local search techniques throughout the search. Failure of SWDE_Success_mBLX on this two functions may be attributed to its unguided local search moves attained by the alternative switching of F to 0.5 and Cr to 0 (inducing a coordinate-wise descent) and through the modified current-to-best/1 strategy. SHADE induces a more finely scheduled local search by maintaining a diverse set of parameters to guide the adaptation of F and Cr as the search progresses. Thus, worse values of F and Cr in any generation does not significantly impact the parameters stored in the memory from previous generations. This, perhaps, helps SHADE to better navigate through the narrow ridge of F3 and the smooth local irregularities of F4 in the 30D search space. However, as will be evident from Table 3A, the performance of SHADE on F3 deteriorates with the increase of search space dimensionality to 50 and SWDE_Success_mBLX again appears as the best performing algorithm for F3 at 50D.

We can see that over the unimodal functions the proposed algorithm performed best on 3 functions, where one or more competitors also obtained a statistically equivalent result. Thus, on unimodal small dimensional problems our algorithm can perform equivalent if not better than the state-of-the-arts. On the multimodal functions, SWDE_Success_mBLX performed significantly better than any other algorithm on 11 functions, and for the rest, it ended up in a tie with any competing algorithm or becoming statistically equivalent to the best in terms of optimizing performance. Among the 8 composite functions SWDE_Success_mBLX performed better than any other in 7 cases, and only for F25, where SaDE performed better, the result is still

not significantly different. Hence, on the low dimensional problems SWDE_Success_mBLX can be a reasonably good DE variant, especially for solving harder multi-modal problems. We plot the average ranks of the algorithms with the help of a column graph, in Fig. 5A. As is evident from the plot, SWDE_Success_mBLX achieves best rank in average over all the 28 functions, followed by SHADE and jDE.

Comparative performance of SWDE_Success_mBLX with respect to the six state-of-the-art variants of CMA-ES are shown in Table 2E for 30*D* CEC 2013 functions. We can see that for functions F2-F4, all the CMA-ES variants outperform the proposed method as in general CMA-ES based methods are strong in elliptical-shaped and ill-conditioned search spaces. However, on the Rotated Schaffer' s function F7 as well as on rotated Ackley's function F8, SWDE_Success_mBLX perform statistically better than all the CMA-ES variants. This indicates that the randomized operators of our proposed method provides DE with better search capability on multimodal, non-separable, and asymmetric fitness landscapes even in presence of huge number of local optima. Furthermore, on more complicated functions F17 - F24, F26, and F27, the gross performance of SWDE_Success_mBLX (even with constant population size) remains superior to all the CMA-ES variants, which use the restart strategy and other auxiliary search mechanisms. This indicates the effectiveness of the balanced search moves of SWDE_Success_mBLX on hybrid composition functions (F21 - F24, F26, F27), some of which exhibit different landscape properties around different optima. Table 2E also indicates that SWDE_Success_mBLX achieves best average rank of 2.71 being followed by NBIPOP-aCME-ES (rank 3.58) and BIPOP-CMA-ES (rank 4.01) respectively.

5.2.4 Results and Discussions on 50D CEC 2013 Benchmark Suite

We now present the summary of the comparison of SWDE_Success_mBLX with 8 competitor methods, on 50-dimensional versions of the IEEE CEC 2013 benchmark optimization problems, in Tables 3A-3D. We can observe that among the 28 functions the proposed method performed best in 26 cases. However, in 3 cases the results are equal, and in 2 cases the difference in performance is statistically insignificant with other algorithms. In the 2 cases of F4 and F16 respectively, SHADE performed better than SWDE_Success_mBLX, though only in 1 case the improvement is significant. Compared to the 30D problems, SaDE and jDE performed poor with the increase in dimensions, as they reached the best-obtained minimum in 1 and 3 cases respectively, and all of them are tie scenarios. We can thus conclude, that SWDE_Success_mBLX exhibits better performance on both the small and moderate dimensional problems of diverse nature, than the current state-of-the-art. Moreover, SWDE_Success_mBLX shows a very noteworthy improvement over the hard multimodal and specially composite optimization problems. Furthermore, we draw the column graph for the average ranks of the algorithms in Fig. 5B, which clearly shows the SWDE_Success_mBLX to be the best on average over the CEC 2013 50 dimensional optimization problems, followed by SHADE, JADE, and jDE.

Comparative performance of SWDE_Success_mBLX with respect to the 6 state-of-the-art variants of CMA-ES are shown in Table 3E for 50D CEC 2013 functions. This table shows a similar trend as Table 2E. Here also the proposed algorithm is able to achieve statistically best results on 15 out of 28 functions as compared to all the CME-ES variants. Moreover, SWDE_Success_mBLX also finished with the best average rank of

2.53, closely followed by NBIPOP-aCMA-ES (rank 3.5) and BIPOP-CMA-ES (rank 3.53) respectively. This indicates that proposed algorithm can retain its performance level significantly despite the growth of dimensionality.

5.3 Comparison on Large-scale Benchmarks: The CEC 2010 Suite

In order to exhibit scalability of SWDE-Success_mBLX, CEC 2010 (Tang *et al*., 2009) large-scale Single Objective Global Optimization (LSGO) suite is considered. The benchmark suite consists of total 20 functions, each of which is 1000 dimensional and has a different degree of separability among the dimensions. Among the 20 functions, we can form five groups as follows; the first group containing 3 separable functions, the second group consisting 5 single group *m*-non-separable problems (*m* is a specified number of variables), 5 are *D*/2*m*- group *m*-non-separable, forming the third group. In the fourth group, there are 5 *D/m*- group *m*-non-separable functions and the last 2 non-separable problems create the fifth group. In terms of modality, the functions can be again divided into two groups, where the first group contains 8 unimodal problems, and the rest of the 12 functions create the second group of multimodal optimization problems. Hence, CEC 2010 test-suite not just validates the scalability of SWDE_Success_mBLX but also evaluates its resilience to the separability of dimensions.

5.3.1 Compared Algorithms

We compare the performance of SWDE_Success_mBLX with eleven state-of-art evolutionary optimization algorithms, specially tailored to perform large-scale global optimization. These algorithms are namely: Self-adaptive DE for large-scale global optimization (jDElsgo) (Brest *et al.*, 2010), DE with Multi-Level cooperative co-evolution (DECC-ML) (Omidvar *et al*., 2010), Differential Ant-Stigmergy Algorithm (DASA) (Korošec and Šilc, 2008), Dynamic Multi-Swarm Particle Swarm Optimizer with Sub-region based Harmony Search (DMS_PSO_SHS) (Zhao *et al.*, 2008), Sequential DE enhanced by Neighborhood Search (SDENS) (Wang *et al*., 2010), two-stage based Ensemble Optimization Evolutionary Algorithm (EOEA) (Wang and Li, 2010), Cooperative Co-evolution based DE with random Grouping (DECC-G) (Yang *et al.*, 2008), Multi-Level Cooperative Co-evolution for large-scale global optimization (MLCC) (Yang *et al.*, 2008), Memetic Algorithm based on local search chains for large-scale global optimization (MA-SW-Chains) (Molina *et al*. 2010), DE enhanced with Cooperative Co-evolution and Differential Grouping (DECC-DG) (Omdivar *et al.*, 2014) and DE/ best/1/bin with $F = 0.8$, $Cr = 0.9$ (standard settings). Parametric settings of the other peer algorithms were kept as per their original literature.

5.3.2 Results and Discussions: 1000*D* CEC 2010 LSGO Suite

We summarize the performances of SWDE_Success_mBLX along with 11 state-of-the-art evolutionary algorithms (designed for handling large-scale optimization problems), over the 1000 dimensional CEC 2010 benchmark test suite in Table 5A-5C. We can see that among the 20 functions, the proposed algorithm performed best on 18 problems, none of which is a tie situation. However, among the 18 problems where SWDE_Success_mBLX performed best, the difference of performance from the closest competitor is not

statistically significant in one case. DECC-ML minimized the error to the least among the 12 methods on F11, though SWDE_Success_mBLX is performing equivalently to it. The method jDElsgo performed best on F19, and here the difference of performance from the proposed method is significant. In fact, this is the only function where SWDE_Success_MBLX performs statistically worse than the best algorithm (jDElsgo). Once again we note that it is a unimodal, non-separable function and thus, the success of jDElsgo can be attributed to its intensive (exploitative) search of smaller volume by using a very narrow interval of [0.01,0.1] for randomizing the value of *F* and thus, inducing a *scale-factor local search* (Brest *et al*., 2010) to improve the best population member. Also, better performance in mean value implies the robustness of jDElsgo is improved on F19 due to the dynamic population reduction mechanism, which can be trivially integrated with SWDE_Success_mBLX. To avoid taking any costly feedback from the search landscape, SWDE_Success_mBLX employs uniform randomization while changing the values of *F* between 0.5 and 2. Such switching induces highly explorative search, which is especially beneficial for the multi-modal and rugged functions, a fact that is evident from the results of SWDE_Success_mBLX on rest of the functions.

Thus by looking at the gross performance, we can conclude that SWDE_Success_mBLX can retain its consistent performance even over the higher dimensional problems with group-wise epistasis among the dimensions (decision variables). Moreover, the performance of the proposed method is not much dependent on the degree of separability of a problem. The performance of SWDE_Success_mBLX is also consistent, as observed from the chart of average rank illustrated in Fig. 6. Similar to the case of CEC 2013, here also SWDE_Success_mBLX achieved the lowest average rank of 1.15, whereas the closest follower jDELsgo has a rank 4.3.

5.4 Comparison on 2000 Dimensional Large-scale Benchmarks: The CEC 2008 Suite

The performance of SWDE_Success_mBLX is compared with its own predecessor, SWDE_Success (Das *et al.*, 2015), and other two popular evolutionary algorithm for large-scale optimization, namely CCPSO2 (Cooperative Co-evolutionary Particle Swarm Optimization) (Li and Yao, 2012), and Sep-CMA-ES (Ros and Hansen, 2008) which is a well known variant of Covariance Matrix Adaptation Evolutionary Strategy. Table 7 summarizes the obtained results on 2000*D* three selected function F1 (Shifted Sphere Function, Unimodal, Separable), F3 (Shifted Rosenbrock's Function, Multi-modal, Non-separable) and F7 (Fast Fractal Function, Multi-modal, Non-separable) from IEEE CEC 2008 benchmark suite (Tang *et al*., 2007). SWDE_Success_mBLX performs best in all three test functions.

5.5 Comparison on Different Component Configurations of SWDE_Success_mBLX

The proposed SWDE_Success_mBLX is a modified and extended version of SWDE_Success (Das *et al*., 2015), where a new crossover mBLX is introduced in the framework, and the choice of mutation operators are revised along with the introduction of a modified DE/current-to-best/1 mutation scheme. This requires us to validate if the above-mentioned changes can significantly improve the performance from the previous proposal. In order to check this, we designed a small experiment setup, where we compare SWDE_Success_mBLX with its predecessor SWDE_Success. To check the compatibility of mBLX with the success based revised mutation scheme, we integrate the proposed mBLX crossover with the modified

current-to-best mutation scheme (introduced in this study) and retained the general DE framework, calling it DE/current-to-best/mBLX. Furthermore, to validate whether mBLX alone without binomial crossover can lead to a significant improvement, we coupled mBLX with the popular DE/rand/1 scheme, naming the resultant DE/rand/1/mBLX. To save space, we show the simulation results on 7 representative functions from the CEC 2013 benchmark suite. All the functions are 30 dimensional and distinct in nature (F1 and F4 are unimodal, F7 and F9 are multimodal and F24, F26 and F28 are composite, for further details see Table 2). The experimental procedure is kept similar to the previously mentioned. We demonstrate the performance of the 4 algorithms on the 7 optimization problems in the following Table 6. A closer observation reveals, SWDE_Success_mBLX as the winner in all of the 7 minimization problems. Moreover, the proposed performed significantly better than SWDE_Success over all of the 7 functions, indicating the inclusion of mBLX and modified mutation strategy is indeed fruitful. Furthermore, SWDE_Sucees_mBLX retained a significantly better performance than DE/current-to-best/mBLX and DE/rand/1/mBLX as well. Thus, the individual crossover and mutation scheme can perform significantly better with the addition of the switching, validating the importance of the process. The entire comparison helps us to conclude that the performance of individual components like mBLX, modified current-to-best mutation and switching technique, can be boosted significantly further when they are coupled together, along with the popular techniques like the binomial crossover.

5.6 Sustaining diversity through SWDE_Success_mBLX

We would like to illustrate here that SWDE_Success_mBLX can indeed conserve the diversity of the population to a considerable extent, retaining vital information about interesting regions (probably containing the global optimum) in the process. A way to measure the diversity of a population P_G at generation G, can be the average distance of the population from the mean vector, as proposed by (Hu *et al*., 2007), in the following way:

$$diversity(P_G) = \frac{1}{Np \times L} \sum_{i=1}^{Np} \sqrt{\sum_{j=1}^{D} (x_{j,i} - \bar{x}_j)^2}, \quad (7)$$

where L is the length of the longest diagonal in the search space of dimension D and $\bar{x}_j = 1/Np \sum_{i=1}^{Np} x_{j,i}$ is the average value of the j^{th} dimension of the population members. Variation of population diversity (defined in (7)) with respect to the number of generations, for the population of SWDE_Success_mBLX (indicated by red line) and DE/rand/1/bin (shown by blue line and F=0.8, Cr=0.9) is plotted in Figures 5 (a), (b) and (c), for three 50D functions namely, Sphere function (F1), rotated Rosenbrock's function (F6) and Composite function number 2 (F22), from the CEC 2013 benchmark suite (Liang *et al*., 2013). The Fig. 7 clearly illustrates the fact that the population of SWDE_Success_mBLX never completely loses the diversity, thus limiting the chance of any premature convergence.

5.7 Running Time of SWDE_Success_mBLX: A Comparative View

As can be seen from Section 4, SWDE_Success_mBLX does not induce any significant computational load on plain DE. Thus, if the algorithm is stopped after a fixed number of generations G_{max}, then the asymptotic runtime complexity remains O($Np.D.G_{max}$) (Das *et al*., 2009). This fact is experimentally demonstrated through Fig. 8, where we show the average run time of each algorithm on the CEC 2013 test functions in 30D. As it can be observed, the average run-time of SWDE_Success_mBLX is marginally higher than that of DE/best/1/bin and comparable to DE/rand/1bin and DE/current-to-best/1/bin. Whereas, due to the need for maintaining external archive and sampling from parameterized distributions, the run-time of JADE and SHADE are quite high and almost equal (given that SHADE is a modified version of JADE). jDE and SaDE have their runtimes higher than the basic DE variants but lesser than JADE and SHADE.

6. Conclusion

This paper presents a new variant of DE named as SWDE_Success_mBLX, with enhanced scalability, in an attempt to address the problem of improving the performance of DE at the time of optimizing functions with dimensions varying in the large range between small (30) to large (1000). The proposed variant uses a simple switching scheme (in a uniformly random manner) for the two key control parameters of DE, namely *F* the scale factor and the *Cr* the crossover rate used in binomial crossover technique. SWDE_Success_mBLX uses two mutation strategies (the common DE/rand/1, and a modified DE/current-to-best introduced in this study) and applies a simple success (indicated by the replacement of the target by the generated trial in the selection phase) driven strategy selection process, depending on the ability to generate a better trial vector by a mutation scheme only in the previous generation.

