Using spatial neighborhoods for parameter adaptation: An improved success history based differential evolution

Using spatial neighborhoods for parameter adaptation: An improved success history based differential evolution

Arka Ghosh[a], Swagatam Das[b,*], Asit Kr. Das[c], Roman Senkerik[d], Adam Viktorin[d], Ivan Zelinka[e], Antonio David Masegosa[a,f]

[a]*Deusto Institute of Technology (DeustoTech), Faculty of Engineering, University of Deusto, Bilbao, 48007, Vizcaya, Spain*

[b]*ECSU, Indian Statistical Institute, Kolkata, 700108, West Bengal, India*

[c]*Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Shibpur, Howrah, India*

[d]*Faculty of Applied Informatics, Tomas Bata University in Zlin, Czech Republic*

[e]*Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB-TUO, Czech Republic f KERBASQUE, Basque Foundation for Science, Spain*

**Corresponding author: E-mail address: swagatamdas19@yahoo.co.in (S. Das)*

A B S T R A C T

Differential Evolution (DE) has been widely appraised as a simple yet robust population-based, non-convex optimization algorithm primarily designed for continuous optimization. Two important control parameters of DE are the scale factor F, which controls the amplitude of a perturbation step on the current solutions and the crossover rate Cr, which limits the mixing of components of the parent and the mutant individuals during recombination. We propose a very simple, yet effective, nearest spatial neighborhood-based modification to the adaptation process of the aforesaid parameters in the Success-History based adaptive DE (SHADE) algorithm. SHADE uses a historical archive of the successful F and Cr values to update these parameters and stands out as a very competitive DE variant of current interest. Our proposed modifications can be extended to any SHADE-based DE algorithm like L-SHADE (SHADE with linear population size reduction), jSO (L-SHADE with modified mutation) etc. The enhanced performance of the modified SHADE algorithm is showcased on the IEEE CEC (Congress on Evolutionary Computation) 2013, 2014, 2015, and 2017 benchmark suites by comparing against the DE-based winners of the corresponding competitions. Furthermore, the effectiveness of the proposed neighborhood-based parameter adaptation strategy is demonstrated by using the real-life problems from the IEEE CEC 2011 competition on testing evolutionary algorithms on real-world numerical optimization problems.

Keywords: Differential evolution, SHADE, parameter adaptation, scaling factor, crossover rate

1. Introduction

1.1. Overview

Following the introduction in 1995 [**1**] as a black-box, non-convex, real-parameter optimization algorithm, Differential Evolution (DE) has been appraised as a significant associate of the Evolutionary

Computing (EC) family because of its lucidity, just a small number of adjustable parameters, and still a commendable ability to solve optimization problems in diverse scenarios [**2**]. DE has achieved its popularity among the EC researchers due to its consistent performance and high ranks in various IEEE CEC (Congress on Evolutionary Computation) competitions on real-parameter optimization in complex scenarios like constrained, multi-modal, multiobjective etc. Many decades of research effort has been devoted to enhancing the efficacy, convergence speed, and robustness of DE and this effort is still very much active owing to the growing complexity of the practical optimization problems. A few major research directions along this track include parameter adaptation based on success history (like JADE [**3**], self-adaptive DE (SaDE) [**4**], SHADE [**5**]), enhanced crossover and mutation techniques (e.g. MDE-pBX [**6**], Pro-DE [**7**], crossover based on eigenvector in DE [**8**], multiple mutation strategies and criteria [**9**] etc.), combination of different offspring generation techniques (e.g. EPSDE [**10**], multiple variant coordination framework [**11**]), improved parameter tuning [**12**], and re-use of promising past search moves [**13**]. In the domain of multi-objective optimization, BiasMOSaDE [**14**] uses a set of personal archives to evolve the search process instead of a fixed-size population and multiple biased mutation operators to further enhance the search performance of DE in the multiobjective scenario. Authors in [**15**], introduced Alopex [**16**] based mutation scheme in DE framework and tested its efficiency using the CEC 2013 single-objective real-parameter benchmark suite.

Scale factor (F), crossover rate (Cr), and size of the population (N_{p}) are the three major control parameters of DE. Among these, the first duo affects the search performance of DE significantly by regulating the step-size of perturbation and severity of the recombination respectively. Most of the parameter adaptation schemes aim to adapt these two parameters based on their past records of creating successful offspring (which could replace their parents in successive generation). JADE [**3**] is one of the most popular DE variants, which uses a control parameter adaptation strategy. Instead of using fixed values for F and Cr, JADE samples them for each individual from a Cauchy and a normal distribution respectively. It updates the location parameter μ_F of the Cauchy distribution and mean μ_{Cr} of the normal distribution by using the Lehmer mean and the arithmetic mean of the corresponding set of successful parameters respectively in each generation. Among the existing parameter adaptation schemes, Success History based Adaptive DE (SHADE) [**5**] turns out to be a very competitive extension of JADE, which maintains two memory archives to hold means of the successful values of F and Cr in each generation. It was the DE-based winner of the IEEE CEC 2013 competition on bound-constrained real parameter optimization. Another improved version of SHADE with linear population size reduction termed as L-SHADE [**17**] and it won the IEEE CEC 2014 competition on bound-constrained real-parameter optimization. The DE-based winners of the CEC competitions in the later years (SPS-L-SHADE-EIG, winner of CEC 2015 [**18**], L-SHADE-EpSin, joint winner of CEC 2016 [**19**], and jSO, the DE-based winner of CEC 2017 [**20**]) are mostly extensions of the L-SHADE algorithm.

1.2. Motivation

The use of spatial neighborhoods in swarm and evolutionary algorithms to improve the solution generation process has been extensively researched. A previous study in this area, using the Particle Swarm Optimization (PSO) technique, was published in 1999 [**21**]. In subsequent works, such as [**22**], Euclidean neighbourhoods have been used to improve mutation strategies in DE. Such Euclidean neighbourhoods, on the other hand, haven't been thoroughly examined for the automatic parameter tweaking of DE. As can be observed, in all SHADE-based algorithms, when parameter values are calculated for the next generation, all present members of the memory archive are considered. This is not helpful in practical scenarios, e.g., when the current member, for which the parameters are being generated, is located in an ill-posed region of the functional landscape and the memory archive

consists of parameter values that were responsible for success in a completely different terrain of the landscape. For several complex practical optimization problems, the fitness landscape may have diverse properties like a flat plateau in one region and a funnel-shaped basin of attraction in another region. In such cases, considering all successful values of F and Cr to generate these parameters for another individual without incorporating any information about the fitness landscape of the surroundings of this individual can negatively impact the search. Thus, the parameters may not contribute to the newly emerging difficulties of the current individual and the SHADE-type adaptation is likely to suffer from poor performance.

1.3. Contribution

To address the aforementioned problem, we propose incorporating functional landscape information into the adaptation process by establishing spatial-distance-based neighborhoods around the current individual and only considering successful individuals who are members of the computed neighborhood. The idea is unique in that it uses proximity-based local information to generate control settings for new individuals in DE. This conceptually simple modification results in significant performance enhancement over all the prominent SHADE-based algorithms. We tested this scheme integrated with SHADE on CEC 2013, L-SHADE on CEC 2014, and jSO on CEC 2017 benchmark suites. We compared the proposed scheme with the algorithm proposed by Viktorin et al. [**23**] on CEC 2015 benchmark problems. In all these scenarios, our proposed method shows significant improvement over the original algorithm. In addition to these benchmark suites, the effectiveness of our proposal was further validated on the IEEE CEC 2011 competition on testing evolutionary algorithms on real-world numerical optimization Problems [**24**]. Performance of our method on these 22 problems was compared with the Multi-Parent Crossover based Genetic Algorithm (GA-MPC) [**25**], which was the winner of 2011 CEC competition and DE with multiple strategies (SAMODE) [**26**].

1.4. Organization

The rest of this paper is organized in the following way. **Section 2** overviews the algorithmic developments of DE focusing on the canonical DE, SHADE, and some SHADE-based notable DE variants. **Section 3** discusses the proposed neighborhood-based parameter adaptation in sufficient details. **Section 4** describes the experimental setup, presents the relevant results on well-accepted numerical benchmarks as well as real world problems with pertinent discussions. Finally, the paper is concluded in **Section 5**.

2. Background: canonical DE, SHADE, and variants

In this section,at first we discuss about some basic aspects of the vanilla DE algorithm and in the later part, SHADE and other SHADE based variants are briefly discussed with the focus on the parameter adaptation process.

2.1. The basic DE algorithm

DE iteratively refines a set (called population) of trial solutions to the optimization problem at hand until a stopping criterion, e.g. the exhaustion of a given number of Function Evaluations (FEs) that depends on the computational budget of the user, is met. Following the conventions of the common DE literature, we denote the population size by N_p and refer to an iteration as a $\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, \ldots, x_{D,i,G}]^T$,

generation. Thus, we may indicate the i^{th} population member of the current generation G as the vector , where D is the problem dimension. Basic steps of the DE algorithm are described below.

2.1.1. Initialization

From a coarse knowledge of the problem at hand, we first figure out the minimum and maximum bounds for each variable as: $\vec{X}_{min} = [x_{1,min}, x_{2,min}, \ldots, x_{D,min}]^T$ and $\vec{X}_{max} = [x_{1,max}, x_{2,max}, \ldots, x_{D,max}]^T$

Then we initialize the j^{th} dimension of the i^{th} candidate solution as:

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

where randj j is a uniformly distributed random number drawn from the range [0,1] afresh for each ($_{i,j}$) ordered pair.

