An Adaptive Differential Evolution Algorithm With Novel Mutation and Crossover Strategies for Global Numerical Optimization

Sk. Minhazul Islam, Swagatam Das, *Member, IEEE*, Saurav Ghosh, Subhrajit Roy, and Ponnuthurai Nagaratnam Suganthan, *Senior Member, IEEE*

***Abstract*—Differential evolution (DE) is one of the most powerful stochastic real parameter optimizers of current interest. In this paper, we propose a new mutation strategy, a fitness-induced parent selection scheme for the binomial crossover of DE, and a simple but effective scheme of adapting two of its most important control parameters with an objective of achieving improved performance. The new mutation operator, which we call DE/current-to-gr_best/1, is a variant of the classical DE/current-to-best/1 scheme. It uses the best of a group (whose size is $q\%$ of the population size) of randomly selected solutions from current generation to perturb the parent (target) vector, unlike DE/current-to-best/1 that always picks the best vector of the entire population to perturb the target vector. In our modified framework of recombination, a biased parent selection scheme has been incorporated by letting each mutant undergo the usual binomial crossover with one of the p top-ranked individuals from the current population and not with the target vector with the same index as used in all variants of DE. A DE variant obtained by integrating the proposed mutation, crossover, and parameter adaptation strategies with the classical DE framework (developed in 1995) is compared with two classical and four state-of-the-art adaptive DE variants over 25 standard numerical benchmarks taken from the IEEE Congress on Evolutionary Computation 2005 competition and special session on real parameter optimization. Our comparative study indicates that the proposed schemes improve the performance of DE by a large magnitude such that it becomes capable of enjoying statistical superiority over the state-of-the-art DE variants for a wide variety of test problems. Finally, we experimentally demonstrate that, if one or more of our proposed strategies are integrated with existing powerful DE variants such as jDE and JADE, their performances can also be enhanced.**

***Index Terms*—Derivative-free optimization, differential evolution (DE), evolutionary algorithms (EAs), genetic algorithms (GAs), parameter adaptation, particle swarm optimization (PSO).**

Manuscript received January 20, 2011; revised May 29, 2011 and August 4, 2011; accepted August 11, 2011. Date of publication October 14, 2011; date of current version March 16, 2012. This paper was recommended by Associate Editor H. Ishibuchi.

S. M. Islam, S. Ghosh, and S. Roy are with the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata 700 032, India (e-mail: skminha.isl@gmail.com; saurav_online@yahoo.in; roy.subhrajit20@gmail.com).

S. Das is with the Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata 700 108, India (e-mail: swagatam.das@ieee.org).

P. N. Suganthan is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: epnsugan@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2011.2167966

I. Introduction

THE DIFFERENTIAL evolution (DE) [1]–[4] algorithm emerged as a very competitive form of evolutionary computing more than a decade ago. Since the late 1990s, DE has started to find several significant applications to the optimization problems arising from diverse domains of science and engineering. The DE family of algorithms has been frequently adopted to tackle multiobjective, constrained, dynamic, large-scale, and multimodal optimization problems, and the resulting variants have been achieving top ranks in various competitions held under the IEEE Congress on Evolutionary Computation (CEC) conference series (e.g., see http://www3.ntu.edu.sg/home/epnsugan/index_files/cec-benchmarking.htm). For a detailed survey on the state-of-the-art research with DE, the readers can see [5] and [6] and the references therein.

Although during the last ten years, research on and with DE has reached an impressive state, there are still many open problems, and new application areas are continually emerging for the algorithm (see , Section IX[6] for the open research issues). There is still a need for improving the search performance of DE to face challenges from the modern application areas. Some of the recent publications [7]–[9] indicate that DE may face significant difficulty on certain kind of functions. The performance of DE primarily depends on the mutation strategy, the crossover operation, and the intrinsic control parameters like scale factor (F), crossover rate (Cr), and population size (Np). The DE family now consists of more than five distinct mutation strategies (some of which, like DE/current-to-best/1, may even generate mutated recombinants) and two prominent crossover schemes (exponential and binomial). Each of these mutation and crossover operations may be effective over certain problems but poorly perform over the others, e.g., see works like [10] for claims, counter-claims, and comparisons among these evolutionary operators in DE. Similarly, in order to make the performance of DE more robust, over the past decade, several research efforts have been reported regarding the tuning and adaptation strategies of the control parameters F and Cr. Some objective functions are very sensitive to the proper choice of the parameter settings in DE [11]. Therefore, researchers naturally started to consider techniques to automatically find an optimal set of control parameters for DE [12]–[18]. The most recent trend in this direction is the use of self-adaptive strategies like the ones reported in [12] and [13]. However, self-adaptation

of control parameters as well as mutation strategies (like in [12] and [18]) may make the programming fairly complex and run the risk of increasing the number of function evaluations (FEs).

In this paper, we propose the following three algorithmic components, where one or more of which can be integrated with the DE family of algorithms to improve their performances on complicated fitness landscapes.

1) First, a less greedy and more explorative variant of the DE/current-to-best/1 mutation strategy is proposed. We call it DE/current-to-gr_best/1 (gr stands for group). Unlike DE/current-to-best/1 that always uses the fittest (hence best) member of the entire current population to perturb a target (parent) vector, DE/current-to-gr_best/1 forms a group, corresponding to each target vector, by randomly selecting population members to form a group whose size is, for example, $q\%$ of the total population size. The best member of this dynamic group is used to perturb the target vector.
2) Second, we modify the conventional binomial crossover scheme of DE by introducing a fitness-induced bias in the selection of parents from the current generation. The exploitative crossover scheme, so developed, is referred to here as "p-best crossover." Under this scheme, a mutant vector is allowed to exchange its components not with the parent at the same index but with a randomly selected member from the p top-ranked individuals from the current generation through uniform (binomial) crossover. Note that it is not a new crossover operator but the usual binomial crossover with a biased parent selection strategy.
3) Third, we suggest simple schemes to update the values of F and Cr in each generation, guided by the knowledge of their successful values that were able to generate better offspring (trial vectors) in the last generation.

Note that, in the light of the analysis of the algorithmic role and functioning principle of different modifications introduced to DE as presented in , pp. 39–40[5], the mutation including a ranking selection (the proposed one) tends to increase the exploitation as it makes a fitness-based selection. The p-best crossover also increases the exploitation as it performs a fitness-based selection. DE does not select parents in a fitness-based way, and this can be the origin of the stagnation problems. Finally, the adaptation schemes for the parameters include a certain degree of randomization that is likely to promote the explorative behavior in a controlled manner. In this context, it is worth mentioning that, in [19], an explicit balance between exploration and exploitation is obtained in DE by separate subpopulations.

We develop a stand-alone DE variant by including the aforementioned components in the framework of classical DE algorithm (e.g., see , p. 42[3]). In what follows, we shall refer to this new adaptive DE (ADE) algorithm as MDE_pBX (modified DE (MDE) with p-best crossover). The proposed scheme is compared with DE/rand/1/bin, DE/current-to-best/1/bin and four state-of-the-art ADE variants (self-adaptive DE (SaDE) [12], JADE [20], jDE [13], and DEGL [21]) over 25 standard numerical benchmarks taken from the CEC 2005 competition and special session on real parameter optimization. We also tested the algorithms on two instantiations of the spacecraft trajectory optimization problems [22], [23]. Our experimental results indicate that MDE_pBX is able to meet or beat its nearest competitors over most of the tested instances in a statistically meaningful way. Our experiments indicate that incorporation of one of the proposed mutation, crossover, and parameter adaptation schemes in the structures of recently developed and best-known DE variants can further improve their optimizing abilities by a good extent. In order to save space, in Section IV-H, we experimentally demonstrate this fact by considering two powerful ADE variants: jDE and JADE.