With the increase in the number of dimensions, the search volume also increases exponentially, thus making it more challenging to search for even a near-optimal solution. Thus to perform both of the explorative searches to cover the space as well as exploitative searches for reaching an accurate global optimum a judicious balance is required. Such a demand can be met by randomly choosing *F* and *Cr* between their extreme possible values. Our results demonstrate that a combination of an unconventional large enough value of *F* (= 2) and a very low value (= 0.5) is found to be very useful for optimizing real parameter large-scale problems. In contrast to some of the premier algorithms designed for the purpose (like JADE, SaDE, CoDE etc.) that sample *F* from the interval of (0.4, 1) and *Cr* from (0, 1), our results reveal that *F* and *Cr* values, when taken as the extreme boundaries of their allowable ranges, contain important information useful for better optimization, rather than a value lying inside the range. However, this claim requires investigations (both analytical and empirical) to formally ascertain it as a fact, which can be a future direction of this current research.

Not only SWDE_Success_mBLX restricts the success-based scheme selection in the mutation stage, but also extends it to the crossover phase. The paper introduces a new crossover scheme called mBLX, which is suitable for application in DE, and influenced by the BLX-α-β crossover originally proposed for GAs. The proposed algorithm uses a success-based selection (similar to the method used for mutation) between the

widely used binomial and the proposed mBLX crossover operator to extend the searching capability required for large-scale optimization in DE.

The future directions may also include a complete analytical study of SWDE_Success_mBLX, formally explaining its high performance and dynamics. Also, the parameter switching strategy, success based selection of mutation and crossover schemes, and the proposed mBLX crossover operator, may be further investigated in other optimization scenarios like for multi-objective, constrained and dynamic problems. To improve the robustness, a dynamic population reduction mechanism, similar to jDElsgo can be integrated with SWDE_Success_mBLX since the current framework keeps the population size a constant for the sake of computational simplicity. The suggested modifications can be integrated with other multi-population DE models like the one proposed by Piotrowski *et al*. (2012) for large-scale optimization problems.

References

Brest J., Greiner S., Boskovič B., Mernik M.,& Zumer V. (2006) Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Trans. Evol. Comput*., 10(6), 646–657.

Brest, J., & Maučec, M. S. (2011). Self-adaptive differential evolution algorithm using population size reduction and three strategies. *Soft Computing*, 15(11), 2157-2174.

Brest, J., Zamuda, A., Fister, I., & Maučec, M. S. (2010). Large scale global optimization using self-adaptive differential evolution algorithm. In *2010 IEEE Congress on Evolutionary Computation* (*CEC*), (pp. 1-8). IEEE.

Das, S., Abraham, A., Chakraborthy, U. K., Konar, A., (2009), Differential evolution using a neighborhood-based mutation operator, *IEEE Transactions on Evolutionary Computation*, Vol. 13, pp. 526–553.

Das, S., & Suganthan, P. N. (2011). Differential evolution: a survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1), 4-31.

Das S., Ghosh A. & Mullick S. S. (2015) A switched parameter differential evolution for large scale global optimization–simpler may be better, In R. Matousek (ed.), *MENDEL 2015, Recent Advances in Soft Computing*, Springer, pp. 103-125.

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution–An updated survey. *Swarm and Evolutionary Computation*, 27, 1-30.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1), 3-18.

Draa A., Bouzoubia S., & Boukhalfa I. (2015). A sinusoidal differential evolution algorithm for numerical optimization. *Applied Soft Computing*, 27, 99-126.

Elsayed, S. M., Sarker, R. A., & Essam, D. L. (2011). Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, (pp. 1041-1048).

Elsayed, S. M., Sarker, R. A., & Ray, T. (2013). Differential evolution with automatic parameter configuration for solving the CEC2013 competition on real-parameter optimization. In *2013 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1932-1937.

Epitropakis, M. G., Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P., & Vrahatis, M. N. (2011). Enhancing differential evolution utilizing proximity-based mutation operators. *IEEE Transactions on Evolutionary Computation*, 15(1), 99-119.

Eshelman, L. J., & Schaffer, D. J. (1993). Real-coded genetic algorithms and interval-schemata. *Foundations of Genetic Algorithms*, 2, pp. 187 - 202.

Foli, K., Okabe, T., Olhofer, M., Jin, Y., & Sendhoff, B. (2006). Optimization of micro heat exchanger: CFD, analytical approach and multi-objective evolutionary algorithms. *International Journal of Heat and Mass Transfer,49*(5), 1090-1099.

Gong, W., Cai, Z., Ling, C. X., & Li, H. (2011). Enhanced differential evolution with adaptive strategies for numerical optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(2), 397-413.

Gong, W. and Cai, Z. (2012), Adaptive parameter selection for strategy adaptation in differential evolution for continuous optimization, *Journal of Computers*, Vol. 7 (3), 2012.

Hansen, N., A. Ostermeier and A. Gawelczyk (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. In L. Eshelman (Ed.),*Proceedings of the Sixth International Conference on Genetic Algorithms*, Pittsburgh, pp. 57-64. Morgan Kaufmann

Hansen, N., & Kern, S. (2004, September). Evaluating the CMA evolution strategy on multimodal test functions. In International Conference on Parallel Problem Solving from Nature (pp. 282-291). Springer Berlin Heidelberg.

Hansen, N., & Ros, R. (2010, July). Benchmarking a weighted negative covariance matrix update on the BBOB-2010 noiseless testbed. In Proceedings of the 12th annual conference companion on Genetic and evolutionary computation (pp. 1673-1680). ACM.

Hansen, N. (2009, July). Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers (pp. 2389-2396). ACM.

Hu, J., Zeng, J., & Tan, Y. (2007). A diversity-guided particle swarm optimizer for dynamic environments. In *Bio-inspired computational intelligence and applications* (pp. 239-247). Springer Berlin Heidelberg.

Islam, S. M., Das, S., Ghosh, S., Roy, S., & Suganthan, P. N. (2012). An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(2), pp. 482-500.

Korošec, P. & Šilc, J. (2008). The differential ant-stigmergy algorithm for large scale real-parameter optimization, *Ant Colony Optimization and Swarm Intelligence*, Vol. 5217 of the series Lecture Notes in Computer Science, pp 413-414 Springer Berlin Heidelberg.

Liang, J. J., Qu, B. Y., Suganthan, P. N., & Hernández-Díaz, A. G. (2013). Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 2012*.

Li, X., & Yao, X. (2012). Cooperatively coevolving particle swarms for large scale optimization. *IEEE Transactions on Evolutionary Computation*, *16*(2), 210-224.

Loshchilov, I., Schoenauer, M., & Sebag, M. (2012, September). Alternative restart strategies for CMA-ES. In *International Conference on Parallel Problem Solving from Nature* (pp. 296-305). Springer Berlin Heidelberg.

Mahdavi, S., Shiri, M. E., & Rahnamayan, S. (2015). Metaheuristics in large-scale global continues optimization: A survey. *Information Sciences*, 295, 407-428.

Mallipeddi, R. (2013). Harmony search based parameter ensemble adaptation for differential evolution. *Journal of Applied Mathematics*, Vol. 2013 (2013), Article ID 750819.

Mallipeddi, R., & Suganthan, P. N. (2010). Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies. In *Swarm, Evolutionary, and Memetic Computing Conference* (SEMCCO)(pp. 71 - 78). Springer Berlin Heidelberg.

Mallipeddi, R., Suganthan, P. N., Pan, Q. K., & Tasgetiren, M. F. (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, 11(2), 1679 - 1696.

Mezura-Montes E., Vel´asquez-Reyes J. and Coello Coello C. A. (2006): A Comparative Study of Differential Evolution Variants for Global Optimization, in Maarten Keijzer et al. (editors), 2006 *Genetic and Evolutionary Computation Conference* (*GECCO*'2006), Vol. 1, ACM Press, Seattle, Washington, USA, July, pp. 485–492.

Molina, D., Lozano, M., & Herrera, F. (2010). MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization. In *2010 IEEE Congress on Evolutionary Computation (CEC)*, (pp. 1-8).

Omidvar, M. N., Li, X., & Yao, X. (2010). Cooperative co-evolution with delta grouping for large scale non-separable function optimization. In *2010 IEEE Congress on Evolutionary Computation* (*CEC*), (pp. 1-8).

Omidvar, M. N., Li, X., Mei, Y., & Yao, X. (2014). Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 18(3), 378-393.

Parsopoulos, K. E. (2009). Cooperative micro-differential evolution for high-dimensional problems. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation* (pp. 531-538). ACM.

Picek, S., & Jakobovic, D. (2014). From fitness landscape to crossover operator choice. In *Proceedings of the 2014 conference on Genetic and evolutionary computation* (pp. 815-822). ACM.

Picek, S., Jakobovic, D., & Golub, M. (2013). On the recombination operator in the real-coded genetic algorithms. In *Evolutionary Computation (CEC), 2013 IEEE Congress on* (pp. 3103-3110).

Piotrowski, A. P., Napiorkowski, J. J., Kiczko, A. (2012), Differential Evolution algorithm with Separated Groups for multi-dimensional optimization problems, *European Journal of Operational Research*, Vol. 273, 216 (2012) 33–46.

Pranav, P. and Jeyakumar, G. (2015). Control parameter adaptation strategies for mutation and crossover rates of differential evolution algorithm - an insight. 2015 *IEEE International Conference on Computational Intelligence and Computing Research* (ICCIC), DOI:10.1109/ICCIC.2015.7435788.

Posík, P., Kubalík, J., (2012), Experimental comparison of six population-based algorithms for continuous black box optimization, *Evolutionary Computation*, Vol. 20(4), pp. 483 - 508.

Qin, A. K., Huang, V. L., & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 13(2), 398-417.

Ronkkonen, J., Kukkonen, S., & Price, K. V. (2005). Real-parameter optimization with differential evolution. In *Proc. IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 506-513, Edinburgh, Scotland.

Ros, R., & Hansen, N. (2008). A simple modification in CMA-ES achieving linear time and space complexity. In *International Conference on Parallel Problem Solving from Nature* (pp. 296-305). Springer Berlin Heidelberg.

Sarker, R. A., Elsayed, S. M., & Ray, T. (2014). Differential evolution with dynamic parameters selection for optimization problems. *IEEE Transactions on Evolutionary Computation*, 18(5), 689-707.

Sonoda, T., Yamaguchi, Y., Arima, T., Olhofer, M., Sendhoff, B., & Schreiber, H. A. (2003). Advanced High Turning Compressor Airfoils for Low Reynolds Number Condition: Part 1—Design and Optimization. In *ASME Turbo Expo 2003, collocated with the 2003 International Joint Power Generation Conference* (pp. 437-450). American Society of Mechanical Engineers.

Spears, W. M. (1995). Adapting crossover in evolutionary algorithms. In 4-*th Annual Conference on Evolutionary Programming*, pp. 367 - 384, San Diego, CA.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4), 341-359.

Tanabe, R., & Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. In *2013 IEEE Congress on Evolutionary Computation (CEC)*, pp. 71-78, Cancun, Mexico.

Tang, K., Yáo, X., Suganthan, P. N., MacNish, C., Chen, Y. P., Chen, C. M., & Yang, Z. (2007). Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. *Nature Inspired Computation and Applications Laboratory, USTC, China*, 153-177.

Tang K., Li X., Suganthan P. N., Yang Z. & Weise T. (2009). Benchmark Functions for the CEC'2010 Special Session and Competition on Large Scale Global Optimization, *Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, http://nical.ustc.edu.cn/cec10ss.php*, & Nanyang Technological University.

Uymaz, S. A., Tezel, G., and Yel, E., Artificial algae algorithm with multi-light source for numerical optimization and applications, *Biosystems*, Vol. 138, Pages 25–38, Dec. 2015.

Wang, C., & Gao, J. (2012). High-dimensional waveform inversion with cooperative coevolutionary differential evolution algorithm. *IEEE Geoscience and Remote Sensing Letters*, 9(2), 297-301.

Wang, H., Wu, Z., & Rahnamayan, S. (2011). Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems. *Soft Computing*, *15*(11), 2127-2140.

Wang, H., Wu, Z., Rahnamayan, S., & Jiang, D. (2010). Sequential DE enhanced by neighborhood search for large scale global optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on* (pp. 1-7).

Wang, Y., & Li, B. (2010, July). Two-stage based ensemble optimization for large-scale global optimization. In *2010 IEEE Congress on Evolutionary Computation (CEC)*, (pp. 1-8).

Wang, Y., Cai, Z., & Zhang, Q. (2011). Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation*, 15(1), 55-66.

Wang, Y., & Li, B. (2010). Two-stage based ensemble optimization for large-scale global optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on* (pp. 1-8). IEEE.

Weber, M., Neri, F., & Tirronen, V. (2011). Shuffle or update parallel differential evolution for large-scale optimization. *Soft Computing*, *15*(11), 2089-2107.

Wu, G., Mallipeddi, R., Suganthan, P. N., Wang, R., & Chen, H. (2016). Differential evolution with multi-population based ensemble of mutation strategies. *Information Sciences*, *329*, 329-345.

Yang, Z., Tang, K., & Yao, X. (2008). Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15), 2985-2999.

Yang, Z., Tang, K., & Yao, X. (2008). Multilevel cooperative coevolution for large scale optimization. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on* (pp. 1663-1670). IEEE.

Zaharie, D. (2002). Critical values for the control parameters of differential evolution algorithms. In *Proceedings of MENDEL* (Vol. 2, p. 6267).

Zaharie D. (2009) Influence of crossover on the behavior of the Differential Evolution Algorithm, *Applied Soft Computing*, 9(3), pg. 1126-1138.

Zamuda, A., Brest, J., & Bošković, B. (2008). Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on* (pp. 3718-3725). IEEE.

Zhang, J., & Sanderson, A. C. (2009). JADE: adaptive differential evolution with optional external archive. *Evolutionary Computation, IEEE Transactions on*, *13*(5), 945-958.

Zhao, S. Z., Liang, J. J., Suganthan, P. N., & Tasgetiren, M. F. (2008). Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*(pp. 3845-3852).

Zhao, S. Z., Suganthan, P. N., & Das, S. (2011). Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. *Soft Computing*, 15(11), 2175-2185.

Zhao, Z., Yang, J., Hu, Z., and Che, H. (2016), A differential evolution algorithm with self-adaptive strategy and control parameters based on symmetric Latin hypercube design for unconstrained optimization problems, *European Journal of Operational Research*, Vol. 250, Issue 1, Pages 30-45.

