Explicit Control of Diversity in Differential Evolution

Nayeli Angel, Carlos Segura, Oscar Dalmau Cedeño
Center for Research in Mathematics, Jalisco S/N, Col. Valenciana CP: 36023 Guanajuato, Gto, México
nayeli.angel@cimat.mx, carlos.segura@cimat.mx, dalmau@cimat.mx

***Abstract*—One of the issues that might affect the performance of Differential Evolution (DE) is premature convergence. In such cases, and especially in long-term executions, due to the nature of the reproduction phase of DE, computational resources might not be employed efficiently. In DE this issue is usually tackled by altering the reproduction phase or by attaching an external archive. However, in other fields such as in combinatorial optimization, methods that amend the replacement phase have been very successful. In this paper, two variants of DE are extended by including a replacement phase that explicitly relates the amount of diversity maintained in the population with the number of generations evolved and with the stopping criterion. One of the variants is the classic DE/rand/1/bin whereas the other one is Success-History based Adaptive DE (SHADE), a state-of-the-art approach. In the case of the classic variant, the principles employed in combinatorial optimization could be used straightforwardly. However, when integrating it with state-of-the-art techniques that already incorporate mechanisms to preserve diversity, additional modifications were required. Experimental validation has been performed with the benchmarks provided for the 2013 IEEE Congress on Evolutionary Computation competition on real parameter optimization. The results of the best-ranked DE in such a competition could be improved further. Additionally, some problems that had not been solved to optimality ever by any proposal, could be successfully solved.**

***Index Terms*—Differential evolution, Diversity preservation, Survivor selection, Replacement strategy**

I. Introduction

The popularity of stochastic approximate optimization algorithms has increased considerably in the last decades [16]. Among them, Evolutionary Algorithms (EAs) are one of the most widely applied techniques [29]. Since the inception of EAs, several different variants have been proposed [3]. In the case of continuous optimization, Differential Evolution (DE) was proposed in [24] by Storn and Price. DE is a simple, yet effective continuous numerical population-based optimizer that has been extensively used in the last years [4]. One of the main distinguishing features of DE when compared to other approximate algorithms is that it manipulates the members of the population by taking into account differences among vectors present in the population. Thus, maintaining a diverse population is specially important.

Several different variants of DE have been devised. Recent DE variants have yielded remarkable results in several optimization competitions [5], such as in the *2005 IEEE Congress on Evolutionary Computation* competition on real parameter optimization [19], or in the special issue on *Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems*, recently organized for the *Soft Computing Journal* [11]. However, in the more recent *2013 IEEE Congress on Evolutionary Computation* competition on real parameter optimization, the best variant of DE [25] obtained the fourth position, meaning that state-of-the-art DE techniques require additional modifications to properly deal with some of the benchmark functions incorporated in this set of tests. The classic DE is the *DE/rand/1/bin* [18] and together with the Success-History based Adaptive DE SHADE [25] — which is a state-of-the-art approach — have been the variants taken into account in this paper.

Since the inception of EAs, premature convergence has been recognized as one of their recurrent drawbacks [27]. Premature convergence arises when all the population members are located in a reduced part of the search space and this region differs from the optimal one. In such cases, the probability of moving to other regions of the search space is dramatically decreased, so even if large executions are performed, no promising results might be obtained. For instance, it is obvious that the exploratory capabilities of the DE mutation operator diminishes in such a case. Given the importance of this drawback, many schemes have been devised to prevent premature convergence in EAs [17]. Note that in the case of DE, the issue of premature convergence might appear. In fact, the original authors noted that the survivor selection strategy applied in DE is a greedy scheme based on selecting the best among a parent and an offspring in each step [18], resulting in a fast convergence that might not be suitable in some contexts. In the case of DE, authors usually rely on modifying the reproduction phase with the aim of alleviating these drawbacks [32], whereas other kinds of schemes have not gained popularity.

Note also that in the last years, several advances have been attained in the field of combinatorial optimization by developing population-based metaheuristics with an explicit control of diversity in the replacement phase [23]. These kinds of methods have obtained remarkable results. For instance, they have the current record in the generation of frequency plans for several instances of the frequency assignment problem [22]. The novelty of these schemes relies on providing a dynamic balance between exploration and exploitation by incorporating the stopping criterion as an input to the replacement strategy. In this way, the first stages of the optimization procedure induce a larger degree of exploration, whereas, in the last stages, exploitation is promoted.

The main objective of this paper is to prove that premature

978-1-5090-6017-7/18/$31.00 ©2018 IEEE

convergence appears in DE when facing some benchmark functions and that by applying some of the principles that have been successfully used in combinatorial optimization, this inconvenient might be alleviated. In order to prove this hypothesis, the classic variant of DE and SHADE are extended by incorporating a mechanism to explicitly control the diversity, thus creating the Diversity-based DE (DDE) and the Diversity-based SHADE (DSHADE) schemes. DDE incorporates a mechanism to control the diversity that is mainly the same that the one used in combinatorial optimization. However, in order to properly extend SHADE some additional modifications were required, showing that properly extending state-of-the-art techniques is usually a more complex process. Experimental validation with the benchmarks proposed in the 2013 IEEE Congress on Evolutionary Computation competition on real parameter optimization shows that results attained for both proposals are much superior than their original variants, thus improving further the results attained by the variant of DE that was best ranked in such a competition. Moreover, there are test cases that could be solved to optimality by our proposals in some executions and that to our knowledge had not ever been solved by any proposal. The rest of the paper is organized as follows. Section II offers a review of related works including papers on DE and on control of diversity. Section III describes the proposal to control the diversity and its incorporation in DE. Section IV shows the experimental validation of our proposal. Finally, conclusions and some lines of future work are given in Section V.