2.1.2. Mutation

During the mutation operation, for each population member, a base vector from the current population is perturbed with the scaled difference vector(s) of the form ($\vec{X}_{rl,G} - \vec{X}_{r2,G}$) (created from among the current population members) to produce a new vector, known as the donor or mutant. A scale factor F (usually ∈ [0.4 , 2]) controls the amplitude of the said difference vector(s). Below we show two widely popular DE mutation strategies out of several others proposed over the years [**2,27**]:

$$\text{DE/rand/1} : \vec{V}_{i,G} = \vec{X}_{r1,G} + F \cdot (\vec{X}_{r2,G} - \vec{X}_{r3,G}), \quad (2a)$$

$$\text{DE/best/1} : \vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot (\vec{X}_{r1,G} - \vec{X}_{r2,G}). \quad (2b)$$

Here r_1, r_2, r_3 are randomly selected from {1. 2. . .., N_p}, they are mutually exclusive and different from current running index i. These indices are randomly sampled for each donor vector. $\vec{X}_{bestG}$ is the best performing solution vector of current generation G. DE/best/1 scheme promotes intensification of search around the best performing member thus it is exploitative in nature, whereas DE/rand/1 increases the diversity in the population thus it promotes exploration of the search space.

2.1.3. Crossover/recombination

This step creates a final offspring vector $\vec{U}_{i,G}$ by recombining the parent and the donor vectors of the same index. During the widely popular binomial crossover in DE (which has been used along with the DE variants addressed in the paper), each component of $\vec{U}_{iG}$ can come from either target vector $\vec{X}_{i,G}$ or the mutant vector $\vec{V}_{i,G}$ in the following way:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & if\ rand_{i,j} \leq Cr \parallel j = j_r, \\ x_{j,i,G}, & otherwise, \end{cases} \quad (3)$$

where randtj is a random number in [0,1] instantiated afresh for each pair (i, j). Also, j_r is an index chosen randomly from {1, 2, . D} and it makes sure that the offspring and the mutant vector may differ in at least one variable.

2.1.4. Selection

The ultimate step in DE is a selection that is based on a direct competition between the trial and the target vectors as shown below:

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

where f (.) is the function to be minimized.

2.2. SHADE and variants

SHADE adapts two major control parameters - scale factor F and crossover rate Cr based on the success of the generated offspring over past generations. Below we briefly outline the algorithmic details of SHADE.

2.2.1. Initialization

The memory archive values $M_{Cr,i}$ and $M_{F,i}$ for Cr and F respectively and over i = 1, . .., H are initialized to 0.5. In addition, an external archive of inferior solutions A is initialized. In the beginning, as there is no inferior solution, this archive is kept empty, i.e. $A = \phi$ and its maximum size can be N_p.

2.2.2. Mutation

SHADE implements the "DE/current-to-pbest/1" mutation strategy, proposed in [**3**] as given below:

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F_i \cdot (\vec{X}_{pbest,G} - \vec{X}_{i,G}) + F_i \cdot (\vec{X}_{r1,G} - \vec{X}_{r2,G}). \quad (5)$$

Here, $\vec{X}_{pbest,G}$ is randomly selected from the p.Np individuals in the current generation G. The value of P is generated uniformly at random from $[P_{min}, 0,2]$, where p_{min} = 2/Np. Vector $\vec{X}_{r1G}$ is randomly selected from the current population and $\vec{X}_{r2}$ is picked at random from the union of the current population P and the archive A. The value of F_i is sampled as:

$$F_i = C[M_{F,r}, 0.1], \quad (6)$$

where $M_{F,r}$ is selected randomly (r is the index value) from M_F and C represents the Cauchy distribution with $M_{F,r}$ as the location parameter and 0.1 as the scale parameter. If $F_i >$; it is truncated to 1 and when $F_i \leq 0$; it is regenerated by **Eq. (6)**.

2.2.3. Crossover

SHADE uses binomial crossover as in **Eq. (3)**. Crossover rate Cr_i for the i^{th} individual is sampled separately from a Gaussian distribution with $M_{Cr,r}$ as the mean value, which is selected randomly (by the same index r as mutation) from memory M_{Cr} and 0.1 as the standard deviation:

$$Cr_i = N[M_{Cr,r}, 0.1]. \quad (7)$$

2.2.4. Historical memory adaptation stratégy

After completion of each generation, memory cells of M_F and M_{Cr} are updated. Each of these cells has an index k which runs from 1 to H. The k^{th} cell in each memory is updated according **Eqs. (8)** and **(9)**, and k is then increased by one after each generation and reset to one if bigger than H.

$$M_{F,k} = \begin{cases} mean_{WL}(S_F), & if\ S_F \neq \phi, \\ M_{F,k}, & otherwise. \end{cases} \quad (8)$$

$$M_{Cr,k} = \begin{cases} mean_{WA}(S_{Cr}), & if\ S_{Cr} \neq \phi, \\ M_{Cr,k}, & otherwise. \end{cases} \quad (9)$$

Here $mean_{WL}(\cdot)$ and $mean_{WA}(\cdot)$ are the weighted versions of Lehmer and arithmetic means respectively as shown below:

$$mean_{WL}(S_F) = \frac{\sum_{k=1}^{|S_F|} w_k \cdot S_{F,k}^2}{\sum_{k=1}^{|S_F|} w_k \cdot S_{F,k}}, \quad (10)$$

$$mean_{WA}(S_Cr) = \sum_{k=1}^{|S_{Cr}|} w_k \cdot S_{Cr,k}. \quad (11)$$

Here the weights ω_k are calculated in the following way:

$$w_k = \frac{\triangle f_k}{\sum_{k=1}^{|S_{Cr}|} \triangle f_k}, \text{ where } \triangle f_k = |f(\vec{U}_{i,G}) - f(\vec{X}_{i,G})|. \quad (12)$$

2.2.5. Linear population size reduction in SHADE

In L-SHADE [**17**], the population size is linearly reduced with generations in the following way (while removing worse performing individuals from current population, if needed):

$$Np_{new} = round(Np_{init} - \frac{FEs}{max\_FEs}.(Np_{init} - Np_{min})), \quad (13)$$

where $N\ \mathrm{p}_{\mathrm{init}}$ is initial population size, $N\ p_{\mathrm{min}}$ is the minimum allowed population size $F\ Es$ is the current number of fitness function evaluations, and $max\_F\ Es$ is the maximum limit of $F\ Es$.

2.2.6. Two improvements on SHADE

To alleviate the chances of premature convergence in SHADE, in [**23,28**], the authors proposed a new normalized weight calculation scheme for the Lehmer and arithmetic means, by using the Euclidean distance between a target and the resulting trial vectors. The more the distance between the target and the trial vectors, the more weight is added to the corresponding F and Cr values. The weights are calculated in the following way:

$$w_k = \frac{\sqrt{\sum_{j=1}^{D}(u_{k,j,G} - x_{k,j,G})^2}}{\sum_{m=1}^{|S_{Cr}|}\sqrt{\sum_{j=1}^{D}(u_{m,j,G} - x_{m,j,G})^2}}. \quad (14)$$

The authors compared their proposed schemes called Db-SHADE and Db-L-SHADE with the original SHADE and L-SHADE algorithms by using the CEC 2015 benchmark suite.

Another significant extension of L-SHADE has been jSO [**20**] (it was an upgraded version of iL-SHADE [**29**]). This work used a weighted version of the " current-to-$pbest$/1 " mutation scheme and refers to it as "current-to-$pbest$-w/1", the expression of which can be shown as:

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F_w \cdot (\vec{X}_{pbest,G} - \vec{X}_{i,G}) + F \cdot (\vec{X}_{r1,G} - \vec{X}_{r2}), \quad (15)$$

where F_ω is calculated as:

$$F_w = \begin{cases} 0.7F, & FEs < 0.2 \cdot max\_FEs, \\ 0.8F, & FEs < 0.4 \cdot max\_FEs, \\ 1.2F, & otherwise. \end{cases} \quad (16)$$

3. The neighborhood-based parameter adaptation strategy for SHADE

In this Section, we elaborate the proposed modification of SHADE through the incorporation of spatial neighborhood information in the parameter adaptation framework in adequate details.

3.1. The proposal

In the proposed technique, for any i^{th} population member, a spatial neighborhood is created based on a valid distance measure between the pairs of successful members $S_{i,N}$ of size N = round $\sqrt{Np}$), , around the considered individual ($S_{i,N}$ = [$\vec{X}_{N1,G}$..... $\vec{X}_{NN,G}$ }, where $\vec{X}_{NI\,G}$ is the Ith neighbor of current individual in its neighborhood in generation G). Instead of the conventional Euclidean distance metric (i.e. the l_2 norm induced distance), here we have usedl the l_1 norm induced distance, also known as the Manhattan distance (for two vectors $\vec{X}_i$, $\vec{X}$, $\in R^D$, this distance is given by $d_1(\vec{X}_i\,\vec{X}_j) = \sum^D_{k=1} |x_{k,i} - x_{k,j}|$) to enhance the time efficiency of the algorithm. Although both the l_1 and l_2 norms induce an asymptotic complexity of 0(D) for a D dimensional vector space, l_1 norm uses less arithmetic operations as it does not need to square each term and then find the square root. As a benefit, while keeping similar level of performance, some gain in the run-time can be achieved, as will be evident from our experiments reported in **Section 4.5**.