The remainder of this paper is organized as follows. In Section II, we provide a brief outline of the DE family of algorithms. Section III introduces the proposed MDE algorithm, explaining each of the steps in sufficient detail. The experimental settings for the benchmarks and simulation strategies are explained in Section IV. Results are also presented and discussed in the same section. Finally, Section V concludes this paper with a discussion on potential future research directions.

II. Classical DE Family of Algorithms

DE is a simple real parameter optimization algorithm. It works through a simple cycle of stages of mutation, crossover, and selection, as described in the following.

A. Initialization of the Parameter Vectors

DE searches for a global optimum point in a D-D real parameter space $\Re^D$. It begins with a randomly initiated population of Np D-D real-valued parameter vectors. Each vector, also known as *genome/chromosome*, forms a candidate solution to the multidimensional optimization problem. We shall denote subsequent generations in DE by $G = 0, 1, \ldots, G_{\max}$. Since the parameter vectors are likely to be changed over different generations, we may adopt the following notation for representing the ith vector of the population at the current generation:

$$\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, \ldots\ldots, x_{D,i,G}]. \quad (1)$$

For each parameter of the problem, there may be a certain range within which the value of the parameter should be restricted, often because parameters are related to physical components or measures that have natural bounds (for example, if one parameter is a length or mass, it cannot be negative). The initial population (at $G = 0$) should cover this range as much as possible by uniformly randomizing individuals within the search space constrained by the prescribed minimum and maximum bounds

$$\vec{X}_{\min} = \{x_{1,\min}, x_{2,\min}, \ldots, x_{D,\min}\}$$
$$\vec{X}_{\max} = \{x_{1,\max}, x_{2,\max}, \ldots, x_{D,\max}\}.$$

Hence, we may initialize the jth component of the ith vector as

$$x_{j,i,0} = x_{j,\min} + rand_{i,j}[0,1] \cdot (x_{j,\max} - x_{j,\min}) \quad (2)$$

where $rand_{i,j}[0,1]$ is a uniformly distributed random number lying between 0 and 1 (actually $0 \leq rand_{i,j}[0,1] \leq 1$) and is instantiated independently for each component of the ith vector.

B. Mutation With Difference Vectors

After initialization, DE creates a *donor* vector $\vec{V}_{i,G}$ corresponding to each population member or *target* vector $\vec{X}_{i,G}$ in the current generation through mutation. The five most frequently referred mutation strategies are listed in the following:

"DE/rand/1" :

$$\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F \cdot \left(\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}\right) \tag{3a}$$

"DE/best/1" :

$$\vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot \left(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}\right) \tag{3b}$$

"DE/current-to-best/1" :

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F \cdot (\vec{X}_{best,G} - \vec{X}_{i,G}) + F \cdot \left(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}\right) \tag{3c}$$

"DE/best/2" :

$$\vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot \left(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}\right) + F \cdot \left(\vec{X}_{r_3^i,G} - \vec{X}_{r_4^i,G}\right) \tag{3d}$$

"DE/rand/2" :

$$\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F \cdot \left(\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}\right) + F \cdot \left(\vec{X}_{r_4^i,G} - \vec{X}_{r_5^i,G}\right). \tag{3e}$$

The indices r_1^i, r_2^i, r_3^i, r_4^i, and r_5^i are mutually exclusive integers randomly chosen from the range $[1, Np]$, and all are different from the index i. These indices are randomly generated anew for each donor vector. The scaling factor F is a positive control parameter for scaling the difference vectors. $\vec{X}_{best,G}$ is the best individual vector with the best fitness (i.e., lowest objective function value for a minimization problem) in the population at generation G. In (3c), $\vec{X}_{i,G}$ is known as the target vector, and $\vec{V}_{i,G}$ is known as the donor vector. The general convention used for naming the various mutation strategies is DE/x/y/z, where DE stands for differential evolution, x represents a string denoting the vector to be perturbed, and y is the number of difference vectors considered for the perturbation of x. z stands for the type of crossover being used (*exp*: exponential and *bin*: binomial). The following section discusses the crossover step in DE.

C. Crossover

Through crossover, the donor vector mixes its components with the target vector $\vec{X}_{i,G}$ under this operation to form the trial vector $\vec{U}_{i,G} = [u_{1,i,G}, u_{2,i,G}, u_{3,i,G}, \ldots, u_{D,i,G}]$. The DE family of algorithms uses mainly two kinds of crossover methods—*exponential* (or two-point modulo) and *binomial* (or uniform) [2]. We only elaborate here the binomial crossover as the proposed DE variant uses this scheme. Binomial crossover is performed on each of the D variables whenever a randomly generated number between 0 and 1 is less than or equal to the Cr value. In this case, the number of parameters inherited from the donor has a (nearly) binomial distribution. The scheme may be outlined as

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if } (rand_{i,j}[0,1] \leq Cr \text{ or } j = 1_{rand}) \\ x_{j,i,G}, & \text{otherwise} \end{cases} \tag{4}$$

where, as before, $rand_{i,j}[0,1]$ is a uniformly distributed random number, which is called anew for each jth component of the ith parameter vector. $j_{rand} \in [1, 2, \ldots, D]$ is a randomly chosen index, which ensures that $\vec{U}_{i,G}$ gets at least one component from $\vec{V}_{i,G}$. It is instantiated once for each vector per generation.

D. Selection

Selection determines whether the target or the trial vector survives to the next generation, i.e., at $G = G + 1$. The selection operation is described as

$$\begin{aligned} \vec{X}_{i,G+1} &= \vec{U}_{i,G}, \quad \text{If } f(\vec{U}_{i,G}) \leq f(\vec{X}_{i,G}) \\ &= \vec{X}_{i,G}, \quad \text{if } f(\vec{U}_{i,G}) > f(\vec{X}_{i,G}) \end{aligned} \tag{5}$$

where $f(\vec{X})$ is the objective function to be minimized. Therefore, if the new trial vector yields an equal or lower value of the objective function, it replaces the corresponding target vector in the next generation; otherwise, the target is retained in the population.

III. MDE_pBX Algorithm

In this section, we outline MDE_pBX and discuss the steps of the algorithm in details.

A. DE/current-to-gr_best/1

The oldest of the DE mutation schemes is DE/rand/1/bin, developed by Storn and Price [1], [2], and is said to be the most successful and widely used scheme in the literature [6]. However, [10] and [24] indicate that DE/best/2 and DE/best/1 may have some advantages over DE/rand/1. The authors of [25] are of the opinion that the incorporation of the best solution (with lowest objective function value for minimization problems) information is beneficial and use DE/current-to-best/1 in their algorithm. Compared to DE/rand/k, greedy strategies like DE/current-to-best/k and DE/best/k benefit from their fast convergence by guiding the evolutionary search with the best solution so far discovered, thereby converging faster to that point. However, as a result of such exploitative tendency, in many cases, the population may lose its diversity and global exploration abilities within a relatively small number of generations, thereafter getting trapped to some locally optimal point in the search space. In addition, DE employs a greedy

selection strategy (the better between the target and trial vectors is selected) and uses a fixed scale factor F (typically in [0.4, 1]). Thus, if the difference vector $\vec{X}_{r_1,G} - \vec{X}_{r_2,G}$, used for perturbation is small (this is usually the case when the vectors come very close to each other and the population converges to a small domain), the vectors may not be able to explore any better region of the search space, thereby finding it difficult to escape large plateaus or suboptimal peaks/valleys.

