Information Sciences 323 (2015) 106–129

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Improving the vector generation strategy of Differential Evolution for large-scale optimization

Carlos Segura[a,*], Carlos A. Coello Coello[b], Alfredo G. Hernández-Díaz[c]

[a] Area of Computer Science, Centre for Research in Mathematics (CIMAT), Callejón Jalisco s/n, Mineral de Valenciana, Guanajuato, Guanajuato 36240, Mexico
[b] Evolutionary Computation Group, Department of Computer Science, Center of Research and Advanced Studies, National Polytechnic Institute, Mexico City 07300, Mexico
[c] Department of Economics, Quantitative Methods and Economic History, Pablo de Olavide University, Seville, Spain

ARTICLE INFO

Article history:
Received 29 September 2014
Revised 31 March 2015
Accepted 16 June 2015
Available online 25 June 2015

Keywords:
Differential evolution
Diversity preservation
Global numerical optimization
Large-scale optimization
Vector generation strategy

ABSTRACT

Differential Evolution is an efficient metaheuristic for continuous optimization that suffers from the curse of dimensionality. A large amount of experimentation has allowed researchers to find several potential weaknesses in Differential Evolution. Some of these weaknesses do not significantly affect its performance when dealing with low-dimensional problems, so the research community has not paid much attention to them. The aim of this paper is to provide a better insight into the reasons of the curse of dimensionality and to propose techniques to alleviate this problem. Two different weaknesses are revisited and schemes for dealing with them are devised. The schemes increase the diversity of trial vectors and improve on the exploration capabilities of Differential Evolution. Some important mathematical properties induced by our proposals are studied and compared against those of related schemes. Experimentation with a set of problems with up to 1000 dimensions and with several variants of Differential Evolution shows that the weaknesses analyzed significantly affect the performance of Differential Evolution when used on high-dimensional optimization problems. The gains of the proposals appear when highly exploitative schemes are used. Our proposals allow for high-quality solutions with small populations, meaning that the most significant advantages emerge when dealing with large-scale optimization problems, where the benefits of using small populations have previously been shown.

1. Introduction

Differential Evolution (DE) [43] is an efficient population-based metaheuristic initially designed for continuous optimization. From its inception, it has yielded remarkable results in several optimization competitions [12], such as in the *2005 IEEE Congress on Evolutionary Computation* (CEC) competition on real parameter optimization [39], 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* (SOCO) [21]. In addition, it has been successfully applied to demanding practical optimization problems [16,56].

In spite of the promising results obtained with different variants of DE, several potential weaknesses have been discovered. One of the first weaknesses studied is the large dependency between the DE parameters and the quality of the results. Several

* Corresponding author. Tel.: + 52 473 732 7155.
E-mail addresses: carlos.segura@cimat.mx, csegura@ull.es (C. Segura), ccoello@cs.cinvestav.mx (C.A. Coello Coello), agarher@upo.es (A.G. Hernández-Díaz).

http://dx.doi.org/10.1016/j.ins.2015.06.029
0020-0255/© 2015 Elsevier Inc. All rights reserved.

studies have shown that DE is very sensitive to the setting of its control parameters [15,60]. Furthermore, several DE trial vector generation strategies have been proposed [17,37], hampering the choice of DE's parameters and components. In order to sidestep this drawback, some schemes consider the simultaneous use of several DE components and parameter values [34,55].

Some weaknesses involving the specific way in which new individuals are created in DE have also been identified [19,20]. DE borrows the idea from the Nelder and Mead method [28] of employing information from the vector population to alter the behavior of the variation scheme. Thus, the perturbation performed by the scheme is internally induced, i.e., the strength of the mutation depends on the contents of the population. The main associated problem is that DE has the potential to generate only a limited number of different trial solutions within one generation. In [19], it was shown that this can lead to a situation where DE may stop proceeding toward a global optimum even though the population has not even converged to a local optimum. This situation is called stagnation. The likelihood of stagnation occurring depends on the population size in question, and is more likely to occur when low population sizes are involved. An additional weakness of DE was reported in [20], where the authors generated a set of complex problems by applying genetic programming with the aim of detecting the drawbacks of different metaheuristics. It was shown that, in general, DE is less expansive than other metaheuristics. Thus, DE might be deceived into converging on the wrong peak, and once there, it could be impossible for this approach to escape.

The aforementioned weaknesses are closely related to the high selection pressure introduced by DE's survivor selection scheme and to its premature loss of diversity [2]. In the field of evolutionary computation, several schemes for dealing with such drawbacks have been proposed, some of which have been considered with the aim of improving the capabilities of DE. In fact, since its inception, the hybridization of DE with annealing procedures was studied to reduce the selection pressure [37]. Some more recent schemes include the introduction of stochasticity into the selection process [2] or the use of generational replacement [3].

The papers listed above enumerated some of the potential weaknesses of DE. However, most state-of-the-art DE schemes do not introduce any special mechanisms to address these weaknesses, probably because in most cases their effects are not very significant. In fact, in many cases the main effects of these weaknesses can be bypassed by increasing the population size [19]. Thus, it seems that with the traditional parameterizations of DE, the effect of these potential weaknesses is not very cumbersome. The result is that not much attention has been paid to these weaknesses. On the other hand, it is well-known that DE suffers from the curse of dimensionality, which refers to various phenomena that arise when dealing with high-dimensional spaces that do not occur in low-dimensional settings. In these cases, the results can generally be improved by applying micro-DE [30], i.e., schemes with low population sizes, meaning that inducing additional intensification is useful in these cases. However, such a reduction in the population size can lead to problems in terms of reliability [60].

The main aim of this paper is to show that when dealing with high-dimensional problems, the weaknesses mentioned above might arise and greatly deteriorate the performance of DE. Thus, in this paper, two techniques to address these drawbacks are proposed. Instead of considering general proposals from the field of evolutionary computation, our modifications take into account the particular variation method of DE so as to preserve its basic principles. Then, we show the utility of these schemes when faced with two well-known sets of high dimensional optimization benchmark problems. This is tested with several different non-hybrid variants of DE. Experimental evaluation shows that the benefits are more significant when exploitative configurations of DE are used, such as when considering low population sizes. This indicates that the effects of the weaknesses analyzed herein have a higher likelihood of appearing when large-scale complex problems are involved because of the required balance toward intensification. Note that it is beyond the scope of this paper to design a complete state-of-the art optimization scheme, which usually involves hybrid methods that combine several optimization methodologies. Instead, we want to better explore the reasons for the sub-optimal behavior of DE when applied to large-scale problems, which is why simple baseline and non-hybrid state-of-the-art algorithms are used.

The rest of the paper is organized as follows. A brief summary of DE and a discussion of the relevant background are given in Section 2. Section 3 is devoted to justifying the development of the new scheme by analyzing certain mathematical properties of several related schemes. The new proposal is described in Section 4. Then, our experimental validation is presented in Section 5. Finally, our conclusions and some lines of future work are given in Section 6.

2. State of the art in DE

2.1. Fundamentals

DE was initially proposed as a direct search method for single-objective continuous optimization problems [44]. In continuous optimization, the variables governing the system to be optimized are given by a vector $\vec{X} = [x_1, x_2, x_3, \ldots, x_D]$, where each variable x_i is a real number. The number of variables (D) defines the dimensionality of the optimization problem. Finally, the objective function $f(\vec{X})(f : \Omega \subseteq \mathbf{R}^D \rightarrow \mathbf{R})$ measures the quality of each set of variables. The aim of the optimization—considering a minimization problem—is to find a vector $\vec{X*} \in \Omega$ in which $f(\vec{X*}) \leq f(\vec{X})$ holds for all $\vec{X} \in \Omega$. The problems most typically addressed with DE are box-constrained optimization problems. In these cases, the region Ω is specified with the lower (a_j) and upper (b_j) bounds of each variable (j) in the problem.

DE is a population-based stochastic algorithm that belongs to the broad class of Evolutionary Algorithms (EAS). As with other EAS, it randomly initializes a population (P) with NP individuals ($P = \{\vec{X}_1, \ldots, \vec{X}_{\mathrm{NP}}\}$). Each individual is a vector with D real numbers. The value of the jth variable of individual X_i is denoted by $x_{i,j}$. Then, the population evolves over successive iterations to explore the search space. In DE, the term *vector*, instead of *individual*, is commonly used. At each DE iteration, the following steps are executed. First, for each vector in the population—called *target vector*—a new *mutant vector* is created using a vector generation strategy. Several vector generation strategies have been proposed [29,37]. Then, the mutant vector is combined with the target vector to generate the *trial vector*. After generating NP trial vectors, each one is compared against its corresponding target vector. In each comparison, the best one is selected to survive. In case of a tie, the new generated trial vector survives.

Regardless of the vector generation strategy, the term *base vector* is used to refer to an initial vector that is subsequently perturbed to generate the mutant vector. The perturbation is done by considering one or several differences among other vectors in the population. In order to classify the different variants of DE, the well-known notation DE/*x*/*y*/*z* was introduced in [44]. In this paper, in order to comment some properties of several mutation strategies, the notation DE/*x*/*y* is used. The descriptions given in such cases are independent from the crossover scheme.

One of the most common ways of selecting the base vectors is the "rand" strategy. In the "rand" strategy, any vector in the population different from the target vector is randomly selected as the base vector. Then, this vector is mutated to create the mutant vector (V_i). The most commonly applied crossovers are the binomial (bin) and the exponential (exp) operators. The crossover operation is controlled by means of the crossover rate (CR). In the bin strategy, the trial vector (U_i) is generated using (1). $\text{rand}_{i,j}$ is a uniformly distributed random number in the range [0,1], and $j_{\text{rand}} \in [1, 2, \ldots, D]$ is a randomly chosen index.

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } (\text{rand}_{i,j} \le \text{CR } \textit{or } j = j_{\text{rand}}) \\ x_{i,j} & \text{otherwise} \end{cases} \tag{1}$$

The exp crossover operates as follows. First, an integer n is chosen at random from among the numbers [1, D]. This integer is the starting point in the target vector, from where the crossover starts. In addition, an integer $L \in [1, D]$, which represents the number of mutated components, is generated by considering the truncated geometric distribution (see (2)) [54]. Finally, the mutant vector is generated using (3). In this equation the angular brackets $\langle\,\rangle_D$ denote a modulo function with modulus D.

$$\text{Prob}(L = h) = \begin{cases} (1 - \text{CR})\,\text{CR}^{h-1} & \text{if } 1 \le h < n \\ \text{CR}^{n-1} & \text{if } h = n \end{cases} \tag{2}$$

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } j - 1 \in \{\langle n-1\rangle_D, \ldots, \langle n+L\rangle_D\} \\ x_{i,j} & \text{otherwise} \end{cases} \tag{3}$$

It is also important to note that the vector generation strategy might generate trial vectors outside the feasible region. Several ways of dealing with this scenario have been proposed [37]. A widely used repairing scheme is based on randomly reinitializing the offending values. As this last approach is the most unbiased [37] and has yielded promising results [38], it is the one we will use in this paper.

2.2. Large scale optimization

The large amount of research conducted on DE in the last two decades has shown that the performance of DE quickly deteriorates as the size of the search space increases [12]. In fact, this is a common finding in the field of evolutionary computation [50]. The reasons appear to be two-fold. First, problem complexity usually increases with the size of the search space, meaning that more efficient search strategies might be required. Second, the search space grows exponentially but the available computational resources cannot be scaled in the same way. In fact, in most scalability studies, the number of function evaluations only increases linearly with the dimensionality of the problem.

The new properties of the optimization process imply that the features required of the optimization scheme might differ from those corresponding to low-dimensional cases. For instance, the balance needed between exploration and exploitation might vary. In light of this, the most promising schemes for large-scale problems might not be the same as those used for low-dimensional problems. In some ways, it is similar to those cases where the optimal scheme depends on the number of function evaluations allowed, because the optimal way of exploring the search space might depend on the proportion of the search space to be explored. Moreover, in our opinion, we cannot expect to have schemes as efficient as in low-dimensional spaces precisely because of the reduction in the percentage of the space being explored. We should note that there exist some methods, as Multiple Offspring Sampling (MOS) [21], that have yielded quite good results for a large range of dimensionalities without the requirement to adapt their parameters to the different search space sizes. However, even in these high-quality cases, there exist other approaches that might be preferred for some specific dimensions. For instance, in some problems, Generalized Adaptive Differential Evolution (GADE) [52] yielded better results than MOS at the lowest dimensionalities, but not at the largest ones.

Since the inception of DE, it has been obvious that its optimal setting might depend on the dimensionality of the given optimization problem. For instance, several different recommendations for setting NP based on *D* have been given. Some initial studies [15,60] recommended to establish a linear dependency between NP and *D*. However, when very large search spaces were used [30], it was clear that using such large population sizes is not practical. In fact, in MOS—one of the top-rated schemes for the special issue held in SOCO—the population size was fixed to 15 even for dimensions as high as 1000. Small population sizes induce a higher convergence speed but at the cost of increasing the likelihood of stagnation or premature convergence. Thus, the proper setting of NP does not only depend on *D* but also on the stopping criterion. Finally, some promising schemes use a variable population size so as to adapt the exploitation and exploration capabilities to the requirements of different stages [8,58]. This idea has been successfully adopted to deal with high-dimensional problems [7].

In the case of the crossover scheme, several comparative studies involving low-dimensional spaces [26] have shown that the bin strategy is generally more adequate. However, in some high-dimensional benchmarks, much better results have been reported with exponential crossover than with binomial crossover [21]. The reason seems to be that when too many values from the mutant vector are considered, the perturbation is too disruptive. In [54] it was shown that the difference between the behavior of binomial and exponential crossover is influenced more by the different impact of CR on the number of values taken from the mutant vector than by the different strategies for selecting components. In fact, when dealing with high-dimensional problems, promising results have also been obtained with binomial crossover and very low CR values [30]. However, when operating with other large-scale optimization problems, such as those devised for the competition on Large Scale Global Optimization [45] that was organized at the 2010 IEEE Congress on Evolutionary Computation (CEC'10), some different conclusions can be drawn. For instance, some of the best solutions that have been reported for DE consider the combination of several crossover operators [5,48]. The implications of these combinations are not fully analyzed in the previously indicated papers, but our preliminary experiments with this benchmark show that by only using exp crossover, the perturbations are too limited to avoid stagnation. However, relatively small perturbations are required in some stages, which seems to justify the combination of several crossover operators. The large differences between the best-behaved schemes for the SOCO and CEC'10 benchmark sets—where not only the crossover operators are different—probably indicate that adapting the schemes to the features of the problems at hand is more important when dealing with large-scale problems than when dealing with problems with a low number of variables. In fact, it has been recently shown that some of the best schemes defined for certain benchmarks do not perform well when other functions are considered [23].

Several different alternatives for dealing with large-scale optimization have been devised. The reader is referred to [25] for a detailed review of different frameworks. In [25], adaptations involving several different metaheuristics are reviewed. Most of the ideas presented have also been applied to DE. One of the most popular design decisions when dealing with large-scale optimization is to hybridize the scheme by taking into account different optimization methodologies [34]. Probably the most popular choice is to design memetic schemes by combining DE with an intensification scheme such as a local search strategy. However, note that in some of the most successful memetic schemes for large-scale optimization, the additional components are not used only for intensification. For instance, in MOS, the Multiple Trajectory Search (MTS) can lead to very large movements. Thus, two different schemes where both can do global and local searches are integrated. While this idea is quite promising, the purposes of each method are not independent, meaning that the control of the different schemes involved is not trivial at all. This is also an indication of the poor diversity induced by DE when dealing with large search spaces. In fact, another popular choice is to apply a DE with a structured population [49], which also gives additional control over the diversity.