After creation of such a neighborhood, we extract two arrays $S_{F,N}$ and $S_{Cr,N}$ which holds successful F and Cr values for the corresponding members of $S_{i,N}$ ($S_{F,N}$ = [$F_{N1,G}$..., $F_{NN,G}$} and $S_{Cr,N}$ = [$Cr_{N1,G}$,.... $Cr_{NN,G}$}). Similarly, a fitness improvement array ($\triangle f_N$, = [$\triangle_{fNN,G}$.... $\triangle f_{NN,G}$ }) is created to store the fitness improvements for corresponding individuals of $S_{i,N}$. Thus, the new F and Cr values for the ith member are calculated as shown in (**17**) and (**18**) correspondingly.

$$F_i = \begin{cases} mean_{WL}(S_{F,N}) & if\ S_{F,N} \neq \phi, \\ rand[0.5, 1] & otherwise, \end{cases} \quad (17)$$

$$Cr_i = \begin{cases} mean_{WA}(S_{Cr,N}) & if\ S_{Cr,N} \neq \phi, \\ rand[0.1, 1] & otherwise. \end{cases} \quad (18)$$

Here rand produces a uniformly distributed random number between a given range specified by [.,.]. The overall SHADE algorithm, equipped with the proposed neighborhood-based parameter adaptation scheme is summarized in **Algorithm 1**.

3.2. Intuition behind the proposal - an empirical illustration

The intuition behind the proposed neighborhood-based adaptation scheme can be empirically illustrated by using **Fig. 1**. Let us consider the following aspects of the 6 neighboring members of a current population member shown by using a black pentagon marker in **Fig. 1**, $\triangle_{fN}$ = [25. 20. 100, 50. 150. 100], $S_{F,N}$ = [0. 3. 0. 2. 0.5 . 0.52 , 0. 3 . 0.5], and $S_{Cr,N}$ = [0.4. 0.3. 0.41.0.42. 0.5. 0.43].

If we consider all the 15 successful members from the current generation, then $\triangle f$ = [1000. 500. 700. 800. 1200. 700, 500. 600. 900. 25. 20, 100. 50. 150, 100]. S_F = [0. 8. 0.75. 0.96. 0.56, 0.76. 0.96. 0.8. 0.75, 0.95. 0.3. 0. 2 , 0. 5 . 0.52 . 0. 3 , 0. 5]. and S_{Cr} = [0. 8. 0.9. 0.95. 0.4. 0.96, 0.54. 0.69. 0.78. 0.56. 0.4. 0.3. 0.41. 0.42. 0.5, 0.43]. Now according to (12), $w_{N,k}$ = [0.0562. 0.0449. 0. 2247. 0. 1124. 03371.0.2247] and w_k = [0. 1361.0.0681,0.0953. 0. 1089. 0. 1634. 0.0953. 0.0681. 0.0817. 0. 1225. 0.0034. 0.0027. 0.0136. 0.0068. 0.0204. 0.0136].

Evidently, if we adapt F and Cr according to the proposed neighborhood scheme, the values are 0.4387 and 0.4404 respectively. However, if all the successful members from the population are considered, then F becomes 0.8160 and Cr equals 0.7186, which are considerably higher and can cause overshoot of the resulting offspring beyond the optimal basin. **Fig. 2** illustrates the above discussed overshooting problem for a larger F value on a toy fitness landscape. In this context, one particular situation can be of interest, when all the local neighbors of an individual are stuck inside a locally optimal basin of attraction. In such case, for successive generations, the successful neighborhood will remain empty, thus causing a re-initialization of F and Cr according to (**17**) and (**18**) respectively. This may, in turn, provide opportunities to escape the local basin. Note that in the proposed strategy, there is no need to maintain a historical archive of parameter values for all the previously successful candidates, as well.

To analyze further, let us consider as an example, the 30 dimensional function F_{10} from the CEC 2013 benchmark suite here. **Figs. 3, 4** depict the distribution of F and Cr values corresponding to the population members for three different search scenarios, namely when 10% of the maximum allowable Function Evaluations (FEs) are exhausted indicating the beginning phase of the search, when 50% are exhausted, representing the middle phase of search and finally, and when 100% allowable FEs are exhausted, implying the termination of the search, as it has exhausted its resources. From **Fig. 3**, we can see that with the progress of the search, both of the control parameters for respective population members got stuck in a certain value range for SHADE and hence, a horizontal cluster line is visible for each of F and Cr. Such constriction of F and Cr values during the later stages of search implies similar perturbation of all the population members, thus indicating a lack of diversity and higher chances of stagnation in a sub-optimal basin of attraction. On the other hand, **Fig. 4** shows that for N-SHADE, no such stagnation occurs as the search progresses. In the final stage of the search, formation of small clusters of F and Cr values are visible, but they are due to the neighborhood generation and as is clear, not all parameter values get stipulated in a small range, like what happens for the SHADE algorithm. Our experiments with other benchmark functions from CEC 2013 (not reported here due to lack of space) also indicate a similar trend for SHADE and N-SHADE.

Algorithm 1 The N-SHADE algorithm

Input: The optimization problem, population size N_p, maximum permitted generation MAX_GEN.
Output: The solution of the optimization problem.

Generate N_p candidate solutions uniform randomly within the given bounds for dimensions as the initial population.
Initialize gen_count to 0.
Initialize all values of M_F and M_{Cr} to 0.5.
Set Archive $A = \phi$.
$S_{Cr,N} = \phi$, $S_{F,N} = \phi$.
while $gen\_count < MAX\_GEN$ **do**
  for $i = 1$ to N_p **do**
    $\vec{X}_{i,G} = P_{i,G}$.
    Create spatial neighborhood of $round(\sqrt{N_p})$ members around $\vec{X}_{i,G}$ by using the l_1 norm induced distance.
    p_i = select randomly from $[p_{min}, 0.2]$.
    Set F_i by (17)
    Set Cr_i by (18).
    Create $\vec{V}_{i,G}$ by mutation (5).
    Create $\vec{U}_{i,G}$ by crossover (3).
    if $f(\vec{U}_{i,G}) \leq f(\vec{X}_{i,G})$ **then**
      $\vec{X}_{i,G+1} = \vec{U}_{i,G}$.
      $\vec{X}_{i,G} \rightarrow A$.
      $F_i \rightarrow S_{F,N}$, $Cr_i \rightarrow S_{Cr,N}$.
    else
      $\vec{X}_{i,G+1} = \vec{X}_{i,G}$.
    end if
    if $|A| > Np$ **then**
      randomly delete an $|A| - N_p$ individuals from A.
    end if
  end for
  Set $gen\_count = gen\_count + 1$
end while
Return the best member $\vec{X}_{best}$ of the final population.

Algorithm 2 Experimental Complexity Analysis

for $i = 1$ to 1000000 **do**
  $x = x + x$
  $x = \sqrt{x}$
  $x = \log(x)$
  $x = e^x$
  $x = x/(x + 2)$
end for

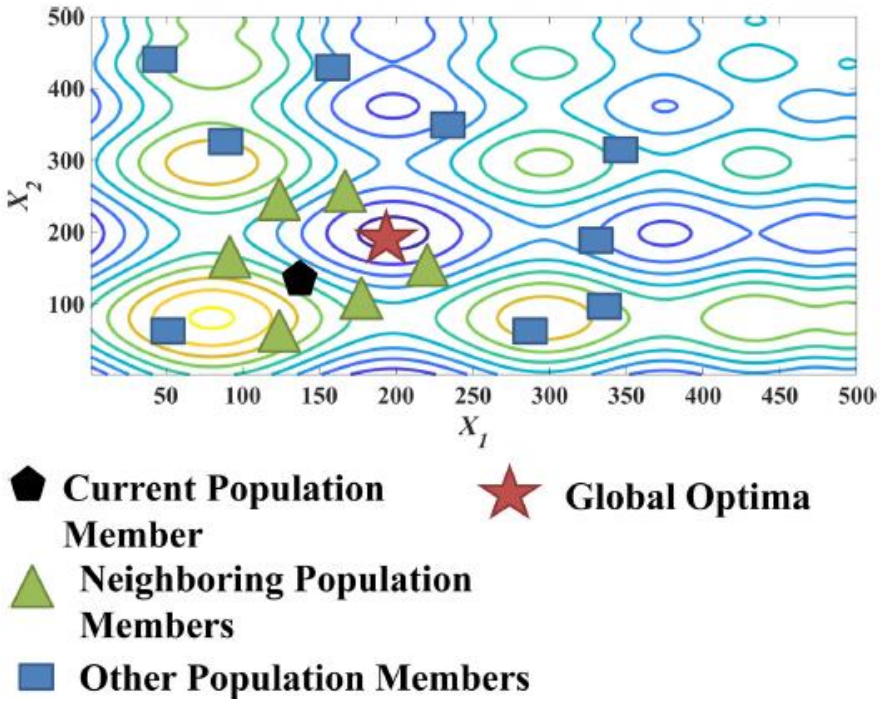

Fig. 1. Neighborhood based parameter adaptation scheme.

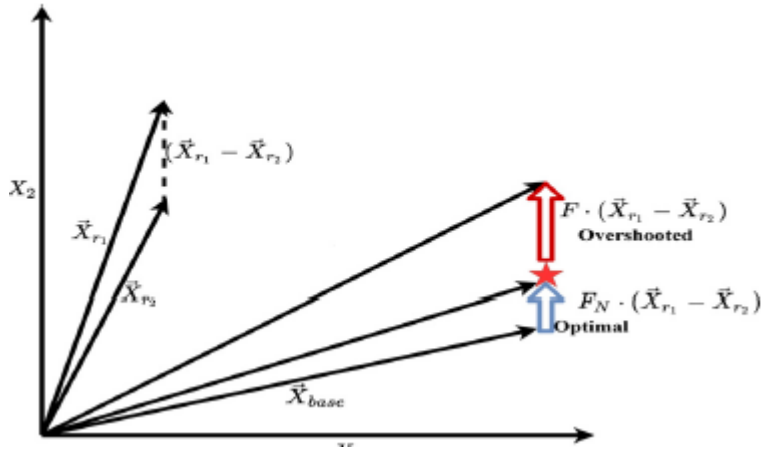

Fig. 2. Overshooting scenario in mutation.