Taking into consideration these facts and to overcome the limitations of fast but less reliable convergence performance of DE/current-to-best/1 scheme, in this paper, we propose a less greedy and more explorative variant of the DE/current-to-best/1 mutation strategy by utilizing the best vector of a dynamic group of $q\%$ of the randomly selected population members for each target vector. The new scheme, which we call DE/current-to-gr_best/1, can be expressed as

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F_i \left(\vec{X}_{gr\_best,G} - \vec{X}_{i,G} + \vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G} \right) \tag{6}$$

where $\vec{X}_{gr\_best,G}$ is the best of the $q\%$ vectors randomly chosen from the current population, whereas $\vec{X}_{r_1^i,G}$ and $\vec{X}_{r_2^i,G}$ are two distinct vectors picked up randomly from the current population, and none of them is equal to $\vec{X}_{gr\_best,G}$ or the target vector to ensure that none of the vectors is equal in (6). Under this scheme, the target solutions are not always attracted toward the same best position found so far by the entire population, and this feature is helpful in avoiding premature convergence at local optima. It is seen that keeping the group size equal to 15% of the population size provides very good results on majority of the tested benchmarks. The effect of the variation of q on the performance of the algorithm is discussed in Section IV-E.

B. p-Best Crossover

The p-best crossover operation incorporates a greedy parent selection strategy with the conventional binomial crossover scheme of DE. Here, for each donor vector, a vector is randomly selected from the p top-ranking vectors (according to their objective function values) in the current population, and then, normal binomial crossover is performed as per (5) between the donor vector and the randomly selected p-best vector to generate the trial vector at the same index. Parameter p is linearly reduced with generations in the following way:

$$p = ceil \left[\frac{Np}{2} \cdot \left(1 - \frac{G-1}{G_{\max}} \right) \right] \tag{7}$$

where G is the current generation number, $G_{\max}$ is the maximum number of generations, $G = [1, 2, \ldots, G_{max}]$, and $ceil(y)$ is the "ceiling" function returning the lowest integer greater than its argument y. The reduction routine of p favors exploration at the beginning of the search and exploitation during the later stages by gradually downsizing the elitist portion of the population, with a randomly selected member from where the component mixing of the donor vector is allowed for formation of the trial vector.

C. Parameter Adaptation Schemes in MDE_pBX

The parameter adaptation schemes in MDE_pBX are inspired by those used by Zhang and Sanderson in their JADE algorithm [20], but the former ones are also distinct due to their own characteristics.

Scale Factor Adaptation: At every generation, the scale factor F_i of each individual target vector is independently generated as

$$F_i = Cauchy(F_m, 0.1) \tag{8}$$

where $Cauchy(F_m, 0.1)$ is a random number sampled from a Cauchy distribution with location parameter F_m and scale parameter 0.1. The value of F_i is regenerated if $F_i \leq 0$ or $F_i > 1$. Denote F_{success} as the set of the successful scale factors, so far, of the current generation generating better trial vectors that are likely to advance to the next generation. Also, let $mean_A(F_{G-1})$ be the simple arithmetic mean of all scale factors associated with population members in generation $G-1$. Location parameter F_m of the Cauchy distribution is initialized to be 0.5 and is then updated at the end of each generation in the following manner:

$$Fm = w_F \cdot Fm + (1 - w_F) \cdot mean_{\text{Pow}}(F_{\text{success}}). \tag{9a}$$

The weight factor w_F varies randomly between 0.8 and 1 in the following way:

$$w_F = 0.8 + 0.2^* rand(0, 1) \tag{9b}$$

where $rand(0, 1)$ stands for a uniformly distributed random number in (0, 1) and $mean_{\text{Pow}}$ stands for the power mean [26] given by

$$mean_{\text{Pow}}(F_{\text{success}}) = \sum_{x \in F_{\text{Success}}} (x^n / |F_{\text{success}}|)^{\frac{1}{n}} \tag{10}$$

with $|F_{\text{Success}}|$ denoting the cardinality of the set F_{Success}. We take $n = 1.5$ as it gives the best results on a wide variety of tested problems. Small random perturbations to the weight terms of F_m and $mean_{\text{pow}}$ put slightly varying emphasis on the two terms each time an F is generated and improve the performance of MDE_pBX as revealed through our parameter tuning experiments.

Crossover Probability Adaptation: At every generation, the crossover probability Cr_i of each individual vector is independently generated as

$$Cr_i = Gaussian(Cr_m, 0.1) \tag{11}$$

where $Gaussian(Cr_m, 0.1)$ is a random number sampled from a Gaussian distribution accordingly with mean Cr_m and standard deviation 0.1. Cr_i is regenerated if it falls outside the interval [0, 1]. Denote Cr_{success} as the set of all successful crossover probabilities Cr_is at the current generation. The mean of the normal distribution Cr_m is initialized to be 0.6 and is then updated at the end of each generation as

$$Cr_m = w_{Cr} \cdot Cr_m + (1 - w_{Cr}) \cdot mean_{\text{Pow}}(Cr_{\text{success}}) \tag{12a}$$

with the weight being uniformly randomly fluctuating between 0.9 and 1

$$w_{Cr} = 0.9 + 0.1 * rand(0, 1). \quad (12b)$$

The power mean is calculated as

$$mean_{\text{Pow}}(Cr_{\text{success}}) = \sum_{x \in Cr_{\text{Success}}} (x^n / |Cr_{\text{success}}|)^{\frac{1}{n}} \quad (13)$$

where $|Cr_{\text{Success}}|$ denotes the cardinality of the set Cr_{Success}. We also choose here $n = 1.5$ based on the experimental results.

Explanation of the Parameter Adaptation: Earlier theoretical studies on DE [27], [28] have indicated that the scale factor F has a big role in controlling the population diversity and the explorative power of DE. During the adaptation of F_m, the usage of power mean leads to a higher value of F_m that accounts for larger perturbation to the target vectors, thus avoiding premature convergence at local optima. The essence of F_{success} is that it memorizes the successful scale factors in the current generation, thereby glorifying the chance of creating better donor vectors as more and more target vectors are processed. F_m is used as a location parameter of Cauchy distribution, which diversifies the values of F more as compared to the traditional normal distribution.

The fact that the Cauchy distribution has a far wider tail than traditional Gaussian distribution [29] is beneficial when the global optima is far away from the current search point as the values of F taken from the tail region give sufficient perturbation so that premature convergence can be avoided. We would like to mention that other probability distributions like the beta distribution of any shape (which was used in context to DE for memetic coordination in [30]) or exponential distribution (which was used with the memetic DE in [31]) can also be constructed by using two shape factors for such adaptation purpose. We wish to investigate the effect of such distributions on the algorithmic performance in the future. Note that our motivation for adapting the scale factor over the generations is inspired by the work of Weber *et al.* [32], where the scale factor over multiple populations is also self-adaptively determined and is found that a proper scale factor is dynamic over time (during a run) in addition to being problem dependent. According to the analyses and empirical results presented in [5] and [33], DE has a limited amount of search moves, and the effectiveness and amplitudes of such moves explicitly depend on how the scale factor values are being adjusted over generations. In this respect, the adaptations of the DE scheme proposed here (e.g., the sampling of the scale factor F from a Cauchy distribution) allow an enhancement of the performance.