Several other authors are working on non-hybrid DE schemes. While these schemes are not yet able to yield the high-quality solutions obtained with hybrid methods, they are usually easier to analyze and control. Advances in non-hybrid variants are important, as they might yield future hybrid schemes that are easier to control, along with further improvements. For instance, if the capabilities of DE as a global searcher are improved, it might be used only as a global optimizer and combined with a "real" local search that is based only on making small movements. Some of the most popular non-hybrid alternatives that have been devised in the DE field are based on the application of adaptive schemes [52] and on the application of opposition-based learning [47]. The principle of the former is to exploit the strengths of different components and/or parameter values. The latter is based on diversifying the search by incorporating an opposition-based population. GADE [52] is an example of the former, while Generalized Opposition-based Differential Evolution (GODE) [47] is an example of the latter. Both schemes were successfully applied to the SOCO tests. While these schemes are not yet competitive when compared to hybrid approaches, they are clearly superior to the basic variant of DE. In the case of the CEC'10 tests, some similar ideas have also been adopted. In this case, however, several adjustments must be made in order to obtain competitive results. For instance, in jDEsps [5], in addition to parameter adaptation, a varying population size, ageing, a local search and several other mechanisms were included.

Finally, another relatively unexplored topic for large-scale optimization with DE is the use of cooperative co-evolution [35]. The principle of co-evolution is to exploit the separability of problems [50]. In these kinds of schemes the problem is decomposed by splitting the variables into different subcomponents, each of which is then optimized and the findings combined. A major difficulty in applying co-evolution is the choice of a suitable decomposition [31,42], though some strategies for automatically decomposing problems have been proposed [32,51]. Note that even with the use of co-evolution, considering small population sizes seems very promising [33].

2.3. Adaptation of scale factor

In DE, one of the parameters that must be set by the user is the mutation scale factor (F). Several adaptive proposals have been devised to avoid having to manually tune this parameter [12]. Since these schemes are related to the new proposals put forth in this paper, they are briefly summarized in this section. The first proposals that considered a non-static F value did not take into account the feedback obtained from the optimization process. In some cases, a static schedule for F was defined [11]. In other cases, the F value was randomly set [37]. The update process can be done at the generation, vector (dither) or variable level (jitter). Several random distributions have been proposed for generating the new values, with some of the most widely used being the Gaussian, Log-normal and Cauchy distributions [36]. Empirical studies have revealed that the results are highly dependent on the distribution used, although no single distribution has been shown to be superior to the others.

In order to better adapt the scheme to the given optimization problem, the feedback obtained in the optimization process can be used to set F [46,53]. Interestingly, in many of these schemes a large amount of randomness is considered in the generation of new *F* values. In fact, proposals where less randomness is considered have not yielded such promising results [9]. Some of the most successful schemes have been jDE [6] and JADE [55]. In jDE, each individual has its own value for *F*. When a new individual is created, a new random *F* value in the range $[F_{\min}, F_{\max}]$ is used with ratio τ. Otherwise, the *F* value of the target vector is used. In JADE, the *F* value is generated based on a Cauchy distribution with location factor μ_F and scale parameter 0.1. If the generated value is lower than 0, it is regenerated. If it is higher than 1, it is truncated to 1. The location factor is initialized to 0.5 and updated after each generation by considering the Lehmer mean of the successful *F* values, the previous location factor and a parameter *c*, which represents the location factor's velocity of adaptation.

Finally, it is interesting to note that some researchers have claimed that the benefits obtained by considering feedback when setting the *F* values are not significant [59]. They assert that the advantages obtained with these methods stem from the increase in diversity and that random schemes could be used as well. In fact, it is worth noting that in SaDE [38]—one of the most promising variants of DE—*F* is set randomly using a Gaussian distribution with mean 0.5 and standard deviation 0.3. However, the study presented in [59] was carried out with a basic DE variant. In [41] it was shown that there is a relationship between the likelihood of success of the adaptation schemes and the balance between exploration and intensification caused by the trial vector generation strategy. Thus, these kinds of adaptive schemes are useful but they are not as general as expected; thus, designing more general methods is still an open area of research.

3. Mathematical analysis of related approaches

3.1. Number of potential trial vectors

The appearance of stagnation in DE is closely related to the limited number of potential trial vectors that can be generated in each generation [19,20]. In [19], an analysis of the number of potential trial vectors that can be created in different contexts is carried out by considering binomial crossover. As previously mentioned, in the case of high-dimensional spaces, the use of exponential crossover has led to promising results [21]. By adapting the equation developed in [19] to binomial crossover (4), the number of potential trial vectors of exponential crossover can be calculated (5). A DE/rand/1/exp strategy is considered.

$$n_{\text{trial}} = (\text{NP}^3 - 3\text{NP}^2 + 2\text{NP}) \times 2^D \times \text{NP} \tag{4}$$

$$n_{\text{trial}} = (\text{NP}^3 - 3\text{NP}^2 + 2\text{NP}) \times \text{NP} \times (D^2 - D + 1) \tag{5}$$

The number of potential trials that can be generated with exponential crossover is much smaller than the number generated with binomial crossover. In fact, in the exponential case, the number of potential trials only grows polynomially with *D*. This means that as *D* grows, the difference between the search space size and the number of potential trials increases exponentially. For this reason, our hypothesis is that as *D* grows, the exploration capabilities of DE deteriorate and stagnation is more likely to appear. The above equations also reveal that by increasing NP, the number of trial vectors can be increased. Doing so, however, reduces the convergence speed, so expanding NP could significantly increase the number of function evaluations required, which might be non-viable for expensive, large-scale optimization problems.

In the cases where the binomial strategy is used, the number of trial solutions grows exponentially. However, when dealing with high-dimensional spaces, the most promising results have been achieved by using very low CR values [30]. This means that the generation of each potential trial solution is not equiprobable. For instance, a vector containing the whole variables of a mutant vector is generated with a very low probability (CR^D). Thus, in practice, the number of potential trial vectors with no insignificant probabilities of being generated does not grow as much as the search space.

3.2. Effects of adapting the mutation scale factor

Several schemes that consider a variable *F* value have been devised in an effort to increase the robustness of DE. The number of potential trial vectors can be indirectly increased by using these schemes, which could alleviate stagnation problems. As previously described, the methods that consider a non-static *F* value can be separated into those schemes that take into account feedback to set the value of *F* and those that do not do it. Among the schemes that do not consider any feedback, the most popular strategies set *F* by using a random (e.g., Gaussian or Cauchy) distribution. One of the first distributions tested was the $N(0, 1)$ [1], i.e., a Gaussian distribution with $\mu = 0$ and $\sigma = 1$. This distribution has a large standard deviation, and as such it could generate very low and very high *F* values frequently. This means that the size of the vector differences and the strength of the perturbations are not as correlated as in the basic version of DE. For instance, in the first stages of the optimization, even if all the differences that appear in the population are large, some small perturbations might be carried out. In some way, this goes against the principles of DE. Related to this drawback, it is important to note that in most of the distributions considered so far, the probabilities of generating values near zero are not negligible. Thus, whereas all of these schemes could improve the exploitation capabilities of DE, they would do so at the cost of accepting small movements even in the first stages of the optimization, where the original DE does not generate small perturbations. In the cases in which large-scale spaces are explored and a limited amount of resources are considered, these small movements could significantly deteriorate performance. In the same way, when using distributions with long tails, DE might generate highly disruptive mutations even at the end of the optimization.

In order to better understand the behavior of these schemes, it is interesting to show the conversions performed by these methods for a one-dimensional problem. Let us consider a problem with one variable (v) whose lower and upper bounds are 0

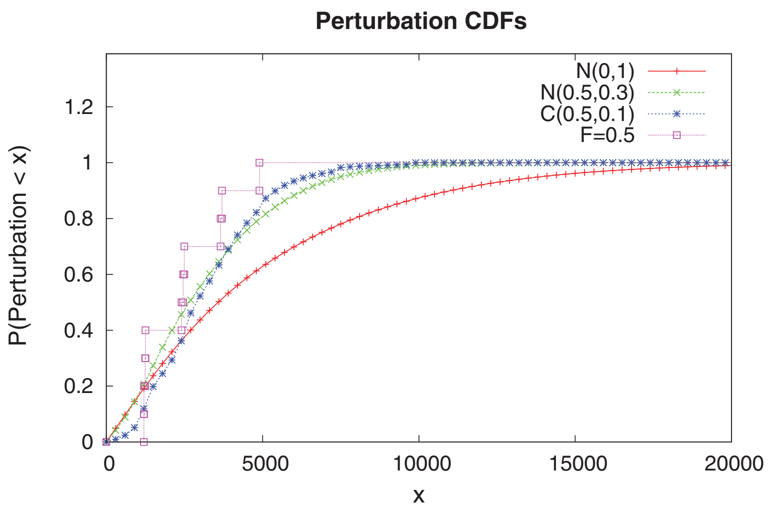

Fig. 1. CDFS used for mutating in different DE variants.

and 10,000, respectively, and assume that the N individuals in the current population are almost equidistributed in the search space. Specifically, five individuals with the following v values are considered: 20, 2505, 5008, 7419 and 9819. Fig. 1 shows the cumulative distribution function (CDF) used to perturb the individuals in different DE variants, i.e. the probability of using perturbations lower than the value given on the horizontal axis. Specifically, it shows the CDFS obtained with a DE that considers a fixed scale factor ($F = 0.5$) and with three typically applied distributions: the distribution $N(0,1)$, used in [1]; the $N(0.5,0.3)$, applied in SaDE [38]; and finally, the Cauchy distribution with location factor 0.5 and scale parameter 0.1, which is the initial distribution considered in JADE [55]. For this last case, the truncation mechanism applied in JADE is considered. Given the symmetry of DE, absolute values are considered.

DE with the distributions tested tends to yield small perturbations with a larger probability than the basic DE. In fact, the original DE does not yield perturbations smaller than 1200 units. However, in the rest of the schemes, the probability of producing a perturbation lower than 1200 units is larger than 12%. In the same way, larger perturbations than in the original DE might be generated. For instance, DE with a fixed F value does not produce any perturbation larger than 5000 units. However, the probabilities of having such perturbations with the remaining distributions are not insignificant. Thus, many of the perturbations might generate individuals outside the feasible region. In fact, several perturbations larger than the search space size can be generated. These undesired movements might deteriorate the performance of DE, especially when dealing with large spaces where the percentage of space explored by DE is smaller than in low-dimensional spaces. Considering all these properties, it is clear that using such random distributions might have a large impact on the results. However, it is not clear whether these properties are beneficial when high-dimensional spaces are involved.

In other promising schemes, the feedback obtained during the execution is used to set the value of F. In these cases, it is not easy to develop a theoretical analysis because the behavior depends on the function to be optimized. However, it is important to note that the advantages of using feedback are not as robust as expected [41,59]. In any case, since some state-of-the-art DE schemes use feedback to set the value of F, the experimental evaluation developed in this research takes into account some of the most well-known adaptive schemes.

4. Our proposals

In this paper, a novel DE scheme is proposed whose aim is to avoid the aforementioned weaknesses. The scheme incorporates two main modifications. Both proposals change the way of perturbing solutions in DE. Specifically, when the new modifications are considered, the way of calculating the perturbation size is modified. Thus, instead of using the scaled difference vector to perturb a given base vector, alternative methods are applied. The principle of the first modification is to increase the number of potential trial solutions while at the same time preserving the basic behavior of DE. The aim of the second modification is to improve the exploration capabilities of DE so as to correctly deal with large search spaces and enable additional ways of avoiding stagnation and premature convergence. Both modifications are used when a trial vector that considers only one mutant variable is going to be created, i.e., when the value L generated by the exponential crossover is equal to one. These modifications might also be applied when several mutant variables are inherited by the trial vector. This can be done in several ways, however, and requires a more in-depth study considering the relationships among the distributions that arise for the different variables. We carried out some initial experiments in which the distributions were regarded as independent, though this resulted in unsuccessful schemes. This could probably be fixed by taking into account the relationships among the distributions that arise in the different variables. One option might be to apply copula functions, as was done in Estimation of Distribution Algorithms [40]. However, this would

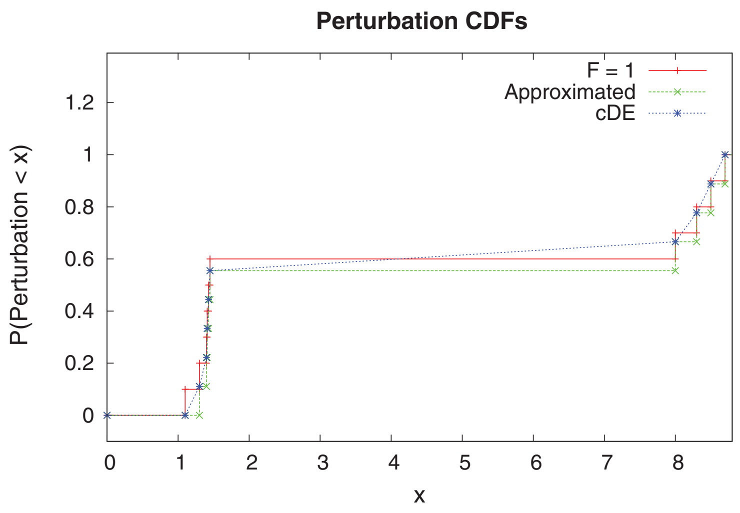

Fig. 2. CDF generated with CDE.

result in a quite complex scheme. Since quite promising results are obtained without the incorporation of these more complex schemes, this analysis is beyond the scope of this paper.

4.1. Continuation scheme

The aim of the first modification is to increase the number of potential trial vectors that can be generated with DE. In order to define the proposal, the mathematical analysis developed for the DE schemes presented before for dealing with random F values was considered. One of the main strengths of these models is that any perturbation between the maximum and the minimum admissible perturbations has a non-zero probability of being generated. Thus, if for a given variable i the maximum and minimum perturbations that can be generated with a given distribution are max_i and min_i, respectively, any modification in the range $[min_i, max_i]$ can be generated. In other words, the CDF is monotonically increasing in this range. This does not happen when a fixed value of F is used. In contrast, the main weaknesses detected involve the very small or large mutations that are occasionally carried out.

Our proposal is based on generating a monotonically increasing CDF by slightly modifying the CDFs that appear in those DE schemes that consider a fixed F value. Then, these CDFs are used to generate random numbers that are multiplied by F and used to mutate the population vectors, just as in the original DE. The scheme functions as follows. First, an approximation of the CDF that models the absolute values of the perturbation performed by a DE with $F = 1$ is calculated for the variable that is going to be mutated. This is done by considering (6), where $minor(x)$ is the number of differences that appear in the population that are lower than x. The denominator is the amount of vector differences appearing in the population minus one. This ensures that the maximum value of (6) is one. The generated CDFs have the form of a step function. Then, for each difference (d) that does not appear in the current population and that is located between the minimum and maximum existing differences, CDF(d) is calculated using a linear interpolation. The interpolation is done by considering the differences $d1$ and $d2$, where $d1$ is the highest difference lower than d that appears in the population, and $d2$ is the lowest difference higher than d that appears in the population.

$$\mathrm{CDF}(x) = \frac{minor(x)}{0.5 \times (\mathrm{NP}) \times (\mathrm{NP} - 1) - 1} \tag{6}$$

As an example, assume that the set of differences appearing in a given population with NP = 5 is: 1.1, 1.3, 1.4, 1.41, 1.43, 1.45, 8, 8.3, 8.5, 8.7. Fig. 2 shows the CDF corresponding to a DE with $F = 1$, the approximate CDF calculated using the above steps—function (6) is evaluated for each of the differences appearing in the population—and the CDF corresponding to the new model, i.e., the previous one with interpolation. The scheme that considers the use of this new CDF is called Continuous Differential Evolution (CDE), in reference to the fact that any mutation step in the range $[min_i, max_i]$ can be generated. As we can see, the CDF in CDE has a larger slope in the zones where a larger number of differences appear and the values min_i and max_i remain intact. Thus, differently to the cases that apply a variable F value, the DE principle that correlates the strength of the mutation to the size of the differences is better respected in our transformation. At the same time, the number of potential trials is augmented with the aim of avoiding stagnation.

