Noname manuscript No.
(will be inserted by the editor)

An Advanced Strategy for Maintaining Diversity in Differential Evolution

Joel Chacón Castillo · Carlos Segura

Received: date / Accepted: date

Abstract Differential evolution is a popular evolutionary algorithm widely used in complex optimization problems. However, it is well known that the performance of differential evolution is seriously affected by the premature convergence. An alternative to alleviate this drawback is to explicitly try to maintain proper diversity. In this paper is proposed a replacement strategy that preserves useful diversity. The novelty of our method is that it induces a balance between exploration and exploitation to various optimization stages. Specifically, in the initial phases, larger amounts of diversity are accepted. The strategy proposed is oriented in the speciation concept, which is based on calculate distances to the closest surviving individual. The experimental validation is carry out with several benchmark tests and the top-rank algorithms of the competitions organized in the Congress on Evolutionary Computation. The results illustrate the usefulness of the proposal. The new method significantly improves on the best results of the state-of-the-art.

Keywords Diversity · Differential Evolution · Evolutionary

1 Introduction

Evolutionary Algorithms (EAs) are built to deal with optimization problems, which are designed from many scientific and application fields, such as science, economic and engineering [1, 2]. Principally, EAs can be classified into following categories, Genetic Algorithms (GAs) [3,4] , Evolutionary Strategies (ESs) [5], Genetic Programming (GP) [6], Evolutionary Programming (EP) [7],

Joel Chacón
Guanajuato, Gto.
E-mail: joel.chacon@cimat.mx

Carlos Segura
Guanajuato Gto.
E-mail: carlos.segura@cimat.mx

Differential Evolution (DE) [8] and other natural-inspired algorithms [9]. DE was introduced by Storn and Price [8], also is considered as one of the most effective EAs used to deal with real-world optimization problems, mainly for its convergence properties. Similarly than with other EAs, DE follows the natural evolution process which involves mutation, recombination and selection to evolve a population through an iterative progress until the criteria stop is reached. However, the peculiarity of DE resides in that it considers the difference between the vectors to explore the search space, being very similar than its precursor algorithms namely the Nelder-Mead [10] and the Controlled Random Search (CRS) [11]. In spite of the popularity and effectiveness of DE, there exists several weakness that had been partially solved through learning techniques. One of the first weakness and possibly the most important, is the performance of DE which is very sensitive to choice of the strategy parameters depending in the objective function [12]. Several strategies as adaptive and self-adaptive have been proposed to alleviate this drawback [13, 14]. However, none of them has shown superior results than the rest.

A second weakness of DE algorithms resides in the reproduction phase. In DE this phase involves the distribution of the vector differences, therefore it highly depends on the content of the population that migth provoke premature convergence in several scenarios. In fact, this issue can lead to the converge into a local optima or the lost of diverse solutions better known as premature convergence [15]. On the other hand, there exist situations where the search process could not progress and the population remains diverse, this phenomena is known as stagnation [16]. Its well known that stagnation occurs with a small population size. Although that large populations are not prone to stagnate, it involves more evaluation functions and in some scenarios might not converge, also in certain situations is not available a large population e.g. expensive optimization problems [17].

The last one drawback is highly related with the diversity of the population. Generally speaking, the search process of all the EAs involves two process: exploration and intensification. A desirable behavior of an algorithm is to produce a proper balance between both of them. So that first it induces an exploration in the search space and subsequently is induced an exploitation based in the knowledge gathered during the search process [18]. Both exploration and exploitation are equally important, since that with an excessive exploitation, the population loses its diversity and the populations members can be located in a reduced sub-optimal region of the search space. On the other hand, if the exploration is dominant, the algorithm waste resources on uninteresting regions, resulting in too slow convergence and in poor quality-solutions. Principally, DE algorithms are very likely to prematurely converge, since it introduces a high selection pressure [15]. Several strategies have been proposed in DE to deal with premature convergence through controlling the population diversity, such as parameter adaptation [18], auto-enhanced population diversity mechanisms [19] and alternative selection strategies [15].

Over the last years, a novel approach to deal with these diversity issues has been proposed, which implements a sophisticated replacement strategy that explicitly preserves the diversity [20]. This method transforms a single-objective problem into a multi-objective one, by considering diversity as an explicit objective, based in the idea of adapting the balance induced between exploration and exploitation to various optimization stages. Thus, in this way an ideal balance is reached through the criteria stop of the algorithm.

Our proposal follows a similar guideline, where it aims an ideal balance between exploration and exploitation considering the criteria stop. However, we keep the single-objective context and focus in DE algorithms.

The rest of the paper is organized as follows. The basic concepts of the classic DE and a review of the related work of diversity with DE is described in the section 2. In the section 3 is showed the tendency of algorithms of the Congress on Evolutionary Computation among the last years. Our proposal based in diversity is described in the section 4. In the section 5 are showed the experimental results including some of the most popular EAs. Finally, our conclusions and some lines of future work are given in section 6.

2 Literature Review

2.1 Differential Evolution: Basic Concepts

Although that in the literature are present several variants of DE, for simplicity in this work is used the classic DE scheme [9]. Originally DE was proposed as direct search method for single-objective continuous optimization problems. Usually, the parameters governing the system performance are presented in a vector like $\mathbf{X} = [x_1, x_2, ..., x_D]^T$, which is identified as an individual. Particularly, for real parameter optimization each parameter x_i is a real number.