II. Literature review

This section is devoted to review some of the most relevant related literature. First, some of the most recent advances on the field of DE related to our proposal are summarized. Then, some of the most popular techniques that have been used to deal with the premature convergence drawback are reviewed.

A. Differential Evolution

DE has been mainly applied to continuous optimization. In this paper, single-objective continuous box-constrained problems are taken into account. These problems are defined in the following way. Given a function $f(X) : \Omega \subset \mathbb{R}^M \rightarrow \mathbb{R}$, the task is to find a vector $X = (x_1, x_2, \ldots, x_M)$ that minimizes the function f and fulfills a set of box constraints. The box constraints restricts the feasible search space (Ω) by establishing the maximum and minimum values of each x_i, i.e., $\Omega = \prod_{j=1}^{M}[l_j, u_j]$. In the previous definition, X is the vector of variables, f is the objective function, M is the dimensionality of the problem, and l_j and u_j are the lower and upper bounds of the values that can take the variables.

DE has shown remarkable performance in several test cases. Usually, main performance advantages of DE over other EAs are attributed to floating-point encoding, good combination of adaptive evolutionary operators and use of elitism [31]. However, it is also known that for some test cases several issues might affect its performance [5]. One of the most well-known issue somewhat related to diversity is stagnation. DE borrows the idea from the Nelder & Mead method [15] of employing information from the vector population to alter the variation scheme. Thus, the strength of the mutation depends on the contents of the population. The main associated problem is that DE, in its classic variant, has the potential to generate only a limited number of different trial solutions within one generation. In [9], it was shown that this can lead to a situation where DE may stop proceeding towards a global optimum even though the population has not even converged to a local optimum. This situation is called stagnation. Although stagnation might appear even with diverse populations, authors noted that actions that are taken to preserve diversity usually have positive effects on avoiding stagnation. An additional weakness of DE was reported in [10]. It was shown that, in general, DE is less expansive than other metaheuristics. Thus, DE might be deceived into converging on the wrong region, and once there, it could be impossible for this approach to escape.

The aforementioned weaknesses are related to the high selection pressure introduced by DE's survivor selection scheme and to its premature loss of diversity [20]. Thus, this problem has been treated in several ways. The most successful approaches are based on altering the reproduction phase. First, the effects of its different parameters (population size, scale factor, etc.) have been deeply studied [31], [30] including several issues related to the loss of diversity [13]. These studies have lead to several recommendations depending on the type of optimization function. Additionally, some authors have devised alternative reproduction operators which are used as ensembles of operators in adaptive ways [2]. Adapting the parameter values to provide additional exploration capabilities has also been a quite common approach [32]. Finally, incorporating an external archive with the aim of increasing diversity and maintaining promising search directions has also provided important benefits [32].

A quite different alternative is to alter the replacement phase. Since its inception, the hybridization of DE with annealing procedures was studied to reduce the selection pressure [18]. Some more recent schemes include the introduction of stochasticity into the selection process [20] or the use of generational replacement [1]. However, these last schemes have not been very successful in terms of the results obtained in competitions.

Regarding the *2013 IEEE Congress on Evolutionary Computation* competition on real parameter optimization, SHADE [25] was the best-ranked variant of DE. Two of the most important features of SHADE is that it incorporates adaption of the crossover rate and mutation scale factor, and it uses an external archive.

B. Premature Convergence

Since premature convergence is one of the most typical issues that appears in the application of EAs, a large number of techniques have been devised to deal with this drawback [17]. The proposals are classified in [27] taking into account the component of the EA that they modify. Specifically, the following categories are identified: *selection-based*, *population-based*, *crossover/mutation-based*, *fitness-based*, and *replacement-based*.

Since this paper deals with a replacement-based technique, some of the most popular approaches belonging to this group are described. The basic principle of these schemes is to induce larger levels of exploration in successive generations by diversifying the survivor of the population [21]. For instance, in *crowding* the basic principle is that offspring should replace similar individuals from the previous generation, which can be implemented in several ways [12]. In other methods, diversity maximization is considered as an objective that is combined with the original objective to calculate the fitness of each individual [28]. Finally, some alternatives consider both the original objective and the diversity contribution of each individual as functions to optimize and apply multi-objective concepts [14].

In the combinatorial optimization area this last principle was used to develop a replacement phase, the MULTI_DYN survivor selection scheme [21], that has provided important benefits. The basic principle of MULTI_DYN is that individuals that contribute too little to diversity should not survive regardless of their original objective value. Specifically, this is performed by incorporating a dynamic penalty approach. This penalty method avoids the simultaneous survival of too close individuals. In order to determine if two individuals are too close to each other, a distance metric is defined. Initially, larger distances are enforced to promote exploration but gradually, the boundary distance used to penalize is reduced, meaning that as the optimization procedure progresses, more exploitation is induced. Given the proper performance of MULTI_DYN in combinatorial optimization, in this paper this method has been extended and integrated in different variants of DE. The details of the MULTI_DYN strategy and extensions developed in this research are given in the next section.