Note that in order to calculate the CDF, $\frac{\mathrm{NP}\times(\mathrm{NP}-1)}{2}$ distances must be taken into account. In our implementation, the only CDFs that are calculated are those that are used in each generation. In the worst case, and taking into account that for large-scale optimization, NP is usually much smaller than D, NP CDFs should be calculated, meaning that in the worst case about NP^3 operations are required. However, in the average case fewer operations are used. In fact, when the “exp” crossover is taken

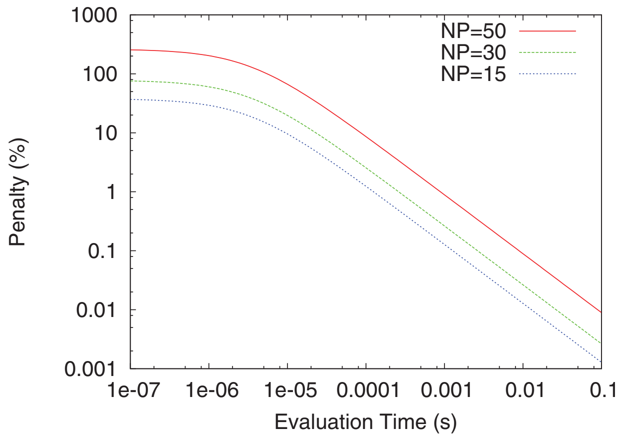

Fig. 3. Approximation of the penalty (%) induced by the continuation scheme.

into account, on average only $(1 - \text{CR}) \times \text{NP}$ CDFs are calculated in each generation, meaning that the time associated with the continuation scheme is shorter. When using benchmark problems, this task can consume a non-negligible proportion of the total time. For instance, when DE with $\text{NP} = 15$ and $\text{CR} = 0.9$ is used to solve the F1 problem of the soco benchmark, the model with the continuation scheme uses about 35% more time. Moreover, the penalty grows for larger population sizes. Specifically, the penalty is 75% when NP is set to 30 and 255% when NP is set to 50. However, when more expensive functions are used, these penalties are drastically decreased. Once a case is analyzed, it is not difficult to predict the penalty obtained for other problems. The most important features are the evaluation time associated with the new problem and the value of NP. For instance, if for a given problem whose evaluation cost is *EvCost*, the basic DE variant takes $t1$ seconds to complete *StopEv* evaluations and the model that incorporates the continuation scheme takes $t2$ seconds, the penalty percentage can be approximated using (7). In this equation, t represents the evaluation time associated with the problem whose penalty is predicted. Note that the numerator represents the increase in time caused by our proposal, which is independent of the evaluation cost, while the denominator is an estimation of the time of an execution of the DE variant that does not include the continuation scheme, for a problem whose evaluation cost is t.

$$\text{Penalty}(t) = \frac{t2 - t1}{t1 - StopEv \times EvCost + StopEv \times t} \times 100 \tag{7}$$

Fig. 3 shows an approximation of the penalty by using the data obtained for F1 with three different values of NP. Note that for short evaluation times, the penalty is large. However, as the evaluation time increases, the penalty quickly diminishes. In order to evaluate the accuracy of the approximation, the penalties associated with F17 were calculated. Experimentally, the penalties were 0.87%, 2.02%, and 4.3% when NP was set to 15, 30, and 60, respectively. In problem F17, the evaluation takes about 3×10^{-4} s, so the predicted values are 0.42%, 0.87%, and 2.93%. While these values are not exact, they are proper approximations, meaning (7) can be used to accurately estimate the penalties. In any case, the key is that even with relatively inexpensive evaluation functions such as F17, the penalty is not too large, so in computationally demanding problems the penalty term is probably negligible. As a result, in this paper the computational study is carried out considering the number of evaluations instead of the time.

4.2. *Avoiding a large reduction in diversity*

Initial experimentation—which did not consider any technique to promote large perturbations—has shown that large mutations were not required in most of the executions. However, in some of the worst cases, the maximum difference between the values appearing for one variable might be highly reduced from one generation to the next. This might leave large portions of the search space unexplored, yielding unsatisfactory results. In addition, our initial experiments showed that the likelihood of this occurrence increases with the number of dimensions considered. Our analysis showed that this was related to the *fake large* moves studied in [27], which are a set of moves that appears when there are some low differences in the population. Note that, given two randomly generated individuals (X_i and X_j), the probability (P) that the difference between the values of their corresponding vth variables will be lower than ϵ is given by (8). Obviously, as the number of dimensions considered grows, the probability associated with the appearance of low differences in at least one variable increases sharply. Specifically, the expected number of variables where this happens is $D \times P$, meaning that as the number of dimensions grows, there will be a larger number of variables where low differences will appear. In fact, the probability associated to this event tends to one when the number of variables considered tends to infinity.

$$P(|x_{i,v} - x_{j,v}| < \epsilon) = \frac{2\epsilon}{b_v - a_v} - \frac{\epsilon^2}{(b_v - a_v)^2} \tag{8}$$

The previous formula applies to any type of EA. However, DE is affected by the *fake large* moves, which is why special actions must be taken in DE. Considering this and the promising results that have been obtained with complex multimodal problems by using distributions with long tails to generate the F values [36], we decided to design an adaptive scheme that promotes large perturbations in a controlled fashion. Since the scheme promotes large perturbations, using it frequently might provoke a too disruptive scheme. For this reason, the scheme is only used when the value L generated by the exponential crossover is equal to one and with a ratio equal to HMR—high mutation ratio. In the remaining cases (ratio 1−HMR), the original DE or the continuation scheme previously described is applied.

The scheme operates as follows. Each time that it is applied to a given variable i, a large random number (R) is generated. Then, R is used to perturb such a variable in the base vector, i.e., it acts as the scaled difference vector in the original DE. In half of the cases it is added and in the remaining cases it is subtracted. In order to calculate R, our scheme stores the maximum admissible mutation ($FalseMax_i$) for each variable i. Initially, they are set using (9). Thus, a maximum R equal to at least 20% of the variable range size is initially allowed. Then, R is generated by producing a random number in the range $[Max_i, FalseMax_i]$, where Max_i represents the maximum perturbation that can be generated for such a variable by using the CDF of CDE. In each application of this scheme, if the value of Max_i is higher than $FalseMax_i$, $FalseMax_i$ is reset to Max_i. This usually happens in the first generations, meaning that our scheme is not sensitive to the initialization of $FalseMax_i$. Even so, this initialization should be maintained in order to ensure a minimum amount of perturbation, even if an improper initial population is generated. After perturbing the base vector, the value $FalseMax_i$ is updated. In cases where the trial vector generated was successful, i.e., better than the target vector, the value is updated with (10). Otherwise, it is updated with (11). The principle of the update mechanism is similar to the one that governed the design of the Win or Learn Fast approach [4]. Specifically, in the cases where large mutations are successful, even larger mutations are promoted. In contrast, when mutations are not successful, the maximum admissible mutation for this variable is shortened. The *UpdateDenom* parameter can be used to specify how quickly $FalseMax_i$ is adapted. Specifically, the adaptation is slower when larger values are used. As a result, higher values should be used for cases where a higher balance toward exploration is required. Note that the denominators in (10) and (11) might even take different values. However, since the experimental evaluation shows that the new proposal is not too sensitive to this parameter, devoting great efforts to tuning the update mechanism in this way seems unproductive.

$$FalseMax_i = \frac{b_i - a_i}{5} \tag{9}$$

$$FalseMax_i = FalseMax_i + \frac{(FalseMax_i - Max_i)}{UpdateDenom} \tag{10}$$

$$FalseMax_i = FalseMax_i - \frac{(FalseMax_i - Max_i)}{UpdateDenom} \tag{11}$$

Finally, we would like to mention that considering how schemes that adapt the F value usually improve on the intensification capabilities of DE, incorporating some modifications to enhance the exploitation features of DE seems encouraging. In fact, some of the most promising schemes published in the literature are hybrid approaches that combine DE with local search mechanisms [23]. As mentioned earlier, in some of the most promising hybrid schemes, such as MOS [21], several components are used to carry out the global search. We did some initial tests by incorporating a local search mechanism at the end of the executions. However, these schemes were not as successful as other hybrid approaches where several components are in charge of the global search. This means that while our schemes clearly improve the global search capabilities—as demonstrated by the experimental validation—, additional work will be required to develop a competitive DE for large-scale optimization where no other components are involved in the global search. Thus, any further improvement to the global search capabilities, as well as hybridizing the new DE schemes with some of the highly efficient local search mechanisms proposed in the literature, is left for future work.

5. Experimental evaluation

In this section, the experiments conducted with the newly designed DE scheme are described. Most of the analyses were performed with the benchmark problems devised for soco [24], which are a set of 19 scalable continuous optimization problems to be minimized. The parameter D allows setting the number of variables in the problems. These problems have different features and combine different properties involving modality, separability, and ease of optimization dimension by dimension. In order to analyze the schemes, six sets of experiments were carried out with such a benchmark. In every case, each execution was repeated 1000 times, unless otherwise stated. In addition, and so as to illustrate how our schemes can provide benefits for problems with very different features, we also conducted some experiments with the CEC'10 test problems [45]. As previously described, the schemes that have reported promising results for each of these benchmarks are quite different. Specifically, the selection of the proper crossover operator seems very important. In the case of the soco test problems, and considering the results obtained in [21], exp crossover was used. In the case of the CEC test problems, similarly to the scheme devised in [10], in our proposal we decided to alternate between the use of bin and exp crossover operators, using the bin operator in even generations and the exp operator in the odd ones. The other properties of DE were the same for both benchmarks.

Since stochastic algorithms were considered in this study, comparisons were carried out by applying a set of statistical tests. A similar guideline as the one applied in [13] was taken into account. Specifically, the following tests were applied, assuming a significance level of 5%. First, 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 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 for assessing whether samples are drawn from the same distribution. Note that some researchers [18] have suggested that the obtained *p*-values might be normalized by taking into account the number of independent executions carried out. However, in our experiments the *p*-values were either very high or very low, so such scaling does not affect the results of our statistical tests. In this work, the sentence "algorithm A is better than algorithm B" means that the differences between them are statistically significant, and that the mean and median obtained by A are lower—one of the metrics might be equal—than the mean and median achieved by B. In the experiments below, the tables provided show the results of certain statistical tests used to compare a variant of DE with the same scheme, but incorporating our proposals. In these tables the following symbols, and their associated meanings, are used:

- $\uparrow$: The model that incorporates our proposals is better.
- $\downarrow$: The model that incorporates our proposals is worse.
- $*$: The differences are statistically significant, and the model that incorporates our proposals achieves a higher mean and lower median.
- $**$: The differences are statistically significant, and the model that incorporates our proposals achieves a lower mean and higher median.
- $\leftrightarrow$: The differences are not statistically significant.

5.1. First set of experiments: Benefits of each proposal

As described in Section 4, in this work we propose two modifications for enhancing DE. Each modification can be included separately into DE or they can be used in conjunction. The aim of the first experiment was to analyze the benefits contributed by each proposal. The DE schemes that incorporate the continuation scheme are referred to as CDE, whereas in those cases where it was not considered, the term DE is used. Regarding the scheme that promotes the use of large perturbations, the *UpdateDenom* parameter was set to 10 and several HMR ratios were considered. Specifically, the following ratios were tested: 0, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32 and 0.64. In the rest of the paper, the models are referred to as DE-HMR or CDE-HMR. If no value for HMR is given, it means that HMR $= 0$ is assumed. The number of variables D was fixed to 50 in a first set of experiments, while the stopping criterion was set to 150 000 function evaluations. Both DE and CDE were parameterized as in [21]. Specifically, the vector generation strategy used was DE/rand/1/exp, while F and CR were set to 0.5. Finally, NP was set to 15.

Fig. 4 shows the median and mean values obtained with the different schemes at the end of the executions for the set of functions where the new schemes caused a larger effect on the results. Specifically, they show the results for DE and CDE with the different values of HMR. Since in some functions the means and medians obtained were quite different, different axes are used to represent each of them. In most of the problems tested, the positive effect of considering an HMR value different from 0 is clear, both for DE and CDE. The effects caused by the low values of HMR on the mean are more significant than on the median. The reason is that in most of the problems, premature convergence only appears occasionally, so the large perturbations resulting from the use of a non-zero HMR have the main effect of improving the worst-case behavior, which significantly reduces the mean but not the median. For instance, some cases where this effect is clear are F4 and F9, where even large HMR values can be successfully applied. However, in F13 and F15, making large perturbations does not provide benefits, probably because these problems are not as heavily affected by premature convergence. Also worth noting is the fact that there are some problems, e.g., F4 and F17, where the best results are obtained by considering very high HMR values. This indicates that a large loss of diversity is emerging in these problems, so using large perturbations frequently is helpful. However, in most of the test cases, using a too high value for HMR deteriorates the quality of the results because the scheme loses its intensification capabilities. Some cases where this happens are F10, F12, F14 and F15. Note than in these last problems, the medians are much lower than the mean values. This happens in several problems where the eventual executions that suffer from premature convergence govern the resulting mean value. In these cases, using a large HMR does not significantly deteriorate the mean value, at least when compared to the deterioration caused by the use of HMR $= 0$. The reason is that as HMR increases, the likelihood of premature convergence does not increase. However, since many large perturbations are involved, the typical executions are affected, meaning that the medians are more heavily penalized. Considering the overall results, the most promising schemes consider low, but non-zero, values of HMR; specifically, the CDE scheme, where HMR $= 0.04$ seems very promising.

It is also remarkable that in several cases, the differences between the errors obtained by using different HMR values are not too large. This happens because relatively low errors are obtained for several of the functions with low dimensionalities. DE was executed with the previously tested HMR values for the following values of D: 50, 100, 200, 500, 750 and 1000. The stopping criterion was set at $5000D$ in every case. Fig. 5 shows, for the first four problems, the differences between the maximum and minimum mean fitness values obtained by using the different HMR. It is quite clear that as the dimensionality grows, the effect of HMR is larger. However, probably due to the stochastic nature of the schemes, the graphics are not monotonically increasing. For instance, in the case of F3, when $D = 500$ was considered, DE-0 exhibited quite a large mean error because one execution attained a very low quality. Such a high error did not appear when considering larger dimensions, which is the reason why the differences between the maximum and minimum mean values decreased for these larger dimensionalities. However, in general,

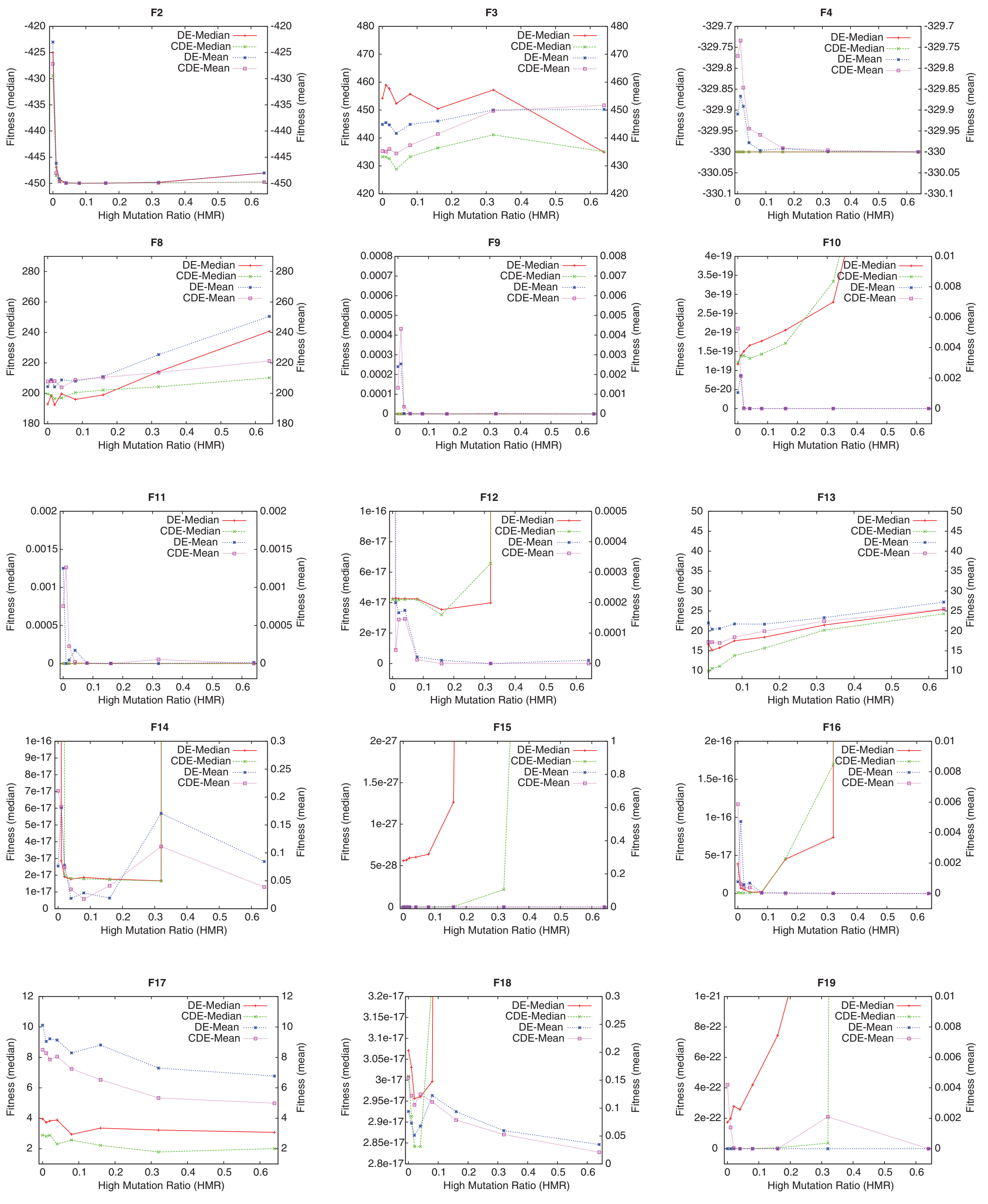

Fig. 4. Mean and median obtained by DE and CDE with different values of HMR in 150 000 function evaluations for several functions.