In single-objective optimization, is aimed to obtain the vector $\mathbf{X}^*$ which minimizes (or maximize) a defined objective function, mathematically denoted by $f(\mathbf{X})(f : \Omega \subseteq \Re^D \rightarrow \Re)$, i.e., $f(\mathbf{X}^*) < f(\mathbf{X})$ for all $\mathbf{X} \in \Omega$, where Ω is a non-empty large finite set identified as the domain of the search.

The basic scheme of DE consists that given the target parameter vectors (each vector of the population), a new mutant (or donant) vector is created using a vector generation strategy. After that, the mutant vector is combined with the target vector to generate the trial vector. Then, each one of the trial vectors is compared with its corresponding target vector, where the vector with the best fitness is selected to survive as the trial vector of the next generation. In case of tie, the new generated trial vector survives.

2.1.1 Initialization

The DE algorithms as is usual begins with a randomly initiated population of NP parameter vectors. Subsequent generations in DE are denoted by $G = 0, 1, ..., G_{max}$. The ith vector of the population at the current generation is denoted as:

$$\mathbf{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, ..., X_{D,i,G}]. \tag{1}$$

The initial population should cover a bounded range and it is reached by uniformly randomizing individuals within the search space constrained by the prescribed minimum and maximum bounds. Hence, each jth component of the ith vector is initialized as follows:

$$X_{j,i,0} = x_{j,min} + rand_{i,j}[0,1](x_{j,max} - x_{j_m in}) \tag{2}$$

where $rand_{i,j}[0,1]$ is an uniformly distributed random number lying between 0 and 1.

2.1.2 Mutation

The mutation can be seen as a change or perturbation with a random element. Particularly, in DE a parent vector called *target* vector is combined through a defined strategy to form the *donor* vector. In one simple form, a mutant vector $V_{i,G}$ is created from the ith target vector and is established as follows:

$$\mathbf{V}_{i,G} = \mathbf{X}_{r1,G} + F(\mathbf{X}_{r2,G} - \mathbf{X}_{r3,G}) \quad r1 \neq r2 \neq r3 \tag{3}$$

The indices $r1, r2, r3 \in [1, NP]$ are mutually exclusive integers randomly chosen from the range $[1, NP]$. It is important to take into account that the difference of any two vectors is scaled by a scalar number F and usually is defined in the interval $[0.4, 1]$, also the scale difference is added to the third one.

2.1.3 Crossover

In order to increase the diversity of the perturbed parameter vectors, a crossover operation is applied to the generated donor vector. Accordingly this, the target vector is mixed with the mutated vector to form the trial vector $\mathbf{U_{i,G}} = [u_{1,i,G}, u_{2,i,G}, ..., u_{D,i,G}]$. In the DE-context are present two kinds of crossover methods −*exponential* and *binomial*(or uniform), however in this paper is only considered the binomial crossover. In the binomial crossover strategy, the *trial* vector $\mathbf{U}_{i,G}$ is generated as follows:

$$\mathbf{U}_{j,i,G} = \begin{cases} \mathbf{V}_{j,i,G}, & \text{if}(rand_{i,j}[0,1] \leq CR \quad \textit{or} \quad j = j_{rand}) \\ \mathbf{X}_{j,i,G}, & \text{otherwise} \end{cases} \tag{4}$$

where $rand_{i,j}[0,1]$ is a uniformly distributed random number, which is generated for each jth component of the ith vector parameter. j_{rand} is a randomly chosen index, which ensures that $\mathbf{U}_{i,G}$ has at least one component from $\mathbf{V}_{i,G}$. CR is the crossover constant $\in [0,1]$, which has to be determined by the user.

2.1.4 Selection

Once generated the trial vectors, is performed a greedy selection scheme. This selection determines whether the target or the trial vector survives to the next generation, and it is described as follows:

$$\mathbf{X}_{j,i,G+1} = \begin{cases} \mathbf{U}_{i,G}, & \text{if} \quad f(\mathbf{U}_{i,G}) \leq f(\mathbf{X}_{i,G}) \\ \mathbf{X}_{i,G}, & \text{otherwise} \end{cases} \tag{5}$$

where $f(\mathbf{X})$ is the objective function to be minimized. Hence, the population either gets better or remains the same fitness status, but never deteriorates.

The mutation scheme described with the crossover proposed is referred as DE/rand/1/bin. The general convention is DE/x/y/x, where DE indicates "Differential Evolution", x denotes the base vector to be perturbed, y is the number of difference vectors considered for perturbation and z is the type of crossover to use.

2.2 Differential Evolution and Diversity techniques

DE is highly susceptible to the diversity drawbacks generally seen in EAs. This fact resides in the selection operator due that it is a greedy strategy. However through the last decade has been developing several analyses and strategies to deal with these drawbacks. One of them is proposed in 2003 by Zaharie et al. [18] where is estimated the theoretical variance and is developed a parameter adaptation based in two critical equations oriented on the idea of controlling the population diversity. In 2009 James Montgomery [21] analyses the effect of the difference vectors showing that small differences vectors applied to solutions in one cluster can produce improvements, thus movements produced by large difference vectors are wasted. After that in 2010 Montgomery shows a study where are analyzed the DE parameters, principally it shows empirically the effect of the crossover probability. In 2012 Montgomery et al. [22] proposed a strategy that prevents the movements vectors that could provoke premature convergence, specifically it dismiss the movements that are below a threshold and it decreases over the algorithm's run, however this strategy only slow down the convergence since that only is considered the distance between the *base* vectors and the *trial* vectors but the distance between the *trial* vector and the *target* vector is not considered. After that, in 2013 Antonio Boluf et al. proposed an adaptive threshold convergence mechanism [23].