III. PROPOSALS

In the present work, we have extended the classic DE/rand/1/bin strategy [18] as well as SHADE [25]. They share several features. First, they start with a population with N candidate solutions that are initialized by taking into account a uniform random distribution. Then, a set of generations are evolved until a stopping criterion is reached. This stopping criterion is usually set to a maximum number of generations. In each generation, a mutant vector (V_i) is built and recombined with the target vector (X_i) to generate a new trial vector or offspring (U_i). Specifically, the binomial recombination is used both in DE/rand/1/bin and in SHADE. This operator requires setting the crossover rate (CR) parameter, which controls the fraction of values that are copied to the trial vector from V_i. The remaining variables are taken from the target vector. Finally, the replacement phase takes place. Specifically, the target vector — which is in the current population — is replaced by the trial vector (U_i) if the trial vector has an equal or lower (for the minimization case) objective function than the target vector X_i (Equation 1).

$$X_i = \begin{cases} U_i & \text{if } f(U_i) \leq f(X_i) \\ X_i & \text{otherwise} \end{cases} \tag{1}$$

Note that the difference between SHADE and DE/rand/1/bin appears in the process to create the mutant vector.

Regarding the replacement operator, which is the most relevant phase to our discussion, DE operates in a greedy way, i.e. it just checks the objective function to perform the selection. This might lead to a fast convergence which might be unsuitable for long-term executions. However, it is also true that this scheme is not so greedy as other proposals of the evolutionary computation field. For instance, the replace-worst strategy [7] joins parents and offspring in a single set and selects the fittest ones which results in a much faster convergence. Moreover, the combination of the reproduction operator and replacement phase applied in DE has some additional advantages related to diversity. Each trial vector takes some variables directly from the target vector. However, since only one individual between the target vector and the trial vector survives, this avoids the repetition of such a value in the next population, meaning that convergence is delayed.

Our proposals are based on altering the replacement phase of DE. In the following section, the specific replacement phase to better control the convergence and the integration with both variants of DE are described.

A. Replacement phase with explicit control of diversity

MULTI_DYN [21] is a replacement strategy that has been applied successfully in combinatorial optimization problems. MULTI_DYN has two principles. First, it considers the maximization of diversity contribution of an individual as an objective. Thus, each individual has two values associated, the one corresponding to the fitness function and the contribution to diversity. Second, it adjusts the selection to induce a larger degree of exploration in the first stages and to balance towards exploitation as the optimization process evolves. In order to adjust this balance, the stopping criterion and the amount of period of optimization that has already elapsed are taken into account. In this paper, since the stopping criterion is set to a maximum number of generations, the stopping criterion is referred to as Gen_{End}, whereas the elapsed generations is referred to as $Gen_{Elapsed}$.

The basic principle that motivates the replacement strategy used in our proposal (Algorithm 1) is that individuals that contribute too little to diversity should not be accepted regardless of their original objective value. The algorithm operates as follows. First, the current population and offspring, i.e., trial vectors are joined in a single population. Then, the individual with best performance in the optimization function is selected to survive. Afterwards, $N - 1$ individuals must be selected to survive. In each iteration an additional individual is selected. First, the contribution to the diversity of each of the pending individuals is calculated. The contribution is measured by the distance to the closest neighbor (DCN) in the set of already selected individuals. This is performed with the Euclidean distance, i.e. for each pending individual the Euclidean distance to each already selected individual is calculated and the minimum value is considered as its contribution to diversity. Then, a boundary value (D) is calculated. The process to calculate D is detailed later. This boundary value is used to distinguish

Algorithm 1 Modified MULTI_DYNAMIC Survivor Selection

```
1:  CurrentMembers = Population ∪ Offspring
2:  Best = Individual with best $f(X)$ in CurrentMembers
3:  NewPop = { Best }
4:  CurrentMembers = CurrentMembers - { Best }
5:  **while** $|NewPop| < N$ **do**
6:      Calculate DCN of CurrentMembers, considering as reference NewPop
7:      $D = \max(0, D_I - \frac{D_I}{\alpha Gen_{End}} Gen_{Elapsed})$ with $\alpha \in (0, 1)$
8:      Penalized = {X ∈ CurrentMembers | X.DCN < D}
9:      CurrentMembers = CurrentMembers - {Penalized}
10:     ND = Non-dominated individuals of CurrentMembers (without repetitions)
11:     **if** $|ND| == 0$ **then**
12:         Selected = Individual with the largest DCN in Penalized
13:     **else**
14:         Selected = Select the individual (BEST, RAND_BIAS) from ND
15:     **end if**
16:     NewPop = NewPop ∪ Selected
17:     CurrentMembers = CurrentMembers - {Selected}
18: **end while**
19: Population = NewPop
```

between individuals that contribute too little to be selected and those individuals whose contribution is large enough. This is illustrated in Figure 1. All the individuals (points in the figure) that are at the left of the dashed line are transferred to a set of penalized individuals. If all the pending individuals are penalized, the method selects that individual with the largest DCN value. Otherwise, considering the maximization of DCN and minimization of the original function to optimize, the non-dominated set of the current non-penalized individuals is calculated. Non-dominated individuals are shown in blue color in Figure 1. Then, two different alternatives are considered to select among the non-dominated individuals. In the *BEST* strategy, the individual with best performance in the function to optimize is selected. This individual is tagged with a B in Figure 1. In the *RAND_BIAS* strategy, a percentage of the best elements — taking into account the original function — is identified, and then a randomly selected individual of this set is selected to survive. Note that the set of elements taken into account for the selection are those located from left to right in the non-dominated set, i.e. starting from the penalty boundary line. Finally, the selected individual is transferred to the set of survivors, so in the iteration to select the next individual, the contributions to diversity are updated conveniently.