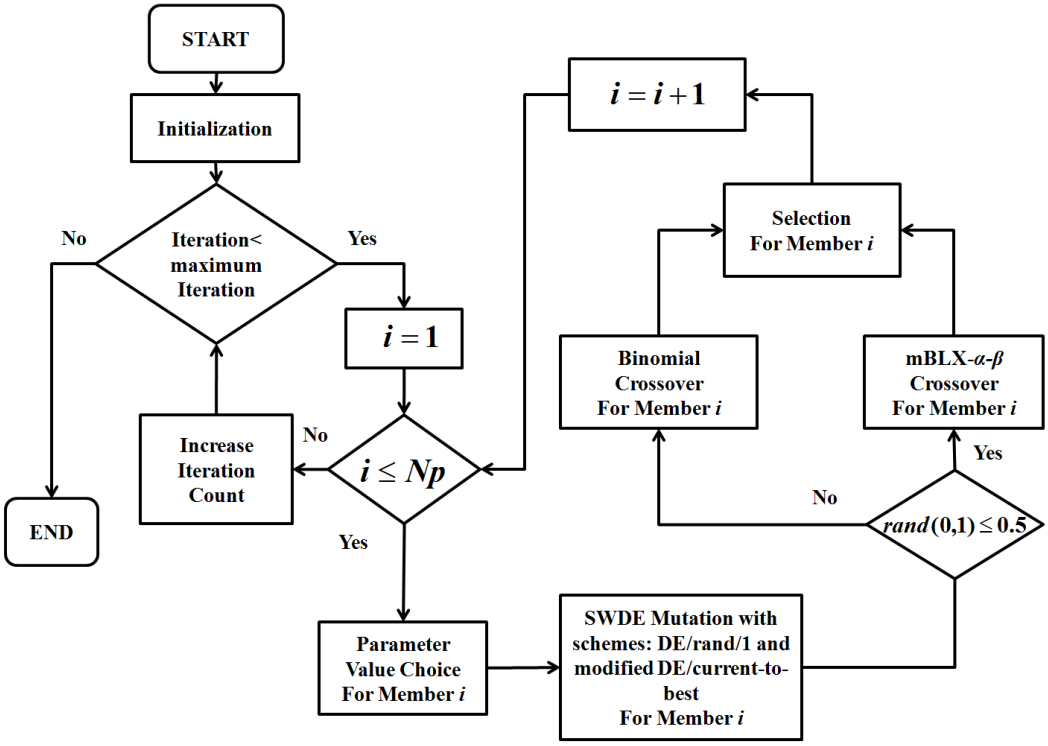
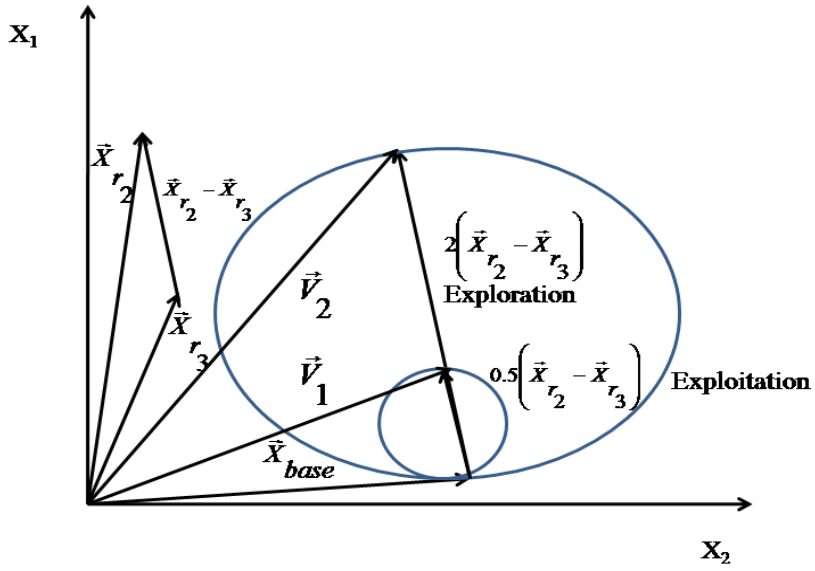
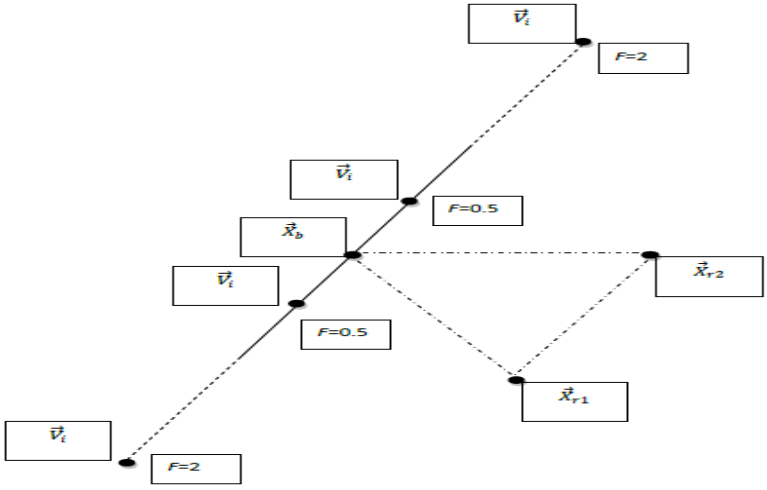


Fig. 1: Flow diagram of the complete SWDE_Success_mBLX method

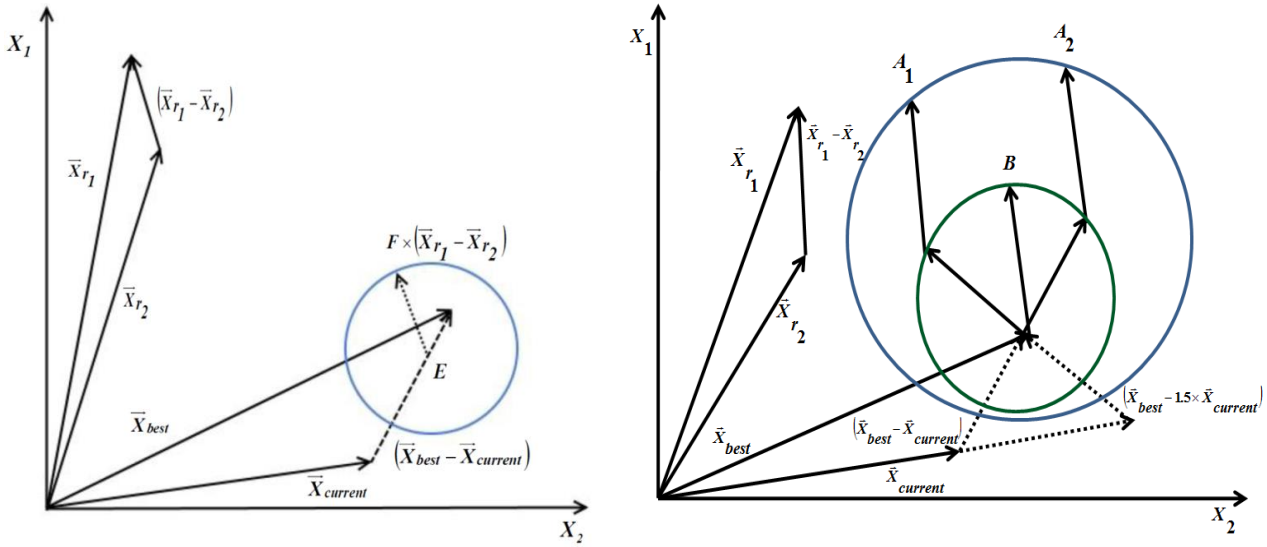


a) Donor formation with respect to an actual 2D search space



(b) Possible donor formation as an affine combination of the base vector with two other vectors for extreme values of F

Fig. 2: Effect of $F = 2$ and $F = 0.5$ in DE mutation



(a) the DE/current-to-best/1 scheme

(b) the modified DE/current-to-best/1 scheme

Fig. 3. Search regions of the DE/current-to-best/1 scheme and its proposed variant.

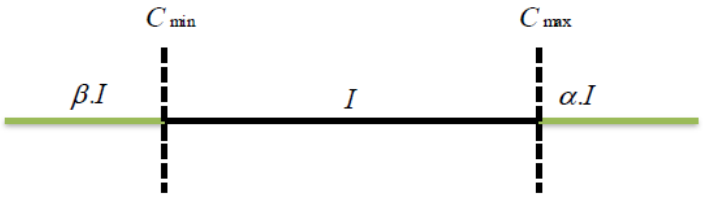


Fig. 4: mBLX-α-β recombination scheme in DE

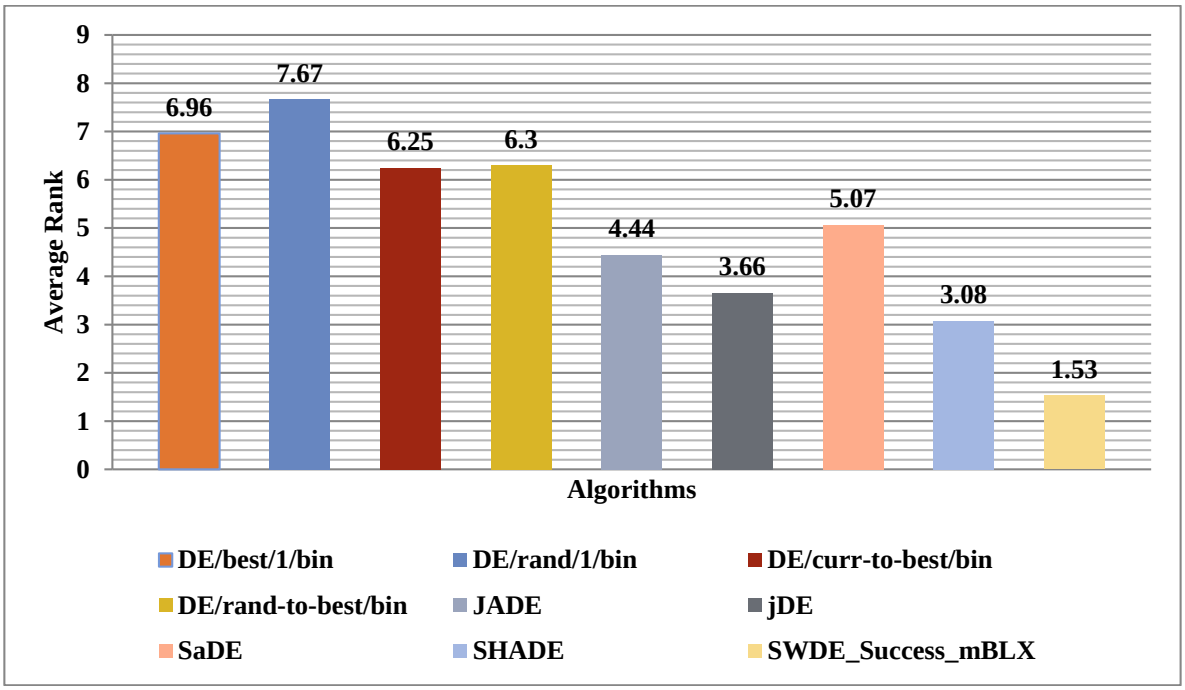


Fig. 5A. Average ranks of SWDE_Success_mBLX and other peer algorithms for CEC 2013 30*D* problems.

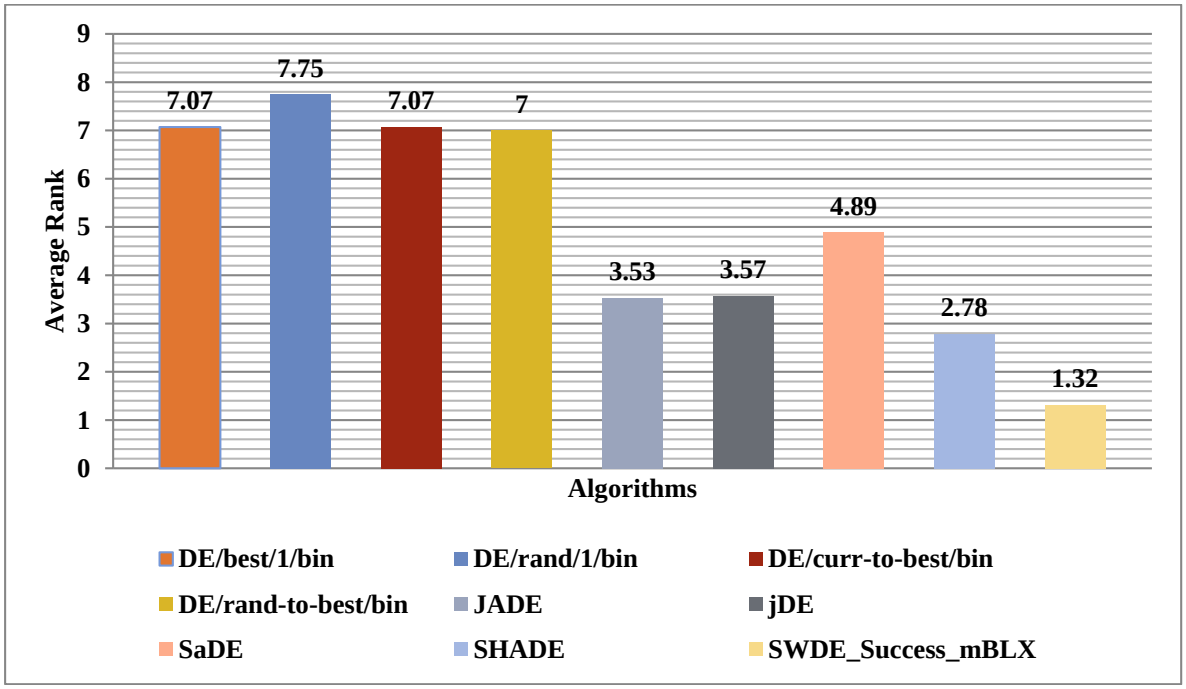


Fig. 5B. Average ranks of SWDE_Success_mBLX and other peer algorithms for CEC 2013 50*D* problems.

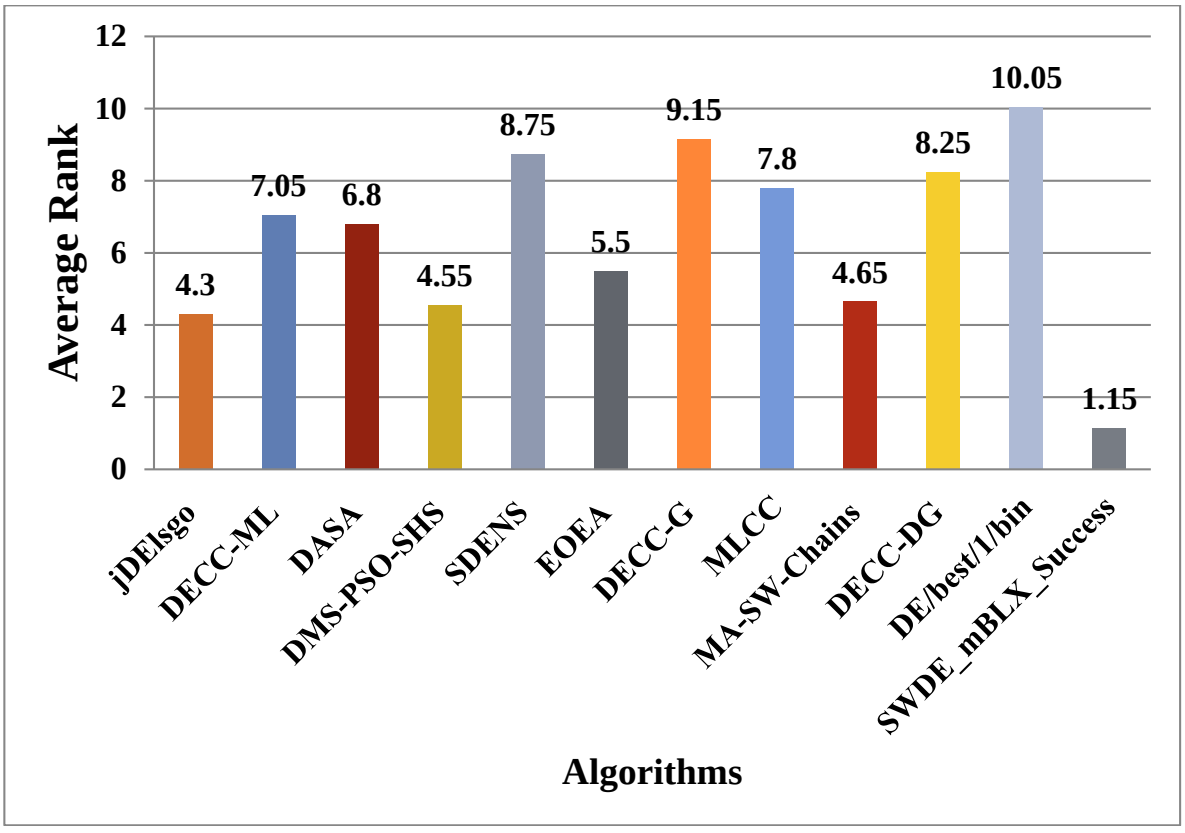
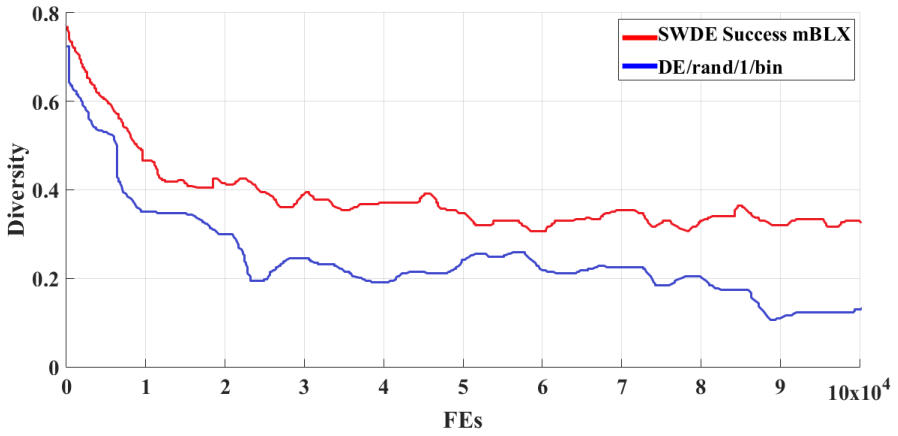
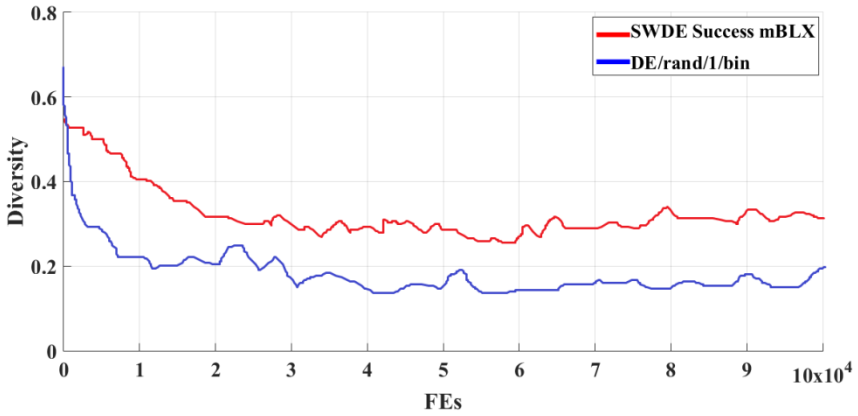