3.3. A note on time complexity of the resulting DE variants

A conventional DE with population size N_p, running on a D-dimensional search space for G generations has an asymptotic com- plexity of $0(N_p.D.G)$. In the case of the proposed neighborhood-based parameter adaptation scheme, in each generation, if N_s population members become successful, we have to calculate $N^2{}_s$ number of distances where each calculation takes $O(D)$ time. Thus, in the worst possible scenario (from the perspective of time complexity) N_s = N_p and the complexity of the parameter adaptation scheme over G generations is 0(N2.D.G). Thus, the worst-case complexity of the DE with our proposed adaptation strategy turns out to be $0(\max(N^2{}_p.D.G,\ N_p.D.G)) = 0(N^2{}_p\ .D.G)$.

Thus, asymptotic bounds on the complexity are scaled by a factor the same as the population size in our proposal. Since, most often for medium to large scale problems, a population size of N_p = 100 is sufficient, this constant scaling on the asymptotic bound does not impose a very serious restriction on the complexity. This will also be evident from our experiments reported in **Section 4.5**.

3.4. An empirical analysis of proposed spatial neighborhood-based parameter adaptation with SHADE

In this section, we do an empirical analysis of the SHADE framework equipped with our spatial neighborhood-based parameter adaptation scheme, concerning a few desirable aspects of an evolutionary optimizer. In particular, here we explain how the proposed parameter adaptation scheme may help in a better contour fitting property and more improved saddle crossing behavior concerning the existing SHADE framework.

3.4.1. Enhancing the contour matching property

An intriguing characteristic of DE is contour matching [**30,31**]. Contour matching refers to the progressive adaptation of the difference vector distribution in DE to the local shapes of the functional landscape. The integration of spatial neighborhood information in parameter adaptation of SHADE enables the algorithm to efficiently cope up with diverse local properties of the functional landscape as shown in Figures 3 and 4 of the supplementary document for Rosenbrock's function. As can be seen from **Fig.4** (supplementary document), when the population of DE variants drives along a curved valley, for N-SHADE, proposed spatial neighborhood-based parameter adaptation scheme guides the population with local functional landscape information and thus, the adaptation of the local shape of the objective function can occur much earlier than the original SHADE algorithm.

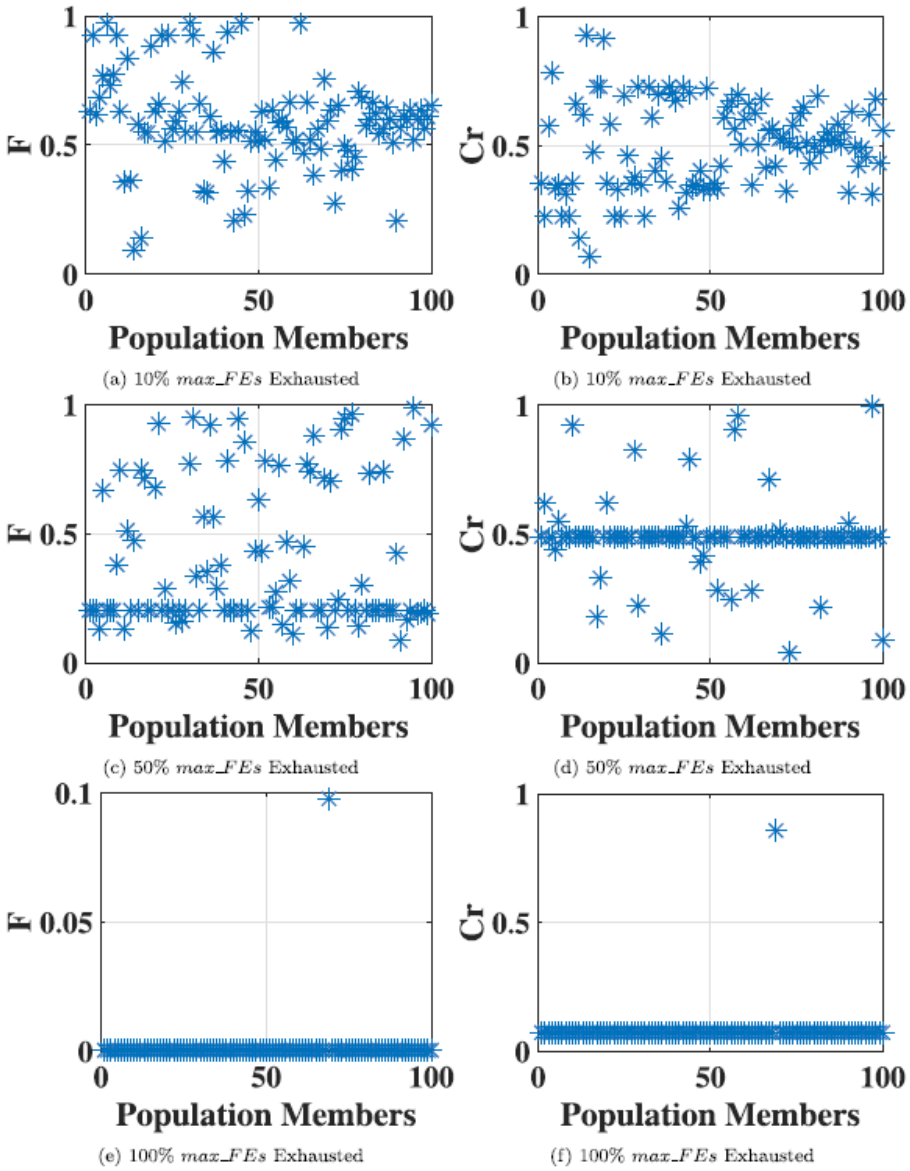

Fig. 3. Distribution of F and Cr over the indices of the population members for SHADE. Here max F Es denote the maximum number of F Es allowed.

3.4.2. Enhancing the saddle crossing behaviour

Saddle crossing [**32**] is another desirable property, which refers to the phenomenon of crossing a ridge-like saddle between the basins. Such basin-to-basin transfer is a favorable property of DE as it enhances the ability to quickly capture the global minima over a fitness landscape with multi-modality. In proposed scheme, neighborhood information while adapting parameters can steer the population towards global optima. This fact is illustrated on the peaks function with two minimal basins of attraction through **Figs. 1** and **2** (from the supplementary document). We can observe that starting from an asymmetric initialization, N-SHADE can transfer population members to the globally minimal basin much earlier than original SHADE. For more details regarding these figures, see the description in the supplementary document.

4. Experimental results & discussions

This section deals in the empirical study of the performance for proposed method with respect to other competitors for selected IEEE CEC benchmark suites.

4.1. Peer algorithms, benchmark suite, and computational protocols

We consider five well-known DE variants: SHADE and L-SHADE (SHADE [5], L-SHADE [**17**], Db-SHADE and Db-L-SHADE [**23**], and jSO [**20**]), all of which use the SHADE type parameter adaptation scheme, to experimentally validate our proposal. In this section, we present a detailed performance comparison between these algorithms and their counterparts (denoted by prefixing the name of the original algorithm with 'N', denoting Neighborhood) employing the proposed adaptation strategy. We evaluate the performance of the neighborhood-based parameter adaptation scheme with SHADE on the IEEE CEC 2013 competition and special session on single-objective real parameter numerical optimization suite [**33**] in 10, 30, and 50 dimensions, with L-SHADE on CEC 2014 suite [**33**] in 10, 30, 50, and 100 dimensions, with Db-SHADE and Db-L-SHADE on CEC 2015 suite [**34**] in 10 and 30 dimensions, and finally, with jSO on CEC 2017 [**20**] in 10, 30, and 50 dimensions. All the peer algorithms except Db-SHADE and Db-L-SHADE are the DE-based winners for the respective CEC competitions. We also test L-SHADE equipped with our neighborhood-based adaptation scheme on the IEEE CEC 2011 Competition on testing evolutionary algorithms on real-world optimization problems [24] to demonstrate the usefulness of our proposal. On this problem, we compare the proposed N-L-SHADE against L-SHADE and GA-MPC [**25**] as the latter won the 2011 CEC competition on testing evolutionary algorithms on real-world optimization problems [**24**].