In the previous description the way to calculate D was omitted. The original version of MULTI_DYN starts with $D = D_I$ in the first generation and D is reduced linearly so that at the final generation (Gen_{End}) of the optimization, D is set to 0. Initial experimentation showed that with DE it is preferable to reach the zero value prior to the last generation. In this way, more intensification is promoted in the last generations. Since the linear decrease of D is maintained, the D update mechanisms is expressed as shown in line 7 of Algorithm 1.

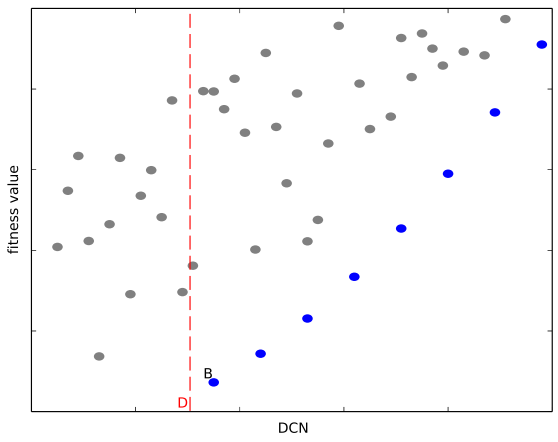

Fig. 1. Effect of the penalty approach with boundary distance D

In this formula, the α value is used to denote the instant where D reaches the 0 value, i.e. if α is set to 0.90, it means that D reaches 0 after evolving the 90% of the generations.

Finally, another important issue is how to set the initial boundary distance (D_I). In the initial generations, the aim is to distribute each one of the N individuals throughout the hypervolume of the search space (H_E). Since each selected individual provokes a penalty in base of an Euclidean distance, we can consider that there is a hypersphere centered in each individual. Thus, we calculate D_I so that the sum of the hypervolume of the N hyperspheres ($H_H(D_I)$) is equal to the hypervolume of the search space. Thus, $H_H(D_I) \cdot N$ must be equal to H_E. Since $H_E = \prod_{j=1}^{M}(u_j - l_j)$, $H_H(D_I) = \frac{\pi^{M/2} D_I^M}{\Gamma(M/2+1)}$ and $\Gamma(n) = (n-1)!$, the initial D_I value is given in (2).

$$D_I = \sqrt[M]{\frac{H_E \cdot \Gamma(M/2+1)}{N * \pi^{M/2}}} \qquad (2)$$

Finally, we would like to note that, in the original MULTI_DYN scheme, the selection among non-dominated individuals is completely random. The reason is that in the cases where it was applied, very long executions were performed, so a slow convergence was required. In our case, no so long executions are performed, so inducing a larger selection pressure is preferable.

Due to the changes performed on the original survivor selection algorithm, the algorithmic complexity of the approach per each generation is $O(N^2 * (M + \log N))$. Note that the added cost is given by the incorporation of the calculation of the distance between individuals and the calculation of the non-dominated individuals.

B. Integration with Differential Evolution

Our first proposal (Algorithm 2) is called DDE and it integrates the classic DE/rand/1/bin strategy with the MULTI_DYN replacement. The only modification that is required is to change the standard replacement phase used in DE by the MULTI_DYN strategy. In fact, it is clear that other variants of

Algorithm 2 DDE: Classic Differential Evolution (DE/rand/1/bin) integrated with MULTI_DYN

```
Population initialization $P = (X_1, X_2, \ldots, X_N)$
evaluate $f(X_i), i = 1, 2, \ldots, N$
while termination criterion not met do
    // build the offspring set Y
    for i=1 to N do
        generate a trial vector $Y_i$
        evaluate $f(Y_i)$
    end for
    P = MULTI_DYN replacement scheme ($P \cup Y$)
end while
```

DE that rely on the standard replacement can also be integrated with our proposal in a similar way. However, while developing our second proposal (DSHADE), which is based on integrating MULTI_DYN with SHADE, some drawbacks arose.

SHADE is an adaptive DE that uses a historical memory in order to adapt the control parameters F and CR. In addition, in the attempt to maintain diversity in the set of solutions it uses an external archive that is used in the generation of trial vectors. Specifically, this bounded external archive is filled with those parent vectors which are replaced by a trial vector. Thus, all the modifications are related to the reproduction phase, so MULTI_DYN was initially incorporated straightforwardly. However, initial validation showed that no proper performance was obtained. Due to the internal operation of the crossover operator, some variables are inherited intact from the previous population. The standard replacement avoids the simultaneous selection of a target vector and its corresponding trial vector, so no duplicate values are generated in this way. However, in our strategy, a trial vector and its corresponding target vector might simultaneously survive, so duplicate values might appear. This can accelerate convergence at least in some of the variables. In the cases where CR is high, only few values are inherited intact, so this problem is not so hard. However, since SHADE uses adaptive values, in some cases many values are generated intact which impacts the performance. In order to avoid this drawback the calculation of the DCN value was modified. In the cases where for a non-selected individual, there are more than three solutions already selected whose distance in a given variable is less than 10^{-1}, the value of its DCN is set to 0. Note that this issue did not impact our basic variant of DE with a fixed CR because a large CR was used.