The adaptation of Cr_m is also based on the record of recent successful crossover probabilities and uses them to guide the generation of new Cr_is. Therefore, for the adaptation of Cr_m, the usage of Cr_{success} again records the successful Cr values, thus generating better individuals as offspring, which are more likely to survive. A normal distribution with mean Cr_m and standard deviation of 0.1 is used to generate the Cr values. The usage of power mean instead of arithmetic mean in the adaptation of Cr_m leads to higher values of Cr, which eliminates the implicit bias of Cr toward small values during its self-adaptation. Here, the Cauchy distribution is avoided, and Gaussian distribution is selected because the long tail property of the former is not needed in case of the crossover probability adaptation. If the Cauchy distribution were used, the long tail property of the Cauchy distribution would lead to higher values of Cr, which would have to be truncated to unity. Consequently, values of Cr would become independent of the Cauchy distribution. However, the usage of the Gaussian distribution provides the opportunity to generate most of the Cr values within unity because of its short tail property.

The weight terms in (9a) and (12a) control the average life span of successful F and Cr values [20]. Small values of the weights cause false convergence due to the lack of sufficient information to smoothly update Cr and F. Our experimental results indicate that small random perturbations to the weight term w in (9a) and (12a) are very effective in improving the performance of the corresponding DE algorithm on a wide variety of functions.

IV. Experiments and Results

A. Numerical Benchmarks

The MDE_pBX algorithm is tested using a set of standard benchmark functions from the special session and competition on real parameter optimization held under the IEEE CEC 2005 [34], [35]. These functions span a diverse set of problem features, including multimodality, ruggedness, noise in fitness, ill-conditioning, nonseparability, and interdependence (rotation), and are based on classical benchmark functions such as Rosenbrock's, Rastrigin's, Swefel's, Griewank's, and Ackley's functions. A detailed description of these functions appears in [34] and [35], and so, it will not be repeated here for the sake of space economy. In summary, functions 1–5 are uni-modal, functions 6–12 are multimodal, and functions 13–25 are hybrid composition functions, implementing a combination of several well-known benchmark functions.

B. Algorithms Compared and Parametric Setup

The performance of the MDE_pBX algorithm is compared with the following algorithms that include (in the order of appearance) two classical DE variants and four state-of-the-art ADE variants:

1) DE/current-to-best/1/bin with $F = 0.8$ and $Cr = 0.9$;
2) DE/rand/1/bin with $F = 0.8$ and $Cr = 0.9$;
3) JADE with $c = 0.1$, $p = 0.05$, and optional external archive [20];
4) jDE with $F_l = 0.1$, $F_u = 0.9$, and $\tau_1 = \tau_2 = 0.1$[13];
5) SaDE [12];
6) DEGL/SAW [21] with $\alpha = \beta = F = 0.8$, $Cr = 0.9$, and neighborhood size $= 0.1 * Np$.

Unless otherwise stated, for all the contestant algorithms, we employ the best suited parametric setup chosen with guidelines from their respective literatures. The population size Np for the DE variants has been kept equal to 100 irrespective of the problem dimension D.

TABLE I
MEAN AND STANDARD DEVIATION OF THE ERROR VALUES FOR $f_1 - f_{15}(30D)$. THE BEST ENTRIES ARE MARKED IN BOLDFACE

Func / Algorithms	f_1 Mean(Std)	f_2 Mean(Std)	f_3 Mean(Std)	f_4 Mean(Std)	f_5 Mean(Std)
DE/rand/1/bin	2.4536e-05 (3.4756e-05)	5.4975e-02 (1.2754e-02)	2.8921e+05 (1.9383e+05)	5.0435e-01 (8.5897e-01)	1.2719e+03 (2.8364e+02)
DE/current-to-best/1/bin	2.4536e-25 (3.4756e-25)	6.2953e-08 (1.2927e-07)	1.8918e+05 (1.0392e+05)	1.0714e-02 (2.7821e-02)	5.7628e+02 (1.2476e+02)
JADE	1.3258e-54 (9.2436e-54)	2.5146e-26 (3.4269e-26)	4.7421e+04 (1.6213e+04)	5.1159e-07 (4.0194e-07)	3.2792e+02 (1.8494e+02)
jDE	3.8652e-29 (4.5732e-29)	7.5064e-06 (7.3804e-06)	2.2663e+05 (1.6085e+05)	2.7305e-01 (2.7305e-01)	1.1108e+03 (3.7238e+02)
SaDE	6.7843e-30 (2.3879e-30)	9.7191e-08 (4.8596e-07)	5.0521e+04 (1.5754e+05)	5.8160e-06 (1.4479e-05)	7.8803e+02 (1.2439e+03)
DEGL	2.3462e-20 (5.6234e-20)	1.1757e-07 (6.5592e-08)	2.3114e+05 (1.032e+05)	1.5746e+03 (9.501e+00)	5.0692e+02 (5.803e+02)
MDE_*p*BX	**1.3429e-62 (2.4352e-61)**	**1.9981e-26 (2.4429e-26)**	**2.0977e+03 (1.2699e+03)**	**6.9268e-08 (8.9742e-08)**	**2.2057e+02 (1.6754e+02)**
Func / Algorithms	f_6 Mean(Std)	f_7 Mean(Std)	f_8 Mean(Std)	f_9 Mean(Std)	f_{10} Mean(Std)
DE/rand/1/bin	2.7764e+01 (1.0186e+01)	9.6575e-01 (9.1476e-02)	2.0941e+01 (6.2596e-02)	4.3742e+01 (9.7342e+00)	1.5643e+02 (4.5674e+01)
DE/current-to-best/1/bin	9.2736e+00 (8.2835e+00)	8.4253e-01 (9.1442e-02)	2.0942e+01 (5.0063e-02)	2.3465e+01 (8.6395e+00)	1.9617e+02 (2.1683e+01)
JADE	5.6094e+00 (1.9445e+01)	6.9598e-03 (4.4845e-03)	2.0929e+01 (2.6628e-02)	**5.9272e-22 (8.9543e-22)**	3.0313e+01 (8.3551e+00)
jDE	1.1196e+01 (1.3987e+00)	9.8597e-03 (3.4824e-03)	2.0931e+01 (2.5067e-02)	8.3264e-16 (2.3645e-15)	5.2547e+01 (4.4660e+00)
SaDE	2.1248e+01 (1.3413e+01)	8.2727e-03 (1.1445e-02)	2.0140e+01 (5.7258e-02)	2.2737e-15 (1.1369e-14)	3.5758e+01 (6.0809e+00)
DEGL	4.7840e-01 (1.3222e+00)	6.9990e-01 (4.5371e-03)	2.0097e+01 (2.3029e-02)	1.7591e+01 (3.0222e+00)	3.7410e+01 (5.2883e+00)
MDE_*p*BX	**3.9870e-01 (1.0815e+00)**	**6.6472e-03 (9.0313e-03)**	**2.0000e+01 (6.7185e-07)**	1.0342e-09 (3.2346e-10)	**1.4890e+01 (8.9159e-01)**
Func / Algorithms	f_{11} Mean(Std)	f_{12} Mean(Std)	f_{13} Mean(Std)	f_{14} Mean(Std)	f_{15} Mean(Std)
DE/rand/1/bin	3.2648e+01 (1.0954e+01)	8.4331e+04 (6.2535e+04)	4.5130e+00 (2.2662e+00)	1.3347e+01 (3.4764e-01)	4.8432e+02 (2.1467e+01)
DE/current-to-best/1/bin	3.1683e+01 (7.5983e+00)	9.8363e+04 (6.2262e+03)	3.5182e+00 (2.2693e+00)	1.3453e+01 (3.4795e-01)	3.8424e+02 (5.1434e+01)
JADE	2.6456e+01 (1.9169e+01)	2.6978e+04 (6.8003e+03)	1.6285e+00 (4.8739e-02)	1.2771e+01 (2.2057e-01)	2.8884e+02 (9.0503e+01)
jDE	3.1370e+01 (2.3952e+00)	3.8376e+04 (6.5374e+03)	1.8568e+00 (1.0313e-01)	1.3545e+01 (9.9402e-02)	3.5642e+02 (1.8711e+01)
SaDE	2.6562e+01 (1.1275e+00)	**8.7345e+02 (9.3383e+02)**	1.2070e+00 (1.3420e-01)	1.2760e+01 (2.5936e-01)	3.2775e+02 (9.6450e+01)
DEGL	2.7278e+01 (1.5739e+00)	2.5359e+04 (2.8883e+03)	2.3595e+00 (5.2823e-01)	1.2961e+01 (4.1146e-01)	3.4400e+02 (5.0662e+01)
MDE_*p*BX	**1.7590e+01 (6.0615e+00)**	1.5793e+03 (8.1383e+02)	**1.1051e+00 (5.6060e-02)**	**1.2429e+01 (3.4320e-01)**	**2.5653e+02 (9.7542e+01)**