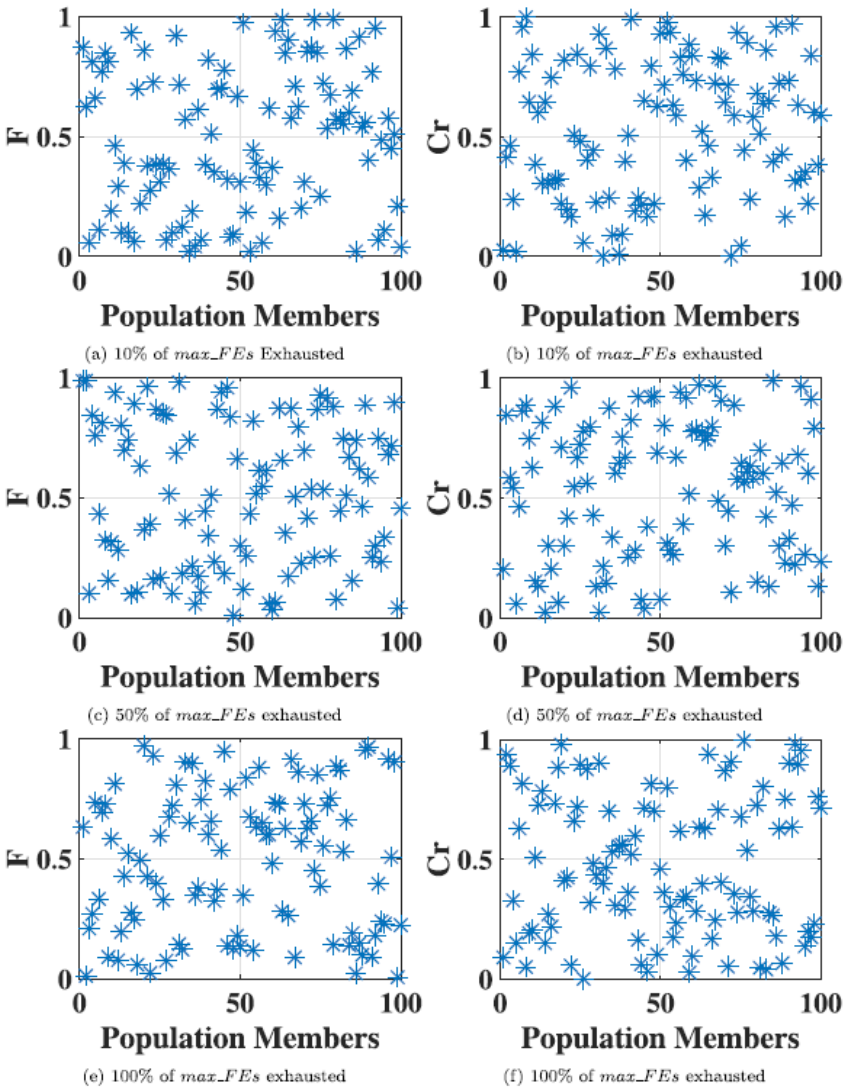

Fig. 4. Distribution of F and Cr over the indices of the population members for N-SHADE. Here max F Es denote the maximum number of F Es allowed.

We report the main results in terms of mean best-of-the-run error values over 51 runs (as recommended in [**20**]) which are recorded in an independent manner. The error is measured as $|f_{best} - f^*|$, where fbest is the best objective function value returned by an algorithm in a run and f^* is the known optimum value of that function. A result less than 10^{-8} is rounded to 0.0 in this simulation process. The maximum budget of FEs (max_FEs) is taken as D x 10, 000 as mentioned in [**20,33,34**]. Only for the CEC 2011 problems, max_FEs for all the competing algorithms are fixed to 150000. The control parameters of all the competitor algorithms are kept similar to their original literature. Note that our choice of 4 different artificial benchmark suites spanning over the CEC competitions for years 2013, 2014, 2015, and 2017 along with the collection of real-world problems from CEC 2011 competition are in accordance with the suggestions given in a very interesting work of Piotrowski [**35**], where the author cautioned on possible ranking bias among the peer algorithms due to the use of only one specific test suite.

A well-known non-parametric statistical test, called Wilcoxon's rank-sum test [**36**] is conducted at 5% significance level to judge the signifi- cance of the reported results. In all the summary result tables, the results marked with " + "," - " and "≈" indicate that DE variant with the proposed neighborhood-based adaptation is significantly better than, worse than, or equivalent to the corresponding competitor respectively. See the supplementary document for the fully detailed results on each function.

All the simulations are carried out on a workstation having two Intel Xeon gold processor having 10 cores and 20 threads each, both of them clocked at 3.9 GHz, associated with 128 GB of DDR4 ECC RAM at 2400 MHz, running Linux based operating system with MATLAB r2017a.

4.2. Effect of neighborhood size on search performance

Table-VI (see supplementary document) summarizes the results corresponding to the effect of different neighborhood sizes on the search performance. In this study, seven 10D and 50D test functions (each of different characteristics like unimodal, multi-modal, composite etc.) are selected from the CEC 2013 suite and we run N-SHADE with the following seven different neighborhood sizes: $N_p/8, N_p/6, N_p/4$, proposed $\sqrt{N_p}$, $2 \cdot \sqrt{N_p}$, $4 \cdot \sqrt{N_p}$ and $8 \cdot \sqrt{N_p}$. Table-VIA shows a performance comparison between the above-said neighborhood sized in 10.D and Table-VIB shows a performance comparison between the above-said neighborhood sized in 50D. It can be seen that for both of the dimensions N-SHADE with the proposed neighborhood size of $\sqrt{N_p}$ performs consistently better than the N-SHADE variants with other sizes and it is also indicative from Table-VI that size of the neighborhood does not depend upon the problem dimension. Testing across other suites and in other dimensions (not reported here due to space economy) leaves our conclusion unaltered.

4.3. Effect of Euclidean distance metric to create neighborhood on search performance

Table-VII (see supplementary document) summarizes the comparative performance between proposed l_1 norm induced distance metric and the traditional l_2 norm-based metric. It is apparent from this comparison that the choice of distance metric has no adverse effect on search performed by the algorithm but l1 norm has a lower computational overhead as discussed in **Section 3.1** and empirically shown in **Table 7** (Section 4.5) in terms of execution time. This comparison empirically justifies our choice of using Manhattan distance in the neighborhood formation stage of the proposed algorithm.

4.4. Main results and discussions

IEEE CEC 2013 benchmark suite incorporates total 28 black-box optimization problems and testing dimensions for these problems are D = 10, 30, and 50. Among these 28 functions, $f_1 - f_5$ are unimodal, $f_6 - f_{20}$ are basic multimodal and $f_{21} - f_{30}$ are composite functions. Our proposed neighborhood-based parameter strategy is coupled with the competition winner SHADE (N-SHADE) and its performance is compared with original SHADE algorithm in CEC 2013 benchmark suite.

Table 1 summarizes the comparative performance between N-SHADE and SHADE. In 10D problems, N-SHADE significantly outperforms SHADE on 16 problems, gives statistically equivalent results in 12 problems, and does not looses to SHADE in any of the functions. For 30D problems, N-SHADE outperforms SHADE in 17 cases, gives statistically equivalent results in 8 cases, and looses to SHADE in only 3 cases, among which, f_3 and f_4 are unimodal in nature, whereas only f_{14} is multimodal.

Table 1 N-SHADE vs SHADE summary results on the CEC 2013 Suite

	10D	30D	50D
+	16	17	21
≈	12	8	4
-	0	3	3

Table 2 N-L-SHADE 2014 Suite vs L-SHADE summary results on the CEC

	10D	30D	50D	100D
+	22	17	22	18
≈	6	6	2	10
-	2	7	6	2

Table 3 N-SHADE vs. SHADE & Db-SHADE summary results on the CEC 2015 Suite

	vs SHADE		vs Db-SHADE	
	10D	30D	10D	30D
+	8	10	10	8
≈	7	5	5	4
-	0	0	0	3

In the last case, for 50D problems, N-SHADE outperforms SHADE in 21 instances, gives statistically equivalent results in 4 cases, and cannot perform significantly better in 3 cases, these 3 cases are same problems, in which SHADE dominated N-SHADE previously. Note that N-SHADE performs better than SHADE on unimodal, multimodal, and composition functions of widely varying characteristics including quadratic ill-conditioning (e.g. on f_2), smooth local irregularities (e.g. on f_3), presence of huge local optima (e.g. functions like f_7 , f_{11} , and f_{12}), different properties around different local minima (functions f_{21} -f_{28}) and so on. However, a ridge-like structure in the fitness landscape (as in the case of the function f_3) or a particularly sensitive direction (as in f_4) may attenuate the performance of N-SHADE.

In CEC 2014 benchmark suit, there are 30 functions, among which $f_1 - f_3$ are unimodal, $f_4 - f_{16}$ are basic multimodal, $f_{17} - f_{22}$ are hybrid functions and $f_{23} - f_{30}$ are composite functions. For this suite, we summarize the comparative performance between L-SHADE and N-L-SHADE algorithms in **Table 2**.

On the CEC 2014 suite, N-L-SHADE significantly outperforms L-SHADE in 22, 17, 22, and 18 problems and looses to it in 2, 7, 6, and 2 problems while yielding equivalent results in 6, 6, 2, and 10 problems for the 10.D, 30.D, 50D, and 100.D functions respectively. Our experiments reveal that the performance of N-L-SHADE remains statistically consistent on diverse nature of the functions as before. Also, N-L-SHADE performs reasonably well in higher dimensions (D = 100) compared to D = 30 and 50. In fact for 100D functions, N-L-SHADE dominates L-SHADE on all the functions except a multi-modal function f_{15} and on another hybrid composition function f_{21}.

Next, on the CEC 2015 test suite, we compare N-SHADE with SHADE and Db-SHADE. We also compare N-L-SHADE with L-SHADE and Db-L-SHADE using the same suite. Note that the CEC 2015 test suite was used by the authors in [**23**] to test their proximity induced weight vector adaptation in SHADE.

Table 3 indicate that out of the 15 benchmark functions, the proposed scheme significantly outperforms SHADE on 8 10D functions and 10 30D functions, while giving equivalent results in 7 and 5 cases for the 10D and 30D functions respectively. The proposed method outperforms Db-SHADE in 10 and 8 cases for 10D and 30D functions respectively. In 30D comparison, N-SHADE looses to Db-SHADE in only 3 cases and these are f1 (unimodal function), f5 (multi-modal), and f6 (hybrid). In comparison with L-SHADE, N-L-SHADE gives statistically better result in 6 cases for 10.D and 10 cases in 30D problems and interestingly, never looses to L-SHADE, as it yields statistically comparable results to L-SHADE in 9 10D functions and 5 30D functions. **Table 4** indicate that N-L-SHADE significantly outperforms Db-L-SHADE in 5 10.D functions and 12 30.D functions, while giving comparable results in 9 and 3 cases for 10D and 30.D problems respectively. Only on the multi-modal function f_5 in 10D, Db-L-SHADE is able to produce better average error as compared to N-L-SHADE. For 30.D instances, N-L-SHADE never yields statistically insignificant result compared to Db-L-SHADE.