Additionally, note that SHADE maintains diversity through the incorporation of an external archive. We noted that DSHADE presented a too slow convergence in the initial generations and no proper improvement was obtained. Thus, an additional modification was to start with a more reduced D_I value. Particularly, the D_I value was calculated as in (2) but multiplied by 0.2.

IV. Experimental validation

In this section, the performance of the proposed mechanisms is analyzed by applying them to the benchmark proposed in the IEEE Congress on Evolutionary Computation 2013 Special Session on Real Parameter Optimization. This benchmark set comprises 28 test functions, which are classified into three groups, functions 1 to 5 are unimodal, functions 6 to 20 are basic multimodal, and functions 21 to 28 are composed functions. The mechanism proposed was tested in 3 different dimensions, i.e. the number of variables M is 10, 30 and 50 for each benchmark problem. For each test problem and each tested mechanism, 30 runs have been executed and the error is reported, i.e. the value $f(Best) - f^*(X)$, where $f^*(X)$ is the global minimum of a function and $Best$ is the best individual evaluated in the search process. Therefore, a Results Set (RS) is built by the final optimum values of the set of 30 runs for each mechanism configuration tested. As a result, the RS is used to analyze the quality of a mechanism. In addition to reporting some statistics about the error attained, different approaches are also pairwise compared by applying some statistical tests. A similar guideline as the one applied in [6] was used. Specifically, the following tests were applied, assuming a significance level of 5%. First, over the RS a *Shapiro-Wilk test* was performed to check whether or not the values of the results followed a Gaussian distribution. If so, the *Levene test* was used over the RS 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 RS with non-Gaussian distributions, the non-parametric *Kruskal-Wallis* test was used to test whether samples are drawn from the same distribution. In this work, the sentence "algorithm A is superior to algorithm B” means that differences between their RS are statistically significant and that the mean and median of the error obtained by A are lower than the mean and median achieved by B.

We conduct two kinds of experiments, in that way we have divided this section into two main parts. Firstly, the robustness of the more basic DDE is tested and compared against DE/rand/1/bin. Secondly, both DDE and DSHADE are numerically analyzed against the state-of-the-art in DE.

A. Robustness of schemes with explicit control of diversity

In combinatorial optimization, one of the advantages that has reported those schemes that use the MULTI_DYN replacement is that more robust results are obtained, i.e. with several parameters, instances with quite different features can be treated successfully [21]. The aim of this set of experiments is to validate the stability of DDE, especially in comparison to the same scheme without diversity control, i.e., the DE/rand/1/bin. The two variants previously described of MULTI_DYN are considered, that is, both the *RAND_BIAS* and *BEST* selection are analyzed. The results shown in the following sections correspond to the analysis on dimension 10 of each benchmark problem.

The parameter settings used to run the three schemes are as follows. First, since the main advantages of using diversity control appear usually in the long-term, the number of generations was set to 30,000. Both the population size and CR used quite standard values. The population size was set to 50 [26], [8], and CR was set to 0.9 [18], [8]. Finally, α was set to 0.9, meaning that the penalty for individuals is omitted from generation 27,000 to 30,000, whereas for *RAND_BIAS*,

TABLE I
STABILITY OF THE MECHANISMS FOR DIFFERENT VALUES OF MUTATION SCALE FACTOR (F)

F	BEST	RAND_BIAS	Classic
0.1	7	10	0
0.2	13	10	1
0.3	12	11	1
0.4	9	10	3
0.5	10	8	3
0.6	**13**	**11**	5
0.7	**12**	**12**	7
0.8	**12**	9	**8**
0.9	**12**	8	**8**
1.0	**13**	6	3
1.1	7	1	7
1.2	4	1	6
1.3	2	1	5
1.4	1	1	3
1.5	1	1	3
1.6	1	1	1
1.7	1	1	1
1.8	1	1	1
1.9	1	1	1
2.0	1	1	1

the individual is chosen among the 40% of best individuals. These values were used to build the different mechanism configurations and experiments in this paper.

The stability of the mechanism was analyzed by varying the mutation scale factor (F). Specifically, the range [0.1, 2.0] is used with an increment of 0.1 to establish the set of tested values. First, a set of different mechanism configurations is built — i.e. a combination of method and value of F — Then, for each problem, the configuration that obtains the RS with minimum mean error is identified, called Best Configuration (BC). Finally, each configuration and BC are statistically compared with the statistical mechanism described above. Results of the experiments are shown in Table I. Each cell shows for a configuration, the amount of problems where it was the best scheme or whose differences were not statistically significant when compared against the BC. It is noticeable that DDE with the BEST scheme is stable for a large range of values.

The mechanism that implements the RAND_BIAS scheme reports the largest number of resolved functions with 11 and 12 with $F \in [0.6, 0.7]$ and $F = 0.3$, which confirms that being greedy in the selection is preferable. Finally, the classic mechanism just reaches the best results in 8 functions with $F \in [0.8, 0.9]$ and results are more sensitive to the value of F. This confirms that one of the advantages of incorporating an explicit control of diversity is the increase of robustness. Moreover, not only the robustness is increased, but also solutions with a largest quality are obtained.

Taking into account the previous analysis, in the following analyses the DDE with the BEST selection is considered.