A similar work is proposed by Angela et al. [15], which suggests a modification of the selection operator of the classical DE. Particularly, the selection pressure is relaxed through a probabilistic selection to maintain the population diversity and consequently to allow escape from basin of attraction of local optima, however it considers the fitness in a defined probabilistic selection, thus it depends in the cost function.

Ming Yang et al. (2013) [19] proposed a mechanism named *Auto-Enhanced Population Diversity* (AEPD) where are identified the moments when a population becomes converging or stagnating by measuring the distribution of the population in each dimension and diversifying at the dimensional level. Similar strategies has been proposed as the one by Zhao Li et al. in 2016 [24] where are varied the assembling positions of the premature individuals by mutation operation.

Our proposal is based in several ideas of the previously mentioned works, specifically the following:

- Is considered a threshold to control explicitly the convergence of the solutions.
- This threshold decreases over the algorithm's run.
- The selection operator is relaxed.

3 Differential Evolution Through the Years

In the last decade, DE has been recognized as one of the most promising EAs, likely for its efficient and simple approach to solve optimization problems. Specifically, the DE variants have been highly present in several optimization competitions, principally in the Congress on Evolutionary Computation (CEC). In fact DE occupied the top places in several optimization scenarios as are single-objective, multi-objective, constrained problems, large scale problems, dynamic problems, multi-niche landscape problems and learned based problems. In this work we are interested in the design tendency of DE algorithms in the CEC competitions problems through the last years.

In CEC 2005 competition on real parameter optimization [25], on 10-D problems classical DE secured 2nd rank and a self-adaptive DE variant called SaDE secured third rank although they performed poorly over 30-D problems. Later in CEC 2006 on constrained problems [26] DE algorithms obtained first place with ϵ constrained Rank-based Differential Evolution (ϵ RDE) and third place with SaDE.

Multi-objective optimization problems were proposed in CEC 2007 [27] competition, where DE obtained the second place with the based Generalized Differential Evolution 3 (GDE3), it is important take into account that later in CEC 2009 the first place was reached by the Multi-Objective Evolutionary Algorithm Based in Decomposition (MOEA/D) which implements the DE operators instead of its old version that use the genetic operators (Simulated Binary Crossover).

However, in the large scale global optimization (CEC 2008) [28] a Self-adaptive DE (jDEdynNP-F) reached the third place, unfortunately in later competitions (CEC 2010) DE algorithms did not reach the top rank, this could be an indicator of the weakness of DE in large scale problems [29].

In CEC 2010 competition on constrained real-parameter optimization [30] the first place was reached by the ϵ Constrained DE with gradient based mutation (ϵ Deg) and the third place by the Self-adaptive DE for solving constrained optimization (jDEsoco).

In CEC 2011 competition with real world optimization problems [31], the second and third places were reached by Hybrid DE (DE-Λ_{CR}) and Self-adaptive Multi-Operator DE (SAMODE) respectively. Later in CEC 2014 [32], the first place was reached by the Linear Population Size Reduction Success-History Based Adaptive DE with Linear Population Size Reduction (L-SHAE) in the single objective real parameter optimization scenario. In CEC 2015 with the scenario of learned based single-objective [33] DE obtained the first three places, Successful-Parent Selecting L-SHADE with Eigenvector-Based Crossover (SPS-L-SHADE-EIG), DE with success Parameter Adaptation (DesPA), Mean Variance Mapping Optimization (MVMO) and Neuro-dynamic L-SHADE (L-SHADE-ND), being placed the last two in third place. DE was also ranked in the scenario of multi-niche single objective optimization in the third place with Neighborhood based Speciation Differential Evolution (NSDE). In CEC 2016 competition in single objective optimization [33] the first place was reached with the United Multi-Operator Evolutionary Algorithm (UMOEAs-II), the second place was reached by Ensemble Sinusoidal Differential Covariance Matrix Adaptation with Euclidean Neighborhood (L-SHADE-EpSin) and in the third place Improved L-SHADE (iL-SHADE), all of them applied DE operators. In the scenario of learned based single objective optimization [34], the second and third places were reached by Cooperative Co-evolution L-SHADE with restarts (CCL-SHADE) and L-SHADE with four strategies (L-SAHDE44) respectively.

In CEC 2017 single objective optimization competition [35] the first three places were obtained by DE variants which are Effective Butterfly Optimizer with Covariance Matrix Adapted Retreat Phase (EBOwithCMAR considered as an improvement of UMOEAs-II), jSO (improvement of iL-SHADE) and L-SHADE-EpSin, being the first, second and third places respectively.

It is important take into account two dominant approaches in the described competitions: Multi-operator EAs (EBOwithCMAR) and adaptive (family of SHADE). Also it seems that in the last algorithms the criteria stop is considered to control the convergence level either explicitly or implicitly, such as Linear Population Size Reduction (LPSR), decreasing pBest mutation strategy, local search in the last stages, among others.