C. Simulation Strategies

Functions f_1 to f_{25} are tested in 30 and 50 dimensions. In order to evaluate how the performance of MDE_pBX changes with the scaling of the search space, we also provide comparative results of all the algorithms in functions f_1 to f_{14} in 100 dimensions. We exclude the last ten hybrid composition functions in order to save space and also because they incur excessive computational time in 100 dimensions. The maximum number of FEs was set to 3×10^5 for 30-D, 5×10^5 for 50-D, and 1×10^6 for 100-D functions according to the guidelines provided in CEC 2005 special session [34], [35]. All of the simulations were done on a Pentium core 2 duo machine with 2-GB RAM and 2.23-GHz speed. All of the DE variants were started from the same initial population in each run so that any difference in their performances may be attributed to their internal search operators only.

D. Results on Numerical Benchmarks

Tables I–IV show the mean and the standard deviation of the best-of-run errors for 50 independent runs of each of the seven algorithms on 25 numerical benchmarks for $D = 30$ and $D = 50$, respectively. Table V reports the same values for functions f_1 to f_{14} tested in $D = 100$. Note that the best-of-the-run error corresponds to the absolute difference between

TABLE II
MEAN AND STANDARD DEVIATION OF THE ERROR VALUES FOR $f_{16}-f_{25}(30D)$. THE BEST ENTRIES ARE MARKED IN BOLDFACE

Algorithms \ Func	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}
	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)
DE/rand/1/bin	2.8228e+02 (1.1328e+01)	3.0942e+02 (1.5698e+01)	9.1342e+02 (8.4336e-01)	9.1984e+02 (1.2190e+00)	9.1317e+02 (1.1642e+00)
DE/current-to-best/1/bin	2.2282e+02 (1.6392e+02)	2.3465e+02 (1.5837e+02)	9.5042e+02 (2.1063e+01)	9.5182e+02 (2.1142e+01)	9.4110e+02 (2.9314e+01)
JADE	7.4383e+01 (3.9432e+01)	8.4619e+01 (3.5763e+01)	8.1679e+02 (1.6523e-01)	8.1644e+02 (1.5419e-01)	8.1697e+02 (1.7231e-01)
jDE	1.2854e+02 (4.0730e+01)	1.6189e+02 (4.7251e+01)	8.6111e+02 (1.8705e+00)	8.4801e+02 (3.1790e+00)	8.5466e+02 (9.54e-01)
SaDE	1.379e+02 (1.702e+01)	1.509e+03 (9.363e+02)	9.544e+02 (3.438e+01)	8.458e+02 (6.215e+01)	2.040e+03 (8.768e+02)
DEGL	2.0350e+02 (1.2068e+02)	1.4519e+02 (7.3247e+01)	9.2253e+02 (1.5761e+00)	9.1960e+02 (1.3383e+00)	9.1989e+02 (3.0719e+01)
MDE_pBX	**5.2307e+01 (3.8872e+00)**	**8.2328e+01 (3.9757e+01)**	**7.1626e+02 (1.5209e-01)**	**8.0625e+02 (1.5340e-01)**	**6.1942e+02 (1.6990e-01)**
Algorithms \ Func	f_{21}	f_{22}	f_{23}	f_{24}	f_{25}
	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)
DE/rand/1/bin	5.8135e+02 (2.6247e+01)	9.6425e+02 (1.1439e+01)	6.2131e+02 (3.0647e+01)	3.1434e+02 (3.2249e+01)	9.8643e+02 (2.1775e+01)
DE/current-to-best/1/bin	8.3154e+02 (2.6246e+02)	9.3438e+02 (4.1464e+01)	8.5963e+02 (2.8782e+02)	3.0493e+02 (3.5563e+01)	9.6754e+02 (1.306e+01)
JADE	8.5838e+02 (1.1013e+00)	5.0762e+02 (1.6550e+00)	8.6558e+02 (6.6102e-01)	2.1141e+02 (1.5664e+01)	2.1052e+02 (2.5584e+00)
jDE	8.6002e+02 (1.1361e+00)	5.0340e+02 (2.9115e+00)	6.1835e+02 (4.5481e+00)	2.1081e+02 (2.8842e+00)	9.761e+02 (2.409e+01)
SaDE	1.730e+03 (5.118e+02)	1.582e+03 (4.252e+02)	5.506e+02 (2.489e+01)	2.0985e+02 (1.2455e-04)	5.0012e+02 (5.683e-02)
DEGL	7.5734e+02 (1.0132e+01)	9.2154e+02 (9.841e+01)	7.524e+02 (4.001e+01)	6.5600e+02 (5.0123e+01)	9.889e+02 (3.015e+01)
MDE_pBX	**5.0000e+02 (0)**	**5.0021e+02 (4.5755e-01)**	**5.3416e+02 (7.8384e-04)**	**2.0000e+02 (0)**	**2.0962e+02 (3.6271e+00)**

the best-of-the-run value $f(\vec{X}_{\rm best})$ and the actual optimum f^* of a particular objective function, i.e., $|f(\vec{X}_{\rm best}) - f^*|$. A nonparametric statistical test called Wilcoxon's rank-sum test for independent samples [36], [37] is conducted at the 5% significance level in order to judge whether the results obtained with the best performing algorithm differ from the final results of rest of the competitors in a statistically significant way. The P-values obtained through the rank sum test between the best algorithm and each of the remaining algorithms over all the benchmark functions are presented in Table VI for $30D$ problems, Table VII for $50D$ problems, and Table VIII for $100D$ problems. In these tables, NA stands for *not applicable* and occurs for the best performing algorithm itself in each case. If the P-values are less than 0.05 (5% significance level), it is a strong evidence against the null hypothesis, indicating that the better final objective function values achieved by the best algorithm in each case are statistically significant and have not occurred by chance [37].

Tables I and II indicate that, considering the mean of the error values for $30D$ problems, MDE_pBX outperformed all the contestant algorithms in a statistically significant fashion (as revealed by Table VI) over 22 functions. It managed to remain second best for function f_{24} being outperformed by SaDE alone. In function f_8, the performance of MDE_pBX remained comparable to that of DEGL, with the former, however, achieving the lowest standard deviation once again. A close scrutiny of Tables III and IV reveals that the performance of MDE_pBX was not affected in a worse way with the enhancement of search space dimensionality to 50. It achieved statistically superior performance compared to all other competitor algorithms for 23 benchmark instances in $50D$ problems as revealed by Table VII. It has managed to remain second best for function f_{12} being outperformed by SaDE alone. In function f_{24}, the performance of MDE_pBX remained comparable to those of JADE, jDE, and SaDE, with the former, however, achieving the lowest standard deviation once again. Note that MDE_pBX has shown its superiority in case of rotated and hybrid functions as well as functions with noise in fitness. Therefore, function rotation, incorporating multiplicative noise in them as well as composition of functions, does not hamper the performance of the proposed algorithm significantly. Table V indicates that, in 100 dimensions, in 12 out of 14 cases, MDE_pBX with the same parametric setup could outperform all other DE variants in a statistically meaningful way. Only for functions f_9 and f_{12}, it managed to remain the second best algorithm, being beaten by JADE and SaDE, respectively.