B. Analysis of diversity

In order to better understand the behavior of the proposal, the evolution of the diversity of different configurations with several functions was analyzed. Specifically, the diversity was measured by considering the mean Euclidean distance between all the members of the population. The evolution of the diversity is shown in Fig. 2 for the function F9 and in Fig. 3 for F18. In both cases, the DE and DDE with BEST selection are considered by setting the F values to 0.3 and 0.8. The value 0.8 was selected because both schemes reported their maximum values with such a value in the previously reported analysis. The value 0.3 is a scale factor where DDE obtained promising results but the classic DE failed. The functions were selected because they show two different behaviors that appeared in most of the functions. In the function F18, regarding the DDE scheme, a stable and almost linear descent is attained. Thus, the MULTI_DYN scheme achieve its purpose. In the case of F9, we can appreciate that a too large diversity is maintained for a large period, and then, a faster loss of diversity appears after $20,000$ generations. While in the cases with this behavior, competitive results were obtained, this analysis shows that probably introducing some adaptive mechanisms to measure the diversity and alter the behavior of the replacement phase might bring additional benefits. In the case of the basic DE it is clear that diversity is lost very fast, meaning that part of the resources are not used adequately.

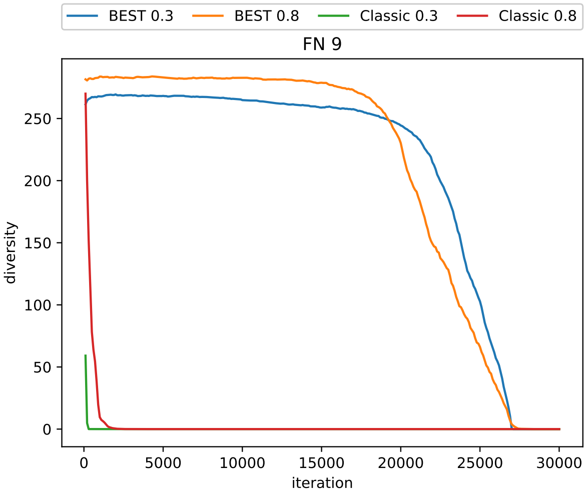

Fig. 2. Evolution of diversity for function F9

C. Comparison against state-of-art algorithms

Finally, DDE and DSHADE with BEST selection are compared against SHADE, which was the DE variant that obtained the best rank in the IEEE Congress on Evolutionary Computation 2013 Special Session on Real Parameter Optimization. Table II presents the results in terms of the error values obtained by DDE, SHADE and DSHADE. The mean, best and standard deviation of the attained errors is shown. The parameterization of SHADE and DSHADE was the standard one, i.e. the one used in the competition and the one that is considered in the source code of the authors when no special parameterization is specified. This means that both the population size and archive size were set to 100. In the case of the DDE mechanism, the mutation scale factor (F) is set to 0.8. The number of function evaluations was set to 1,500,000 in every case.

TABLE II
ERROR ATTAINED BY DDE, SHADE AND DSHADE FOR M = 10

	DDE BEST			**SHADE**			**D-SHADE**		
FN	**best**	**mean**	**std**	**best**	**mean**	**std**	**best**	**mean**	**std**
1	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
2	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
3	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	0.0000E+00	2.105E-01	1.1336E+00	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
4	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
5	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
6	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	0.0000E+00	7.8499E+00	3.9250E+00	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
7	**4.4930E-06**	**7.6569E-05**	**6.6657E-05**	**0.0000E+00**	**1.3915E-03**	**4.6769E-03**	**0.0000E+00**	**3.6053E-03**	**9.8071E-03**
8	2.0125E+01	2.0217E+01	4.5742E-02	2.0000E+01	2.0013E+01	3.0586E-02	**2.0000E+01**	**2.0006E+01**	**1.5487E-02**
9	0.0000E+00	6.4724E-01	4.4019E-01	1.5253E+00	2.9319E+00	6.5185E-01	**0.0000E+00**	**4.5703E-01**	**5.3417E-01**
10	1.9719E-02	3.5149E-01	1.6083E-01	**0.0000E+00**	**8.2343E-04**	**2.4918E-03**	0.0000E+00	6.6780E-02	4.5826E-02
11	0.0000E+00	1.4738E-01	3.4172E-01	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
12	1.9899E+00	1.2183E+01	5.8007E+00	1.1107E-04	2.7861E+00	9.749E-01	**9.9496E-01**	**2.5871E+00**	**7.9603E-01**
13	7.2606E+00	1.7019E+01	5.7721E+00	9.9499E-01	2.6967E+00	1.1085E+00	**9.9496E-01**	**1.7751E+00**	**6.5888E-01**
14	1.2491E-01	3.8454E+00	5.3381E+00	**0.0000E+00**	**8.3273E-03**	**2.1230E-02**	0.0000E+00	2.4287E-02	3.3570E-02
15	5.8620E+02	9.5915E+02	1.9412E+02	1.2857E+02	3.0324E+02	9.5706E+01	**6.9517E+00**	**9.3514E+01**	**6.4297E+01**
16	4.922E-01	7.8668E-01	1.1469E-01	**0.0000E+00**	**1.5900E-01**	**7.5287E-02**	**5.0309E-02**	**1.4914E-01**	**4.3507E-02**
17	1.6365E+00	1.2016E+01	2.5406E+00	1.0122E+01	1.0122E+01	3.5527E-15	**0.0000E+00**	**9.1183E+00**	**3.0344E+00**
18	2.5830E+01	3.5707E+01	5.0283E+00	**1.1039E+01**	**1.2697E+01**	**7.275E-01**	1.2689E+01	1.5447E+01	1.4003E+00
19	3.0025E-01	9.2731E-01	4.3087E-01	1.2832E-01	1.6377E-01	1.8421E-02	**1.0792E-02**	**1.0407E-01**	**4.6669E-02**
20	1.2589E+00	2.0647E+00	2.9785E-01	1.0871E+00	1.8418E+00	2.8907E-01	**4.8108E-01**	**1.3958E+00**	**4.6123E-01**
21	**0.0000E+00**	**9.0000E+01**	**3.0000E+01**	4.0019E+02	4.0019E+02	1.1369E-13	4.0019E+02	4.0019E+02	1.1369E-13
22	2.0281E+01	1.0591E+02	2.7101E+01	0.0000E+00	6.291E-01	1.9716E+00	**0.0000E+00**	**3.1212E-02**	**6.9055E-02**
23	5.3877E+02	9.6655E+02	2.4712E+02	1.1969E+02	3.5323E+02	1.3332E+02	**1.6428E+01**	**6.6447E+01**	**6.1537E+01**
24	1.0670E+02	2.0269E+02	1.7864E+01	**1.4329E+02**	**1.9846E+02**	**1.0354E+01**	2.0000E+02	2.0121E+02	2.2768E+00
25	2.0000E+02	2.0348E+02	1.9202E+00	**1.0388E+02**	**1.9704E+02**	**1.7351E+01**	**2.0000E+02**	**2.0024E+02**	**8.616E-01**
26	**1.0398E+02**	**1.1807E+02**	**1.6354E+01**	1.0199E+02	1.2629E+02	4.0703E+01	1.0199E+02	1.3513E+02	4.5887E+01
27	3.0000E+02	3.5797E+02	8.1985E+01	**3.0000E+02**	**3.0000E+02**	**0.0000E+00**	**3.0000E+02**	**3.0000E+02**	**0.0000E+00**
28	**1.0000E+02**	**1.0000E+02**	**0.0000E+00**	3.0000E+02	3.0000E+02	0.0000E+00	1.0000E+02	2.9333E+02	3.5901E+01