In the competitions of the last years SHADE's family algorithms seems to be more participatory, however based in that the ability exploration of DE is highly affected by the population size, usually the search is complemented with a Covariance Matrix Adaptation variant as is showed in the UMOEAs-II and L-SHADE-EpSin algorithms.

4 Proposal

Principally, our proposal[1] is based in the following two works. The first one that is delimited for DE algorithms, it is shown by Montgomery et al. [22] where is suggested a strategy to prevented the premature caused by the displacement of the mutation operator. The second one is a generalization of EAs, it is proposed by Carlos Segura et al. [20] and it transforms the single objective optimization problem to multiple objectives where one of them is the fitness and the other one is a diversity measurement, similarly is used a threshold which is decreased as the criteria stop is reached.

Particularly, our proposal induces a balance between exploration and exploitation that is automatically adjusted on the given stopping criterion. Thus, the stopping criterion, as well as the elapsed time or the evaluations already executed, are used as inputs to the replacement strategy. In this way, for shorter stopping criteria the method induces a faster reduction in diversity than for longer stopping criteria. To achieve such balance are considered three populations, parent vectors, offspring vectors and elite vectors, being one of the novelties of the new design.

One of the basic principles behind the development of the replacement strategy devised in this paper is that individuals that contribute too little to diversity –the contribution is measured with the Distance to Closest Neighbor (DCN) value– should not be accepted as be part of the parent vectors, instead it could replace one of the elite vectors.

In our approach, the vectors that contribute too little to the diversity are penalized. The value D_t[2] represents the minimum DCN required to avoid being penalized. Any vector whose DCN value is lower than this threshold value is penalized. The key principle resides in how to evaluate whether an vector contributes enough or not, i.e., how to set the value D_t. The value of D_t should depend on the optimization stage. Specifically, this value should be reduced as the stopping criterion is approached. In our scheme, an initial D_I value must be set. Then, a linear reduction of D_t is done. Particularly, in this work, the stopping criterion is set by function evaluations (nfes). The reduction is calculated in such way that by the 95% of maximum number of evaluations the resulting D_t value is 0, and the rest is present a similar behavior of the classical DE. Thus, if max_nfes is the maximum number of evaluations and $nfes$ the elapsed number of evaluations, D_t can be calculated as $D_t = D_I - D_I * (nfes/(0.95 * max\_nfes))$. According to Segura et al. [20] updating D_t is more appropriate through a linear reduction.

Specifically, the previously strategy is implemented in the replacement phase (algorithm 1) where is used a popular niche-strategy known as *Speciation* [36]. Initially, based in a niche-radius (D_t) and a defined distance[3]

[1] The code in C++ can be consulted in the next link `https://github.com/joelchaconcastillo/Diversity_DE_Research.git`

[2] Do not confuse the threshold distance D_t with the dimension D.

[3] For simplicity we use euclidean distance, however can be user other distance as the mahalanobis distance.

(equation 6), in an iterative process the seeds (or survivors) are identified, these are the vectors with best fitness and whose minimal DCN is not lower than the one determined by the D_t value. It is important to remark that should be considered the normalized distance in such way that each dimension is equally important and the maximum distance is the unity, and as is suggested in previous works the initial niche-radius (D_I) is the fraction of the main space diagonal.

$$distance(x_{seed}, x_j) = \frac{\sqrt{\sum_{d=1}^{D} \left(\frac{x_{seed}^d - x_j^d}{max_d - min_d} \right)^2}}{\sqrt{D}} \tag{6}$$

Therefore, the vectors that have a lowest distance to any seed than D_t are moved to the penalized set. In this way are preserved the best fitness vectors and simultaneously the diversity is maintained in some level. It is important to take into account that if the niche-radio is too high, just one seed or survivor will be selected. In this scenario the rest of parent vectors are selected from the penalized vectors. Thus, are selected the penalize vectors that have the maximum contribution to diversity considering the selected seeds vectors. Although that in the literature exist several diversity measures, we consider the DCN. According this, in an iterative process is selected as survivor the penalized vector that has the maximum DCN.

Algorithm 1 Replacement Phase

Input: D_I, $Population$, $Offspring$ and $Elite$
$Survivors = Penalized = \emptyset$.
$Current = Population \cup Offspring \cup Elite$.
Sort $Current$ according to fitness.
while $Survivors < pop\_size$ **do**
    Select the best individual $Current_{best}$ of $Current$ as a new seed.
    Find the other individuals nearest according to Eq. (6) and move to $Penalized$.
    Move the best individual $Current_{best}$ to $Survivors$.
while $Survivors < pop\_size$ **do**
    Select the individual $Penalized$ with maximum distance to closest $Survivor$.
    Move individual $Penalized$ to $Survivors$.
return $Current$

On the other hand, since that the diversity in the parent vectors should be kept, the selection operator indicated in the equation (5) is modified. Thus, instead of made a comparison between the target or parent vectors and the trial or offspring vectors, is applied a comparison between the offspring vectors with the elite vectors. Hence, the elite vectors record the best individuals obtained among the optimization process.