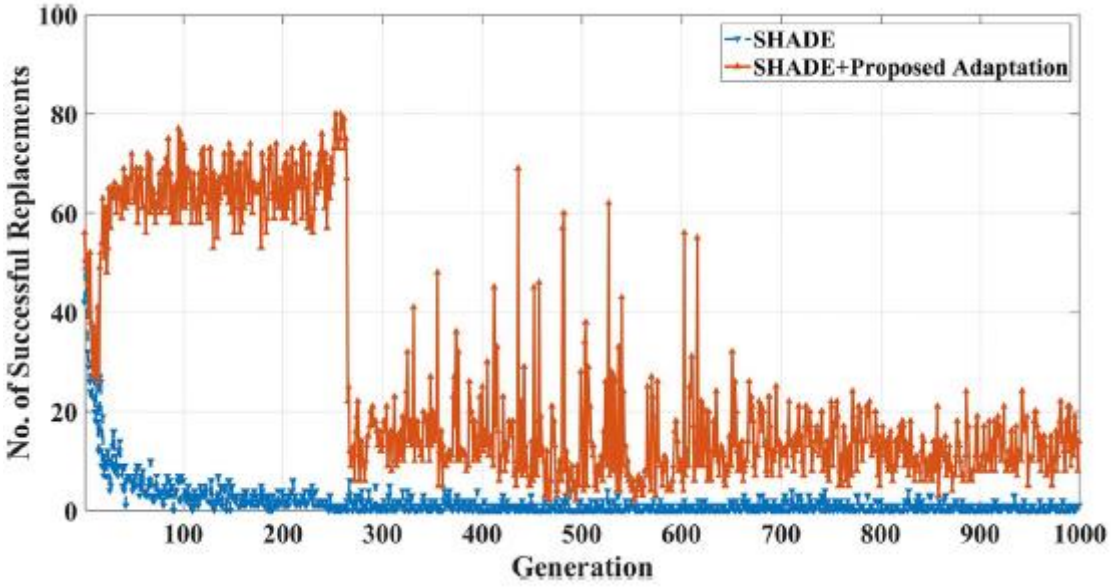

Fig. 5. Offspring replacement comparison between N-SHADE and SHADE for f_{28} of CEC 2013 suite.

Table 4 N-L-SHADE vs. L-SHADE & Db-L-SHADE summary results on the CEC 2015 Suite

	vs	L-SHADE	vs	Db-L-SHADE
	10D	30D	10D	30D
+	6	10	5	12
≈	9	5	9	3
-	0	0	1	0

Lastly, we integrate the proposed scheme with the SHADE-based winner of the CEC 2017 competition - jSO. The corresponding results are summarized in **Table 6**. We observe that, out of the 30 benchmark instances, N-jSO outperforms jSO in 17, 20, and 21 cases, gives equivalent result in 9, 5 and 4 cases and looses to it in 4, 5, and 5 cases for the 10D, 30.D, and 50.D problems respectively. In 10.D problems, N-jSO looses to jSO in f_{12}, f_{18}, f_{27}, and f_{30}. Note that f_{12} and f_{18} are hybrid functions composed of 3 and 4 basic functions and interestingly, both involve the ill-conditioned elliptic function with very high condition number. Functions f_{21} and f_{30} are both composition functions and f_{21} also involves the high-conditioned elliptic function as one component. Thus, the spatial neighborhood-based adaptation seems to be not so effective if the iso-contours of the function are severely squeezed due to ill-conditioning. For 30.D functions, in addition to the 4 functions mentioned for the 10D instances, jSO also outperforms N-jSO on the shifted and rotated Rastrigin's function f_5, which is multi-modal in nature. The huge distance between the global optimum and the second better optimum for this function is likely to deceive the neighborhood-based adaptation scheme of our proposal. However, for majority of functions with multi-modality, non-separability, heterogeneous landscape properties around different local optima, and non-uniform nature of the subcomponents for the hybrid functions, N-jSO still retains its statistically superior performance over jSO. A good heuristic black-box optimizer is expected to maintain such consistent performance.

The effectiveness of proposed neighborhood based parameter adaptation technique is further tested on CEC 2017 benchmark suite and results are summarized in in the **Table 5**. In this comparison, we have tested the performance of proposed scheme with L-SHADE, Db-L-SHADE. In the N-L-SHADE vs L-SHADE comparison, proposed enhancement boosts L-SHADE algorithm's performance for 19, 23 and 24 functions out of total 30 functions in 10.D, 30.D and 50D respectively. It can be seen that for only 2 functions in 10.D and 50D, N-L-SHADE performs worst than original L-SHADE. **Table 5**, shows the performance comparison of N-L-SHADE with Db-L-SHADE, in this case, L-SHADE with the proposed neighborhood-based parameter adaptation scheme outperforms its competitor in 19, 20 and 24 cases in 10.D, 30.D and 50D respectively. From this comparison, we observe that the proposed scheme does not loses its effectiveness with increasing problem dimension.

Table 5 N-L-SHADE vs. L-SHADE & Db-L-SHADE summary results on the CEC 2017 Suite

	vs	L-SHADE		vs	Db-L-SHADE	
	10D	30D	50D	10D	30D	50D
+	19	23	24	19	20	24
≈	9	7	4	10	9	5
-	2	0	2	1	1	1

Table 6 N-jSO vs. jSO summary results on the CEC 2017 Suite

	10D	30D	50D
+	17	20	21
≈	9	5	4
-	4	5	5

4.5. Algorithm complexity: an empirical analysis

To illustrate the fact that the proposed neighborhood-based parameter adaptation scheme does not impose a serious computational overhead to the already existing algorithmic framework of SHADE, we measure the complexity of jSO with proposed scheme as instructed for the IEEE CEC 2017 competition [**20**]. In **Table 7**, the required time to run following code is shown as T_0,

Here, We compute the complexity for D = 10, 30, and 50. Tt is the time required to execute 200,000 FEs of f_{18} from CEC 2017 suite and T_2 is the time to execute jSO with the proposed adaptation scheme (i.e. N-jSO with neighborhoods formed by using the norm induced distance) for 200,000 FEs for the same fis in dimension D. $T_{2\text{-}0}$ is the time to execute normal jSO for 200,000 FEs on fls in dimension D. $T_{2\text{-}E}$ is the time to execute N-jSO with the Euclidean distance-based neighborhood for 200,000 FEs on fis in dimension D. **Table 7** indicates that the complexity of N-jSO with proposed scheme is only marginally higher than the original jSO and is lesser than that of N-jSO with Euclidean neighborhoods.

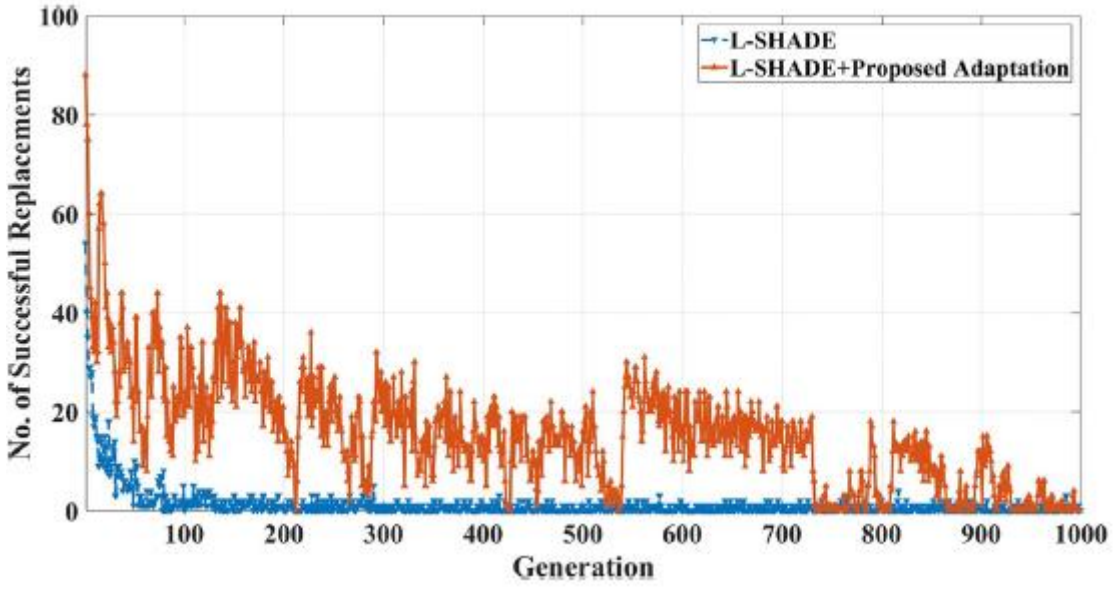

Fig. 6. Offspring replacement comparison between N-L-SHADE and L-SHADE for f_{25} of CEC 2014 suite.