For further illustration, in Fig. 1, we show the convergence graphs for the median run of (where the runs were sorted according to the final best error values achieved in each) the algorithms on six benchmarks in 30D. As evident from the convergence characteristics, the overall convergence speed of MDE_pBX is the best among the contestant algorithms. We restrained from giving all the graphs in order to save space.

TABLE III
MEAN AND STANDARD DEVIATION OF THE ERROR VALUES FOR $f_1 - f_{15}(50D)$. THE BEST ENTRIES ARE MARKED IN BOLDFACE

Func / Algorithms	f_1	f_2	f_3	f_4	f_5
	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)
DE/rand/1/bin	9.6015e-05 (1.2837e-05)	3.960e+03 (9.307e+02)	5.469e+07 (1.328e+07)	1.180e+04 (3.332e+03)	8.709e+03 (6.938e+02)
DE/current-to-best/1/bin	2.1443e-06 (6.1254e-06)	2.136e+03 (1.128e+03)	1.030e+07 (6.375e+06)	8.5675e+03 (2.8624e+03)	7.462e+03 (1.326e+03)
JADE	7.4651e-14 (2.4190e-14)	5.6310e-04 (7.8233e-06)	8.7156e+04 (3.6847e+04)	3.160e+03 (4.134e-01)	3.055e+03 (5.485e+02)
jDE	3.1544e-09 (4.9946e-09)	5.202e+03 (1.486e+03)	2.977e+07 (5.744e+06)	1.0194e+04 (2.1828e-01)	4.206e+03 (5.088e+02)
SaDE	1.4872e-11 (2.8335e-12)	2.280e-03 (8.545e-03)	7.179e+05 (1.007e+06)	9.778e+04 (9.835e+01)	5.992e+03 (4.464e+02)
DEGL	6.4679e-10 (9.864e-11)	1.2960e-05 (9.5612e-06)	2.311e+05 (1.032e+05)	2.8851e+04 (1.8932e+01)	6.093e+03 (6.840e+02)
MDE_*p*BX	**4.9303e-32 (8.3354e-20)**	**4.4563e-06 (8.7963e-11)**	**3.5438e+04 (1.7823e+04)**	**1.2075e+02 (2.3453e-06)**	**2.0758e+03 (1.9111e+02)**
Func / Algorithms	f_6	f_7	f_8	f_9	f_{10}
	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)
DE/rand/1/bin	4.9162e+01 (1.182e+01)	6.1953e+03 (4.594e-12)	2.1142e+01 (3.330e-02)	3.468e+02 (1.199e+01)	3.763e+02 (1.578e+01)
DE/current-to-best/1/bin	1.0782e+07 (1.237e+07)	6.6691e+03 (1.795e+02)	2.1133e+01 (4.841e-02)	2.406e+02 (2.939e+01)	2.5467e+02 (7.6634e+01)
JADE	1.5413e+01 (1.0642e+01)	6.1932e+03 (1.840e+00)	2.1136e+01 (3.251e-02)	1.352e+02 (2.591e+00)	1.935e+02 (2.060e+01)
jDE	4.1758e+01 (8.910e+00)	6.3114e+03 (1.596e+01)	2.1132e+01 (3.807e-02)	1.716e+02 (1.409e+01)	1.9597e+02 (5.6236e+01)
SaDE	1.1337e+01 (1.044e+01)	6.1951e+03 (4.594e-12)	2.1132e+01 (3.458e-02)	1.148e+02 (1.266e+01)	6.342e+01 (1.287e+01)
DEGL	1.3452e+01 (1.108e+01)	6.1953e+03 (4.594e-12)	2.1131e+01 (3.917e-02)	1.620e+02 (1.743e+01)	1.0217e+02 (3.5590e+01)
MDE_*p*BX	**7.9745-01 (1.0112e+00)**	**6.1835e+03 (2.0912e+00)**	**2.0422e+01 (3.1241e-02)**	**1.0642e+02 (1.2346e+01)**	**3.6818e+01 (1.3496e+01)**
Func / Algorithms	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)	Mean(Std)
DE/rand/1/bin	7.264e+01 (1.212e+00)	2.049e+06 (5.925e+05)	3.596e+01 (1.446e+00)	2.339e+01 (1.486e-01)	5.090e+02 (7.981e+01)
DE/current-to-best/1/bin	4.950e+01 (4.451e+00)	2.505e+05 (1.137e+05)	3.412e+01 (8.9042e+00)	2.286e+01 (4.091e-01)	4.989e+02 (5.001e+01)
JADE	6.208e+01 (1.744e+00)	1.768e+05 (7.105e+04)	2.3112e+01 (4.784e-01)	2.284e+01 (2.5486e-01)	3.769e+02 (8.764e+01)
jDE	7.330e+01 (1.008e+00)	1.473e+05 (1.928e+05)	2.5603e+01 (1.322e+00)	2.309e+01 (2.8437e-01)	4.000e+02 (0.000e+00)
SaDE	6.634e+01 (1.485e+00)	**8.781e+03 (7.092e+03)**	2.771e+01 (4.112e+00)	2.284e+01 (2.0634e-01)	3.8827e+01 (1.0755e+02)
DEGL	6.290e+01 (1.360e+01)	5.781e+04 (4.566e+04)	3.063e+01 (4.361e+00)	2.262e+01 (3.3750e-01)	3.8982e+02 (4.9284e+01)
MDE_*p*BX	**4.1328e+01 (1.545e+00)**	1.0779e+04 (8.4902e+03)	**2.0628e+01 (1.2654e+00)**	**2.1720e+01 (2.0801e-01)**	**3.6670e+02 (6.1264e+01)**

E. Choosing a Group Size in the DE/Current-to-gr_best/1 Scheme

The appropriate selection of the group size (q) influences the balance between exploration and exploitation, which is inevitable for any evolutionary population-based stochastic search technique such as DE. The performance of MDE_*p*BX is dependent on the selection of q, which remains an open problem when solving any given single-objective optimization problem. Some empirical guidelines may, however, be provided based on the fact that, if the value of group size (q) is large (near the population size), the proposed mutation scheme DE/current-to-gr_best/1 basically becomes identical to the current-to-best scheme. This hampers the explorative power, and the algorithm may be trapped at some local optimum that is not the actual global optimum. The reason is that, if q is at par with the population size, the probability that the best of randomly chosen $q\%$ vectors is similar to the globally best vector of the entire population will be high. This drives most of the vectors toward a specific point in the search space resulting in premature convergence. Again, a very small value of q runs the risk of losing the exploitative capacity of the algorithm. This is due to the fact that, with the value of q being small, the best of randomly chosen $q\%$ vectors may not be a fitter agent of the population, resulting in the creation of poor donor vectors, and the convergence performance may get hampered.