Algorithm 2 General scheme of DE considering diversity

1: Randomly initialize the population of NP individuals, where each one is uniformly distributed.
2: Update $D_t = D_I - D_I * (nfes/(0.95 * max\_nfes))$
3: **while** stopping criterion is nor satisfied **do**
4: **for** $i = 1$ to NP **do**
5: Mutation: Generate the donor vector according Eq. (3)
6: Crossover: Recombine the mutate vector according Eq. (4)
7: Selection: Update the parent vector according Eq. (5)
8: Replacement: Select the parent vectors according to algorithm 1

An advantage of our proposal is that it alleviates one critical weakness of the DE algorithms. These are the control parameters both crossover probability (CR) and mutation factor (F). Based in several studies showed by Montgomery et al. [37], CR is perhaps the most important. Extremes CR values leads to vastly different search behaviors. Low values of CR result in a search that is not just aligned with a small number of search space axes, but which is gradual, slow and robust. High values of CR result in searches where fewer generated solutions may be improving, but the improvements can be large. According this, we employ both high and low CR values showed in the equation 7.

$$CR = \begin{cases} Norm(0.2, 0.1), & \text{if} \quad rand[0,1] \leq 0.5 \\ Norm(0.9, 0.1), & \text{otherwise} \end{cases} \tag{7}$$

On the other hand, the mutation factor F is computed as follows. For each vector is sampled a F value with a Cauchy distribution $Cauchy(0.5, 0.5 * nfes/max\_nfes)$. In this way the shape of the distribution increases with the function evaluations and therefore are generated more extreme values at the end of execution, this aims avoid stagnation in different stages of the algorithm.

5 Experimental Study

In this section the experimental validation is carried out. Specifically is showed that controlling the diversity in a classic DE, is a way to improve further some of the results obtained by the state-of-the-art algorithms. Particularly, the benchmarks of CEC 2016 and CEC 2017 are considered, each one of them is composed of thirty different problems. The state-of-the-art is composed by the algorithms that correspond to the first places of each year. Thus, the algorithms considered from the CEC 2016 are UMOEAs-II [38] and L-SHADE-EpSin [39] that are the first and second place respectively. Also the top algorithms from CEC 2017 are EBOwithCMAR [40] and jSO [41]. It is interesting to take into account that EBOwithCMAR is an improvement of the UMOEAs-II. Also, jSO and L-SHADE-EpSin are considered from the SHADE's family.

Given that all of them are stochastic algorithms, each execution was repeated 51 times with different seeds. The stopping criterion was set to 25,000,000 functions evaluations. We performed our evaluation following the guidelines of CEC benchmark competitions. According this, if the gap between the values of the best solution found and the optimal solution was 10^{-8} or smaller the error is treated as 0. The specific parameterization of each one tested algorithm is as follows:

- **EBOwithCMAR**: For EBO maximum population size of $S_1 = 18D$, minimum population size of $S_1 = 4$, maximum population size of $S_2 = 146.8D$, minimum population size of $S_2 = 10$, historical memory size H=6. For CMAR Population size $S_3 = 4+3log(D)$, $\sigma = 0.3$, CS = 50, probability of local search $pl = 0.1$ and $cfe_{ls} = 0.4 * FE_{max}$.
- **UMOEAs-II**: For MODE, maximum population size of $S_1 = 18D$, minimum population size of $S_1 = 4$, size memory H=6. For CMA-ES Population size $S_2 = 4 + \lfloor 3log(D) \rfloor$, $\mu = \frac{PS}{2}$, $\sigma = 0.3$, CS = 50.For local search, $cfe_{ls} = 0.2 * FE_{max}$.
- **jSO**: Initial population size (N) = $25log(D)\sqrt{D}$, historical memory size H= 5, initial mutation memory $M_F = 0.5$, initial probability memory $M_{CR} = 0.8$, maximum population size = N, minimum population size = 4, initial p-best = $0.25 * N$, final p-best = 2.
- **L-SHADE-EpSin**: Initial population size (N) = $25log(D)\sqrt{D}$, historical memory size H= 5, initial mutation memory $M_F = 0.5$, initial probability memory $M_{CR} = 0.5$, initial memory frequency $\mu_F = 0.5$, maximum population size = N, minimum population size = 4, initial p-best = $0.25 * N$, final p-best = 2, generations of local search $G_{LS} = 250$.
- **Diversity-DE**: Initial niche radius $D_I = 0.3 * \sqrt{D}$, population size = 250, $F = Cauchy(0.5, nfes/max\_nfes)$.

Our experimental analyzes has been performed in base of the error between the true optimal and the optimal obtained. In order to statistically compare the results, a similar guideline than the one proposed in [42] was used. First a Shapiro-Wilk test was performed to check whatever or not the values of the results followed a Gaussian distribution. If, so, the Levene test was used to check for the homogeneity of the variances. If samples had equal variance, an ANOVA test was done; if not, a Welch test was performed. For non-Gaussian distributions, the non parametric Kruskal-Wallis test was used to test whether samples are drawn from the same distribution. An algorithm X is said to win algorithm Y when the differences between them are statistically significant, if the mean and median obtained by X are higher than the mean and median achieved by Y.