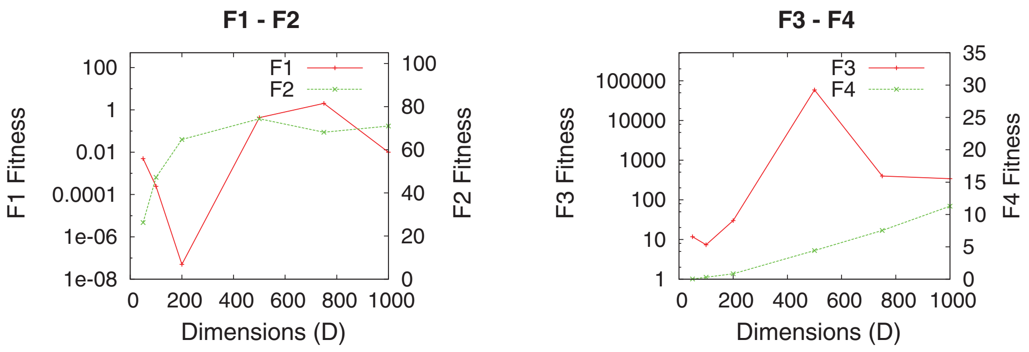

Fig. 5. Differences between the maximum and minimum obtained means with different HMR values for several functions.

Table 1
Statistical comparison of DE vs. CDE in 150.000 evaluations.

	HMR = 0	HMR= 0.01	HMR= 0.02	HMR= 0.04	HMR= 0.08	HMR= 0.16	HMR= 0.32	HMR= 0.64
F1	↔	↔	↔	↔	↔	↔	↔	↔
F2	↑	↑	↑	↑	↓	↓	↑	↑
F3	↑	↑	↑	↑	↑	↑	↔	↔
F4	↓	↓	↓	↓	↓	↔	↔	↔
F5	↔	↔	↔	↔	↔	↔	↔	↔
F6	↑	↑	↔	↔	↔	↔	↔	↔
F7	↔	↔	↔	↔	↔	↔	↑	↔
F8	↔	↔	↔	↔	↔	↔	↑	↑
F9	↑	↔	↔	↔	↑	↑	↑	↔
F10	↔	↔	↑	↑	↑	↑	↓	↓
F11	↔	↔	↔	↔	↑	↑	↔	↔
F12	↑	↑	↑	↑	↑	↑	↓	↑
F13	↑	↑	↑	↑	↑	↑	↔	↑
F14	↓	↔	↔	↔	↔	↔	↔	↑
F15	↑	*	↑	↑	↑	↑	↑	↑
F16	↔	↑	↑	↔	↔	**	↓	↑
F17	↑	↑	↑	↑	↑	↑	↑	↑
F18	↔	↔	↔	↔	**	**	**	↑
F19	*	*	*	↑	↑	↑	*	↑

Table 2
Statistical comparison of DE vs. CDE-0.04.

Median	Mean	75.000 evaluations	150.000 evaluations
↑ =	↑	F2, F3, F6, F7, F13, F14, F14, F15, F17, F19	F2, F3, F6, F9, F11, F12, F13, F14, F15, F16, F17, F19
↓ =	↓	F4	
↑ =	↓		F18
↓ =	↑	F9, F10, F11, F12, F16, F18	F10
↔		F1, F5, F8	F1, F4, F5, F7, F8

larger dimensions imply larger differences. In fact, in every problem the differences that appear for $D = 1000$ are larger than those that appear for $D = 50$. Several aspects regarding the scalability are analyzed further in subsequent sections.

Further statistical analyses were carried out in order to check the benefits of CDE over DE. Specifically, we statistically compared the results obtained by DE and CDE for each value of HMR with $D = 50$. Table 1 shows the results of these statistical tests. From a total set of 152 statistical tests, CDE is better in 65, while it is worse in only 12, demonstrating its overall superiority. The only problem where DE is clearly superior to CDE is F4. In this case, the optimum values are almost equidistributed in the search space. Thus, DE with an F value equal to 0.5 allows for perturbations that jump from one local optimum to another local optimum with a large probability. This property is not shared by CDE, which explains its inferior behavior.

It is also interesting to perform a statistical comparison of CDE-0.04 with the original DE, i.e., DE with HMR = 0. Table 2 shows the results of these comparisons for a short-time period (75 000 function evaluations) and a longer period (150 000 function evaluations). Even when the differences have been significant, there are some cases where similar medians have been obtained. This has occurred when both schemes have reached the optimal values in most cases. In the case of the median, the symbol

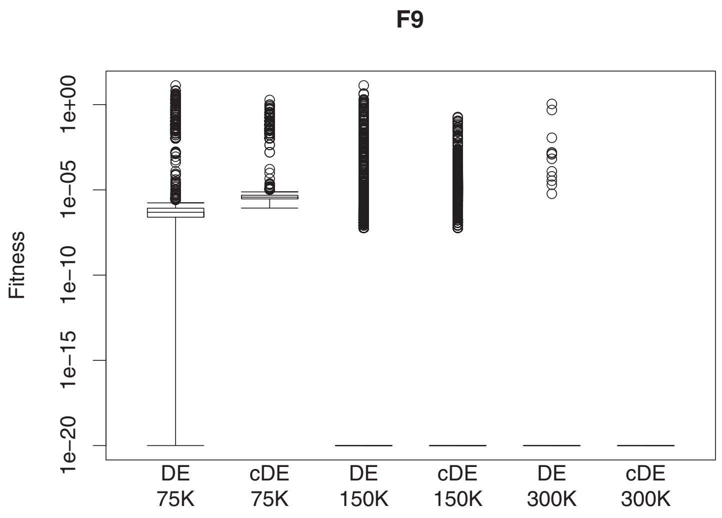

Fig. 6. Boxplots of results obtained by DE and CDE-0.04 for different numbers of function evaluations (F9).

↑= expresses that differences have been significant and that CDE-0.04 obtained a lower or similar median than the original DE. Similarly, ↓= indicates that the original DE obtained a lower or similar median than CDE-0.04. It is interesting to note that:

- In the long term, CDE is not statistically inferior in any of the problems, when considering both the mean and median.
- In the long term, CDE is superior to DE in 12 problems.
- In some of the problems (F4, F7), there are significant differences in the short term but both schemes are similar in the long term. The reason is that both schemes reach optimal values in most of the executions for these problems in the long term.
- In four problems (F9, F11, F12, F16), CDE obtains a lower mean but a higher median in the short term, while in the long term, CDE is a better scheme.

An additional analysis was carried out with the aim of better understanding the last fact described above. This analysis was performed with F9. Specifically, DE and CDE-0.04 were executed considering a stopping criterion of 300 000 function evaluations. This study considered 50 000 executions of each scheme. The reason for using such a large number of executions is that in the long term most of the executions reach optimal values, meaning that in order to also analyze the worst-case behavior, a large number of repetitions is required. Fig. 6 shows the box-plots of the results for different numbers of function evaluations.

In the case where 75 000 function evaluations were considered, DE yielded better results in most executions, hence a lower median was obtained. However, the worst results obtained by DE were worse than those obtained by CDE-0.04, indicating that stagnation or premature convergence might be arising in some cases. Considering the logarithmic scale of the axes, the differences between the worst-case behavior of DE and CDE-0.04 was large. In fact, CDE-0.04 had a lower mean because of its best behavior in the worst cases. After 150 000 function evaluations, both schemes present a similar median because both reached optimal values in most cases. In any case, we can see that the worst results obtained by DE are of very low quality. Finally, in the very long term (300 000 function evaluations), all the executions of CDE-0.04 reached the optimal value, while several DE executions converged prematurely to non-optimal values. Moreover, statistical tests confirm the superiority of CDE-0.04 in the long term.

The previous analyses show the benefits of the new scheme in terms of the quality of the results. However, it is interesting to also analyze the number of function evaluations required to obtain high quality results. In order to carry out this kind of analysis, the first step is to set the quality level (Q) that must be reached. To do this, DE and CDE-0.04 were executed with the stopping criterion fixed at 250 000 function evaluations. Q was set as the higher of the two resulting medians. Then, the number of evaluations required to obtain a success ratio equal to 50% was calculated. Table 3 shows the number of evaluations required by each model, as well as the percentage of evaluations that were saved by using CDE-0.04. Negative values indicate that CDE required a larger number of function evaluations. In those cases where CDE required more function evaluations, the penalty was not very large. However, in several cases the number of function evaluations saved using CDE was very large. For instance, there were nine cases where the number of function evaluations saved by CDE was larger than 20%. Moreover, the mean of the percentages of saved evaluations was 20.2%, demonstrating once again the benefits of the new scheme.

5.2. Second set of experiments: Population size

The likelihood of having stagnation or premature convergence is highly dependent on the population size that we adopt. Specifically, when large population sizes are considered, the appearance of premature convergence and stagnation is less likely. Increasing the population size has the effect of reducing the convergence velocity. As a result, large population sizes are not

Table 3
Evaluations required by DE and CDE-0.04 to obtain a fixed quality level.

	DE	CDE-0.04	Saved (%)
F1	35 000	37 000	−5.40%
F2	248 000	29 000	88.30%
F3	249 000	128 000	48.59%
F4	88 000	90 000	−2.22%
F5	43 000	45 000	−4.44%
F6	217 000	170 000	21.65%
F7	63 000	71 000	−11.26%
F8	250 000	250 000	0%
F9	78 000	86 000	−9.30%
F10	232 000	239 000	−2.92%
F11	79 000	85 000	−7.05%
F12	245 000	250 000	−2%
F13	248 000	152 000	38.70%
F14	249 000	241 000	3.21%
F15	249 000	72 000	71.08%
F16	245 000	145 000	40.81%
F17	247 000	141 000	42.91%
F18	250 000	197 000	21.2%
F19	249 000	115 000	53.81%

Table 4
Comparison of errors obtained by DE and CDE-0.04 with NP = 15 in 150.000 evaluations.

	DE			CDE-0.04		
	Median	Mean	Standard deviation	Median	Mean	Standard deviation
F1	0	5.16×10^{-3}	0.16	0	1×10^{-14}	3.21×10^{-13}
F2	24.96	26.99	15.60	**3.98×10^{-2}**	**4.51×10^{-2}**	**2.28×10^{-2}**
F3	64.22	54.86	57.16	**38.80**	**44.43**	**34.49**
F4	0	9.05×10^{-2}	0.62	0	5.57×10^{-2}	0.47
F5	0	4.51×10^{-5}	9.31×10^{-4}	0	2.62×10^{-5}	4.41×10^{-4}
F6	8.52×10^{-14}	1×10^{-2}	0.20	**8.52×10^{-14}**	**8.52×10^{-14}**	**1.69×10^{-14}**
F7	0	0	0	0	0	0
F8	192.90	204.38	71.72	196.89	203.90	68.01
F9	0	2.39×10^{-3}	7.45×10^{-2}	**0**	**1.79×10^{-5}**	**3.69×10^{-4}**
F10	1.16×10^{-19}	1.04×10^{-3}	3.31×10^{-2}	1.31×10^{-19}	1.65×10^{-19}	1.22×10^{-19}
F11	0	1.25×10^{-3}	3.84×10^{-2}	**0**	**1.99×10^{-5}**	**3.97×10^{-4}**
F12	4.25×10^{-17}	1.08×10^{-2}	0.33	**4.20×10^{-17}**	**1.46×10^{-4}**	**2.32×10^{-3}**
F13	14.14	20.99	23.49	**11.11**	**16.97**	**20.37**
F14	5.34×10^{-15}	7.64×10^{-2}	0.50	**1.81×10^{-17}**	**3.46×10^{-2}**	**0.32**
F15	5.58×10^{-28}	1.70×10^{-13}	5.39×10^{-12}	**5.91×10^{-31}**	**2.47×10^{-29}**	**3.60×10^{-28}**
F16	3.88×10^{-17}	7.71×10^{-4}	8.78×10^{-3}	**1.01×10^{-18}**	**3.92×10^{-4}**	**2.95×10^{-3}**
F17	3.96	10.11	18.05	**2.30**	**8.04**	**19.21**
F18	3.07×10^{-17}	9.39×10^{-2}	0.44	2.84×10^{-17}	0.12	0.33
F19	1.73×10^{-22}	1.60×10^{-14}	4.68×10^{-13}	**1.39×10^{-24}**	**1×10^{-19}**	**5.82×10^{-19}**

commonly used when dealing with large-scale problems. The aim of this experiment is to study the effects of the population size on the new proposal. DE and CDE-0.04 were executed considering the same parameterization as in previous experiments, but with NP = 15 and NP = 50. The stopping criterion was set at 150 000 function evaluations.

Table 4 shows the mean, median and standard deviation obtained for NP = 15. In those problems where one of the schemes was statistically superior to the other, data are shown in **boldface**. There are 12 problems where CDE-0.04 was superior, while DE was not superior in any problem. A similar analysis is presented in Table 5 for NP= 50. In this case, DE is clearly superior to CDE-0.04, showing superior statistical behavior in 12 problems. This means that, when considering a more explorative configuration, further increasing the diversity of the potential trials and promoting large perturbations is counterproductive. The reason might be the loss of intensification capabilities produced by the modifications proposed herein. In any event, when comparing the results obtained with NP = 15 and NP = 50, the superiority of using smaller population sizes is clear. In fact, the statistical tests comparing CDE-0.04 with NP = 15 against DE with NP = 50 show that CDE-0.04 is superior in 13 problems, and inferior in only 2 problems.

Since the use of different population sizes induces distinct convergence velocities, it is also interesting to study the evolution of the fitness values during the executions. Fig. 7 shows the evolution of the mean of the fitness for DE and CDE-0.04 with the aforementioned population sizes. They are shown for four selected problems that are representative of the different behaviors arising from this set of benchmarks. First, it is important to remark that in the short term, the advantages of using small

Table 5
Comparison of errors obtained by DE and CDE-0.04 with NP = 50 in 150.000 evaluations.