TABLE IV
MEAN AND STANDARD DEVIATION OF THE ERROR VALUES FOR $f_{16}-f_{25}(50D)$. THE BEST ENTRIES ARE MARKED IN BOLDFACE

Func / Algorithms	f_{16} Mean(Std)	f_{17} Mean(Std)	f_{18} Mean(Std)	f_{19} Mean(Std)	f_{20} Mean(Std)
DE/rand/1/bin	2.7343e+02 (1.0498e+01)	3.7286e+02 (3.1287e+01)	9.9043e+02 (4.8709e+01)	9.4100e+02 (3.8003e+01)	9.8536e+02 (4.4965e+01)
DE/current-to-best/1/bin	2.5387e+02 (1.4757e+01)	2.5234e+02 (5.7296e+01)	9.2089e+02 (5.6632e+01)	9.2667e+02 (5.9865e+01)	9.3586e+02 (5.3925e+01)
JADE	1.437e+02 (5.2267e+01)	1.896e+02 (3.8745e+01)	9.206e+02 (1.893e+00)	9.6031e+02 (2.5236e+01)	9.8672e+02 (1.8675e+02)
jDE	2.716e+02 (4.7190e+00)	3.059e+02 (1.163e+01)	9.145e+02 (3.163e+01)	9.2090e+02 (1.0406e+01)	9.9121e+02 (1.5365e+01)
SaDE	1.5420e+01 (6.1686e+01)	1.934e+02 (2.9679e+00)	9.041e+02 (5.208e+01)	9.3493e+02 (1.9639e+01)	9.3167e+02 (2.0137e+01)
DEGL	1.3153e+02 (1.9986e+01)	1.7659e+02 (2.3653e+01)	9.6067e+02 (2.8458e+01)	9.1430e+02 (2.0105e+01)	9.2196e+02 (4.5874e+01)
MDE_pBX	**1.1142e+02 (3.5706e+01)**	**1.2502e+02 (1.8344e+01)**	**8.2640e+02 (8.5217e+01)**	**9.0496e+02 (2.8204e+01)**	**9.0280e+02 (2.5078e+01)**
Func / Algorithms	f_{21} Mean(Std)	f_{22} Mean(Std)	f_{23} Mean(Std)	f_{24} Mean(Std)	f_{25} Mean(Std)
DE/rand/1/bin	9.1108e+02 (5.7474e+02)	9.9463e+02 (1.3465e+01)	9.1185e+02 (2.5986e+01)	7.9849e+02 (2.4586e+01)	1.7586e+02 (4.5365e+00)
DE/current-to-best/1/bin	8.9465e+02 (1.7465e+02)	9.3443e+02 (1.1026e+01)	9.2645e+02 (2.5986e+02)	7.9456e+02 (1.3642e+01)	1.7846e+03 (6.7654e+00)
JADE	8.523e+02 (3.5175e+02)	9.1370e+02 (2.4356e+01)	8.103e+02 (2.4572e+02)	**2.000e+02 (0)**	1.6632e+03 (5.5842e+00)
jDE	8.0619e+02 (1.0896e+02)	9.796e+02 (1.4851e+01)	8.3044e+02 (1.0787e+02)	**2.000e+02 (0)**	1.728e+03 (6.2562e+00)
SaDE	8.6400e+02 (1.5779e+02)	9.7245e+02 (3.3383e+01)	8.6405e+02 (1.5266e+02)	**2.0000e+02 (0)**	1.7586e+03 (3.1453e+00)
DEGL	8.3600e+02 (2.1772e+02)	9.4242e+02 (3.5647e+01)	8.3934e+02 (1.6620e+02)	7.2465e+02 (8.3066e+01)	1.671e+03 (6.5096e+00)
MDE_pBX	**5.0000e+02 (0.000e+00)**	**8.987e+02 (9.075e+00)**	**5.000e+00 (0.000e+00)**	**2.0000e+02 (0)**	**9.6149e+02 (6.0040e+00)**

TABLE V
MEAN AND STANDARD DEVIATION OF THE ERROR VALUES FOR $f_1-f_{14}(100D)$. THE BEST ENTRIES ARE MARKED IN BOLDFACE

Algorithms / Func	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	jDE	SaDE	DEGL	MDE_pBX
f_1	2.9467e-05 (3.0947e-04)	8.4735e-06 (9.4927e-05)	6.4825e-10 (4.0249e-09)	7.4627e-07 (3.9371e-09)	8.2615e-08 (5.0445e-09)	9.6844e-07 (4.0937e-08)	**3.8265e-15 (5.9473e-15)**
f_2	8.9371e+04 (7.4925e+04)	9.0927e+03 (8.4736e-05)	3. 4923e+03 (8.4725e-06)	5.9371e+03 (8.4625e-07)	2.9471e+04 (7.7352e-03)	8.9261e+04 (8.4563e-06)	**8.6374e+02 (6.9372-09)**
f_3	9.7635e+07 (8.4625+03)	9.8525e+06 (5.0948e+03)	2.9371e+06 (7.4728e+04)	8.9372e+06 (3.0927e+04)	7.9171e+06 (8.0936e+02)	4.8326e+07 (5.0945e+04)	**7.7823e+05 (7.3812e+05)**
f_4	3.0937e+05 (4.9785e+03)	7.5821e+04 (8.6745e+03)	5.0342e+04 (6.9785e-03)	7.9261e-04 (2.7456e+03)	6.0283e+04 (6.9573e+03)	1.9372e+05 (4.9463e+04)	**3.2545e+04 (3.9637e+03)**
f_5	1.0675e+06 (4.8437e+03)	5.0967e+05 (5.8453e+03)	7.5251e+05 (3.9464e+03)	2.9361e+05 (6.4536e+03)	9.7364e+05 (2.0936e+03)	9.0936e+05 (6.9382e+03)	**1.5285e+05 (4.8463+03)**
f_6	1.8946e+05 (4.9372 e+01)	9.3967e+04 (7.4925 e+01)	7.5329e+04 (7.3725+01)	8.1876e+04 (8.4752e+00)	2.0172e+04 (8.2514+01)	4.2684e+04 (5.8352e+01)	**7.2163e+03 (6.9845e-01)**
f_7	1.8574e+05 (4.0382 e+02)	1.6738e+05 (8.4735 e+02)	9.0417e+04 (8.4735e+02)	1.1625e+05 (7.4637e+02)	9.8463e+04 (8.4623e+02)	1.2383e+05 (9.4573e+02)	**4.8723e+04 (8.6583e+02)**
f_8	2.2497e+01 (3.0423 e+01)	2.2308e+01 (8.4725e+00)	2.1964e+01 (9.4673e+01)	2.2197e+01 (7.4627e+00)	2.2075e+01 (9.3736e+00)	2.2210e+01 (4.8252e+00)	**2.1079e+01 (1.9461e+00)**
f_9	9.3752e+02 (6.7322e+01)	9.4065e+02 (7.6695e+01)	**8.9272e+02 (4.5043e+01)**	9.3264e+02 (7.3645e+01)	9.2737e+02 (5.1369e+01)	9.5591e+03 (3.9274e+01)	8.7123e+02 (5.4873e+00)
f_{10}	9.7631e+02 (1.5787e+01)	7.5467e+02 (5.64334e+01)	5.935e+02 (8.8760e+01)	6.9597e+02 (7.7653e+01)	8.3426e+02 (1.6378e+01)	8.2017e+02 (3.5590e+01)	**4.5629e+02 (7.9876e+01)**
f_{11}	9.2648e+01 (1.9212e+00)	6.9150e+01 (6.6451e+00)	9.1208e+01 (4.7744e+00)	7.3630e+01 (9.4008e+00)	8.4684e+01 (3.485e+00)	8.1290e+01 (1.360e+01)	**5.4264e+01 (5.485e+00)**
f_{12}	9.049e+06 (7.925e+05)	6.505e+05 (4.137e+05)	6.768e+05 (2.105e+04)	7.473e+05 (7.928e+05)	**9.0781e+04 (5.9092e+03)**	8.8781e+05 (7.1566e+04)	2.0932e+05 (3.9865e+04)
f_{13}	5.2926e+01 (4.4346e+00)	4.4132e+01 (3.5042e+00)	3.4141e+00 (4.0834e+00)	3.8650e+01 (4.322e+01)	3.5713e+00 (5.112e+00)	4.063e+01 (2.361e+01)	**3.2723e+01 (6.9843e+00)**
f_{14}	4.4339e+01 (1.4836e-01)	4.5186e+01 (4.4091e-01)	4.2684e+01 (3.8163e-01)	4.3309e+01 (2.7163e-01)	4.2874e+01 (2.5343e-01)	4.2662e+01 (6.9834e-01)	**4.2047e+01 (2.6334e-01)**