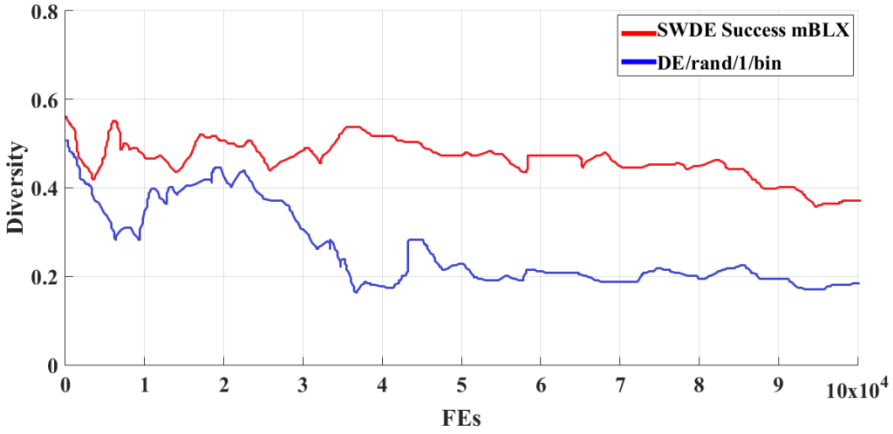
Fig. 6: Comparison of ranks of SWDE_Success_mBLX and other algorithms for CEC 2010 1000*D* problems.



(a) Sphere function (F1)



(b) Rotated Rosenbrock's function (F6)



(c) Composite function 2 (F22)

Fig. 7. Comparison of the variation of population diversity with number of iterations for F1, F6 and F22 of CEC 2013 (D = 50) between SWDE_Success_mBLX and standard DE/rand/1/bin (F = 0.8, Cr = 0.9).

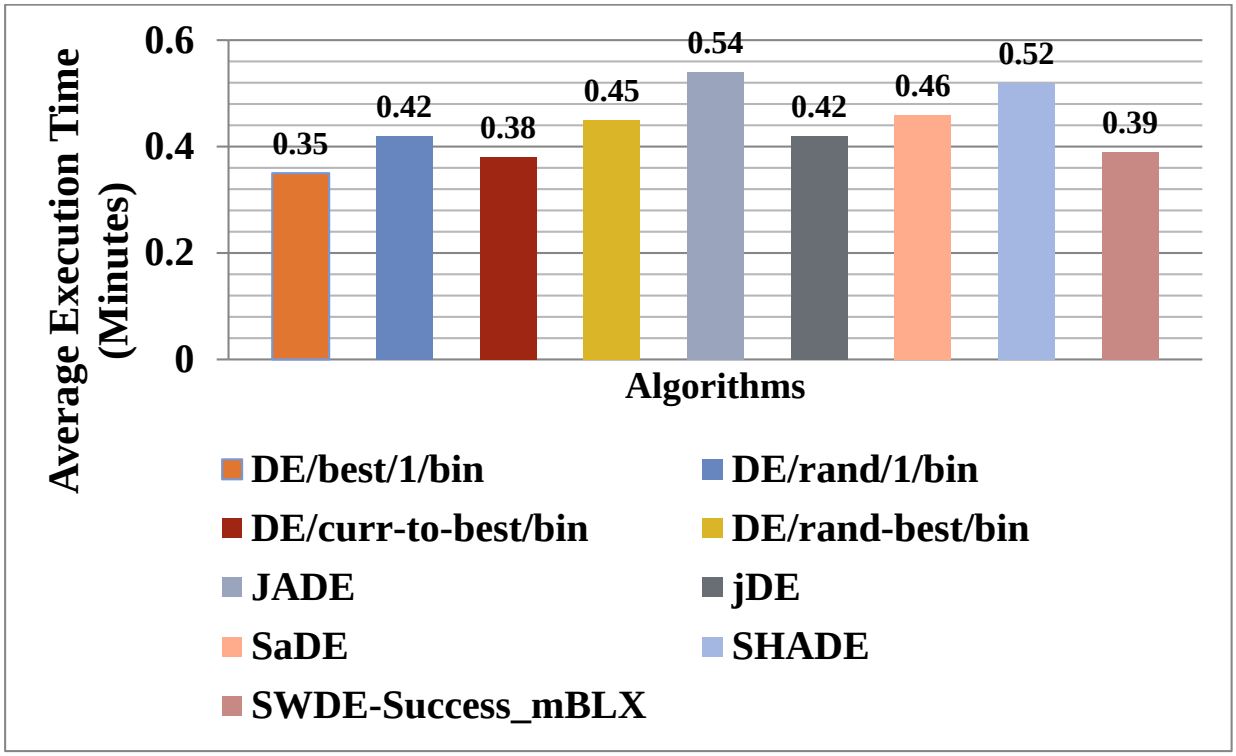


Fig. 8. A comparative view of the per function run time of the DE variants compared.

Table 1: Summary of CEC 2013 benchmark functions (Liang *et al*., 2013; Uymaz *et al*., 2015)

Fn. No.	Functions	Properties
1.	Sphere Function	Unimodal, Separable
2.	Rotated High Conditioned Elliptic function	Unimodal, Non-Separable, Quadratic ill Conditioned, Smooth Local Irregularities
3.	Rotated Bent Cigar Function	Unimodal, Non-Separable, Smooth but Narrow Ridge
4.	Rotated Discus Function	Unimodal, Non-Separable, Asymmetrical, Smooth Local Irregularities
5.	Different Powers Function	Unimodal, Separable
6.	Rotated Rosenbrock's Function	Multi-modal, Non-Separable
7.	Rotated Schaffers F7 Function	Multi-modal, Non-Separable, Asymmetrical
8.	Rotated Ackley's Function	Multi-modal, Non-Separable, Asymmetrical
9.	Rotated Weierstrass Function	Multi-modal, Non-Separable, Asymmetrical, Limited Differentiability
10.	Rotated Griewank's Function	Multi-modal, Non-Separable, Rotated
11.	Rastrigin's Function	Multi-modal, Separable, Asymmetrical
12.	Rotated Ratrigin's Function	Multi-modal, Non-Separable, Asymmetrical
13.	Non-Continuous Rotated Ratrigin's Function	Multi-modal, Non-Separable, Rotated, Asymmetrical, Non- continuous
14.	Schwefel Function	Multi-modal, Non-Separable, Rotated, Asymmetrical, Large Separation between local and global optima
15.	Rotated Schwefel Function	Multi-modal, Non-Separable, Asymmetrical, Large Separation between local and global optima
16.	Rotated Katsuura Function	Multi-modal, Non-Separable, Asymmetrical, Continuous but non differentiable
17.	Lunacek Bi_Rastrigin Function	Multi-modal, Non-Separable, Asymmetrical, Continuous but non differentiable
18.	Rotated Lunacek Bi_Rastrigin Function	Multi-modal, Non-Separable, Rotated, Asymmetrical, Continuous but non differentiable
19.	Expanded Griewank's plus Rastrigin's Function	Multi-modal, Non-Separable,
20.	Expanded Schaffer's F6 Function	Multi-modal, Non-Separable, Asymmetrical
21.	Composite Function 1	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
22.	Composite Function 2	Multi-modal, Separable, Asymmetrical, Each local optima has different properties
23.	Composite Function 3	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
24.	Composite Function 4	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
25.	Composite Function 5	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
26.	Composite Function 6	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
27.	Composite Function 7	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties
28	Composite Function 8	Multi-modal, Non-Separable, Asymmetrical, Each local optima has different properties

Table 2A: Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F1-F7, *D*=30). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithm \ Functions		F1	F2	F3	F4	F5	F6	F7	Win/Tie/ Loss
DE/best/1/ bin	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4.5)**	3.62e+08† 1.31e+08 (8)	4.63e+07† 4.03e+07 (8)	1.43e+05† 2.44e+04 (8)	1.13e+00† 0.00e+00 (9)	2.48e+01† 3.63e-01 (6)	1.49e+02† 8.04e+01 (8)	0/1/6
DE/rand/1 /bin	Mean SD (Rank)	6.35e-09† 2.19e-09 (9)	7.29e+08† 2.05e+08 (9)	4.24e+10† 1.47e+10 (9)	2.10e+05† 4.71e+04 (9)	1.89e-04† 5.04e-05 (6)	2.61e+01† 1.26e-01 (8)	4.89e+02† 5.81e-01 (9)	0/0/7
DE/curr.- to-best/bin	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4.5)**	9.68e+07† 7.09e+07 (6)	1.39e+06† 1.20e+06 (4)	1.26e+05† 1.75e+04 (7)	1.14e-13† 0.00e+00 (8)	2.39e+01† 1.97e-01 (5)	9.12e+01† 5.67e+01 (7)	0/1/6
DE/rand- best/bin	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4.5)**	1.28e+08† 2.05e+07 (7)	3.17e+07† 2.56e+07 (7)	1.15e+05† 6.76e+03 (6)	1.13e-13† 0.00e+00 (7)	2.54e+01† 5.02e-01 (7)	6.43e+01† 4.58e+01 (6)	0/1/6
JADE	Mean SD (Rank)	**0.00e+00**≈ **0.00e+00** **(4.5)**	7.12e+03† 4.59e+03 (2)	4.09e+05† 1.19e +06 (3)	4.60e +03† 1.46e +04 (5)	**0.00e+00** ≈ **0.00e+00** **(3)**	1.53e+00† 6.38 e+00 (3)	3.90e+00† 4.10e+00 (3)	0/2/5
jDE	Mean SD (Rank)	**0.00e+00**≈ **0.00e+00** **(4.5)**	1.14e+05† 7.96e+04 (4)	1.55e+06† 2.16e +06 (5)	2.37e +01† 2.88e +01 (2)	**0.00e+00** ≈ **0.00e+00** **(3)**	1.36e+01† 4.74e+00 (4)	2.80e+00† 2.42e+00 (2)	0/2/5
SaDE	Mean SD (Rank)	**0.00e+00**≈ **0.00e+00** **(4.5)**	3.13e+05† 1.63e+05 (5)	1.25e+07† 2.97e +07 (6)	3.17e +03† 1.67e +03 (4)	**0.00e+00** ≈ **0.00e+00** **(3)**	3.37e+01† 2.59e+01 (9)	2.66e+01† 1.12e+01 (5)	0/2/5
SHADE	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4.5)**	9.00e+03† 7.47e+03 (3)	**4.02e+01** **2.12e+02** **(1)**	**1.92e-04** **3.01e-04** **(1)**	**0.00e+00** ≈ **0.00e+00** **(3)**	5.95e-01† 3.72e+00 (2)	4.60e+00† 5.38e+00 (4)	2/2/3
SWDE- Success_ mBLX	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4.5)**	**1.65e+01** **5.07e+01** **(1)**	2.45e+03† 1.65e +04 (2)	8.91e +01† 9.75e +03 (3)	**0.00e+00** ≈ **0.00e+00** **(3)**	**2.59e-11** **5.83e+01** **(1)**	**7.21e-02** **2.65e-01** **(1)**	**3/2/2**

Table 2B: Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F8 - F14, *D*=30). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F8	F9	F10	F11	F12	F13	F14	Win/Tie/ Loss
DE/best/1/bin	Mean SD (Rank)	2.09e+01† 4.38e-02 (5.5)	3.91e+01† 9.07e-01 (7.5)	2.53e-02† 2.18e-02 (4)	1.98e+01† 7.71e+00 (7)	2.03e+02† 1.09e+01 (8)	2.10e+02† 7.01e+00 (8)	1.05e+03† 1.92e+02 (7)	0/0/7
DE/rand/1/bin	Mean SD (Rank)	2.09e+01† 1.89e-02 (5.5)	3.93e+01† 7.23e-01 (9)	3.16e+02† 1.21e+02 (9)	2.49e+01† 2.90e+00 (8)	2.13e+02† 1.02e+01 (9)	2.25e+02† 7.77e+00 (9)	7.20e+02† 1.22e+02 (6)	0/0/7
DE/curr.-to-best/bin	Mean SD (Rank)	2.09e+01† 2.57e-02 (5.5)	3.91e+01† 9.65e-01 (7.5)	2.13e-02† 9.95e-03 (3)	5.67e+01† 4.20e+01 (9)	1.91e+02† 5.98e+00 (6)	1.95e+02† 3.45e+00 (7)	5.93e+03† 1.62e+02 (9)	0/0/7
DE/rand-best/bin	Mean SD (Rank)	2.09e+01† 1.31e-02 (5.5)	3.87e+01† 4.65e-01 (6)	1.39e-02† 1.13e-02 (2)	1.00e+01† 2.75e+00 (6)	1.97e+02† 2.34e+00 (7)	1.91e+02† 1.14e-01 (6)	4.56e+03† 3.30e+02 (8)	0/0/7
JADE	Mean SD (Rank)	2.69e+01† 9.90e-02 (9)	2.29e+01† 1.44e+00 (3)	3.10e-02† 2.18e-02 (5)	**0.00e+00**≈ **0.00e+00** **(2.5)**	2.49e+01† 5.14e+00 (3)	4.55e+01† 1.34e+01 (2)	3.28e-02† 2.45e-02 (4)	0/1/6
jDE	Mean SD (Rank)	2.06e+01† 4.71e-02 (2)	2.55e+01† 3.88e+00 (4)	3.53e-02† 2.14e-02 (6)	**0.00e+00**≈ **0.00e+00** **(2.5)**	6.05e+01† 9.11e+00 (5)	9.15e+01† 1.65e+01 (4)	**2.45e-03** **6.76e-03** **(1)**	1/1/5
SaDE	Mean SD (Rank)	2.19e+01† 4.25e-02 8)	**1.15e+01** **2.52e+00** **(1)**	2.31e-01† 1.84e-01 (8)	2.34e-01† 6.67e-01 (5)	4.92e+01† 1.91e+01 (4)	9.99e+01† 1.68e+01 (5)	1.42e+00† 1.60e+00 (5)	1/0/6
SHADE	Mean SD (Rank)	2.07e+01† 1.76e-01 (3)	2.74e+01† 1.76e+00 (5)	7.68e-02† 3.57e-02 (7)	**0.00e+00**≈ **0.00e+00** **(2.5)**	2.30e+01† 3.73e+00 (2)	5.03e+01† 1.33e+01 (3)	3.18e-02† 2.33e-02 (3)	0/1/6
SWDE- Success_mBLX	Mean SD (Rank)	**1.05e+01** **8.26e-05** **(1)**	1.78e+01≈ 1.04e+01 (2)	**5.48e-05** **2.50e-04** **(1)**	**0.00e+00**≈ **0.00e+00** **(2.5)**	**9.01e-01** **1.24e-02** **(1)**	**1.16e+01** **5.34e-02** **(1)**	2.96e-03≈ 1.69e+01 (2)	**4/3/0**