In the tables 1 and 2 are showed the summary of CEC 2016 and CEC 2017 respectively. The statistical tests indicate that the diversity DE algorithm provides significantly better results that the state-of-the-art algorithms in both benchmarks. Although that our proposal loses with the functions $\{f_6, f_7, f_{13}, f_{14}, f_{28}\}$ in CEC 2016 and $\{f_{12}, f_{16}, f_{18}\}$ for CEC 2017, it is important to take into account that our proposal provides acceptable and in some problems reach to the optimal. In fact based in a preliminary study this functions are solved at least one time with different configurations (radius niche

Table 1 Summary results - CEC 2016

Algorithm	**Always Solved**	**At least one time solved**	**Score**	**Statistical tests**		
				$\uparrow$	$\downarrow$	$\longleftrightarrow$
UMOEAsII	9	14	41.65	5	31	24
L-SHADE-Epsilon	7	13	45.84	18	14	28
Proposal	13	21	100.00	31	9	20

Table 2 Summary results - CEC 2017

Algorithm	**Always Solved**	**At least one time solved**	**Score**	**Statistical tests**		
				$\uparrow$	$\downarrow$	$\longleftrightarrow$
EBOwithCMAR	11	15	30.6792	11	23	26
JSO	8	19	41.8322	8	29	23
Proposal	21	28	100.0000	36	3	21

and populations). The column named "Always Solved" indicates the number of functions that have a zero error in the 51 runs and the column named "At least one time solved" indicates the number of functions that reach to the optimal at least with one run. Almost all functions were solved in CEC 2017 with our proposal (28 functions) and more than a half in CEC 2016, however the state-of-the-art only were able to reach the optimal values in approximately a half of the functions in both years.

Based in the guideline of the CEC, the "Score" is computed as follows. The evaluation method combines two scores defined in the equation (8). Thus the final score is composed by the sum $Score = Score_1 + Score_2$.

$$\begin{aligned} Score_1 &= \left(1 - \frac{SE - SE_{min}}{SE}\right) \times 50, \\ Score_2 &= \left(1 - \frac{SR - SR_{min}}{SR}\right) \times 50, \end{aligned} \tag{8}$$

Here, SE_{min} is the minimal sum of errors from all the algorithms, and SE is the sum of error values $SE = \sum_{i=1}^{30} error\_f_i$. Also, SR_{min} is the minimal sum of ranks from all the algorithms, namely the sum of each rank in each function for the considered algorithms $SE = \sum_{i=1}^{30} error\_f_i$. Based in the final score the results provided for our proposal are superior in both years. Moreover, in both years the SHADE's algorithms have a superior score than the multi-operator algorithms, although that the multi-operator algorithms were ranked in the first place. This is an indicator that the multi-operator algorithms could suffer more of premature convergence than SHADE's algorithms in long-term executions. Probably, this can be caused by the parameterization, which is difficult since that several parameters need to be assigned.

The error values between the best fitness values found in each run out of 51 runs and true optimal value are calculated and then best, worst, median, mean, standard deviation and success ratio of the error values are presented in each column in the tables 3 and 4. These tables show that the uni-modal functions and almost all the hybrid functions were solved. Approximately a half of the composition functions are solved with at least one run. However our

Table 3 Results for DE based diversity CEC 2016 problems

	Best	**Worst**	**Median**	**Mean**	**Std**	**Succ. Ratio**
f_1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_6	0.00E+00	3.60E-02	4.00E-03	7.39E-03	1.15E-02	3.92E-01
f_7	2.00E-02	1.02E-01	5.90E-02	5.77E-02	4.93E-02	0.00E+00
f_8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{10}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{11}	0.00E+00	6.00E-02	0.00E+00	5.88E-03	1.90E-02	9.02E-01
f_{12}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{13}	1.00E-02	8.00E-02	5.00E-02	4.67E-02	2.60E-02	0.00E+00
f_{14}	1.00E-02	5.00E-02	3.00E-02	2.82E-02	2.13E-02	0.00E+00
f_{15}	0.00E+00	4.70E-01	2.20E-01	1.99E-01	1.55E-01	1.96E-02
f_{16}	4.00E-02	1.50E-01	8.00E-02	8.47E-02	4.96E-02	0.00E+00
f_{17}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{18}	0.00E+00	2.00E-02	1.00E-02	7.65E-03	6.32E-03	3.14E-01
f_{19}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{20}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{21}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{22}	0.00E+00	3.00E-02	0.00E+00	3.73E-03	2.76E-02	7.65E-01
f_{23}	0.00E+00	1.00E+02	0.00E+00	2.55E+01	5.10E+01	7.45E-01
f_{24}	0.00E+00	6.90E-01	0.00E+00	2.61E-02	1.33E-01	9.61E-01
f_{25}	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00	0.00E+00
f_{26}	8.00E-02	1.00E+02	5.29E+01	5.20E+01	3.19E+01	0.00E+00
f_{27}	2.50E-01	9.10E-01	5.40E-01	5.60E-01	2.92E-01	0.00E+00
f_{28}	0.00E+00	3.57E+02	3.43E+02	2.76E+02	1.60E+02	1.96E-01
f_{29}	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00	0.00E+00
f_{30}	1.84E+02	1.84E+02	1.84E+02	1.84E+02	3.25E-02	0.00E+00

proposal has problems solving the multi-modal functions, this can be provoked since that our proposal does not applies an advanced strategy to deal with the incremented distribution of difference vectors. Since that the algorithm finds some niches through the optimization process, the mutation provokes high displacements, that as result some regions are not analyzed properly. To deal with the previously issue, we suggest apply a matting restriction or implement a local search, which could further a better convergence.