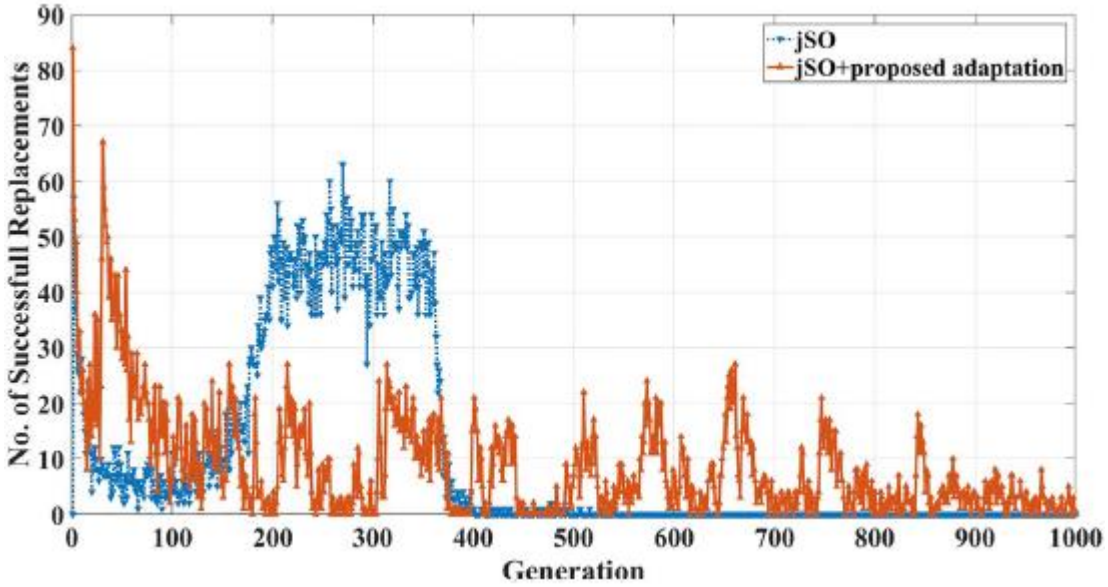

Fig. 7. Offspring replacement comparison between N-jSO and jSO for f_{17} of CEC 2017 suite.

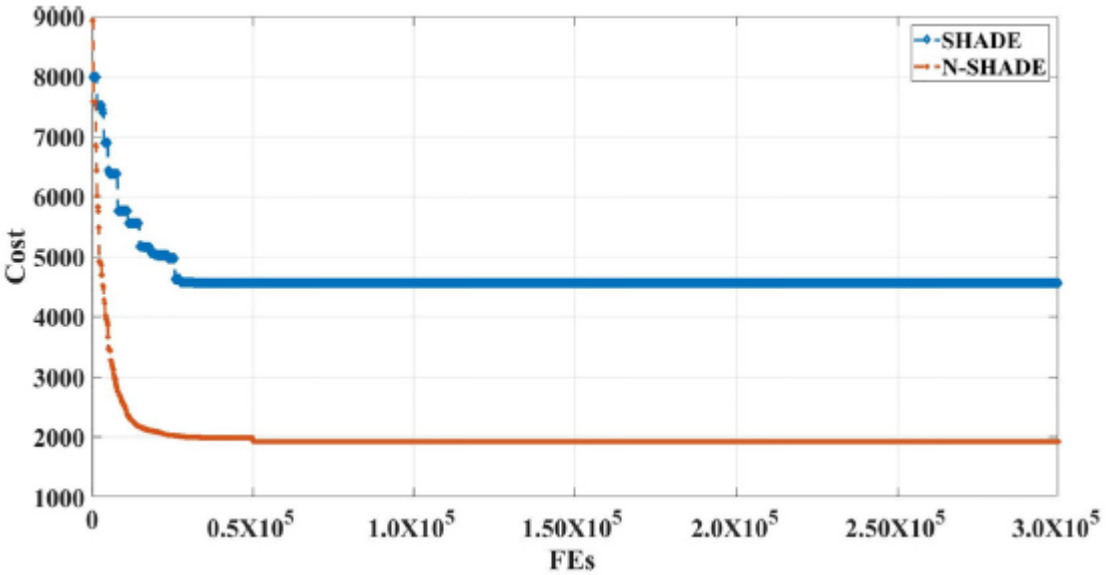

Fig. 8. Convergence comparison between SHADE and N-SHADE on f_{28} of CEC 2013 suite.

4.6. Effect of proposed parameter adaptation scheme on SHADE, L-SHADE, and jSO - a closer look

A population-based optimizer's search performance can be measured by the number of offspring that are replacing their parents in each iteration due to their improved fitness. A progressive increase in this number indicates that the search is progressing in a positive direction, with the discovery of newer interesting answers; however, a decrease in this number, or no change in this number for subsequent generations, may suggest that the search process has come to a halt. In **Figs. 5-7** variation of this number of successful offspring replacements over generations is plotted for a few sample functions: f_{28} (CEC 2013 composite function), f_{25} (CEC 2014 composite function), and f_{17} (CEC 2017 hybrid function).

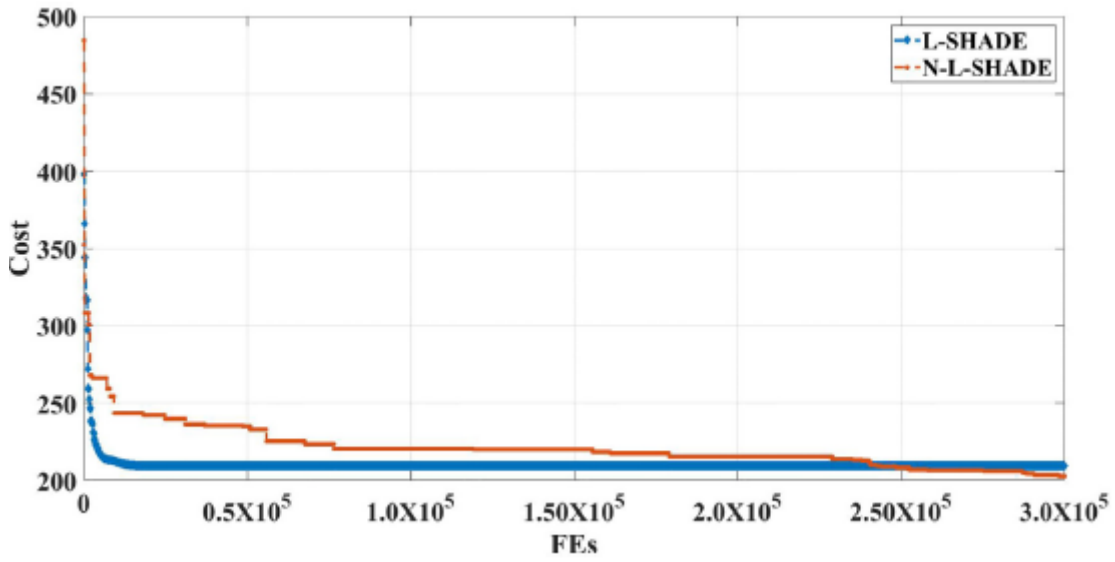

Fig. 9. Convergence comparison between L-SHADE and N-L-SHADE on f_{25} of CEC 2014 suite.

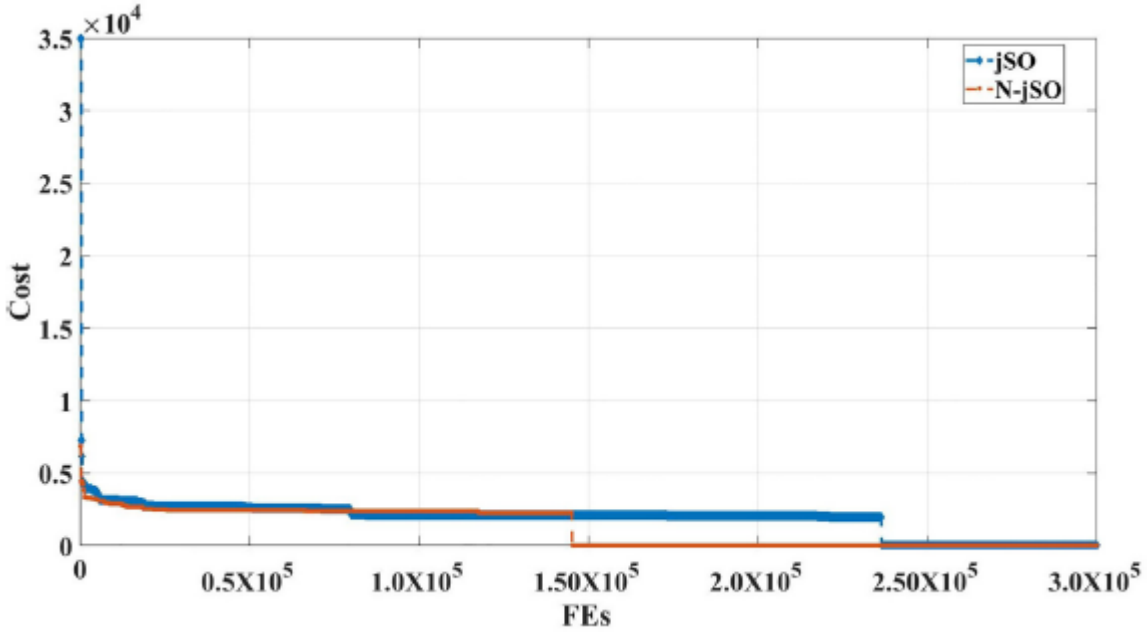

Fig. 10. Convergence comparison between jSO and N-jSO on f_{17} of CEC 2017 suite.

In all the cases, it can be seen that successful replacement is always higher for algorithms coupled with the neighborhood-based parameter adaptation scheme compared to their original counterparts. This implies that the use of the proposed scheme can induce a higher probability of generating better offspring by utilizing neighborhood information in a complicated fitness landscape. **Fig. 8** shows the convergence characteristics of SHADE and N-SHADE (for the median run, when the collection of runs was ordered based on their ”best-of-the-run” errors) on f_{28} , which is a composite function from CEC 2013 suite. The figure suggests that N-SHADE provides faster convergence speed and better final results. **Fig. 9** presents the convergence characteristics of L-SHADE and N-L-SHADE on f_{25} , which is again a composite function from the CEC 2014 suite. In this case, also, N-L-SHADE provides faster

convergence and better final cost value. Finally, **Fig. 10** shows the convergence characteristics of jSO and N-jSO on function f_{17} from the CEC 2017 suite. Here also proposed adaptation based jSO variant discovers better solutions with lower cost earlier than jSO.