	DE			CDE-0.04		
	Median	Mean	Standard deviation	Median	Mean	Standard deviation
F1	0	0	0	0	0	0
F2	**6.28**	**6.30**	**0.54**	6.98	6.96	0.56
F3	**70.34**	**72.96**	**22.46**	76.38	75.47	22.85
F4	0	0	0	0	9.94×10^{-4}	3.14×10^{-2}
F5	0	3.54×10^{-10}	5.65×10^{-9}	0	2.88×10^{-10}	4.34×10^{-9}
F6	$\mathbf{1.17 \times 10^{-10}}$	$\mathbf{1.19 \times 10^{-10}}$	$\mathbf{2.39 \times 10^{-11}}$	3.14×10^{-10}	3.17×10^{-10}	6.19×10^{-11}
F7	$\mathbf{2.21 \times 10^{-11}}$	$\mathbf{2.23 \times 10^{-11}}$	$\mathbf{3.85 \times 10^{-12}}$	6.22×10^{-11}	6.32×10^{-11}	1.06×10^{-11}
F8	4714.86	4708.31	709.18	**4659.60**	**4643.85**	**690.13**
F9	$\mathbf{2.87 \times 10^{-2}}$	$\mathbf{2.87 \times 10^{-2}}$	$\mathbf{2.80 \times 10^{-3}}$	3.77×10^{-2}	3.81×10^{-2}	3.99×10^{-3}
F10	2.59×10^{-18}	3.12×10^{-18}	2.22×10^{-18}	2.57×10^{-18}	3.03×10^{-18}	2.11×10^{-18}
F11	$\mathbf{2.98 \times 10^{-2}}$	$\mathbf{2.99 \times 10^{-2}}$	$\mathbf{3.14 \times 10^{-3}}$	3.99×10^{-2}	4.07×10^{-2}	5.21×10^{-3}
F12	$\mathbf{1.06 \times 10^{-3}}$	$\mathbf{1.19 \times 10^{-3}}$	$\mathbf{5.69 \times 10^{-4}}$	2.18×10^{-3}	2.37×10^{-3}	9.99×10^{-4}
F13	25.36	25.56	8.14	25.70	26.48	9.26
F14	$\mathbf{1.17 \times 10^{-4}}$	$\mathbf{1.28 \times 10^{-4}}$	$\mathbf{5.74 \times 10^{-5}}$	2.10×10^{-4}	2.37×10^{-4}	1.13×10^{-4}
F15	$\mathbf{9.21 \times 10^{-12}}$	$\mathbf{9.45 \times 10^{-12}}$	$\mathbf{2.34 \times 10^{-12}}$	2.38×10^{-11}	2.41×10^{-11}	5.48×10^{-12}
F16	$\mathbf{3.34 \times 10^{-3}}$	$\mathbf{3.48 \times 10^{-3}}$	$\mathbf{1.08 \times 10^{-3}}$	5.82×10^{-3}	5.96×10^{-3}	1.59×10^{-3}
F17	4.97	6.61	5.33	4.93	6.52	5.48
F18	$\mathbf{1.55 \times 10^{-3}}$	$\mathbf{2.59 \times 10^{-3}}$	$\mathbf{3.14 \times 10^{-2}}$	2.61×10^{-3}	2.70×10^{-3}	5.94×10^{-4}
F19	$\mathbf{3.82 \times 10^{-14}}$	$\mathbf{4.10 \times 10^{-14}}$	$\mathbf{1.54 \times 10^{-14}}$	1.47×10^{-13}	1.58×10^{-13}	5.86×10^{-14}

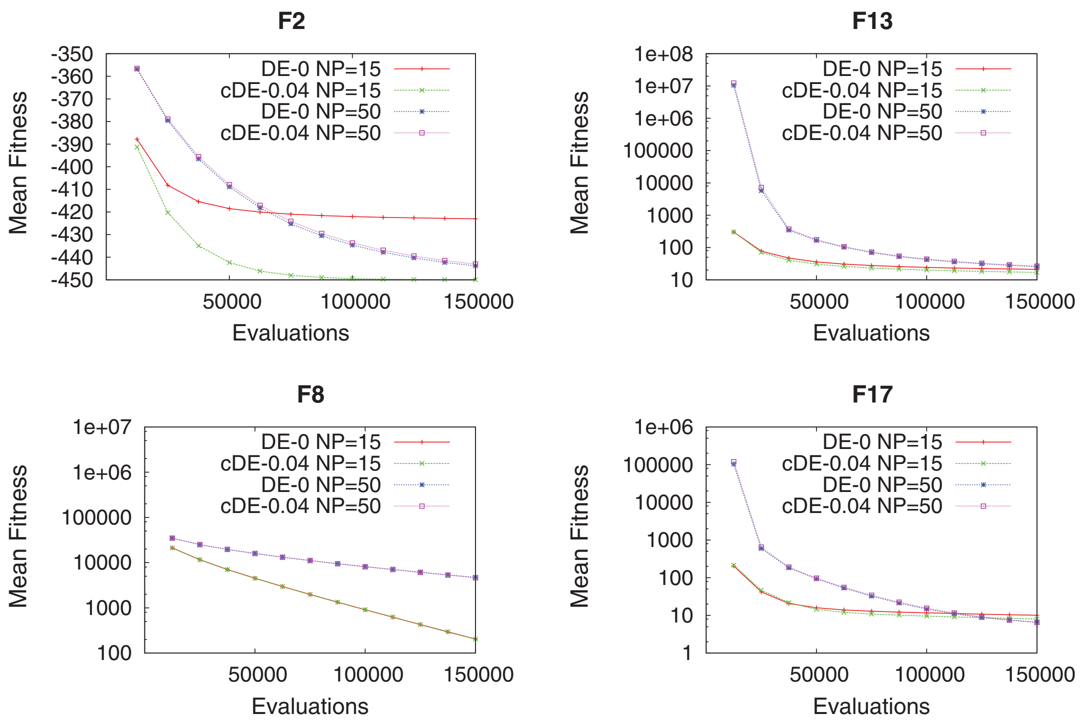

Fig. 7. Evolution of the mean obtained by DE and CDE-0.04 with different population sizes.

population sizes are quite clear, which is a known fact in the field of DE. In fact, in the short term, using NP = 15 is preferable in every problem. However, in the long term, different behaviors appear. In the case of F2, the advantages of using CDE-0.04 with NP = 15 are clear. The low population sizes coupled with the diversity controlling mechanisms designed herein yield high-quality results quickly and robustly. Note that when the original DE is used with NP = 15, fast convergence results in premature convergence, meaning that over the long term, the schemes that consider NP = 50 obtain better results. A somewhat similar situation arises in F13. In this case, CDE-0.04 is the best model, but DE with NP = 15 is also better than the schemes with NP = 50. Problem F8 is a typical case where maintaining a high diversity is not required. In fact, by using an adaptive local search, high-quality results can be obtained in this problem [21]. For this reason, the superiority of the schemes that use a low population size is clear. Note that the inclusion of the diversity controlling mechanisms does not deteriorate performance. Finally, in problem

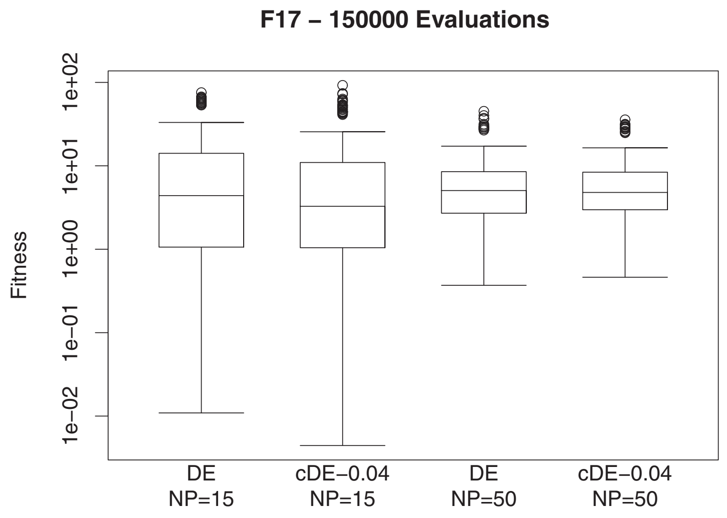

Fig. 8. Boxplots of results obtained by DE and CDE-0.04 with different population sizes (F17).

F17, CDE-0.04 with NP = 15 yields better results than DE with NP = 15. However, the schemes that use NP = 50 provide better results in the long term, meaning that the diversity induced by the increase in the population size is more useful.

In order to better illustrate the differences between the schemes with F17, Fig. 8 shows the boxplots obtained at 150 000 function evaluations. The schemes that consider lower population sizes were able to obtain the best solutions due to the fast convergence, but they also reported the worst solutions. Thus, the preferred population size depends on the risk that can be assumed, the number of repetitions and the stopping criterion established. Note also that using CDE-0.04 with NP = 15 provides some benefits over using DE with NP = 15. Specifically, the minimum, median and first and third quartiles are improved. The amount of improvement is not very large, but the way in which it is obtained is robust. In fact, the statistical tests indicate that the differences are significant.

5.3. *Third set of experiments: Adapting* F

Since the schemes that consider a non-static *F* value have the effect of increasing the number of potential trial vectors created by the vector generation strategies, it is very interesting to compare the new proposal against some of the most popular schemes that consider a variable *F* value. Three different adaptive models were considered. The first one belongs to the group of randomized *F* values. Specifically, we considered the random distribution applied in SaDE [38], i.e., a Gaussian distribution with mean 0.5 and standard deviation 0.3. The DE parameters were fixed as in previous experiments, with NP = 15. In addition, two adaptive schemes that rely on the feedback obtained during the execution were considered. They were the adaptive schemes incorporated in jDE [6] and JADE [55]. In both cases, they were parameterized using the values proposed by their authors. Specifically, in jDE the value τ was set to 0.1, while F_{min} and F_{max} were set to 0.1 and 0.9, respectively. In JADE the parameter *c* was set to 0.1.

Table 6 shows the mean and median obtained by each of the models for the different benchmark functions. In addition, the results of the statistical tests when comparing each model with CDE-0.04 are shown. The comparisons reveal that CDE-0.04 is statistically better than the schemes that employ the adaptation of SaDE and jDE in 10 problems, and worse in one problem. They also show that CDE-0.04 is statistically better than DE with the JADE adaptation scheme in 11 problems and worse in only one problem. The above results show that the way of increasing diversity proposed here is preferable to the methods that adapt the *F* value during the run. Moreover, it also calls into question the robustness of those adaptive schemes that consider feedback to set the value of *F*.

5.4. *Fourth set of experiments: Scalability*

It is also very interesting to check the relationship between the behavior of the new proposal and the dimensionality of the problem. It was shown that as the number of dimensions grows, the number of potential trials also grows. However, while the growth in the search space is exponential, the growth in the potential trials is only polynomial when exp crossover is considered. Thus, it is not easy to predict the overall effect.

Regarding the parameterization of DE, note that according to [57], the optimal CR value usually depends on the dimensionality. In order to avoid having to tune CR for each dimension, the techniques proposed in that paper might be used. However, for the set of problems considered, a fixed CR value equal to 0.5 allows obtaining competitive results for the dimensions considered in this paper [21]. Thus, in order to keep our approach as simple as possible, DE and CDE-0.04 were executed with the same parameterization as in the first set of experiments, but considering several dimensionalities. Specifically, the following values

Table 6
Results obtained with DE by adapting *F* with different schemes (150.000 evaluations).

	SaDE			jDE			JADE		
	Median	Mean	Statistical	Median	Mean	Statistical	Median	Mean	Statistical
F1	0	0	$\leftrightarrow$	0	0	$\leftrightarrow$	0	0	$\leftrightarrow$
F2	1.65	3.88	$\uparrow$	14.49	17.26	$\uparrow$	0.35	0.35	$\uparrow$
F3	28.92	40.61	$\downarrow$	40.53	45.69	$\leftrightarrow$	53.54	58.73	$\uparrow$
F4	0	0.28	$\uparrow$	0	0.45	$\uparrow$	0	9.94×10^{-4}	$\downarrow$
F5	0	4.45×10^{-4}	$\uparrow$	0	2.60×10^{-4}	$\uparrow$	0	2.95×10^{-5}	$\leftrightarrow$
F6	8.52×10^{-14}	8.52×10^{-14}	$\leftrightarrow$	5.68×10^{-14}	4.95×10^{-4}	$\uparrow$	8.52×10^{-14}	1.13×10^{-13}	$\uparrow$
F7	0	0	$\leftrightarrow$	0	0	$\leftrightarrow$	0	3.33×10^{-19}	$\leftrightarrow$
F8	206.31	216.90	$\uparrow$	175.50	184.54	$\downarrow$	526.91	544.28	$\uparrow$
F9	0	1.50×10^{-2}	$\uparrow$	0	9.27×10^{-3}	$\uparrow$	0	2.55×10^{-4}	$\uparrow$
F10	4.65×10^{-20}	3.82×10^{-2}	$**$	4.96×10^{-20}	5.25×10^{-2}	$**$	2.68×10^{-19}	1.04×10^{-3}	$\uparrow$
F11	0	1.61×10^{-3}	$\uparrow$	0	5.03×10^{-3}	$\uparrow$	0	3.35×10^{-4}	$\uparrow$
F12	4.17×10^{-17}	1.91×10^{-4}	$\leftrightarrow$	4.15×10^{-17}	7.90×10^{-3}	$\leftrightarrow$	1.62×10^{-16}	1.44×10^{-4}	$*$
F13	4.05	16.49	$\downarrow$	4.10	18.29	$**$	20.77	22.07	$\uparrow$
F14	1.78×10^{-15}	0.25	$\uparrow$	2.44×10^{-17}	0.28	$\uparrow$	2.58×10^{-17}	0.19	$\uparrow$
F15	4.89×10^{-27}	8.23×10^{-3}	$\uparrow$	5.21×10^{-27}	4.61×10^{-3}	$\uparrow$	8.64×10^{-25}	2.22×10^{-19}	$\uparrow$
F16	4.26×10^{-17}	9.29×10^{-4}	$\leftrightarrow$	1.62×10^{-18}	7.71×10^{-3}	$\leftrightarrow$	3.92×10^{-16}	2.37×10^{-4}	$*$
F17	4.69	11.57	$\uparrow$	6.58	13.49	$\uparrow$	2.48	5.95	$\leftrightarrow$
F18	3.37×10^{-17}	4.28×10^{-2}	$**$	3.23×10^{-17}	0.10	$*$	6.73×10^{-14}	6.21×10^{-2}	$*$
F19	4.03×10^{-23}	2.92×10^{-2}	$\uparrow$	2.62×10^{-23}	3.54×10^{-2}	$\uparrow$	4.26×10^{-21}	3.71×10^{-5}	$\uparrow$

Table 7
Comparison of errors obtained by DE and CDE-0.04 in 1,000,000 evaluations ($D = 200$).

	DE			CDE-0.04			
	Median	Mean	Standard deviation	Median	Mean	Standard deviation	Statistical
F1	0	0	1.79×10^{-14}	0	0	0	$\leftrightarrow$
F2	65.83	65.89	17.21	**1.43**	**1.46**	**0.19**	$\uparrow$
F3	243.92	243.43	59.87	**230.86**	**227.26**	**53.34**	$\uparrow$
F4	0	8.13×10^{-2}	0.71	0	4.34×10^{-2}	0.51	$\leftrightarrow$
F5	0	3.23×10^{-4}	5.38×10^{-3}	0	8.74×10^{-5}	2.04×10^{-3}	$\leftrightarrow$
F6	1.98×10^{-13}	5.62×10^{-3}	4.14×10^{-2}	$\mathbf{1.98 \times 10^{-13}}$	$\mathbf{1.98 \times 10^{-13}}$	$\mathbf{2.18 \times 10^{-14}}$	$\uparrow$
F7	0	0	0	0	0	0	$\leftrightarrow$
F8	4381.43	4412.40	609.19	4404.58	4423.31	617.84	$\leftrightarrow$
F9	0	0	0	0	1.69×10^{-10}	5.36×10^{-9}	$\leftrightarrow$
F10	2.69×10^{-20}	3.12×10^{-20}	4.51×10^{-20}	$\mathbf{2.05 \times 10^{-20}}$	$\mathbf{2.14 \times 10^{-20}}$	$\mathbf{1.03 \times 10^{-20}}$	$\uparrow$
F11	0	0	0	0	2.31×10^{-10}	7.32×10^{-9}	$\leftrightarrow$
F12	8.33×10^{-17}	1.81×10^{-2}	0.56	$\mathbf{8.32 \times 10^{-17}}$	$\mathbf{5.45 \times 10^{-5}}$	$\mathbf{1.07 \times 10^{-3}}$	$\uparrow$
F13	147.99	149.34	63.67	**129.86**	**129.89**	**48.01**	$\uparrow$
F14	1.03×10^{-17}	9.00×10^{-2}	0.56	9.17×10^{-18}	0.11	0.59	$*$
F15	9.94×10^{-30}	1.92×10^{-7}	6.09×10^{-6}	**0**	$\mathbf{1.48 \times 10^{-31}}$	$\mathbf{4.51 \times 10^{-31}}$	$\uparrow$
F16	5.56×10^{-17}	3.52×10^{-4}	8.23×10^{-3}	5.62×10^{-17}	2.40×10^{-5}	4.76×10^{-4}	$\leftrightarrow$
F17	6.07	16.29	23.80	**3.80**	**10.33**	**17.85**	$\uparrow$
F18	6.46×10^{-17}	5.96×10^{-2}	0.28	$\mathbf{5.88 \times 10^{-17}}$	$\mathbf{5.89 \times 10^{-2}}$	**0.30**	$\uparrow$
F19	1.31×10^{-24}	3.81×10^{-3}	0.12	$\mathbf{3.47 \times 10^{-27}}$	$\mathbf{1.23 \times 10^{-26}}$	$\mathbf{4.37 \times 10^{-26}}$	$\uparrow$

of D were considered: 200, 500 and 1000. The stopping criterion was set at $5000D$ in every case. Tables 7–9 show the mean, median and standard deviation of both models for each of the dimensionalities tested. The results of the statistical tests are also shown. These show that both the median and mean errors increase as higher dimensionalities are used. This was expected because while the search space size grows exponentially, the number of function evaluations increases only linearly. In any event, the advantages of CDE-0.04 remain clear and are even greater than for lower dimensionalities. For instance, while for $D = 200$, CDE-0.04 is statistically better than DE in 10 problems, for $D = 1000$ the number of problems where CDE-0.04 is better increases to 14. It is also interesting to note that for the total number of statistical tests conducted in the scalability analysis, CDE-0.04 was superior to DE in 38 out of the 57 cases, being inferior in only one case. This indicates a clear superiority of the CDE-0.04 scheme.