5.1 Sensitive analyses of the initial radius niche

In our proposal the diversity is explicitly promoted through several stages given an initial radius niche or distance factor D_I. Therefore, the robustness of this parameter is analyzed as follows. Based in the configurations of the experimental validation are executed several distance factors configurations ($D_I = \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1\}$).

Table 4 Results for DE based diversity CEC 2017 problems

	Best	Worst	Median	Mean	Std	Succ. Ratio
f_1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_7	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{10}	0.00E+00	1.20E-01	0.00E+00	1.65E-02	3.39E-02	7.45E-01
f_{11}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{12}	0.00E+00	2.20E-01	0.00E+00	6.37E-02	1.76E-01	6.67E-01
f_{13}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{14}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{15}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{16}	0.00E+00	2.10E-01	0.00E+00	2.47E-02	7.27E-02	8.82E-01
f_{17}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{18}	0.00E+00	1.00E-02	0.00E+00	1.96E-03	4.47E-03	8.04E-01
f_{19}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{20}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{21}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{23}	0.00E+00	3.00E+02	0.00E+00	3.49E+01	1.03E+02	8.82E-01
f_{24}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{25}	0.00E+00	1.00E+02	0.00E+00	3.92E+00	2.00E+01	9.61E-01
f_{26}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{27}	0.00E+00	3.87E+02	3.87E+02	2.05E+02	2.68E+02	1.96E-02
f_{28}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E+00
f_{29}	1.45E+02	2.26E+02	2.18E+02	1.99E+02	4.21E+01	0.00E+00
f_{30}	3.95E+02	3.95E+02	3.95E+02	3.95E+02	2.10E-01	0.00E+00

In the figure 1 is showed the average success ratio vs. the initial distance factor D_I. The main conclusions obtained are as follows:

- If the diversity is not promoted ($D_I = 0.0$) the performance of the algorithms is seriously implicated.
- In this scenario the ideal configuration is $D_I = 0.3$, although that the range $[0.1, 0.4]$ also provides quality solutions.
- As the diversity promoted increases the quality of the solutions are implicated.

Finally, its important stand out that the solutions are less affected by the population size, however there is still present a relation between the D_I and the population size.

6 Conclusion

From the experimental results in this paper, several conclusions can be drawn.

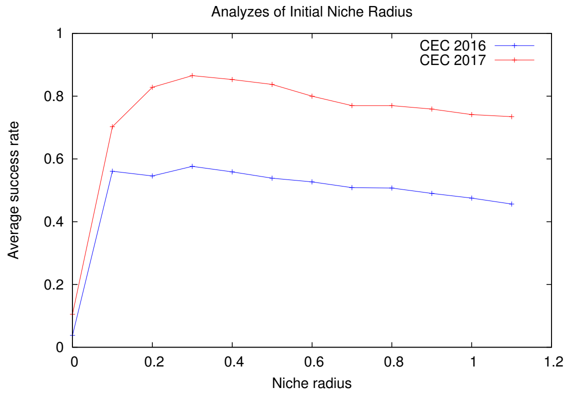

Fig. 1 Average success rate with different initial distance factors in the benchmark of CEC 2016 and CEC 2017, is considered a population size of 250 and 25000000 function evaluations.

Firstly, from experimental investigation on the working mechanism, it can be seen that our proposal is able to relieve the premature convergence to several optimization levels. Secondly, our proposal is able to enhance the performance of DE algorithms, in particular when the search space is large. Third, it is also less sensitive to the parameter of population size, so our proposal can also be competitive even if the population size is small. Fourth, it seems that our proposal has some drawback in relation with the proportion of difference vectors.

For future work of this paper, two interesting issues should be addressed for our proposal. The first one is that explored areas in the search space should be avoided to save computing resources. Development an adaptive strategy for the distance factor should involve a more stable algorithm. Explore the possibility of implement a local search scheme with two goals, save function evaluations and tackle the current multi-modal problem. Applying our proposal to real-world problems should be an interesting topic. Based in several analyzes the mutation factor could be selected inside the distance factor, then develop a strategy where this parameter is no required. Generate a theoretical model to select the adequately population size given a initial distance factor.

References

1. N. Noman, H. Iba, Differential evolution for economic load dispatch problems, Electric Power Systems Research **78**(8), 1322 (2008)
2. U.K. Chakraborty, *Advances in differential evolution*, vol. 143 (Springer, 2008)
3. M. Srinivas, L.M. Patnaik, Genetic algorithms: A survey, computer **27**(6), 17 (1994)
4. H.P. Schwefel, Numerische optimierung von computer-modellen (phd thesis), Reprinted by Birkhuser (1977)
5. H. John. Holland, adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence (1992)
6. J.R. Koza, *Genetic Programming II, Automatic Discovery of Reusable Subprograms* (MIT Press, Cambridge, MA, 1992)