4.7. Performance on the real-world numerical optimization problems from IEEE CEC 2011 competition

To test the effectiveness of the proposed neighborhood-based parameter adaptation scheme, we have considered a benchmark suite for IEEE CEC 2011 competition on testing evolutionary algorithms on real-world numerical optimization problems.

Table 7 A sample algorithm complexity results for jSO and N-jSO

Algorithm	D	10	30	50
Algorithm 2	T_0	0.0958	0.0958	0.0958
F_{18}	T_1	0.6915	1.6859	3.5625
N-jSO	T_2	2.1265	3.8525	5.4531
jSO	T_{2-o}	2.0012	3.5645	5.3925
N-jSO(l_2 norm)	T_{2-E}	2.3289	3.9865	5.9856
jSO	$\frac{T_{2-o}-T_1}{T_0}$	13.6711	19.6096	19.1022
N-jSO	$\frac{T_2-T_1}{T_0}$	14.9791	22.6158	19.7348
N-jSO(l_2 norm)	$\frac{T_{2-E}-T_1}{T_0}$	17.0918	24.0146	25.2933

This suite comprises a total of 22 (by treating the different instances of each problem as separate ones) real-world numerical optimization problems. This benchmark suite holds both unconstrained and constrained optimization problems. Four base algorithms namely, GA-MPC, SAMODE, SHADE, and L-SHADE are considered here for performance comparison purposes. The detailed results for this comparison can be found in Table-V in the companion supplementary file. Summary of this comparison is presented in **Table 8**. SAMODE proposed a DE algorithm with multiple strategies and it was ranked 3rd in the CEC 2011 competition on testing evolutionary algo rithms on real-world numerical optimization problems.

Table 8 Summary results for N-L-SHADE vs SAMODE, GA-MPC, SHADE and L-SHADE on the CEC 2011 set of 22 real-world optimization problems

	SAMODE	GA-MPC	SHADE	L-SHADE
+	18	16	21	19
≈	4	5	1	2
-	0	1	0	1

GA-MPC was ranked 1st in the above-mentioned competition. In this comparison we have reported results for 150000 function evaluations over 25 independent runs as per prescribed guidelines in [**24**]. **Table 8** shows the $+/-/\approx$ comparison between N-L-SHADE and SAMODE, GA-MPC, SHADE and L-SHADE, and we can see that for all of the cases N-L-SHADE, which is basically L-SHADE with the proposed neighborhood-based parameter scheme outperforms its competitors.

5. Conclusion

DE has gone through several modifications to achieve superior search-ability on complex optimization problems. The adaptation of control parameters plays a very crucial role in controlling the performance of DE. SHADE has established itself as a very competitive form of DE that heavily uses parameter adaptation. In this paper, we proposed a spatial neighborhood-based scheme to implicitly incorporate the local landscape information into the parameter adaptation process of SHADE and its variants. Our proposed scheme is validated by using the well-known IEEE CEC 2013, CEC 2014, CEC 2015, and CEC 2017 competitions on real parameter bound-constrained optimization problems. The performance of the proposed scheme is also tested on real-world numerical optimization problems.

There are a few intriguing possibilities for expanding our work. Geodesic and other Non-Euclidean distance metrics, as well as semimetrics (not obeying the triangle inequality), can be used to form the neighborhood, which may better reflect the more complex fitness landscapes. Other creative neighborhood topologies can be coupled with our parameter adaption approach to improve its performance (for a thorough coverage of such proximity structures in connection to DE and PSO, see the recent survey citelynn2018population). The basic concept of neighborhood-based parameter adaptation can be combined with other meta-heuristics such as PSO, where the most promising local parameters (such as the inertia factor and acceleration coefficients) can be saved and reused to guide particle flight in subsequent rounds.

References

[1] R. Storn, Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical report, Int. Comput. Sci. Instit. 11 (1995) .

[2] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution-an updated survey, Swarm Evol. Comput. 27 (2016) 1-30,

[3] J. Zhang, A.C. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945-958.

[4] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2008)398-417 .

[5] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: 2013 IEEE congress on evolutionary computation, IEEE, 2013, pp. 71-78 ,

[6] S.M. Islam, S. Das, S. Ghosh, S. Roy, P.N. Suganthan, An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization, IEEE Trans. Syst. Man Cybern. Part B (Cybernetics) 42 (2) (2011) 482-500,

[7] M.G. Epitropakis, D.K. Tasoulis, N.G. Pavlidis, V.P. Plagianakos, M.N. Vrahatis, Enhancing differential evolution utilizing proximity-based mutation operators, IEEE Trans. Evol. Comput. 15 (1) (2011) 99-119.

[8] S.-M. Guo, C.-C. Yang, Enhancing differential evolution utilizing eigenvector-based crossover operator, IEEE Trans. Evol. Comput. 19 (1) (2014) 31-49.

[9] X.-G. Zhou, G.-J. Zhang, Differential evolution with underestimation-based multimutation strategy, IEEE Trans. Cybern. 49 (4) (2018) 1353-1364.

[10] R. Mallipeddi, P.N. Suganthan, Q.-K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679-1696.

[11] S.X. Zhang, S.Y. Zheng, L.M. Zheng, An efficient multiple variants coordination framework for differential evolution, IEEE Trans. Cybern. 47 (9) (2017) 2780-2793.

[12] X.-F. Liu, Z.-H. Zhan, Y. Lin, W.-N. Chen, Y.-J. Gong, T.-L. Gu, H.-Q. Yuan, J. Zhang, Historical and heuristic-based adaptive differential evolution, IEEE Trans. Syst. Man. Cybern. 49 (12) (2018) 2623-2635.

[13] A. Ghosh, S. Das, A.K. Das, L. Gao, Reusing the past difference vectors in differential evolution-a simple but significant improvement, IEEE transactions on cybernetics (2019) .

[14] X. Wang, Z. Dong, L. Tang, Multiobjective differential evolution with personal archive and biased self-adaptive mutation selection, IEEE Trans. Syst. Man Cybern. (2018) .

[15] M. Leon, N. Xiong, Alopex-based mutation strategy in differential evolution, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 1978-1984.

[16] E. Harth, et al., Alopex: A stochastic method for determining visual receptive fields (1974).

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

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

[19] N.H. Awad, M.Z. Ali, P.N. Suganthan, R.G. Reynolds, An ensemble sinusoidal parameter adaptation incorporated with l-SHADE for solving CEC2014 benchmark problems, in: 2016 IEEE congress on evolutionary computation (CEC), IEEE, 2016, pp. 2958-2965.

[20] J. Brest, M.S. Maučec, B. Boskovic, Single objective real-parameter optimization: Algorithm jSO, in: 2017 IEEE congress on evolutionary computation (CEC), IEEE, 2017, pp. 1311-1318.

[21] P.N. Suganthan, Particle swarm optimiser with neighbourhood operator, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), volume 3, 1999, pp. 1958-1962Vol. 3.

[22] S. Biswas, S. Kundu, S. Das, Inducing niching behavior in differential evolution through local information sharing, IEEE Trans. Evol. Comput. 19 (2) (2015) 246263, doi:**10.1109/TEVC.2014.2313659**.

[23] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, A. Zamuda, Distance based parameter adaptation for differential evolution, in: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2017, pp. 1-7.

[24] S. Das, P.N. Suganthan, Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems, Jadavpur University, Nanyang Technological University, Kolkata (2010) 341-359.

[25] S.M. Elsayed, R.A. Sarker, D.L. Essam, Ga with a new multi-parent crossover for solving ieee-cec2011 competition problems, in: 2011 IEEE congress of evolutionary computation (CEC), IEEE, 2011, pp. 1034-1040.

[26] S.M. Elsayed, R.A. Sarker, D.L. Essam, Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems, in: 2011 IEEE Congress of Evolutionary Computation (CEC), IEEE, 2011, pp. 1041-1048.

[27] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2010) 4-31.

[28] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, A. Zamuda, Distance based parameter adaptation for success-history based differential evolution, Swarm Evol. Com-put. 50 (2019) 100462,

[29] J. Brest, M.S. Maučec, B. Boskovic, il-SHADE: Improved l-SHADE algorithm for single objective real-parameter optimization, in: 2016 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2016, pp. 1188-1195 .

[30] R. Storn, Differential evolution research-trends and open questions, in: U.K. Chakraborty (Ed.), Advances in differential evolution, Springer, 2008, pp. 1-31.

[31] K.R. Opara, J. Arabas, The contour fitting property of differential mutation, Swarm Evol. Comput. 50 (2019) 100441,

[32] J. Arabas, L. Bartnik, K. Opara, Dmeaan algorithm that combines differential mutation with the fitness proportionate selection, in: 2011 IEEE Symposium on Differential Evolution (SDE), IEEE, 2011, pp. 1-8.

[33] J.J. Liang, B.Y. Qu, P.N. Suganthan, A.G. Hernández-Díaz, 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 201212 (34) (2013) 281-295 .

[34] J.J. Liang, B.Y. Qu, P.N. Suganthan, Q. Chen, Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization, Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore 29 (2014) 625-640.

[35] A.P. Piotrowski, Regarding the rankings of optimization heuristics based on artificially-constructed benchmark functions, Inf. Sci. 297 (C) (2015) 191201, doi:**10.1016/j.ins.2014.11.023**

[36] J. Carrasco, S. Garcda, M.M. Rueda, S. Das, F. Herrera, Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review, Swarm Evol. Comput. 54 (2020) 100665, doi:**10.1016/j.swevo.2020.100665**.