It is also remarkable that when dealing with the largest search spaces, there are several cases where DE yields very high mean values, while the behavior of CDE-0.04 does not degrade too much. The reason is that the worst-case behavior of DE is much worse than the worst-case behavior of CDE-0.04. Some of the problems where this happens are F3, F15 and F17 with $D = 500$, and F6, F13 and F15 with $D = 1000$.

Table 8
Comparison of errors obtained by DE and CDE-0.04 in 2,500,000 evaluations ($D = 500$).

	DE			CDE-0.04			
	Median	Mean	Standard deviation	Median	Mean	Standard deviation	Statistical
F1	0	0.43	13.71	**0**	**0**	**0**	↑
F2	90.72	89.82	16.67	**17.07**	**17.10**	**0.75**	↑
F3	658.36	59373.95	1330246	**637.85**	**635.42**	**78.47**	↑
F4	0	0.75	2.92	**0**	**0.22**	**1.42**	↑
F5	0	1.34×10^{-2}	0.25	**0**	**1.23×10^{-5}**	**3.89×10^{-4}**	↑
F6	4.54×10^{-13}	6.61×10^{-3}	2.71×10^{-2}	**4.54×10^{-13}**	**4.54×10^{-13}**	**2.42×10^{-14}**	↑
F7	0	7.56×10^{-9}	2.39×10^{-7}	0	0	0	↔
F8	53737.73	53724.33	3928.97	53335.88	53434.75	3939.66	↔
F9	0	2.39×10^{-4}	6.25×10^{-3}	0	2.21×10^{-5}	4.94×10^{-4}	↔
F10	2.96×10^{-20}	1.10×10^{-2}	0.34	**2.42×10^{-20}**	**1.04×10^{-3}**	**3.31×10^{-2}**	↑
F11	0	5.82×10^{-5}	9.42×10^{-4}	0	2.70×10^{-4}	8.09×10^{-3}	↔
F12	1.66×10^{-16}	0.15	3.59	**1.38×10^{-16}**	**3.90×10^{-4}**	**2.17×10^{-3}**	↑
F13	429.37	440.70	274.80	**405.87**	**404.27**	**82.05**	↑
F14	1.80×10^{-17}	0.39	1.63	**1.64×10^{-17}**	**0.30**	**1.17**	↑
F15	1.95×10^{-29}	3.20×10^{-3}	0.10	**0**	**1.34×10^{-31}**	**4.24×10^{-31}**	↑
F16	1.11×10^{-16}	8.57×10^{-3}	0.23	**1.11×10^{-16}**	**4.08×10^{-4}**	**1.98×10^{-3}**	↑
F17	73.64	294.66	5220.16	**70.60**	**54.09**	**44.35**	↑
F18	3.37×10^{-16}	0.15	0.64	1.88×10^{-16}	0.18	0.74	↔
F19	1.24×10^{-24}	1.05×10^{-3}	3.31×10^{-2}	**3.41×10^{-27}**	**2.54×10^{-22}**	**8.03×10^{-21}**	↑

Table 9
Comparison of errors obtained by DE and CDE-0.04 in 5,000,000 evaluations ($D = 1000$).

	DE			CDE-0.04			
	Median	Mean	Standard deviation	Median	Mean	Standard deviation	Statistical
F1	0	1.38×10^{-5}	4.39×10^{-4}	**0**	**0**	**0**	↑
F2	104.10	102.90	17.89	**44.98**	**44.94**	**1.04**	↑
F3	1380.32	1381.01	109.88	**1353.80**	**1347.02**	**113.79**	↑
F4	0	2.16	6.59	**0**	**0.59**	**3.01**	↑
F5	0	1.40×10^{-2}	0.28	**0**	**6.01×10^{-4}**	**1.32×10^{-2}**	↑
F6	8.81×10^{-13}	9.96×10^{-3}	3.78×10^{-2}	9.09×10^{-13}	8.27×10^{-12}	9.88×10^{-11}	∗∗
F7	0	5.97×10^{-6}	1.88×10^{-4}	**0**	**0**	**0**	↑
F8	281066.30	280617.10	16914.34	281155.30	280555.05	17147.64	↔
F9	0	4.27×10^{-4}	7.70×10^{-3}	0	2.46×10^{-5}	4.64×10^{-4}	↔
F10	3.35×10^{-20}	8.28×10^{-7}	1.86×10^{-5}	**3.33×10^{-20}**	**3.07×10^{-20}**	**3.67×10^{-20}**	↑
F11	0	2.32×10^{-5}	5.21×10^{-4}	**0**	**2.81×10^{-7}**	**8.27×10^{-6}**	↑
F12	3.05×10^{-16}	0.46	12.97	**3.05×10^{-16}**	**7.70×10^{-4}**	**3.66×10^{-3}**	↑
F13	945.00	13076.96	383406.5	**905.31**	**899.26**	**100.83**	↑
F14	4.38×10^{-17}	1.39	3.88	4.95×10^{-17}	0.88	2.60	∗∗
F15	2.55×10^{-29}	5.14×10^{-2}	1.62	**0**	**8.89×10^{-32}**	**1.66×10^{-30}**	↑
F16	2.22×10^{-16}	0.14	1.74	**1.94×10^{-16}**	**1.21×10^{-3}**	**3.42×10^{-3}**	↑
F17	201.52	201.75	293.60	**165.18**	**175.80**	**71.21**	↑
F18	**1.14×10^{-13}**	**0.27**	**1.01**	5.61×10^{-12}	0.65	1.40	↓
F19	1.35×10^{-24}	8.31×10^{-2}	1.11	**3.11×10^{-27}**	**2.10×10^{-3}**	**4.69×10^{-2}**	↑

5.5. *Fifth set of experiments: Other DE variants*

The above experiments have shown the advantages of incorporating our proposals into a basic variant of DE. Since there are several non-hybrid DE variants that have been successfully used to improve on the results obtained with DE in the SOCO tests, analyzing the combination of these DE variants with the new proposals defined in this paper is quite interesting. GADE [52], GODE [47] and jDElscop [7] are three non-hybrid DE variants that have yielded promising results by following significantly different principles. For this reason, they were selected for this study. In every case the integration was done in a similar way. Specifically, when the value L generated by the crossover is equal to one, instead of using the original way to create the trial vector, our proposal was applied. As in previous experiments, the parameter HMR was set to 0.04 and *UpdateDenom* to 10. The remaining details and parameterization of the schemes were as proposed by their authors.

The GADE scheme [52] is based on adapting the mutation strategy, as well as the F and CR parameters. Among the mutation strategies, the greedy DE/current-to-pbest/2 is taken into account, meaning that more intensification is induced. As a result, larger population sizes might be preferred. In fact, in [52], the population size was set to 60. Since the incorporation of our schemes shifts this balance toward exploration, GADE and the new proposal (CGADE-0.04) were tested with two different population sizes (NP = 15 and NP = 60). Table 10 shows the mean and median obtained for each problem. In addition, statistical tests

Table 10
Comparison of errors obtained by GADE and CGADE-0.04 in 5,000,000 evaluations ($D = 1000$).

	NP = 15					NP = 60				
	GADE		CGADE-0.04			GADE		CGADE-0.04		
	Median	Mean	Median	Mean	Statistical	Median	Mean	Median	Mean	Statistical
F1	0	0	0	0	↔	0	0	0	0	↔
F2	39.6	43.8	**10.6**	**11.6**	↑	88.5	86.3	**33.4**	**34.9**	↑
F3	800.0	791.9	**773.2**	**769.0**	↑	945.5	946.4	941.0	966.5	↔
F4	1.9	2.8	1.9	2.4	↔	0	0	0	0	↔
F5	0	2.7×10^{-2}	0	2.9×10^{-2}	↔	0	1.3×10^{-4}	0	2.7×10^{-4}	↔
F6	1.3×10^{-13}	2.3×10^{-13}	1.3×10^{-13}	1.7×10^{-13}	↔	1.8×10^{-14}	1.4×10^{-11}	$\mathbf{1.8 \times 10^{-14}}$	$\mathbf{2.3 \times 10^{-14}}$	↑
F7	0	7.9×10^{-17}	0	0	↔	0	0	0	0	↔
F8	17421.3	17651.7	**11374.6**	**11498.0**	↑	17242.3	17670.5	**15254.1**	**15316.7**	↑
F9	8.3×10^{-2}	1.0×10^{-1}	$\mathbf{3.0 \times 10^{-2}}$	$\mathbf{3.7 \times 10^{-2}}$	↑	**0**	$\mathbf{7.5 \times 10^{-5}}$	0	1.1×10^{-4}	↓
F10	20.9	21.3	**1.0**	$\mathbf{7.9 \times 10^{-1}}$	↑	0	3.9×10^{-1}	**0**	$\mathbf{4.3 \times 10^{-2}}$	↑
F11	8.3×10^{-2}	9.5×10^{-2}	$\mathbf{3.0 \times 10^{-2}}$	$\mathbf{3.7 \times 10^{-2}}$	↑	**0**	$\mathbf{6.4 \times 10^{-5}}$	0	2.5×10^{-4}	↓
F12	10.5	14.8	**0**	$\mathbf{3.5 \times 10^{-2}}$	↑	3.7×10^{-12}	3.4×10^{-2}	1.3×10^{-11}	1.5×10^{-11}	**
F13	717.2	712.5	**638.2**	**629.2**	↑	715.1	723.6	714.7	722.5	↔
F14	1.9	2.2	$\mathbf{9.9 \times 10^{-1}}$	**1.42**	↑	8.4×10^{-11}	19.33	1.6×10^{-10}	1.8×10^{-10}	**
F15	**3.1**	**3.0**	3.1	4.0	↓	**0**	$\mathbf{1.3 \times 10^{-2}}$	0	3.6×10^{-2}	↓
F16	14.4	18.2	$\mathbf{2.1 \times 10^{-2}}$	$\mathbf{4.1 \times 10^{-2}}$	↑	2.4×10^{-12}	1.1×10^{-5}	5.1×10^{-12}	5.2×10^{-12}	**
F17	239.5	244.7	**227.0**	**222.6**	↑	218.1	220.9	219	220.5	**
F18	1.0	9.07×10^{-1}	$\mathbf{4.2 \times 10^{-2}}$	$\mathbf{3.3 \times 10^{-1}}$	↑	1.1×10^{-7}	2.2×10^{-1}	3.1×10^{-7}	6.4×10^{-5}	**
F19	7.3	7.8	**0**	$\mathbf{7.8 \times 10^{-1}}$	↑	0	2.8×10^{-1}	**0**	$\mathbf{8.7 \times 10^{-2}}$	↑

were used to compare GADE and CGADE-0.04 for each NP value. As in previous cases, the symbol ↑ is used to denote the superiority of the scheme that incorporates our proposals, i.e. CGADE-0.04. Note that when NP is set to 15, the superiority of CGADE-0.04 is clear. Since using such a low population size induces a larger degree of exploitation, increasing the diversity of trial vectors with the new proposals provides significant benefits. Moreover, even when NP is set to 60, the proposed schemes provide significant improvements in several cases. However, for some problems using NP = 60 and increasing the diversity produces too much exploration, causing degradation to appear. This happens in problems where very low errors are obtained. In these cases, in order to refine the solutions, small perturbations are required, so the schemes proposed in this paper delay convergence. In light of the results, it seems clear that the population size can be increased in some cases to yield a better balance between exploration and exploitation. However, in other more complex cases, this is not enough and our proposal should be taken into account. In general, in the case of CGADE-0.04, more diversity is induced, so a shorter population can be used, whereas in the case of GADE, more intensification is promoted, and thus larger populations are admissible.

The GODE scheme [47] combines DE with generalized opposition-based learning (GOBL). Similarly to GADE, this scheme balances the search toward exploitation. The reason is that when GOBL is used, the replacement strategy is the replace-worst [14], meaning that diversity can be highly reduced. In fact, in [47] the population size was also set to 60. This is why, as in the previous case, GODE and the new proposal (CGODE-0.04) were tested with two different population sizes (NP = 15 and NP = 60). Table 11 shows the mean, median and the result of the statistical tests obtained for each problem. In this case, the combination of the replace-worst strategy with low population sizes produces a significant degradation in several cases. This can be partially fixed by the use of CGODE-0.04. Moreover, when taking into account larger population sizes, CGODE-0.04 is also useful. In fact, the statistical tests reveal that it produces significant benefits in 12 out of the 19 cases, and that in none of them is the degradation significant.

One of the main features of jDElscop [7] is that it uses a variable population size. Specifically, it starts with a large population size and it is reduced during the run. It also incorporates self-adaptation, adaptation and occasionally changes the sign of *F* so as to select improvement directions with a greater likelihood. Table 12 shows the mean, median and the results of the statistical tests obtained for each problem with jDElscop and cjDElscop-0.04. Note that many of the problems which were solved to optimality by other schemes do not reach such low values in this case. In fact, a mean equal to 0 is not obtained in any problem. This is due to the higher degree of exploration induced by the scheme in the initial phases. As a result, this scheme would require more evaluations to converge. In any event, in most of these cases a very low error is obtained, meaning this drawback might be avoided by including a local search. The inclusion of our proposals yields benefits in eight cases. Probably the most impressive ones are those in which the mean is highly reduced. As in the basic DE variant, there are several cases where the eventual executions that suffer from premature convergence govern the resulting mean value. In these cases, our proposals provide significant advantages. This happens for instance in F12 and F16. In some cases, like F16, this is offset through a low increase in the median value. Note that similar effects also appeared in the other DE variants tested. For instance, this happens in GADE with F14 and F18, and in GODE with F12 and F16.

Finally, we would like to remark that the results obtained with these non-hybrid schemes are not as successful as the hybrid approaches that adaptively incorporate different optimization frameworks [21]. For instance, MOS provides a lower median than CGADE-0.04 with NP = 60 in 8 problems, while CGADE-0.04 provides a lower median only in two problems. We also conducted some experiments by including some simple local search mechanisms at the end of the executions. Since in some cases we

Table 11
Comparison of errors obtained by GODE and CGODE-0.04 in 5,000,000 evaluations ($D = 1000$).