7. D.B. Fogel, L.J. Fogel, J.W. Atmar, in *Signals, systems and computers, 1991. 1991 Conference record of the twenty-fifth asilomar conference on* (IEEE, 1991), pp. 540–545
8. R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization **11**(4), 341 (1997)
9. S. Das, P.N. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE transactions on evolutionary computation **15**(1), 4 (2011)
10. J.A. Nelder, R. Mead, A simplex method for function minimization, The computer journal **7**(4), 308 (1965)
11. W. Price, Global optimization by controlled random search, Journal of Optimization Theory and Applications **40**(3), 333 (1983)
12. R. Gämperle, S.D. Müller, P. Koumoutsakos, A parameter study for differential evolution, Advances in intelligent systems, fuzzy systems, evolutionary computation **10**(10), 293 (2002)
13. J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE transactions on evolutionary computation **10**(6), 646 (2006)
14. J. Zhang, A.C. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Transactions on evolutionary computation **13**(5), 945 (2009)
15. Â.A. Sá, A.O. Andrade, A.B. Soares, S.J. Nasuto, in *AISB 2008 Convention Communication, Interaction and Social Intelligence*, vol. 1 (2008), vol. 1, p. 57
16. J. Lampinen, I. Zelinka, et al., in *Proceedings of MENDEL* (2000), pp. 76–83
17. Q. Chen, B. Liu, Q. Zhang, J. Liang, P. Suganthan, B. Qu, Problem definition and evaluation criteria for cec 2015 special session and competition on bound constrained single-objective computationally expensive numerical optimization, Computational Intelligence Laboratory, Zhengzhou University, China and Nanyang Technological University, Singapore, Technical report (2014)
18. D. Zaharie, in *Proc. of MENDEL*, vol. 9 (2003), vol. 9, pp. 41–46
19. M. Yang, C. Li, Z. Cai, J. Guan, Differential evolution with auto-enhanced population diversity, IEEE transactions on cybernetics **45**(2), 302 (2015)
20. C. Segura, C.A.C. Coello, E. Segredo, A.H. Aguirre, A novel diversity-based replacement strategy for evolutionary algorithms, IEEE transactions on cybernetics **46**(12), 3233 (2016)
21. J. Montgomery, in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on* (IEEE, 2009), pp. 2833–2840
22. J. Montgomery, S. Chen, in *Evolutionary Computation (CEC), 2012 IEEE Congress on* (IEEE, 2012), pp. 1–8
23. A. Bolufé-Röhler, S. Estévez-Velarde, A. Piad-Morffis, S. Chen, J. Montgomery, in *Evolutionary Computation (CEC), 2013 IEEE Congress on* (IEEE, 2013), pp. 40–47
24. L. Zhao, C.j. Sun, X.c. Huang, B.x. Zhou, in *Control Conference (CCC), 2016 35th Chinese* (IEEE, 2016), pp. 2784–2787
25. P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization, KanGAL report **2005005**, 2005 (2005)
26. J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.C. Coello, K. Deb, Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization, Journal of Applied Mechanics **41**(8) (2006)
27. P. Suganthan, in *IEEE conference on evolutionary computation special session, competition on performance assessment of multi-objective optimization algorithms. http://www3. ntu. edu. sg/home/EPNSugan/index_files* (2007)
28. K. Tang, X. Yáo, P.N. Suganthan, C. MacNish, Y.P. Chen, C.M. Chen, Z. Yang, Benchmark functions for the cec2008 special session and competition on large scale global optimization, Nature Inspired Computation and Applications Laboratory, USTC, China **24** (2007)
29. C. Segura, C.A.C. Coello, A.G. Hernández-Díaz, Improving the vector generation strategy of differential evolution for large-scale optimization, Information Sciences **323**, 106 (2015)

30. R. Mallipeddi, P.N. Suganthan, Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization, Nanyang Technological University, Singapore **24** (2010)
31. 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)
32. J. Liang, B. Qu, P. Suganthan, Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore (2013)
33. J. Liang, B. Qu, P. 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 (2014)
34. B. Liu, Q. Chen, Q. Zhang, J. Liang, P. Suganthan, B. Qu, in *Technical Report* (2013)
35. G. Wu, R. Mallipeddi, P. Suganthan, Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization, National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report (2016)
36. Q. Yang, W.N. Chen, Y. Li, C.P. Chen, X.M. Xu, J. Zhang, Multimodal estimation of distribution algorithms, IEEE transactions on cybernetics **47**(3), 636 (2017)
37. J. Montgomery, S. Chen, in *Evolutionary Computation (CEC), 2010 IEEE Congress on* (IEEE, 2010), pp. 1–8
38. S. Elsayed, N. Hamza, R. Sarker, in *Evolutionary Computation (CEC), 2016 IEEE Congress on* (IEEE, 2016), pp. 2966–2973
39. N.H. Awad, M.Z. Ali, P.N. Suganthan, R.G. Reynolds, in *Evolutionary Computation (CEC), 2016 IEEE Congress on* (IEEE, 2016), pp. 2958–2965
40. A. Kumar, R.K. Misra, D. Singh, in *Evolutionary Computation (CEC), 2017 IEEE Congress on* (IEEE, 2017), pp. 1835–1842
41. J. Brest, M.S. Maučec, B. Bošković, in *Evolutionary Computation (CEC), 2017 IEEE Congress on* (IEEE, 2017), pp. 1311–1318
42. J.J. Durillo, A.J. Nebro, C.A.C. Coello, J. Garcia-Nieto, F. Luna, E. Alba, A Study of Multiobjective Metaheuristics When Solving Parameter Scalable Problems, IEEE Transactions on Evolutionary Computation **14**(4), 618 (2010)