Table 2C: Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F15-F21, *D*=30). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F15	F16	F17	F18	F19	F20	F21	Win/Tie/ Loss
DE/best/1/bin	Mean SD (Rank)	8.10e+03† 1.92e+02 (8)	2.48e+00† 1.32e-01 (8)	6.06e+01† 4.96e+00 (7)	1.86e+03† 3.11e+02 (9)	3.39e+00† 4.69e-01 (5)	1.47e+01† 7.96e-02 (8)	2.75e+02† 5.00e+01 (3)	0/0/7
DE/rand/1/bin	Mean SD (Rank)	8.26e+03† 4.09e+02 (9)	2.35e+00† 2.69e-01 (6)	5.94e+01† 3.03e+00 (6)	2.48e+02† 1.09e+01 (8)	1.40e+01† 8.29e-01 (9)	1.42e+01† 1.08e-01 (7)	2.77e+02† 4.44e+01 (4)	0/0/7
DE/curr-best/bin	Mean SD (Rank)	7.98e+03† 1.91e+02 (7)	2.72e+00† 2.35e-01 (9)	1.46e+02† 3.37e+00 (9)	2.26e+02† 1.62e+01 (7)	1.25e+01† 1.85e+00 (8)	1.39e+01† 2.22e-01 (5)	3.00e+02† 1.13e+00 (5)	0/0/7
DE/rand-best/bin	Mean SD (Rank)	7.93e+03† 1.45e+02 (6)	2.33e+00† 2.17e-01 (5)	1.18e+02† 1.05e+01 (8)	2.16e+02† 1.55e+01 (6)	1.24e+01† 1.16e+00 (7)	1.40e+01† 1.21e-01 (6)	3.14e+02† 1.22e+02 (7)	0/0/7
JADE	Mean SD (Rank)	3.10e+03† 3.23e+02 (2)	1.80e+00† 6.39e-01 (3)	3.14e+01† 2.65e-14 (4)	7.16e+01† 7.05e+00 (2)	1.55e+00† 1.31e-01 (3)	1.15e+01† 5.37e-01 (3)	3.19e+02† 7.32e+01 (8)	0/0/7
jDE	Mean SD (Rank)	5.25e+03† 3.96e+02 (5)	2.41e+00† 3.02e-01 (7)	3.06e+01† 9.49e-07 (3)	1.54e+02† 1.56e+01 (4)	1.67e+00† 1.31e-01 (4)	1.17e+01† 3.11e-01 (4)	2.16e+02† 7.38e+01 (2)	0/0/7
SaDE	Mean SD (Rank)	4.67e+03† 1.43e+03 (4)	2.29e+00† 3.60e-01 (4)	3.24e+01† 4.85e-02 (5)	1.68e+02† 4.44e+01 (5)	4.20e+00† 7.34e+01 (6)	1.67e+01† 6.45e-01 (9)	3.50e+02† 7.60e+01 (9)	0/0/7
SHADE	Mean SD (Rank)	3.21e+03† 2.63e+02 (3)	**9.13e-01** **1.85e-01** **(1)**	3.04e+01† 3.83e-14 (2)	7.24e+01† 5.58e+00 (3)	1.35e+00† 1.20e-01 (2)	1.04e+01† 6.04e-01 (2)	3.09e+02† 5.64e+01 (6)	1/0/7
SWDE-Success_mBLX	Mean SD (Rank)	**5.24e+02** **4.75e+00** **(1)**	1.20e+00≈ 3.25e-02 (2)	**1.05e+01** **1.61e-03** **(1)**	**1.25e+00** **1.06e+00** **(1)**	**2.62e-03** **1.35e-01** **(1)**	**2.25e+00** **1.56e+00** **(1)**	**6.25e+01** **2.21e+00** **(1)**	**6/1/0**

Table 2D: Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F22 - F28, *D*=30).). Best results for each function have been marked in boldface Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F22	F23	F24	F25	F26	F27	F28	Win/Tie/ Loss
DE/best/1/bin	Mean SD (Rank)	1.40e+03† 1.86e+02 (5)	8.29e+03† 1.07e+02 (8)	3.01e+02† 2.00e+00 (8)	2.99e+02† 2.66e+00 (7)	3.22e+02† 8.79e+01 (6)	1.31e+03† 1.07e+01 (9)	3.00e+02† 1.13e-13 (5.5)	0/0/7
DE/rand/1/bin	Mean SD (Rank)	2.66e+03† 9.84e+02 (6)	8.58e+03† 3.24e+02 (9)	3.02e+02† 1.77e+00 (9)	3.00e+02† 9.09e-01 (8)	3.94e+02† 7.33e+00 (8)	1.28e+03† 2.94e+01 (6)	3.00e+02† 8.11e-03 (5.5)	0/0/7
DE/curr-to-best/bin	Mean SD (Rank)	7.98e+03† 1.91e+02 (9)	2.72e+03† 2.35e-01 (2)	1.46e+02† 3.37e+00 (2)	2.98e+02† 3.11e+00 (6)	1.25e+02† 1.85e+00 (4)	1.39e+03† 2.22e-01 (8)	3.00e+02† 1.13e+00 (5.5)	0/0/7
DE/rand-best/bin	Mean SD (Rank)	6.20e+03† 1.23e+02 (8)	8.11e+03† 2.56e+02 (6)	2.98e+02† 2.81e+00 (6)	3.01e+02† 2.29e+00 (9)	3.93e+02† 2.56e+02 (7)	1.27e+03† 2.10e+01 (5)	3.00e+02† 0.00e+00 (5.5)	0/0/7
JADE	Mean SD (Rank)	5.04e+03† 2.77e+03 (7)	8.28e+03† 1.83e+02 (7)	2.99e+02† 6.71e-01 (7)	2.65e+02† 9.87e+00 (5)	3.97e+02† 3.45e+00 (9)	1.30e+03† 1.76e+01 (7)	3.00e+02† 0.00e+00 (5.5)	0/0/7
jDE	Mean SD (Rank)	9.10e+01† 2.30e+01 (2)	3.36e+03† 4.54e+02 (3)	2.06e+02† 1.58e+01 (4)	2.33e+02≈ 9.63e+00 (2)	2.17e+02† 5.28e+01 (5)	6.38e+02† 2.40e+02 (3)	3.00e+02† 0.00e+00 (5.5)	0/0/7
SaDE	Mean SD (Rank)	1.04e+02† 1.82e+01 (4)	5.66e+03† 4.15e+02 (5)	2.10e+02† 8.38e+00 (5)	**2.25e+02** **1.66e+01** **(1)**	2.10e+00† 4.73e-03 (3)	6.70e+02† 2.35e+02 (4)	3.00e+02† 0.00e+00 (5.5)	1/0/6
SHADE	Mean SD (Rank)	9.80e+01† 2.52e+01 (3)	3.50e+03† 4.10e+02 (4)	2.05e+02† 5.29e+00 (3)	2.59e+02† 1.96e+01 (4)	2.02e+02† 1.48e+01 (2)	3.87e+02† 1.08e+02 (2)	3.00e+02† 0.00e+00 (5.5)	0/0/7
SWDE-Success_mBLX	Mean SD (Rank)	**2.38e+00** **1.19e+01** **(1)**	**2.48e+01** **1.01e+00** **(1)**	**1.15e+02** **1.29e+00** **(1)**	2.45e+02≈ 1.54e+01 (3)	**1.00e+02** **4.01e+01** **(1)**	**1.25e+01** **4.54e+00** **(1)**	**2.74e+02** **0.00e+00** **(1)**	**6/0/1**

Table 2E: Comparative performance of 6 state-of-the-art variants of CMA-ES and SWDE_Success_mBLX on CEC 2013 benchmark suite (F1 - F28, *D*=30).). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been indicated by † (significantly different from the best algorithm) and ≈ (statistically equivalent to the best algorithm).

Algorithms / Functions		IPOP-CMA-ES	IPOP-aCMA-ES	BIPOP-CMA-ES	BIPOP-aCMA-ES	NIPOP-aCMA-ES	NBIPOP-aCMA-ES	SWDE_Success_mBLX
F1	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**
F2	Mean SD (Rank)	**0.00e+00**≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	1.65e+01† 5.07e+01 (7)
F3	Mean SD (Rank)	1.75e+00† 9.27e+00 (6)	**0.00e+00** ≈ **0.00e+00** **(2.5)**	8.53e-02† 5.09e-01 (5)	**0.00e+00** ≈ **0.00e+00** **(2.5)**	**0.00e+00** ≈ **0.00e+00** **(2.5)**	**0.00e+00** ≈ **0.00e+00** **(2.5)**	2.45e+03† 1.65e+04 (7)
F4	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	**0.00e+00** ≈ **0.00e+00** **(3.5)**	8.91e+01† 9.75e+03 (7)
F5	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**
F6	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**
F7	Mean SD (Rank)	1.69e+01† 1.94e+01 (7)	8.85e+00† 2.25e+01 (5)	9.51e+00 † 1.24e+00 (6)	2.26e+00† 1.09e+01 (2)	4.06e+00† 9.17e+00 (4)	2.32e+00† 6.04e+00 (3)	**7.21e-02** **2.65e-01** **(1)**
F8	Mean SD (Rank)	2.04e+01† 1.04e+00 (2)	2.10e+01† 4.50e-02 (6)	2.08e+01† 5.12e-02 (4)	2.19e+01† 5.55e-02 (3)	2.09e+01† 6.22e-01 (5)	2.07e+01† 4.58e-02 (3)	**1.05e+01** **8.26e-05** **(1)**
F9	Mean SD (Rank)	2.44e+01† 1.64e+01 (6)	2.72e+01† 1.63e+01 (7)	6.49e+00† 2.35e+00 (4)	5.21e+00† 1.94e+00 (3)	**2.89e+00** **1.22e+00** **(1)**	3.35e+00† 1.41e+00 (2)	1.78e+01† 1.04e+01 (5)
F10	Mean SD (Rank)	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**	**0.00e+00** ≈ **0.00e+00** **(4)**
F11	Mean SD (Rank)	2.29e+00† 1.42e+00 (4)	1.18e+00† 1.09e+00 (3)	3.09e+00† 1.59e+00 (6)	3.14e+00† 1.46e+00 (7)	1.03e+00† 1.04e+00 (2)	3.05e+00† 1.41e+00 (5)	**0.00e+00** **0.00e+00** **(1)**
F12	Mean SD (Rank)	1.83e+00† 1.16e+00 (4)	7.05e-01≈ 8.33e-01 (3)	2.42e+00† 1.46e+00 (5)	2.84e+00† 1.57e+00 (6)	6.56e-01≈ 8.26e-01 (2)	2.98e+00† 1.35e+00 (7)	**9.01e-01** **1.24e-02** **(1)**
F13	Mean SD (Rank)	2.45e+00† 2.25e+00 (4)	1.12e+00† 1.25e+00 (2)	2.39e+00† 1.47e+00 (3)	2.48e+00† 1.48e+00 (5)	**9.35e-01** **9.28e-01** **(1)**	2.77e+00† 1.45e+00 (6)	1.16e+01† 5.34e-02 (7)
F14	Mean SD (Rank)	2.87e+02† 2.72e+02 (3)	2.81e+02† 2.20e+02 (2)	6.69e+02† 6.97e+02 (5)	4.95e+02† 2.77e+02 (4)	7.16e+02† 2.44e+02 (6)	8.15e+02† 3.65e+02 (7)	**0.00e+00** **0.00e+00** **(1)**
F15	Mean SD (Rank)	3.37e+02† 2.48e+02 (2)	**3.32e+02** **2.58e+02** **(1)**	6.09e+02† 4.50e+02 (5)	5.45e+02† 4.16e+02 (4)	6.70e+02† 2.80e+02 (6)	7.95e+02† 2.58e+02 (7)	5.24e+02† 4.75e+00 (3)
F16	Mean SD (Rank)	2.58e+00† 2.73e-01 (7)	2.52e+00† 2.56e-02 (6)	7.75e-01† 1.14e+00 (2)	9.50e-01† 1.23e+00 (3)	2.48e+00† 3.14e-01 (5)	**4.49e-01** **9.36e-01** **(1)**	1.20e+00† 3.25e-02 (4)
F17	Mean SD (Rank)	3.45e+01† 1.35e+00 (3)	3.37e+01† 1.42e+00 (2)	3.63e+01† 1.77e+00 (6)	3.75e+01† 1.71e+00 (7)	3.46e+01† 1.71e+00 (4)	3.57e+01† 1.58e+00 (5)	**1.05e+01** **1.61e-03** **(1)**
F18	Mean SD (Rank)	8.16e+01† 6.12e+01 (7)	7.02e+01† 6.25e+01 (6)	5.43e+01† 3.39e+01 (3)	5.89e+01† 4.34e+01 (4)	5.36e+01† 4.45e+01 (2)	6.23e+01† 4.56e+01 (5)	**1.25e+00** **1.06e+00** **(1)**
F19	Mean SD (Rank)	2.49e+00† 4.02e-01 (7)	2.46e+00† 4.49e+00 (6)	2.39e+00† 4.18e-01 (4)	2.29e+00† 3.23e-01 (3)	2.42e+00† 4.65e-01 (5)	2.28e+00† 3.41e-01 (2)	**0.00e+00** **0.00e+00** **(1)**
F20	Mean SD (Rank)	1.46e+01† 3.79e-02 (7)	1.45e+01† 3.24e-01 (6)	1.42e+01† 6.36e-01 (5)	1.40e+01† 7.70e-01 (4)	1.35e+01† 1.26e+00 (3)	1.25e+01† 5.69e-01 (2)	**2.25e+00** **1.56e+00** **(1)**
F21	Mean SD (Rank)	2.49e+02† 5.26e+01 (6)	2.54e+02† 5.02e+00 (7)	2.00e+02† 2.82e+01 (3)	2.13e+02† 3.45e+01 (4)	2.41e+02† 4.96e+01 (5)	1.92e+02† 2.75e+01 (2)	**6.25e+01** **2.21e+00** **(1)**
F22	Mean SD (Rank)	5.03e+02† 3.09e+02 (3)	4.77e+02† 2.93e+02 (2)	8.39e+02† 5.77e+02 (7)	6.62e+02† 3.01e+02 (5)	5.72e+02† 3.41e+02 (4)	8.36e+02† 4.59e+02 (6)	**2.38e+00** **1.19e+01** **(1)**
F23	Mean SD (Rank)	5.76e+02† 3.50e+02 (3)	4.93e+02† 2.95e+02 (2)	7.19e+02† 4.52e+02 (7)	7.06e+02† 3.15e+02 (6)	6.68e+02† 3.56e+02 (5)	6.67e+02† 2.89e+02 (4)	**2.48e+01** **1.01e+00** **(1)**
F24	Mean SD (Rank)	2.85e+02† 3.02e+01 (6)	2.76e+02† 3.46e+01 (5)	1.82e+02† 5.02e+01 (3)	1.86e+02† 4.18e+01 (4)	2.95e+02† 2.35e+01 (7)	1.61e+02† 3.02e+01 (2)	**1.15e+02** **1.29e+00** **(1)**