	NP = 15					NP = 60				
	GODE		CGODE-0.04			GODE		CGODE-0.04		
	Median	Mean	Median	Mean	Statistical	Median	Mean	Median	Mean	Statistical
F1	10132.8	10936.1	**3.2×10^{-1}**	**6.5×10^{-1}**	↑	0	0	0	0	↔
F2	99.1	99.0	**96.1**	**96.0**	↑	92.3	91.8	**85.1**	**84.8**	↑
F3	1.8×10^{9}	2.1×10^{9}	**4271.3**	**4627.0**	↑	970.6	970.7	**970.4**	**970.5**	↑
F4	368.2	371.2	**121.7**	**123.1**	↑	9.9×10^{-1}	1.04	**0**	**3.1×10^{-2}**	↑
F5	93.3	97.8	**1.8×10^{-1}**	**2.8×10^{-1}**	↑	0	0	0	0	↔
F6	5.23	5.27	**1.2×10^{-1}**	**1.1×10^{-1}**	↑	3.3×10^{-13}	3.3×10^{-13}	**3.1×10^{-13}**	**3.1×10^{-13}**	↑
F7	13.10	13.74	**1.5×10^{-1}**	**1.6×10^{-1}**	↑	0	0	0	0	↔
F8	16223.4	16406.3	**14029.8**	**14320.1**	↑	189379.5	189146.1	**185877**	**185541.7**	↑
F9	251.6	253.7	**36.2**	**38.2**	↑	1.6×10^{-4}	1.9×10^{-4}	**3.3×10^{-5}**	**3.3×10^{-5}**	↑
F10	497.1	534.2	**9.2**	**9.4**	↑	0	0	0	0	↔
F11	250.8	253.2	**36.4**	**38.9**	↑	1.6×10^{-4}	1.6×10^{-4}	**3.3×10^{-5}**	**3.3×10^{-5}**	↑
F12	6258.5	6924.5	**62.5**	**67.7**	↑	1.8×10^{-9}	3.8×10^{-2}	**2.6×10^{-10}**	**2.7×10^{-10}**	↑
F13	8.6×10^{8}	1.1×10^{9}	**3233.5**	**3676.3**	↑	730.8	732.4	730.3	732.5	*
F14	258.1	261.6	**82.2**	**84.3**	↑	6.4×10^{-1}	7.7×10^{-1}	**1.2×10^{-8}**	**3.1×10^{-2}**	↑
F15	77.4	92.5	**1.2**	**1.3**	↑	0	0	0	0	↔
F16	2075.5	2278.3	**135.4**	**139.8**	↑	4.5×10^{-9}	5.5×10^{-2}	**6.4×10^{-10}**	**6.4×10^{-10}**	↑
F17	2.2×10^{6}	4.5×10^{7}	**956.2**	**1028.0**	↑	236.2	236.4	**235.8**	**235.8**	↑
F18	110.7	112.3	**29.8**	**31.0**	↑	1.0×10^{-5}	1.3×10^{-1}	**2.3×10^{-6}**	**1.2×10^{-2}**	↑
F19	268.2	292.6	**4.7**	**4.8**	↑	0	0	0	0	↔

Table 12
Comparison of errors obtained by jDElscop and cjDElscop-0.04 in 5,000,000 evaluations ($D = 1000$).

	jDElscop		cjDElscop-0.04		
	Median	Mean	Median	Mean	Statistical
F1	4.1×10^{-16}	4.1×10^{-16}	4.1×10^{-16}	4.1×10^{-16}	↔
F2	**24.3**	**24.8**	26.4	27.2	↓
F3	844.4	849.6	844.5	849.5	↔
F4	0	1.9×10^{-1}	**0**	**9.9×10^{-3}**	↑
F5	2.0×10^{-16}	2.1×10^{-16}	2.0×10^{-16}	2.1×10^{-16}	↔
F6	1.1×10^{-12}	1.1×10^{-12}	1.1×10^{-12}	1.1×10^{-12}	↔
F7	6.6×10^{-16}	8.4×10^{-16}	5.5×10^{-16}	8.3×10^{-16}	↔
F8	31145.0	31980.8	30851.8	31660.8	↔
F9	5.9×10^{-8}	8.8×10^{-4}	**0**	**5.4×10^{-4}**	↑
F10	5.5×10^{-30}	2.0×10^{-29}	**2.6×10^{-30}**	**7.0×10^{-30}**	↑
F11	8.4×10^{-8}	9.1×10^{-4}	**0**	**5.9×10^{-4}**	↑
F12	4.1×10^{-16}	2.0×10^{-2}	**4.1×10^{-16}**	**4.0×10^{-16}**	↑
F13	645.7	659.8	643.5	659.2	↔
F14	4.1×10^{-16}	2.1×10^{-1}	4.5×10^{-16}	1.1×10^{-2}	↔
F15	1.4×10^{-31}	4.0×10^{-18}	3.6×10^{-32}	6.0×10^{-18}	↔
F16	6.0×10^{-16}	2.0×10^{-1}	7.6×10^{-16}	1.0×10^{-4}	**
F17	166.3	173.4	**165.5**	**172.3**	↑
F18	2.6×10^{-12}	8.2×10^{-2}	3.7×10^{-12}	4.2×10^{-3}	**
F19	1.3×10^{-30}	2.2×10^{-30}	**6.2×10^{-31}**	**1.5×10^{-30}**	↑

could not improve the results of [21] even when incorporating this post-processing strategy, this probably means that the global search capabilities should be further improved so as to avoid the requirement of integrating different optimization procedures to perform the global search. In any case, our proposals provide an important advancement in the non-hybrid DE schemes and show the potential of schemes that attempt to control the diversification capabilities of DE.

5.6. Sixth set of experiments: Parameterization of the update mechanism

Our initial experimentation showed that by setting *UpdateDenom* to 10, promising results could be obtained for different benchmark problems in several variants of DE. As a result, in previous experiments this value was used. However, it is interesting to analyze the sensitivity of our scheme to this parameter, so experiments with different *UpdateDenom* values were also carried out. Specifically, CDE-0.04, CGODE-0.04, CGADE-0.04 and cjDElscop-0.04 were tested with the following *UpdateDenom* values: 2, 5, 10, 15, 20, 30, 40, 50. Depending on the problem and scheme, different behaviors were detected.

Fig. 9 shows the mean obtained with the different *UpdateDenom* values in four illustrative cases that appeared repeatedly. In each case, the mean obtained by the model that does not incorporate our proposals is also shown. When applying

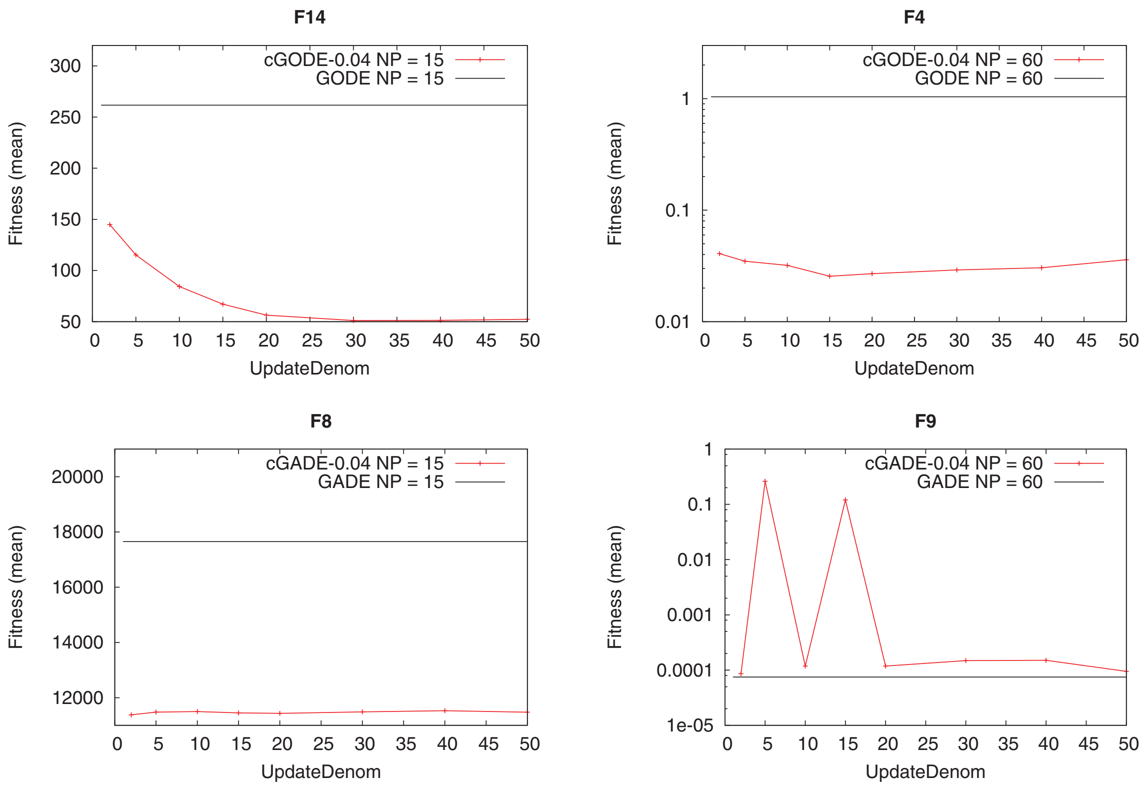

Fig. 9. Mean of the fitness obtained with different *UpdateDenom* values.

cGODE-0.04 with NP = 15 to problem F14, we can see that the largest *UpdateDenom* value yields the lowest mean value. This happens because cGODE with a low population size produces an excessive amount of intensification, so inducing a slow update mechanism results in a useful exploration. In other cases, as expected, it is the intermediate values of *UpdateDenom* that yield a better quality. This happens, for instance, when cGODE with NP = 60 is applied to F4. We also identified some cases where the quality is practically independent of the *UpdateDenom* value. For instance, this arose when applying cGADE with NP = 15 to F8. In this case, the only statistically significant difference appeared when comparing the scheme with $UpdateDenom = 2$ with the one that uses $UpdateDenom = 50$. In cases like this one, most of the benefits come from the continuation scheme and not from the large mutations, so even if the scheme is executed by setting HMR to 0, high-quality values are obtained. In all the previous cases, our proposals provided benefits regardless of the *UpdateDenom* value applied. However, as we have shown, in a minority of cases our proposals do lead to some degradation, as in the case of cGADE with NP = 60 for F9. In these cases, some improvements can be obtained by modifying the *UpdateDenom* value. However, we could not obtain significant improvements when compared to the basic schemes that do not incorporate our proposals. This means that when dealing with new problems, the user can test the schemes with practically any *UpdateDenom* value to check whether our proposals yield any benefits or not. If the scheme provides significant benefits, it might make sense to tune the *UpdateDenom* value. If not, the scheme might already be explorative enough, so testing different *UpdateDenom* values would probably not be productive.

5.7. Seventh set of experiments: Other benchmarks

This last experiment is devoted to demonstrating the generality of the weaknesses of DE that have been analyzed in this paper, and the suitability of our schemes for dealing with other problems of varying complexity. As was explained earlier, the CEC'10 tests were selected for this purpose because the features that characterize them are different from those of the soco tests. For instance, it is known that in the soco tests, "mutating a contiguous sequence of components is somehow more effective" [57], meaning that the exponential crossover is highly suitable. However, in the CEC'10 benchmark, using a combination of bin and exp crossover seems more effective [10,48]. In our scheme, the exp and bin operators considered CR values equal to 0.5 and 0.1, respectively. Several other modifications specifically designed for this benchmark have been devised [5,10]. For instance, in [5] the parameter control strategies were modified by using certain threshold values at specific stages of the optimization process. In addition, ageing, adaptation of the scale factor sign, self-adaptation, local search and several mutation strategies were taken into account. In this experiment, our purpose is merely to show that DE is also affected by the weaknesses analyzed herein when dealing with the CEC'10 benchmark, so we decided to operate with a simple baseline scheme, meaning that these additional

Table 13
Comparison of errors obtained by DE and CDE-0.04 in 3,000,000 evaluations with the CEC'10 benchmark test suite.

	DE			CDE			
	Median	Mean	Standard deviation	Median	Mean	Standard deviation	Statistical
F1	4.17×10^{-17}	2.56×10^{5}	5.39×10^{6}	**6.84×10^{-21}**	**1.33×10^{1}**	**2.55×10^{2}**	↑
F2	1.11×10^{2}	1.11×10^{2}	2.90×10^{1}	**3.48×10^{1}**	**3.53×10^{1}**	**8.40**	↑
F3	1.74×10^{-13}	3.67×10^{-2}	2.10×10^{-1}	**1.03×10^{-13}**	**1.17×10^{-4}**	**7.7×10^{-4}**	↑
F4	4.29×10^{13}	4.30×10^{13}	8.44×10^{12}	4.21×10^{13}	4.25×10^{13}	8.63×10^{12}	↔
F5	1.70×10^{8}	1.70×10^{8}	2.43×10^{7}	1.69×10^{8}	1.69×10^{8}	2.45×10^{7}	↔
F6	1.35×10^{3}	9.84×10^{4}	3.46×10^{5}	**9.52×10^{2}**	**8.58×10^{4}**	**3.38×10^{5}**	↑
F7	1.46×10^{7}	1.80×10^{7}	3.14×10^{7}	**1.42×10^{7}**	**1.71×10^{7}**	**1.22×10^{7}**	↑
F8	4.85×10^{7}	1.33×10^{11}	1.92×10^{12}	**4.68×10^{7}**	**6.69×10^{7}**	**3.63×10^{7}**	↑
F9	8.41×10^{8}	8.39×10^{8}	5.61×10^{7}	**8.35×10^{8}**	**8.35×10^{8}**	**5.67×10^{7}**	↑
F10	5.02×10^{3}	5.01×10^{3}	2.15×10^{2}	**4.99×10^{3}**	**5.00×10^{3}**	**2.13×10^{2}**	↑
F11	2.062×10^{2}	2.05×10^{2}	3.29	**2.061×10^{2}**	**2.04×10^{2}**	**3.65**	↑
F12	7.62×10^{5}	7.60×10^{5}	3.28×10^{4}	**7.55×10^{5}**	**7.54×10^{5}**	**3.09×10^{4}**	↑
F13	1.64×10^{3}	1.45×10^{7}	1.76×10^{8}	**1.15×10^{3}**	**1.62×10^{3}**	**1.59×10^{3}**	↑
F14	1.28×10^{9}	1.28×10^{9}	5.88×10^{7}	**1.27×10^{9}**	**1.26×10^{9}**	**5.90×10^{7}**	↑
F15	1.20×10^{4}	1.20×10^{4}	3.82×10^{2}	1.20×10^{4}	1.20×10^{4}	3.73×10^{2}	↔
F16	4.142×10^{2}	4.127×10^{2}	3.58	**4.141×10^{2}**	**4.124×10^{2}**	**3.85**	↑
F17	1.72×10^{6}	1.72×10^{6}	5.23×10^{4}	**1.70×10^{6}**	**1.69×10^{6}**	**5.06×10^{4}**	↑
F18	6.55×10^{3}	3.09×10^{7}	1.53×10^{8}	**3.25×10^{3}**	**4.53×10^{3}**	**3.26×10^{3}**	↑
F19	4.64×10^{6}	4.63×10^{6}	2.63×10^{5}	**4.62×10^{6}**	**4.61×10^{6}**	**2.73×10^{5}**	↑
F20	2.07×10^{3}	4.10×10^{7}	2.54×10^{8}	**1.71×10^{3}**	**1.73×10^{3}**	**3.76×10^{2}**	↑

modifications were not included. Thus, the only modification with respect to the basic DE used in our initial experiments is the integration of bin crossover.

Table 13 shows the mean, median and standard deviation of DE and CDE-0.04 after 3 000 000 function evaluations. This stopping criterion was the one proposed during the competition held in CEC'10, so it was selected with the aim of facilitating the comparisons between different models. The format of the table is similar to that used in previous sections. We can see that the new model provides significant benefits in most problems. In fact, statistical tests report that CDE-0.04 is significantly better than DE in 17 out of 20 problems. As in previous experiments, there are several cases where DE has a much larger mean value than the one attained by CDE-0.04—for instance F1, F8, F13 and F18—meaning that the drawbacks analyzed also appear in this benchmark, and that they can be alleviated using the schemes proposed in this paper.

Finally, we would like to note that while our proposals provided significant benefits in the baseline DE, the results are not as good as those reported in complex hybrid schemes [5,22]. In fact, in none of the problems our simple scheme achieved lower mean or median values than those reported in [5]. This was expected because we have not made an effort to adapt our scheme to the specific nature of this benchmark, while in [5] several adaptations specifically designed for these problems were included. We did some additional experiments with the schemes used for the SOCO tests. Similarly to what happened in our previous experiments, the results could be improved upon and our proposals provided clear benefits. Thus, conclusions similar to those for the SOCO tests can be drawn, although the results obtained were far from those attained in [5]. This is because most of the methods used for the SOCO tests must be adapted to perform properly when the CEC tests are considered [23].