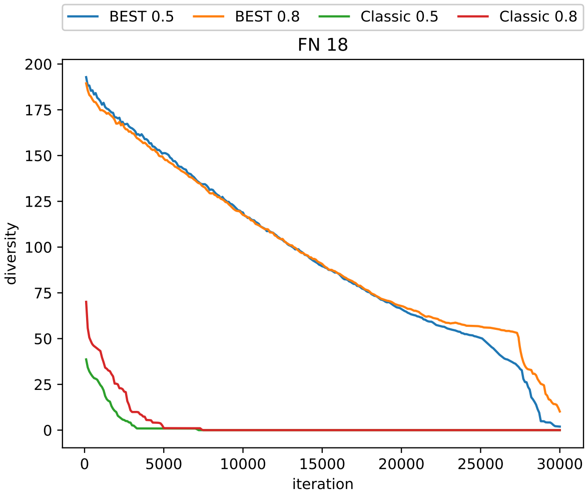

Fig. 3. Evolution of diversity for function F18

In Table II, for each problem, the method that obtained the lowest mean is shown in boldface. Additionally, any of other schemes that reported results whose differences were not statically significant when compared to the best one, are also highlighted. Note that DDE is highlighted in 10 cases, SHADE in 13 cases and DSHADE in 21, meaning that the DSHADE variant is the most robust one. It is also remarkable that DDE was the exclusive winner in three cases, so even if it uses very simple operators, in some cases the combination of simple operators with explicit control of diversity was enough to reach so high-quality results. The advantages of incorporating an explicit control of diversity are clearer when SHADE and DSHADE are compared, meaning that the MULTI_DYN scheme have reported important benefits even when integrated in state-of-the-art schemes. Moreover, while SHADE is better than DSHADE in some problems, the differences in such cases are short. However, in several problems where DSHADE is superior to SHADE, the differences are really large, such as in F15 and F23.

In addition, some tests for higher dimensions were developed — $M = \{30, 50\}$ — and we could detect that while DSHADE could obtain a proper performance, the advantages when compared to SHADE diminished. It is left for future work to extend the proposal to improve its scalability in terms of the number of variables.

V. CONCLUSIONS

Premature convergence is one of the most typical issues that affects the performance of different EAs. In the case of DE this is especially important because the exploration capabilities of its reproduction operators highly depend on the diversity maintained in the population. In most cases, the premature convergence drawback has been tackled in DE by altering the reproduction phase. However, in other fields, such as in combinatorial optimization it is more typical to use replacement-based strategies. In this paper, the MULTI_DYN replacement strategy has been incorporated into two different variants of DE to explicitly control the diversity in the replacement phase. The main aim of the method is to provide a gradual alternation between exploration and exploitation. The integration with classic DE was straightforward, whereas applying it to state-of-the-art schemes, such as SHADE, is a more complex task. Experimental validation shows the important benefits of the proposals both in terms of robustness and quality. The best-ranked variant of DE could be improved further.

Several lines of future work can be explored. Among them, considering adaptive schemes to attain a better control of the diversity seems a plausible approach. Integrating MULTI_DYN with alternative principles that have been used to improve DE might bring additional benefits.

ACKNOWLEDGEMENT

Authors acknowledge the financial support from Consejo Nacional de Ciencia y Tecnología. (CONACyT) through the ”Cienca Básica” projects no. 285599 and no. 258033.