F25	Mean SD (Rank)	2.86e+02† 2.84e+01 (6)	2.89e+02† 2.60e+01 (7)	2.35e+02† 2.12e+01 (3)	2.26e+02† 1.35e+01 (2)	2.78e+02† 3.56e+01 (5)	**2.19e+02** **1.10e+01** **(1)**	2.45e+02† 1.54e+01 (4)
F26	Mean SD (Rank)	3.14e+02† 2.85e+01 (6)	3.28e+02† 2.36e+01 (7)	1.65e+02† 3.25e+01 (3)	1.68e+02† 3.05e+01 (4)	2.51e+02† 5.75e+01 (5)	1.58e+02† 2.99e+01 (2)	**1.00e+02** **4.01e+01** **(1)**
F27	Mean SD (Rank)	1.15e+03† 2.90e+02 (7)	1.07e+03† 3.44e+02 (6)	5.04e+02† 7.01e+01 (3)	5.16e+02† 9.36e+01 (4)	8.70e+02† 4.22e+02 (5)	4.68e+02† 7.37e+01 (2)	**1.25e+01** **4.54e+00** **(1)**
F28	Mean SD (Rank)	3.00e+02† 0.00e+00 (6)	3.00e+02† 0.00e+00 (6)	2.92e+02† 3.69e+01 (4)	2.84e+02† 5.45e+01 (3)	3.00e+02† 0.00e+00 (6)	**2.68e+02** **7.36e+01** **(1)**	2.74e+02† 0.00e+00 (2)
Win/Tie/Loss		0/6/22	1/8/19	0/6/22	0/7/21	2/8/18	3/7/18	**15/4/9**
Avg. Rank		4.71	4.37	4.25	4.01	4.05	3.58	**2.71**

Table 3A. Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F1-F7, *D*=50). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (statistically equivalent to the best algorithm).

Functions / Algorithms		F1	F2	F3	F4	F5	F6	F7	Win/Tie/ Loss
DE/best/1 /bin	Mean SD (Rank)	2.26e-13† 0.00e+00 (6)	1.72e+08† 1.73e+08 (8)	2.63e+10† 2.74e+10 (8)	2.48e+05† 5.28e+04 (8)	1.13e-14† 0.00e+00 (7)	4.49e+01† 1.73e+00 (6)	2.33e+02† 7.75e+01 (7)	0/0/7
DE/rand/1 /bin	Mean SD Rank)	2.74e-06† 1.35e-06 (9)	2.29e+09† 1.22e+08 (9)	2.62e+11† 2.44e+10 (9)	2.90e+05† 4.59e+04 (9)	4.26e-03† 7.53e-04 (8)	4.66e+01† 3.39e+01 (8)	6.49e+02† 6.66e+01 (8)	0/0/7
DE/curr.-to-best/1/bin	Mean SD (Rank)	2.47e-13† 0.00e+00 (8)	2.64e+07† 3.14e+06 (6)	1.82e+08† 3.02e+08 (6)	2.10e+05† 2.96e+04 (6)	1.13e-01† 0.00e+00 (9)	4.43e+01† 6.11e-01 (5)	9.43e+02† 4.34e+01 (9)	0/0/7
DE/rand-best/1 /bin	Mean SD (Rank)	2.28e-13† 0.00e+00 (7)	7.59e+07† 2.31e+07 (7)	2.66e+08† 1.46e+08 (7)	2.25e+05† 9.99e+03 (7)	1.13e-18† 0.00e+00 (6)	4.53e+01† 1.0e+00 (7)	1.20e+02† 7.62e+01 (6)	0/0/7
JADE	Mean SD (Rank)	**0.00e+00≈** **0.00e+00** **(2.5)**	2.65e+04† 1.54e+04 (3)	2.76e+06† 5.23e+06 (3)	8.00e+03† 1.90e+04 (5)	**0.00e+00≈** **0.00e+00** **(3)**	4.36e+01† 1.11e+00 (3)	2.31e+01† 9.99e+00 (3)	0/2/5
jDE	Mean SD (Rank)	**0.00e+00≈** **0.00e+00** **(2.5)**	5.10e+05† 1.65e+05 (4)	3.98e+06† 4.52e+06 (4)	1.77e+02† 1.82e+02 (2)	**0.00e+00≈** **0.00e+00** **(3)**	4.39e+01† 8.65e-01 (4)	1.62e+01† 6.23e+00 (2)	0/2/5
SaDE	Mean SD (Rank)	8.90e-30† 3.85e-29 (5)	7.33e+05† 2.45e+02 (5)	6.47e+07† 6.36e+07 (5)	4.66e+03† 1.66e+03 (4)	**0.00e+00≈** **0.00e+00** **(3)**	5.18e+01† 2.08e+01 (9)	4.86e+01† 1.01e+01 (5)	0/1/6
SHADE	Mean SD (Rank)	**0.00e+00≈** **0.00e+00** **(2.5)**	2.60e+04 † 1.12e+04 (2)	8.81e+05† 1.96e+06 (2)	**1.63e-03** **1.39e-03** **(1)**	**0.00e+00≈** **0.00e+00** **(3)**	4.29e+01† 5.51e+00 (2)	2.34e+01† 9.33e+00 (4)	1/2/4
SWDE-Success_mBLX	Mean SD (Rank)	**0.00e+00≈** **0.00e+00** **(2.5)**	**2.56 e+01** **2.53e-01** **(1)**	**3.84e+02** **1.25e +01** **(1)**	2.85e +02† 1.42e +03 (3)	**0.00e+00≈** **0.00e+00** **(3)**	**9.54e-13** **1.57e+00** **(1)**	**2.18e-01** **1.42e+00** **(1)**	**4/2/1**

Table 3B. Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F8-F14, *D*=50). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (statistically equivalent to the best algorithm).

Functions / Algorithms		F8	F9	F10	F11	F12	F13	F14	Win/Tie/ Loss
DE/best/1/ bin	Mean SD (Rank)	2.12e+01† 7.31e-03 (4)	7.32e+01† 8.25e-01 (9)	3.46e+02† 1.48e-02 (9)	7.27e+01† 1.38e+01 (9)	4.05e+02† 8.22e+00 (8)	4.08e+02† 8.83e+00 (8)	2.62e+03† 6.60e+02 (8)	0/0/7
DE/rand/1/ bin	Mean SD (Rank)	2.17e+01† 3.45e-02 (6)	7.27e+01† 4.20e-01 (8)	1.04e+02† 1.81e+01 (8)	4.46e+01† 2.10e+00 (7)	4.17e+02† 1.48e+01 (9)	4.20e+02† 1.29e+01 (9)	1.46e+03† 7.19e+01 (7)	0/0/7
DE/curr-best/1/ bin	Mean SD (Rank)	2.80e+01† 4.93e-02 (9)	7.23e+01† 1.74e+00 (6)	4.67e-02† 2.14e-02 (4)	5.12e+01† 4.18e+01 (8)	3.79e+02† 9.90e+00 (7)	3.92e+02† 2.77e+01 (7)	1.20e+03† 4.99e+02 (6)	0/0/7
DE/rand-best/1/ bin	Mean SD (Rank)	2.15e+01† 1.88e-02 (5)	7.29e+01† 5.82e-01 (7)	6.40e-02† 2.59e-02 (5)	2.32e+01† 1.51e+00 (6)	3.76e+02† 5.16e+00 (6)	3.86e+02† 5.73e+00 (6)	9.67e+03† 1.83e+02 (9)	0/0/7
JADE	Mean SD (Rank)	2.41e+01† 7.14e-02 (8)	5.34e+01† 2.78e+00 (3)	3.55e-02† 1.75e-02 (2)	**0.00e+00≈ 0.00e+00 (2.5)**	5.65e+01† 1.11e+01 (2)	1.33e+02† 2.44e+01 (2)	4.38e-02† 2.60e-02 (4)	0/1/6
jDE	Mean SD (Rank)	2.11e+01† 4.56e-02 (3)	5.35e+01† 4.75e+00 (4)	4.66e-02† 4.20e-02 (3)	**0.00e+00≈ 0.00e+00 (2.5)**	1.08e+02† 1.70e+01 (4)	1.82e+02† 2.79e+01 (4)	5.94e-03† 1.34e-02 (2)	0/1/6
SaDE	Mean SD (Rank)	2.21e+01† 4.54e-02 (7)	3.90e+01† 4.21e+00 (2)	2.70e-01† 1.29e-01 (7)	1.91e+00 1.46e+00 (5)	1.22e+02† 2.19e+01 (5)	2.50e+02 3.90e+01 (5)	6.03e+01† 3.76e+01 (5)	0/0/7
SHADE	Mean SD (Rank)	2.05e+01† 1.68e-01 (2)	5.55e+01† 1.98e+00 (5)	7.37e-02† 3.65e-02 (6)	**0.00e+00≈ 0.00e+00 (2.5)**	5.87e+01† 1.10e+01 (3)	1.46e+02 1.96e+01 (3)	3.46e-02† 1.93e-02 (3)	0/1/6
SWDE-Success_mBLX	Mean SD (Rank)	**1.86e+00 2.25e+01 (1)**	**1.25e+00 2.05e+00 (1)**	**7.12e-03 1.68e-05 (1)**	**0.00e+00≈ 0.00e+00 (2.5)**	**9.59e+00 2.89e-02 (1)**	**5.35e+00 9.25e+01 (1)**	**8.95e-04 2.89e+03 (1)**	**6/1/0**

Table 3C. Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F15 - F14, *D*=50). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (statistically equivalent to the best algorithm).

Functions / Algorithms		F15	F16	F17	F18	F19	F20	F21	Win/Tie/ Loss
DE/best/1/ bin	Mean SD (Rank)	1.46e+04† 9.68e+02 (6)	3.56e+00† 6.64e-02 (9)	1.33e+02† 9.82e+00 (7)	4.56e+02† 1.66e+01 (9)	5.75e+00† 1.30e+00 (5)	2.40e+01† 1.84e-01 (7)	8.20e+02† 4.34e+02 (7)	0/0/7
DE/rand/1/ bin	Mean SD (Rank)	1.55e+04† 2.42e+02 (9)	3.11e+00† 1.23e-01 (5)	1.03e+02† 3.05e+00 (6)	4.54e+02† 2.27e+01 (8)	2.90e+01† 1.17e+00 (8)	2.41e+01† 1.55e-01 (8)	6.76e+02† 5.15e+02 (3)	0/0/7
DE/curr-best/1/ bin	Mean SD (Rank)	1.48e+04† 1.33e+02 (7)	3.31e+00† 2.79e-01 (8)	3.28e+02† 1.30e+01 (9)	4.41e+02† 5.85e+00 (7)	2.64e+01† 1.29e+00 (7)	2.39e+01† 6.14e-02 (6)	8.14e+02† 5.32e+02 (5)	0/0/7
DE/rand-best/1/ bin	Mean SD (Rank)	1.52e+04† 2.04e+02 (8)	3.22e+00† 4.25e-01 (7)	2.83e+02† 1.62e+01 (8)	4.27e+02† 1.06e+01 (6)	2.82e+02† 9.96e-01 (9)	2.37e+02† 2.79e-01 (9)	7.19e+02† 4.72e+03 (4)	0/0/7
JADE	Mean SD (Rank)	7.00e+03† 3.77e+02 (3)	2.15e+00≈ 8.14e-01 (2)	5.08e+01† 4.58e-14 (3)	**1.**40e+02† 1.09e+01 (3)	2.75e+00† 1.69e-01 (3)	1.96e+01† 5.35e-01 (3)	8.19e+02† 3.84e+02 (6)	0/0/7
jDE	Mean SD (Rank)	9.92e+03† 5.30e+02 (5)	3.18e+00† 3.64e-01 (6)	5.09e+01† 7.48e-14 (4)	2.84e+02† 2.36e+01 (5)	2.89e+00† 2.32e-01 (4)	2.14e+01† 4.60e-01 (5)	5.74e+02† 4.35e-02 (2)	0/0/7
SaDE	Mean SD (Rank)	8.53e+03† 2.25e+03 (4)	3.00e+00≈ 2.77e-01 (4)	5.14e+01† 5.18e-01 (5)	1.57e+02† 5.80e+01 (4)	1.13e+01† 2.25e+00 (6)	1.97e+01† 1.13e+00 (4)	8.58e+02† 3.34e+02 (9)	0/0/7
SHADE	Mean SD (Rank)	6.82e+03† 4.40e+02 (2)	**1.29e+00 2.08e-01 (1)**	5.07e+01† 4.26e-14 (2)	1.36e+02† 1.29e+01 (2)	2.65e+00† 2.89e-01 (2)	1.93e+01† 7.69e-01 (2)	8.45e+02† 3.63e+02 (8)	1/0/6
SWDE-Success_mBLX	Mean SD (Rank)	**2.45e+02 4.98e+01 (1)**	2.89e+00≈ 4.50e-02 (3)	**2.78e+01 2.16e-05 (1)**	**1.29e+00 1.29e+01 (1)**	**3.99e-01 1.55e-01 (1)**	**3.25e+00 1.56e+01 (1)**	**1.26e+02 1.25e+00 (1)**	**6/1/0**

Table 3D. Performance of 9 moderate scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2013 benchmark suite (F22 - F28, *D*=50). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Functions / Algorithms		F22	F23	F24	F25	F26	F27	F28	Win/Tie/ Loss
DE/best/1/ bin	Mean SD (Rank)	3.01e+03† 1.67e+02 (7)	1.51e+04† 3.91e+02 (7)	3.82e+02† 5.05e+00 (7)	3.78e+02† 5.00e+00 (6)	4.75e+02† 2.22e+01 (6)	2.15e+03† 4.25e+01 (8)	4.00e+02† 1.13e+00 (2)	0/0/7
DE/rand/1/ bin	Mean SD (Rank)	2.62e+03† 5.66e+02 (6)	1.58e+04† 1.36e+02 (9)	3.88e+02† 2.74e+00 (9)	3.80e+02† 6.43e+00 (7)	4.90e+02† 2.56e+00 (9)	2.16e+03† 4.99e+01 (9)	1.21e+03† 1.63e+03 (7)	0/0/7
DE/curr-best/1/ bin	Mean SD (Rank)	1.22e+04† 2.19e+02 (9)	1.50e+04† 2.15e+02 (6)	3.81e+02† 1.69e+00 (6)	3.84e+02† 2.92e+00 (9)	4.84e+02† 2.88e+00 (8)	2.14e+03† 2.00e+01 (7)	1.40e+03† 1.74e+03 (8)	0/0/7
DE/rand-best/1/ bin	Mean SD (Rank)	1.06e+04† 1.89e+03 (8)	1.52e+04† 5.70e+02 (8)	3.84e+02† 2.80e+00 (8)	3.83e+02† 2.78e+00 (8)	4.83e+02† 5.73e+00 (7)	2.12e+03† 4.61e+01 (6)	1.42e+03† 1.77e+03 (9)	0/0/7
JADE	Mean SD (Rank)	1.30e+02† 7.12e+01 (5)	7.31e+03† 8.24e+02 (2)	2.49e+02† 2.10e+01 (3)	3.55e+02† 1.71e+01 (5)	3.65e+02† 9.76e+01 (5)	1.39e+03† 3.33e+02 (5)	5.73e+02† 6.99e+02 (5)	0/0/7
jDE	Mean SD (Rank)	2.79e+01† 2.23e+01 (4)	9.69e+03† 8.21e+02 (5)	2.57e+02† 1.39e+01 (4)	3.05e+02† 2.02e+01 (2)	2.89e+02† 9.89e+01 (4)	1.17e+03† 2.97e+02 (3)	4.02e+02† 5.27e-14 (3)	0/0/7
SaDE	Mean SD (Rank)	2.68e+01† 3.42e+01 (3)	8.24e+03† 1.90e+03 (4)	2.77e+02† 1.01e+01 (5)	3.45e+02† 1.17e+01 (4)	2.86e+02† 9.35e+01 (3)	1.19e+03† 1.01e+02 (4)	5.96e+02† 7.93e+02 (6)	0/0/7
SHADE	Mean SD (Rank)	1.33e+01† 7.12e+00 (2)	7.63e+03† 6.57e+02 (3)	2.33e+02† 1.01e+01 (2)	3.39e+02† 3.08e+01 (3)	2.58e+02† 8.07e+01 (2)	9.37e+02† 3.07e+02 (2)	4.58e+02† 4.14e+02 (4)	0/0/7
SWDE-Success_mBLX	Mean SD (Rank)	**1.45e+00 2.45e+01 (1)**	**3.42e+02 2.22e+02 (1)**	**1.95e+02 1.23e+00 (1)**	**1.74e+02 1.26e+01 (1)**	**3.10e+01 2.15e+00 (1)**	**1.56e+02 1.04e+02 (1)**	**3.25e+02 0.00e+00 (1)**	**7/0/0**

Table 3E: Comparative performance of 6 state-of-the-art variants of CMA-ES and SWDE_Success_mBLX on CEC 2013 benchmark suite (F1 - F28, *D*=50).). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been indicated by † (significantly different from the best algorithm) and ≈ (statistically equivalent to the best algorithm).