6. Conclusions and future work

DE is a highly efficient and popular metaheuristic especially tailored for continuous optimization problems. The analyses developed with DE in recent decades have shown that this metaheuristic suffers from the curse of dimensionality, i.e., its performance deteriorates rapidly when dealing with large-scale problems. In this paper, the relationship between certain weaknesses that had been detected by several authors and the use of large dimensionalities is explored. The weaknesses are related to the limited number of trial vectors that can be generated in DE, and to the less expansive behavior that DE shows with respect to other metaheuristics.

The main effects of these weaknesses when using low-dimensional problems can be avoided by increasing the diversity induced by DE, which is usually done by increasing the population size. However, when dealing with large-scale problems, increasing the population size is not a proper choice as faster convergence is required, given the fact that the portion of space being explored is much lower. In addition, the mathematical analysis presented in this paper provides better insight into the reasons for the appearance of these drawbacks when dealing with high-dimensional spaces. The mathematical analysis also shows that by using a variable mutation scale factor, these problems can be partially alleviated. However, these schemes introduce large changes into the basic behavior of DE, the benefits of which are not so clear.

Several different ways of avoiding these weaknesses might be proposed. In this paper, two schemes that can be used simultaneously have been devised. While these schemes represent by themselves, an important advance in the field, the main contribution of this paper is that with the help of the new methods, we have shown that the drawbacks analyzed herein have a significant effect on the behavior of DE, especially when high-dimensional spaces are involved. For this reason, the findings detailed in this paper should be taken into account in the future when designing new DE schemes. The first devised method has the effect of increasing the number of potential vectors that can be generated in DE while at the same time, respecting the basic principles of DE. The second modification increases the explorative behavior of DE by promoting large perturbations with an adaptive scheme.

Computational results from a large set of scalable problems with various complexities show that the new proposal is much better than the original DE scheme. The benefits obtained in terms of the quality of the solutions are clear. Moreover, the number of resources that can be saved with the new scheme is significant. Comparisons with several DE variants that consider a variable scale factor showed the higher effectiveness of the new scheme. Along the same line as other research, this latest study also calls into question the robustness of the adaptive schemes that use feedback to set the value of the mutation scale factor. Experiments with several complex non-hybrid DE variants have also been included. In these cases, the benefits remain intact, meaning that the state-of-the-art non-hybrid DE schemes for the SOCO tests could be improved. In the basic DE variant, the benefits of the proposals vanish when dealing with larger population sizes. However, when more complex DE variants were applied, the benefits of our proposals were also evident when dealing with large populations. The reason is that these state-of-the-art schemes induce a larger degree of exploitation. Thus, the advantages reported depend on this aspect, and are clearer when exploitative schemes are considered. In addition, the scalability study demonstrates that as the dimensionality of the problems involved increases, the advantages of the proposals are more significant. In these cases, the worst-case behavior of DE can be significantly improved by considering the schemes proposed herein. Finally, it is important to remark that the schemes devised were not as successful as other hybrid approaches where several components are in charge of the global search. This means that while our schemes clearly improve the global search capabilities of DE, additional work will be required to develop a more competitive DE for large-scale optimization where no other components are involved in the global search. Similar conclusions can be drawn when quite different problems, such as the CEC benchmarks, are taken into account.

Several lines of future work might be explored. First, similar modifications to those we propose might be applied when the trial vectors take several variables from the mutant vectors. Since the distributions associated with the different variables are not independent, their dependencies should be modeled. One choice might be to use copula functions. Another line of future work is to combine the proposals described here with some other methods that have been proposed in the literature. First, since in our opinion the global search capabilities of DE can already be improved upon, we would like to combine our methods with some general diversity-preservation strategies. Specifically, since our schemes operate on the variation stage, incorporating some of the methods that operate in the replacement phase seems truly promising. Once this is done, merging the scheme with co-evolution and with local search schemes seems quite encouraging. The incorporation of these techniques might result in more robust and competitive schemes. In addition, since the optimal *UpdateDenom* value depends on the problem at hand, schemes with automatic adaptation of this parameter might be in order. Finally, since some questions concerning the robustness of using feedback to set the mutation scale factor have emerged both in this work and in other related research, conducting more detailed analyses on this topic might be highly beneficial to the DE community.

Acknowledgments

The first author acknowledges the financial support from CONCYTEG as part of the plan “Investigadores Jóvenes - DPP-2014” (project 14-IJ-DPP-Q182-11). The second author is also affiliated to the UMI LAFMIA 3175 CNRS at CINVESTAV-IPN. He also acknowledges the financial support from CONACyT project no. 221551. The third author acknowledges the financial support from Ministerio de EconomÃa y Competitividad (project ECO2013-47129-C4-1-R).

References

[1] H. Abbass, The self-adaptive pareto differential evolution algorithm, IEEE Congr. Evol. Comput. 1 (2002) 831–836.

[2] A. Angela, A. Andrade, A. Soares, Exploration vs exploitation in differential evolution, in: Convention in Communication, Interaction and Social Intelligence, University of Aberdeen, 2008, pp. 57–63.

[3] J. Arabas, L. Bartnik, K. Opara, Dmea—An algorithm that combines differential mutation with the fitness proportionate selection, in: 2011 IEEE Symposium on Differential Evolution (SDE’11), IEEE, 2011, pp. 1–8.

[4] M. Bowling, M. Veloso, Rational and convergent learning in stochastic games, in: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI’01), Vol. 2, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001, pp. 1021–1026.

[5] J. Brest, B. Bošković, A. Zamuda, I. Fister, M. Maučec, Self-adaptive differential evolution algorithm with a small and varying population size, in: 2012 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2012, pp. 1–8.

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

[7] J. Brest, M. Maučec, Self-adaptive differential evolution algorithm using population size reduction and three strategies, Soft Comput. 15 (11) (2011) 2157–2174.

[8] J. Brest, M.S. Maučec, Population size reduction for the differential evolution algorithm, Appl. Intell. 29 (3) (2008) 228–247.

[9] J. Brest, A. Zamuda, B. Bošković, S. Greiner, V. Žumer, An analysis of the control parameters adaptation in DE, in: U. Chakraborty (Ed.), Advances in Differential Evolution, Studies in Computational Intelligence, vol. 143, Springer, Berlin Heidelberg, 2008, pp. 89–110.

[10] J. Brest, A. Zamuda, I. Fister, M. Maučec, Large scale global optimization using self-adaptive differential evolution algorithm, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.

[11] S. Das, A. Konar, U.K. Chakraborty, Two improved differential evolution schemes for faster global search, in: The 2005 Conference on Genetic and Evolutionary Computation (GECCO’05), ACM, 2005, pp. 991–998.

[12] S. Das, P. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.

[13] J. Durillo, A. Nebro, C.A.C. Coello, J. Garcia-Nieto, F. Luna, E. Alba, A study of multiobjective metaheuristics when solving parameter scalable problems, IEEE Trans. Evol. Comput. 14 (4) (2010) 618–635.

[14] A. Eiben, J. Smith, Introduction to evolutionary computing, in: Natural Computing Series, Springer, 2003, p. 66.

[15] R. Gämperle, S. Müller, P. Koumoutsakos, A parameter study for differential evolution, in: A. Grmela, N. Mastorakis (Eds.), Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, WSEAS Press, 2002, pp. 293–298.
[16] M. Ghasemi, M.M. Ghanbarian, S. Ghavidel, S. Rahmani, E.M. Moghaddam, Modified teaching learning algorithm and double differential evolution algorithm for optimal reactive power dispatch problem: A comparative study, Inf. Sci. 278 (0) (2014) 231–249.
[17] W. Gong, A. Fialho, Z. Cai, H. Li, Adaptive strategy selection in differential evolution for numerical optimization: An empirical study, Inf. Sci. 181 (24) (2011) 5364–5386.
[18] I.J. Good, The bayes/non-bayes compromise: A brief review, J. Am. Stat. Assoc. 87 (419) (1992) 597–606.
[19] J. Lampinen, I. Zelinka, On stagnation of the differential evolution algorithm, in: 6th International Mendel Conference on Soft Computing (MENDEL'00), 2000, pp. 76–83.
[20] W. Langdon, R. Poli, Evolving problems to learn about particle swarm optimizers and other search algorithms, IEEE Trans. Evol. Comput. 11 (5) (2007) 561–578.
[21] A. LaTorre, S. Muelas, J.-M. Peña, A mos-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test, Soft Comput. 15 (11) (2011) 2187–2199.
[22] A. LaTorre, S. Muelas, J.M. Peña, Multiple offspring sampling in large scale global optimization, in: 2012 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2012, pp. 1–8.
[23] A. LaTorre, S. Muelas, J.-M. Peña, A comprehensive comparison of large scale global optimizers, Inf. Sci. 316 (0) (2015) 517–549.
[24] M. Lozano, D. Molina, F. Herrera, Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems, Soft Comput. 15 (11) (2011) 2085–2087.
[25] S. Mahdavi, M.E. Shiri, S. Rahnamayan, Metaheuristics in large-scale global continues optimization: A survey, Inf. Sci. 295 (0) (2015) 407–428.
[26] E.n. Mezura-Montes, J. Velázquez-Reyes, C.A.C. Coello, A comparative study of differential evolution variants for global optimization, in: 8th Annual Conference on Genetic and Evolutionary Computation (GECCO'06), ACM, New York, NY, USA, 2006, pp. 485–492.
[27] J. Montgomery, Differential evolution: Difference vectors and movement in solution space, in: IEEE Congress on Evolutionary Computation, IEEE, 2009, pp. 2833–2840.
[28] J.A. Nelder, R. Mead, A simplex method for function minimization, Comput. J. 7 (1965) 308–313.
[29] F. Neri, V. Tirronen, Recent advances in differential evolution: A survey and experimental analysis, Artif. Intell. Rev. 33 (1-2) (2010) 61–106.
[30] M. Olguin-Carbajal, E. Alba, J. Arellano-Verdejo, Micro-differential evolution with local search for high dimensional problems, in: IEEE Congress on Evolutionary Computation 2013 (CEC'13), IEEE, 2013, pp. 48–54.
[31] O. Olorunda, A. Engelbrecht, Differential evolution in high-dimensional search spaces, in: IEEE Congress on Evolutionary Computation 2007 (CEC'07), IEEE, 2007, pp. 1934–1941.
[32] M.N. Omidvar, X. Li, Y. Mei, X. Yao, Cooperative co-evolution with differential grouping for large scale optimization, IEEE Trans. Evol. Comput. 18 (3) (2014) 378–393.
[33] K.E. Parsopoulos, Cooperative micro-differential evolution for high-dimensional problems, in: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO'09), ACM, New York, NY, USA, 2009, pp. 531–538.
[34] A.P. Piotrowski, Adaptive memetic differential evolution with global and local neighborhood-based mutation operators, Inf. Sci. 241 (2013) 164–194.
[35] M.A. Potter, K.A.D. Jong, A cooperative coevolutionary approach to function optimization, in: Proceedings of the Third Conference on Parallel Problem Solving from Nature (PPSN III), Springer-Verlag, London, UK, 1994, pp. 249–257.
[36] K. Price, I. Zelinka, V. Snášel, A. Abraham, Differential evolution, in: Handbook of Optimization, in: Intelligent Systems Reference Library, vol. 38, Springer, Berlin Heidelberg, 2013, pp. 187–214.
[37] K. Price, R. Storn, J. Lampinen, Differential evolution: A practical approach to global optimization, in: Natural Computing Series, U.S. Government Printing Office, 2005.
[38] A.K. Qin, V.L. Huang, P. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[39] A.K. Qin, P. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, IEEE Congr. Evol. Comput. 2 (2005) 1785–1791.
[40] R. Salinas-Gutiérrez, A. Hernández-Aguirre, E.R. Villa-Diharce, Using copulas in estimation of distribution algorithms, in: A.H. Aguirre, R.M. Borja, C.A.R. García (Eds.), MICAI 2009: Advances in Artificial Intelligence, Lecture Notes in Computer Science, vol. 5845, Springer, Berlin Heidelberg, 2009, pp. 658–668.
[41] C. Segura, C.A.C. Coello, E. Segredo, C. León, On the adaptation of the mutation scale factor in differential evolution, Optim. Lett. 9 (1) (2015) 189–198.
[42] Y.-j. Shi, H.-f. Teng, Z.-q. Li, Cooperative co-evolutionary differential evolution for function optimization, in: L. Wang, K. Chen, Y. Ong (Eds.), Advances in Natural Computation, Lecture Notes in Computer Science, vol. 3611, Springer, Berlin Heidelberg, 2005, pp. 1080–1088.
[43] R. Storn, K. Price, Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces, International Computer Science Institute, Berkeley, 1995. Tech. Rep. TR95012
[44] R. Storn, K. Price, Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (1997) 341–359.
[45] K. Táng, X. Lǐ, P.N. Suganthan, Z. Yáng, T. Weise, Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization, School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL), University of Science and Technology of China (USTC), Héféi, Ānhuī, China, 2010.
[46] J. Tvrdík, R. Poláková, J. Veselský, P. Bujok, Adaptive variants of differential evolution: Towards control-parameter-free optimizers, in: I. Zelinka, V. Snášel, A. Abraham (Eds.), Handbook of Optimization, Intelligent Systems Reference Library, vol. 38, Springer, Berlin Heidelberg, 2013, pp. 423–449.
[47] H. Wang, Z. Wu, S. Rahnamayan, Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems, Soft Comput. 15 (11) (2011) 2127–2140.
[48] H. Wang, Z. Wu, S. Rahnamayan, D. Jiang, Sequential de enhanced by neighborhood search for large scale global optimization, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–7.
[49] M. Weber, F. Neri, V. Tirronen, Shuffle or update parallel differential evolution for large-scale optimization, Soft Comput. 15 (11) (2011) 2089–2107.
[50] T. Weise, R. Chiong, K. Táng, Evolutionary optimization: Pitfalls and booby traps, J. Comput. Sci. Technol. 27 (5) (2012) 907–936. Special Issue on Evolutionary Computing.
[51] Z. Yang, K. Tang, X. Yao, Large scale evolutionary optimization using cooperative coevolution, Inf. Sci. 178 (15) (2008) 2985–2999.
[52] Z. Yang, K. Tang, X. Yao, Scalability of generalized adaptive differential evolution for large-scale continuous optimization, Soft Comput. 15 (11) (2011) 2141–2155.
[53] D. Zaharie, Control of population diversity and adaptation in differential evolution algorithms, in: R. Matousek, P. Osmera (Eds.), Proceedings of Mendel 2003, 9th International Conference on Soft Computing, Brno, Czech Republic, 2003, pp. 41–46.
[54] D. Zaharie, Influence of crossover on the behavior of differential evolution algorithms, Appl. Soft Comput. 9 (3) (2009) 1126–1138.
[55] J. Zhang, A. Sanderson, Jade: Adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[56] J. Zhao, Y. Xu, F. Luo, Z. Dong, Y. Peng, Power system fault diagnosis based on history driven differential evolution and stochastic time domain simulation, Inf. Sci. 275 (2014) 13–29.
[57] S.-Z. Zhao, P.N. Suganthan, Empirical investigations into the exponential crossover of differential evolutions, Swarm Evol. Comput. 9 (2013) 27–36.
[58] W. Zhu, Y. Tang, J.a. Fang, W. Zhang, Adaptive population tuning scheme for differential evolution, Inf. Sci. 223 (2013) 164–191.
[59] K. Zielinski, X. Wang, R. Laur, Comparison of adaptive approaches for differential evolution, in: G. Rudolph, T. Jansen, S. Lucas, C. Poloni, N. Beume (Eds.), Parallel Problem Solving from Nature—PPSN X, Lecture Notes in Computer Science, vol. 5199, Springer, Berlin Heidelberg, 2008, pp. 641–650.
[60] K. Zielinski, P. Weitkemper, R. Laur, K.D. Kammeyer, Parameter study for differential evolution using a power allocation problem including interference cancellation, in: IEEE Congress on Evolutionary Computation (CEC'06), IEEE, 2006, pp. 1857–1864.