REFERENCES

[1] J. Arabas, ÅĄ. Bartnik, and K. Opara. Dmea - an algorithm that combines differential mutation with the fitness proportionate selection. In *2011 IEEE Symposium on Differential Evolution (SDE)*, pages 1–8, April 2011.

[2] S. Biswas, S. Kundu, S. Das, and A. V. Vasilakos. Teaching and learning best differential evoltuion with self adaptation for real parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 1115–1122, June 2013.

[3] Carlos A Coello Coello, Carlos Segura, and Gara Miranda. History and philosophy of evolutionary computation. In *HANDBOOK ON COMPUTATIONAL INTELLIGENCE: Volume 2: Evolutionary Computation, Hybrid Systems, and Applications*, pages 509–545. World Scientific, 2016.

[4] L. d. S. Coelho, H. V. H. Ayala, and R. Z. Freire. Population's variance-based adaptive differential evolution for real parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 1672–1677, June 2013.

[5] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, Feb 2011.

[6] J.J. Durillo, A.J. Nebro, Carlos A. Coello Coello, J. Garcia-Nieto, F. Luna, and E. Alba. A study of multiobjective metaheuristics when solving parameter scalable problems. 14(4):618–635, Aug 2010.

[7] A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Natural Computing Series. Springer, 2003.

[8] S. M. Elsayed, R. A. Sarker, and T. Ray. Differential evolution with automatic parameter configuration for solving the cec2013 competition on real-parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 1932–1937, June 2013.

[9] Jouni Lampinen, Ivan Zelinka, et al. On stagnation of the differential evolution algorithm. In *Proceedings of MENDEL*, pages 76–83, 2000.

[10] W. B. Langdon and R. Poli. Evolving problems to learn about particle swarm optimizers and other search algorithms. *IEEE Transactions on Evolutionary Computation*, 11(5):561–578, Oct 2007.

[11] Antonio LaTorre, Santiago Muelas, and José-María Peña. A mos-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test. *Soft Computing*, 15(11):2187–2199, Nov 2011.

[12] Ole J. Mengshoel, Severino F. Galán, and Antonio de Dios. Adaptive generalized crowding for genetic algorithms. *Information Sciences*, 258:140 – 159, 2014.

[13] J. Montgomery and S. Chen. An analysis of the operation of differential evolution at high and low crossover rates. In *IEEE Congress on Evolutionary Computation*, pages 1–8, July 2010.

[14] Jean-Baptiste Mouret. Novelty-based multiobjectivization. In Stéphane Doncieux, Nicolas Bredèche, and Jean-Baptiste Mouret, editors, *New Horizons in Evolutionary Robotics*, pages 139–154, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[15] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[16] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[17] Hari Mohan Pandey, Ankit Chaudhary, and Deepti Mehrotra. A comparative review of approaches to prevent premature convergence in ga. *Appl. Soft Comput.*, 24(C):1047–1077, November 2014.

[18] Kenneth Price, Rainer M. Storn, and Jouni A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[19] A. K. Qin and P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1785–1791 Vol. 2, Sept 2005.

[20] Ângela AR Sá, Adriano O Andrade, Alcimar B Soares, and Slawomir J Nasuto. Exploration vs. exploitation in differential evolution. In *AISB 2008 Convention Communication, Interaction and Social Intelligence*, volume 1, page 57, 2008.

[21] C. Segura, C. A. Coello Coello, E. Segredo, and A. H. Aguirre. A novel diversity-based replacement strategy for evolutionary algorithms. *IEEE Transactions on Cybernetics*, 46(12):3233–3246, Dec 2016.

[22] C. Segura, A. HernÃąndez-Aguirre, F. Luna, and E. Alba. Improving diversity in evolutionary algorithms: New best solutions for frequency assignment. *IEEE Transactions on Evolutionary Computation*, 21(4):539–553, Aug 2017.

[23] Carlos Segura, Arturo Hernández Aguirre, Sergio Ivvan Valdez Peña, and Salvador Botello Rionda. *The Importance of Proper Diversity Management in Evolutionary Algorithms for Combinatorial Optimization*, pages 121–148. Springer International Publishing, Cham, 2017.

[24] Rainer Storn and Kenneth Price. Differential evolution &ndash; a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, December 1997.

[25] R. Tanabe and A. Fukunaga. Success-history based parameter adaptation for differential evolution. In *2013 IEEE Congress on Evolutionary Computation*, pages 71–78, June 2013.

[26] J. Tvrdík and R. Poláková. Competitive differential evolution applied to cec 2013 problems. In *2013 IEEE Congress on Evolutionary Computation*, pages 1651–1657, June 2013.

[27] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.*, 45(3):35:1–35:33, July 2013.

[28] Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Comput. Oper. Res.*, 40(1):475–489, January 2013.

[29] Xinjie Yu and Mitsuo Gen. *Introduction to Evolutionary Algorithms*. Springer Publishing Company, Incorporated, 2012.

[30] Daniela Zaharie. Control of population diversity and adaptation in differential evolution algorithms. In *Proc. of MENDEL*, volume 9, pages 41–46, 2003.

[31] A. Zamuda, J. Brest, and E. Mezura-Montes. Structured population size reduction differential evolution with multiple mutation strategies on cec 2013 real parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 1925–1931, June 2013.

[32] Jingqiao Zhang and Arthur C. Sanderson. Jade: Adaptive differential evolution with optional external archive. *Trans. Evol. Comp*, 13(5):945–958, October 2009.