Algorithms Functions		IPOP-CMA-ES	IPOP-aCMA-ES	BIPOP-CMA-ES	BIPOP-aCMA-ES	NIPOP-aCMA-ES	NBIPOP-aCMA-ES	SWDE_ Success_ mBLX
F1	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**
F2	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	2.56 e+01† 2.53e-01 (7)
F3	Mean SD (Rank)	6.98e+03† 2.74e+03 (6)	**5.46e+00 2.21e+00 (1)**	8.56e+03† 3.25e+02 (7)	2.86e+01† 1.52e+01 (4)	1.85e+01† 1.16e+00 (3)	1.81e+01† 1.21e+00 (2)	3.84e+02† 1.25e +01 (5)
F4	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	**0.00e+00 ≈ 0.00e+00 (3.5)**	2.85e +02† 1.42e +03 (7)
F5	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**
F6	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**
F7	Mean SD (Rank)	2.96e+01† 1.96e+01 (6)	1.45e+01† 1.02e+01 (4)	4.35e+00† 7.95e-01 (7)	2.06e+01† 3.61e+00 (5)	1.18e+01† 3.25e+00 (3)	4.91e+00† 5.41e-01 (2)	**2.18e-01 1.42e+00 (1)**
F8	Mean SD (Rank)	2.21e+01† 5.26e+00 (7)	2.12e+01† 3.25e-02 (3)	2.13e+01† 3.96e-02 (4)	2.17e+01† 3.45e-02 (5)	2.11e+01† 4.45e-02 (2)	2.19e+01† 2.35e+00 (6)	**1.86e+00 2.25e+01 (1)**
F9	Mean SD (Rank)	5.98e+01† 2.56e+01 (7)	5.69e+01† 2.36e+01 (6)	1.37e+01† 4.52e+00 (5)	1.25e+01† 3.77e+00 (4)	6.85e+00† 1.7e+00 (2)	7.22e+00† 2.65e+00 (3)	**1.25e+00 2.05e+00 (1)**
F10	Mean SD (Rank)	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00 ≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**	**0.00e+00≈ 0.00e+00 (4)**
F11	Mean SD (Rank)	8.59e+00† 5.56e+00 (6)	8.54e+00† 4.69e+00 (5)	9.25e+00† 5.31e+00 (7)	7.32e+00† 3.56e+00 (4)	1.95e+00† 1.25e-01 (2)	5.50e+00† 2.24e+00 (3)	**0.00e+00 0.00e+00 (1)**
F12	Mean SD (Rank)	6.12e+00† 4.36e+00 (4)	3.53e+00† 3.59e+00 (2)	8.16e+00† 3.70e+00 (6)	6.75e+00† 2.89e+00 (5)	**1.36e+00 1.22e+00 (1)**	5.31e+00† 2.32e+00 (3)	9.59e+00† 2.89e-02 (7)
F13	Mean SD (Rank)	1.08e+01† 2.21e+00 (7)	6.06e+00† 1.02e+00 (3)	7.65e+00† 6.00e+00 (5)	7.13e+00† 4.18e+00 (4)	**1.48e+00 1.00e+00 (1)**	7.59e+00† 5.42e+00 (6)	5.35e+00† 9.25e+01 (2)
F14	Mean SD (Rank)	1.65e+03† 2.36e+02 (6)	1.13e+03† 2.35e+02 (2)	1.69e+03† 2.35e+02 (7)	1.26e+03† 8.56e+02 (4)	1.25e+03† 2.56e+02 (3)	1.34e+03† 5.66e+02 (5)	**0.00e+00 0.00e+00 (1)**
F15	Mean SD (Rank)	1.35e+03† 2.35e+02 (4)	1.07e+03† 2.56e+02 (2)	1.32e+03† 1.17e+02 (3)	1.39e+03† 9.23e+02 (6)	1.37e+03† 2.38e+01 (5)	1.53e+03† 5.32e+02 (7)	**2.45e+02 4.98e+01 (1)**
F16	Mean SD (Rank)	3.31e+00† 2.77e-01 (5)	3.35e+00† 2.71e-01 (6)	1.56e+00† 1.62e+00 (2)	1.63e+00† 1.69e+00 (3)	3.37e+00† 2.96e-01 (7)	**8.78e-01 2.35e-02 (1)**	2.89e+00† 4.50e-02 (4)
F17	Mean SD (Rank)	5.81e+02† 4.37e+00 (6)	5.86e+01† 4.15e+00 (4)	6.16e+02† 5.47e+00 (7)	6.01e+01† 3.60e+00 (5)	5.73e+01† 2.35e+00 (2)	5.74e+01† 2.65e+00 (3)	**2.78e+01 2.16e-05 (1)**
F18	Mean SD (Rank)	2.28e+02† 1.35e+02. (7)	1.64e+02† 1.29e+01 (5)	1.38e+02† 1.17e+02 (3)	1.58e+02† 1.23e+02 (4)	1.93e+02† 1.35e+02 (6)	1.33e+02† 5.36e+01 (2)	**1.29e+00 1.29e+01 (1)**
F19	Mean SD (Rank)	4.41e+00† 7.89e-01 (4)	4.46e+00† 5.53e-01 (6)	4.37e+00† 5.63e-01 (3)	4.24e+00† 5.72e-01 (2)	4.47e+00† 5.21e-01 (7)	4.45e+00† 2.63e-02 (5)	**3.99e-01 1.55e-01 (1)**
F20	Mean SD (Rank)	2.50e+01† 0.00e+00 (6)	2.51e+01† 1.02e-02 (7)	2.35e+01† 6.42e-01 (5)	2.16e+01† 6.26e-01 (2)	2.28e+01† 1.34e+00 (4)	2.25e+01† 2.35e-01 (3)	**3.25e+00 1.56e+01 (1)**
F21	Mean SD (Rank)	5.17e+02† 4.28e+01 (6)	6.58e+02† 4.07e+02 (7)	2.24e+02† 1.25e+02 (3)	2.67e+02† 2.11e+02 (4)	3.56e+02† 1.24e+01 (5)	1.98e+02† 1.42e+01 (2)	**1.26e+02 1.25e+00 (1)**
F22	Mean SD (Rank)	1.83e+03† 2.58e+02 (7)	1.40e+03† 2.36e+02 (3)	1.75e+03† 1.23e+02 (6)	1.62e+03† 1.02e+02 (4)	1.05e+03† 1.24e+02 (2)	1.72e+03† 5.26e+02 (5)	**1.45e+00 2.45e+01 (1)**
F23	Mean SD (Rank)	2.98e+03† 4.09e+02 (7)	1.21e+03† 1.69e+02 (3)	1.92e+03† 1.65e+02 (6)	1.75e+03† 2.59e+02 (5)	1.18e+03† 6.91e+02 (2)	1.46e+03† 2.56e+02 (4)	**3.42e+02 2.22e+02 (1)**
F24	Mean SD	3.75e+02†	3.81e+02†	2.45e+02†	2.43e+02†	3.70e+02†	2.39e+02†	**1.95e+02 1.23e+00**

	(Rank)	3.33e+01 (6)	2.74e+01 (7)	3.26e+01 (4)	3.26e+01 (3)	1.45e+01 (5)	4.65e+01 (2)	**(1)**
F25	Mean SD (Rank)	3.25e+02† 3.36e+01 (5)	3.69e+02† 4.56e+01 (7)	2.58e+02† 3.36e+01 (4)	2.56e+02† 3.36e+01 (3)	3.54e+02† 2.56e+01 (6)	2.47e+02† 5.64e+01 (2)	**1.74e+02** **1.26e+01** **(1)**
F26	Mean SD (Rank)	3.82e+02† 1.29e+02 (7)	3.75e+02† 1.26e+01 (6)	1.75e+02† 2.36e+01 (2)	1.84e+02† 2.45e+01 (3)	2.88e+02† 2.56e+01 (5)	1.96e+02† 1.48e+01 (4)	**3.10e+01** **2.15e+00** **(1)**
F27	Mean SD (Rank)	1.96e+03† 4.54e+02 (6)	2.74e+03† 1.26e+02 (7)	7.36e+02† 1.01e+02 (4)	7.16e+02† 1.41e+02 (2)	1.89e+03† 5.20e+02 (5)	7.27e+02† 1.44e+02 (3)	**1.56e+02** **1.04e+02** **(1)**
F28	Mean SD (Rank)	1.96e+03† 4.54e+02 (7)	8.59e+02† 1.02e+02 (6)	4.01e+02† 1.25e-02 (3)	4.02e+02† 1.25e-02 (4)	5.71e+02† 6.93e+01 (5)	4.00e+02† 0.00e+00 (2)	**3.25e+02** **0.00e+00** **(1)**
Win/Tie/Loss		0/6/22	1/6/21	0/6/22	0/6/22	2/6/20	1/6/21	18/4/6
Avg, Rank		5.53	6.96	4.5	3.53	3.78	3.5	**2.53**

Table 4. Summary of the CEC 2010 LSGO Competition Benchmark Functions

Fn. No.	Functions	Properties	Search Range
1.	Shifted Elliptic Function	Unimodal, Separable, Shifted	$[-100, 100]^D$
2.	Shifted Rastrigin's Function	Multimodal, Shifted, Separable	$[-5, 5]^D$
3.	Shifted Ackley's Function	Multimodal, Shifted, Separable	$[-32, 32]^D$
4.	Single-group Shifted and *m*-rotated Elliptic Function	Unimodal, single group *m*-Non-Separable, Shifted, Single group *m*-rotated	$[-100, 100]^D$
5.	Single-group Shifted and *m*-rotated Rastrigin's Function	Multimodal, single group *m*-Non-Separable, Shifted, Single group *m*-rotated	$[-5, 5]^D$
6.	Single-group Shifted and *m*-rotated Ackley's Function	Multimodal, single group *m*-Non-Separable, Shifted, Single group *m*-rotated	$[-32, 32]^D$
7.	Single-group Shifted and *m*-dimensional Schwefel's Problem 1.2	Unimodal, single group *m*-Non-Separable, Shifted, Single	$[-100, 100]^D$
8.	Single-group Shifted and *m*-dimensional Rosenbrock Function	Multimodal, single group *m*-Non-Separable, Shifted	$[-100, 100]^D$
9.	*D*/2*m* - group Shifted and *m*- rotated Elliptic Function	Unimodal, Shifted, *D*/2*m*-group m-rotated, *D*/2*m*-group *m*-nonseparable	$[-100, 100]^D$
10.	*D*/2*m* - group Shifted and *m*- rotated Rastrigin's Function	Multi-modal, Shifted, *D*/2*m*-group m-rotated, D/2*m*-group *m*-nonseparable	$[-5,5]^D$
11.	*D*/2*m* - group Shifted and *m*- rotated Ackley's Function	Multi-modal, Shifted, *D*/2*m*-group *m*-rotated, *D*/2*m*-group *m*-nonseparable	$[-32,32]^D$
12.	*D*/2*m* - group Shifted and *m* -dimensional Schwefel's Problem 1.2	Unimodal, *D*/2*m*-group *m*-nonseparable, Shifted	$[-100, 100]^D$
13.	*D*/2*m* - group Shifted and *m* -dimensional Rosenbrock Function	Multi-modal,Shifted, *D*/2*m*-group *m*-nonseparable	$[-100, 100]^D$
14.	*D*/*m* - group Shifted and *m*- rotated Elliptic Function	Unimodal, Shifted, *D*/*m*-group *m*-rotated, *D*/*m*-group *m*-non-separable	$[-100, 100]^D$
15.	*D*/*m* - group Shifted and *m*- rotated Rastrigin's Function	Multi-modal, Shifted, *D*/*m*-group *m*-rotated, *D*/*m*-group *m*-non-separable	$[-5,5]^D$
16.	*D*/*m* - group Shifted and *m*- rotated Ackley's Function	Multi-modal, Shifted, *D*/*m*-group *m*-rotated, *D*/*m*-group *m*-non-separable	$[-32,32]^D$
17.	*D*/*m* - group Shifted and *m* -dimensional Schwefel's Problem 1.2	Unimodal, *D*/*m*-group *m*-non-separable, Shifted	$[-100, 100]^D$
18.	*D*/*m* - group Shifted and *m* -dimensional Rosenbrock Function	Multi-modal,Shifted, *D*/*m*-group *m*-non-separable	$[-100, 100]^D$
19.	Shifted Schwefel's Problem 1.2	Unimodal, Fully-non-separable, Shifted	$[-100, 100]^D$
20.	Shifted Rosenbrock Function	Multimodal, Fully-non-separable, Shifted	$[-100, 100]^D$

Table 5A. Performance of 12 large-scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2010 benchmark suite (F1-F7, D=1000). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F1	F2	F3	F4	F5	F6	F7	Win/Tie/ Loss
jDElsgo	Mean SD (Rank)	8.86e-20† 4.51e-20 (6)	1.25e-01† 3.45e-01 (2)	3.81e-12† 5.02e-12 (6)	8.06e+10† 3.08e+10 (2)	9.72e+07† 1.44e+07 (5)	1.70e-08† 4.03e-08 (2)	1.31e-02† 6.38e-02 (2)	0/0/7
DECC-ML	Mean SD (Rank)	1.93e-25† 1.86e-25 (3)	2.17e+02† 2.98e+01 (7)	2.18e-13† 8.22e-15 (2)	3.58e+12† 1.54e+12 (7)	2.99e+08† 9.31e+07 (10)	7.93e+05† 3.97e+06 (8)	1.39e+08† 7.72e+07 (10)	0/0/7
DASA	Mean SD (Rank)	1.52e-21† 2.33e-21 (5)	8.48e+00† 2.52e+00 (5)	7.20e-11† 8.27e-12 (8)	5.05e+11† 2.22e+11 (5)	6.20e+08† 7.87e+07 (12)	1.97e+07† 4.45e+04 (12)	7.78e+00† 3.10e+00 (3)	0/0/7
DMS-PSO-SHS	Mean SD (Rank)	5.51e-15† 4.00e-14 (7)	8.51e+01† 2.06e+01 (6)	5.51e-11† 3.21e-10 (7)	2.41e+11† 3.31e+10 (3)	8.35e+07† 6.15e+06 (4)	8.25e-02† 9.95e-01 (4)	1.95e+03† 1.55e+02 (6)	0/0/7
SDENS	Mean SD (Rank)	5.72e-06† 4.42e-06 (10)	2.22e+03† 8.92e+01 (10)	2.70e-05† 1.52e-05 (9)	5.12e+12† 2.12e+12 (9)	1.12e+08† 2.23e+07 (6)	2.23e-04† 4.56e-05 (3)	1.21e+08† 6.52e+07 (9)	0/0/7
EOEA	Mean SD (Rank)	2.21e-23† 2.81e-23 (4)	3.61e-01† 6.71e-01 (3)	1.61e-13† 1.11e-14 (3)	3.07e+12† 1.66e+12 (6)	2.26e+07† 5.96e+06 (3)	3.86e+06† 4.96e+05 (9)	1.21e+02† 1.51e+02 (5)	0/0/7
DECC-G	Mean SD (Rank)	2.94e-07† 8.64e-08 (9)	1.34e+03† 3.24e+01 (9)	1.38e+00† 9.75e-02 (10)	1.75e+13† 5.34e+12 (11)	2.64e+08† 8.44e+07 (9)	4.91e+06† 8.01e+05 (10)	1.61e+08† 1.31e+08 (11)	0/0/7
MLCC	Mean SD (Rank)	1.55e-27† 7.62e-27 (2)	5.52e-01† 2.22e+00 (4)	9.82e-13† 3.72e-12 (5)	9.60e+12† 3.40e+12 (10)	3.80e+08† 6.90e+07 (11)	1.61e+07† 4.91e+06 (11)	6.81e+05† 7.31e+05 (8)	0/0/7
MA-SW-Chains	Mean SD (Rank)	2.09e-14† 1.98e-14 (8)	8.11e+02† 5.81e+01 (8)	7.21e-13† 3.41e-13 (4)	3.52e+11† 3.12e+10 (4)	1.67e+08† 1.07e+08 (8)	8.13e+04† 2.83e+05 (7)	1.03e+02† 8.71e+01 (4)	0/0/7
DECC-DG	Mean SD (Rank)	5.41e+03† 2.02e+04 (11)	4.33e+03† 1.93e+02 (11)	1.61e+01† 3.31e-01 (11)	4.78e+12† 1.48e+12 (8)	1.51e+08† 2.12e+07 (7)	1.63e+01† 2.72e-01 (5)	1.12e+04† 7.41e+03 (7)	0/0/7
DE/best /1/bin	Mean SD (Rank)	3.52e+05† 2.69e+05 (12)	9.30e+03† 1.3e+04 (12)	3.53e+03† 2.23e+02 (12)	5.25e+19† 9.23e+17 (12)	5.23e+06† 9.63+05 (2)	2.55e+01† 1.21e+01 (6)	1.08e+11† 9.08e+10 (12)	0/0/7
SWDE-Success_mBLX	Mean SD (Rank)	**0.00e+00** **0.00e+00** **(1)**	**1.01e-03** **2.23e-03** **(1)**	**5.11e-14** **2.35e-09** **(1)**	**2.50e+05** **2.44e+01** **(1)**	**5.49e+03** **2.17e+03** **(1)**	**1.03e-08** **1.21e-06** **(1)**	**1.03e-08** **1.21e-06** **(1)**	**7/0/0**

Table 5B. Performance of 12 large-scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2010 benchmark suite (F8-F14, *D*=1000). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F8	F9	F10	F11	F12	F13	F14	Win/Tie/ Loss
jDElsgo	Mean SD (Rank)	3.12e+06† 3.23e+06 (2)	3.13e+07† 5.02e+06 (5)	2.59e+03† 3.18e+02 (5)	2.21e+01† 1.53e+01 (4)	1.22e+04† 2.05e+03 (6)	7.12e+02† 1.39e+02 (2)	1.67e+08† 2.09e+07 (6)	0/0/7
DECC-ML	Mean SD (Rank)	3.41e+07† 3.52e+07 (7)	5.93e+07† 4.71e+06 (9)	1.21e+04† 2.60e+02 (12)	**1.79e-13** **9.55e-15** **(1)**	3.56e+06† 1.35e+05 (12)	1.12e+03† 4.30e+02 (3)	1.70e+08† 1.45e+07 (7)	1/0/6
DASA	Mean SD (Rank)	4.97e+07† 8.94e+07 (9)	3.61e+07† 4.78e+06 (6)	7.26e+03† 2.61e+02 (10)	1.97e+02† 1.52e-01 (10)	1.70e+03† 2.24e+02 (4)	1.20e+03† 7.34e+02 (4)	1.01e+08† 7.85e+06 (4)	0/0/7
DMS-PSO-SHS	Mean SD (Rank)	1.25e+07† 1.95e+06 (4)	8.55e+06† 6.55e+05 (2)	5.59e+03† 5.19e+02 (8)	3.29e+01† 2.99e+00 (6)	6.15e+02† 6.05e+01 (3)	1.25e+03† 1.06e+02 (6)	1.76e+07† 1.56e+06 (2)	0/0/7
SDENS	Mean SD (Rank)	5.15e+07† 2.15e+07 (10)	5.61e+08† 5.71e+07 (12)	6.81e+03† 5.61e+02 (9)	2.22e+02† 5.02e-01 (11)	4.12e+05† 4.22e+04 (10)	2.13e+03† 1.03e+03 (9)	1.83e+09† 2.33e+08 (11)	0/0/7
EOEA	Mean SD (Rank)	1.01e+07† 1.21e+07 (3)	4.62e+07† 4.72e+06 (7)	1.02e+03† 6.92e+01 (3)	3.82e+01† 1.62e+01 (8)	1.57e+04† 2.50e+03 (7)	1.54e+03† 4.19e+02 (7)	1.64e+08† 8.94e+06 (5)	0/0/7
DECC-G	Mean SD (Rank)	6.43e+07† 2.81e+07 (11)	3.20e+08† 3.36e+07 (11)	1.05e+04† 2.94e+02 (11)	2.30e+01† 1.75e+00 (5)	8.91e+04† 6.81e+03 (9)	5.11e+03† 3.91e+03 (10)	8.01e+08† 6.02e+07 (10)	0/0/7
MLCC	Mean SD (Rank)	4.37e+07† 3.44e+07 (8)	1.22e+08† 1.29e+07 (10)	3.45e+03† 8.75e+02 (6)	1.91e+02† 6.91e-01 (9)	3.41e+04† 4.21e+03 (8)	2.01e+03† 7.21e+02 (8)	3.11e+08† 2.71e+07 (8)	0/0/7
MA-SW-Chains	Mean SD (Rank)	1.41e+07† 3.62e+07 (5)	1.42e+07† 1.12e+06 (3)	2.02e+03† 1.42e+02 (4)	3.81e+01† 7.31e+00 (7)	3.61e-06† 5.91e-07 (2)	1.21e+03† 5.71e+02 (5)	3.11e+07† 1.93e+06 (3)	0/0/7
DECC-DG	Mean SD (Rank)	3.01e+07† 2.11e+07 (6)	5.91e+07† 8.12e+06 (8)	4.53e+03† 1.42e+02 (7)	1.01e+01† 1.02e+00 (3)	2.52e+03† 4.85e+02 (5)	4.58e+06† 2.18e+06 (12)	3.46e+08† 2.46e+07 (9)	0/0/7
DE/best/1/bin	Mean SD (Rank)	2.46e+11† 2.22e+10 (12)	3.03e+07† 1.00e+06 (4)	4.65e+01† 5.11e+01 (2)	2.35e+03† 1.39e+03 (12)	2.36e+06† 1.59e+06 (11)	3.59e+06† 1.09e+05 (11)	3.25e+17† 1.59e+15 (12)	0/0/7
SWDE-Success_mBLX	Mean SD (Rank)	**1.15e+05** **2.19e+05** **(1)**	**2.02e+04** **1.19e+02** **(1)**	**2.05e+00** **1.01e+01** **(1)**	1.46e-12≈ 1.55e-09 (2)	**2.25e-07** **1.45e-06** **(1)**	**2.30e+02** **1.24e+00** **(1)**	**7.14e+06** **1.12e+04** **(1)**	6/0/1

Table 5C. Performance of 12 large-scale evolutionary optimizers including SWDE_Success_mBLX on CEC 2010 benchmark suite (F15 - F20, Average Rank, D=1000). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms \ Functions		F15	F16	F17	F18	F19	F20	Win/Tie/ Loss
jDElsgo	Mean SD (Rank)	5.85e+03† 4.46e+02 (6)	1.40e+02† 3.42e+01 (7)	1.00e+05 1.25e+04 (7)	1.86e+03† 3.11e+02 (3)	**2.73e+05** **2.12e+04** **(1)**	1.51e+03 1.32e+02 (7)	1/0/5
DECC-ML	Mean SD (Rank)	1.54e+04† 3.51e+02 (11)	5.07e-02† 2.54e-01 (2)	6.56e+06 4.61e+05 (11)	2.42e+03† 1.11e+03 (5)	1.59e+07 1.71e+06 (11)	9.92e+02 3.55e+01 (3)	0/0/6
DASA	Mean SD (Rank)	1.44e+04† 3.66e+02 (10)	3.94e+02† 2.12e-01 (9)	1.04e+04 8.9e+02 (4)	4.42e+03† 2.18e+03 (7)	8.14e+05 5.56e+04 (4)	1.03e+03 1.49e+02 (5)	0/0/6
DMS-PSO-SHS	Mean SD (Rank)	4.68e+03† 2.15e+02 (4)	6.95e+01† 4.25e+00 (3)	3.23e+03† 4.05e+02 (3)	2.26e+03† 1.16e+02 (4)	1.16e+06† 1.06e+05 (7)	3.52e+02† 4.02e+01 (2)	0/0/6
SDENS	Mean SD (Rank)	7.36e+03† 9.63e+01 (8)	4.03e+02† 2.53e+00 (10)	1.02e+06 1.12e+05 (10)	3.01e+04† 1.21e+04 (10)	8.82e+05 1.52e+05 (5)	9.93e+02 1.63e+01 (4)	0/0/6
EOEA	Mean SD (Rank)	2.14e+03† 1.24e+02 (2)	8.26e+01† 1.68e+01 (5)	7.93e+04† 8.80e+03 (6)	2.94e+03† 6.92e+02 (6)	1.84e+06† 9.97e+04 (10)	1.97e+03† 2.35e+02 (8)	0/0/6
DECC-G	Mean SD (Rank)	1.26e+04† 8.91e+02 (9)	7.65e+01† 8.15e+00 (4)	2.85e+05† 1.98e+04 (9)	2.45e+04† 1.05e+04 (9)	1.15e+06† 5.15e+04 (6)	4.05e+03† 3.66e+02 (10)	0/0/6
MLCC	Mean SD (Rank)	7.11e+03† 1.31e+03 (7)	3.72e+02† 4.72e+01 (8)	1.52e+05† 1.42e+04 (8)	7.02e+03† 4.72e+03 (8)	1.32e+06† 7.32e+04 (8)	2.02e+03† 1.82e+02 (9)	0/0/6
MA-SW-Chains	Mean SD (Rank)	2.86e+03† 1.25e+02 (3)	9.95e+01† 1.45e+01 (6)	1.25e+00† 1.25e-01 (2)	1.34e+03† 4.34e+02 (2)	2.84e+05† 1.69e+04 (2)	1.05e+03† 7.25e+01 (6)	0/0/6
DECC-DG	Mean SD (Rank)	5.84e+03† 1.04e+02 (5)	7.32e+03† 5.72e-14 (12)	4.06e+04† 2.86e+03 (5)	1.18e+10† 2.08e+09 (12)	1.78e+06† 9.58e+04 (9)	4.89e+07† 2.28e+07 (12)	0/0/6
DE/best/1/bin	Mean SD (Rank)	6.66e+05† 5.69e+05 (12)	2.33e+03† 1.11e+03 (11)	3.95e+07† 6.69e+06 (12)	3.52e+07† 1.09e+07 (11)	2.55e+09† 1.59e+07 (12)	3.54e+0† 9.23e+04 (11)	0/0/6
SWDE_Success_mBLX	Mean SD (Rank)	**2.15e+02** **1.09e+01** **(1)**	**4.50e-09** **1.50e-01** **(1)**	**1.05e+00** **2.53e-01** **(1)**	**4.05e+02** **1.52e+01** **(1)**	5.50e+05† 2.00e+05 (3)	**2.86e+00** **2.11e-02** **(1)**	**5/0/1**

Table 6. Performance improvement by the proposed algorithm (SWDE_Success_mBLX) from SWDE_Success and other two conventional DE variants using modified BLX crossover (D=30). Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Function / Algorithm	Unimodal		Multimodal		Composite		
	F2	F4	F7	F19	F24	F26	F28
SWDE_Success (Das *et al*., 2015)	1.54e+02† 2.03e+00	7.91e +02† 6.43e +03	5.64e-01† 3.89e-01	3.45e-01† 2.65e-01	1.65e+02† 1.49e+00	2.00e+02† 8.01e+01	2.89e+02† 1.32e-25
DE/current-to-best/mBLX	6.80e+04† 2.23e+05	1.53e+04† 1.54e+02	9.16e+02† 1.56e+03	4.48e+00† 2.35e+01	2.99e+02† 4.56e+02	3.89e+02† 5.14e+03	3.00e+02† 0.00e+00
DE/rand/1/ mBLX	1.00e+05† 6.39e+01	2.63e+04† 2.31e+02	9.68e+01† 1.12e+03	8.62e-01† 2.32e+00	2.60e+02† 3.54e+02	3.28e+02† 2.98e+01	3.00e+02† 0.00e+00
SWDE_Success_ mBLX	**1.65e+01 5.07e+01**	**8.91e +01 9.75e +03**	**7.21e-02 2.65e-01**	**2.62e-03 1.35e-01**	**1.15e+02 1.29e+00**	**1.00e+02 4.01e+01**	**2.74e+02 0.00e+00**

Table 7. Performance improvement by the proposed algorithm (SWDE_Success_mBLX) from SWDE_Success , CCPSO2 and Sep-CMA-ES on 3 functions from the CEC 2008 test suite (D = 2000) . Best results for each function have been marked in boldface. Results of the Wilcoxon's rank sum test (at 5 % significance level) have been shown by † (significantly different from the best algorithm) and ≈ (Statistically equivalent to the best algorithm).

Algorithms	SWDE_Success _mBLX	SWDE_Success	CCPSO2	Sep-CMA-ES
Functions	Mean (Std. Dev)	Mean (Std. Dev)	Mean (Std. Dev)	Mean (Std. Dev)
F1	**1.56e-32 2.03e-25**	1.23e-14≈ 1.01e-06	1.03e-12† 2.56e-13	7.12e-15≈ 6.24e-15
F3	**7.58e+00 1.25e+00**	1.00e+01† 2.10e+00	2.91e+03† 6.43e+02	1.73e+03† 7.77e+01
F7	**-3.46e+04 1.12e+01**	-2.20e+04† 1.01e+02	-2.28e+04† 1.90e+02	-2.46e+04† 1.57e+02