TABLE VI
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST FOR ALL THE BENCHMARK PROBLEMS ($30D$)

P-values	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	*j*DE	SADE	DEGL	MDE_pBX
f_1	2.535e-07	4.565e-08	1.356e-21	3.475e-18	4.563e-18	3.672e-16	NA
f_2	3.311e-20	8.473e-20	1.254e-21	3.380e-11	7.066e-18	3.311e-20	NA
f_3	6.595e-08	4.375e-08	2.321e-04	3.311e-02	5.130e-05	3.982e-02	NA
f_4	5.885e-12	9.746e-11	8.985e-03	3.583e-05	3.895e-05	7.066e-18	NA
f_5	8.645e-20	3.238e-08	5.628e-04	3.311e-03	9.241e-04	3.311e-20	NA
f_6	3.311e-10	6.475e-10	2.424e-10	3.380e-11	3.311e-10	3.311e-10	NA
f_7	3.311e-20	8.438e-20	3.238e-03	3.380e-11	9.957e-06	3.311e-10	NA
f_8	7.066e-18	8.348e-18	1.518e-04	7.616e-10	3.775e-10	7.616e-02	NA
f_9	3.311e-20	5.475e-20	NA	2.345e-08	3.312e-20	3.312e-20	1.245e-22
f_{10}	7.066e-18	8.437e-18	7.066e-18	4.066e-18	1.066e-18	5.138e-18	NA
f_{11}	2.658e-20	5.475e-20	7.066e-18	2.658e-07	7.066e-18	7.066e-18	NA
f_{12}	7.066e-18	4.576e-18	7.066e-18	7.066e-18	NA	7.551e-06	6.284e-05
f_{13}	1.009e-16	2.385e-16	1.390e-16	1.009e-16	3.6271e-06	5.23e-03	NA
f_{14}	5.319e-04	7.374e-04	7.066e-18	5.319e-04	7.066e-18	1.514e-16	NA
f_{15}	1.128e-07	3.485e-06	2.391e-03	1.128e-07	1.584e-09	7.066e-18	NA
f_{16}	2.949e-11	3.575e-10	3.077e-07	2.949e-11	7.066e-18	9.913e-09	NA
f_{17}	2.083e-18	8.876e-12	5.294e-03	2.083e-13	7.066e-18	3.194e-05	NA
f_{18}	7.066e-18	6.485e-17	1.597e-02	7.066e-18	1.349e-16	7.066e-18	NA
f_{19}	2.291e-15	2.485e-14	1.842e-11	2.291e-15	5.370e-10	7.066e-18	NA
f_{20}	7.066e-18	8.395e-16	7.101e-03	7.066e-18	1.349e-16	7.066e-18	NA
f_{21}	4.425e-09	8.625e-08	5.258e-10	4.425e-09	1.378e-06	3.016e-04	NA
f_{22}	5.288e-14	7.183e-12	6.618e-02	5.288e-14	1.070e-16	1.524e-11	NA
f_{23}	5.922e-11	9.812e-09	7.066e-18	5.922e-11	3.583e-05	7.066e-18	NA
f_{24}	4.576e-13	5.465e-16	7.066e-18	5.885e-06	NA	4.065e-02	5.231e-02
f_{25}	7.066e-18	8.374e-16	7.066e-18	7.066e-18	7.066e-18	7.066e-18	NA

TABLE VII
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST FOR ALL THE BENCHMARK PROBLEMS($50D$)

P-values	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	*j*DE	SADE	DEGL	MDE_*p*BX
f_1	1.234e-23	4.936e-16	5.675e-15	2.987e-16	1.234e-18	3.465e-15	NA
f_2	3.465e-18	8.734e-18	2.4398e-14	3.380e-17	6.798e-16	3.654e-15	NA
f_3	6.595e-18	9.015e-08	4.5867e-02	3.311e-06	7.251e-14	3.4527e-16	NA
f_4	2.873e-18	7.063e-18	2.654e-22	3.583e-15	3.895e-21	7.066e-18	NA
f_5	3.515e-17	5.384e-16	4.675e-22	3.311e-20	3.456e-15	2.654e-12	NA
f_6	3.311e-20	4.364e-22	2.424e-17	3.380e-11	3.311e-20	3.311e-20	NA
f_7	3.311e-20	7.384e-20	3.238e-03	3.380e-11	9.957e-06	3.311e-20	NA
f_8	7.066e-18	8.374e-17	1.518e-10	7.616e-10	3.775e-02	2.616e-02	NA
f_9	3.311e-20	7.894e-19	5.678e-22	2.453e-17	3.312e-04	3.465e-15	NA
f_{10}	7.066e-18	3.384e-17	7.066e-18	4.066e-18	1.066e-18	5.138e-18	NA
f_{11}	2.658e-20	5.348e-18	7.066e-18	2.658e-07	7.066e-18	7.066e-18	NA
f_{12}	6.983e-07	6.495e-06	8.163e-09	7.066e-08	NA	5.981e-17	5.974e-13
f_{13}	1.009e-16	9.365e-14	1.390e-04	1.009e-16	1.9876e-18	5.243e-03	NA
f_{14}	5.319e-04	1.267e-08	7.066e-18	5.319e-04	7.066e-18	1.514e-16	NA
f_{15}	1.128e-07	2.354e-06	2.391e-03	1.128e-07	1.584e-09	6.284e-09	NA
f_{16}	2.949e-11	5.475e-10	3.077e-07	2.949e-11	7.066e-18	9.913e-09	NA
f_{17}	2.083e-03	8.864e-04	5.294e-03	2.083e-03	7.066e-18	3.194e-15	NA
f_{18}	5.346e-18	9.464e-17	1.597e-02	7.066e-18	1.835e-18	7.066e-18	NA
f_{19}	2.291e-15	7.365e-14	1.842e-11	2.291e-15	5.370e-10	7.066e-18	NA
f_{20}	7.856e-18	9.854e-17	7.101e-03	7.126e-18	1.349e-16	6.154e-18	NA
f_{21}	4.425e-09	8.264e-10	5.258e-10	4.425e-09	1.378e-06	3.016e-02	NA
f_{22}	5.288e-14	9.536e-13	6.618e-02	5.288e-14	1.070e-16	1.524e-11	NA
f_{23}	5.922e-11	8.673e-10	7.066e-18	5.922e-11	3.583e-14	7.066e-18	NA
f_{24}	4.885e-05	4.586e-06	NA	5.417e-03	NA	5.065e-03	NA
f_{25}	7.066e-18	6.384e-16	1.835e-18	7.066e-18	7.066e-18	7.066e-18	NA

In Fig. 2, we show the overall success rate of the algorithm MDE_pBX for a group size varying from 5% to 65% of Np (population size) over 30-D multimodal functions f_6, f_7, f_{11}, and f_{14}. Success rate means the percentage of runs that successfully yielded a final accuracy below a certain tolerance within the allotted number of FES for different values of q. The tolerance for both f_6 and f_7 is kept at $1.0e-02$, while the threshold values for f_{11} and f_{14} are kept at 25 and 13,

TABLE VIII
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST FOR ALL THE BENCHMARK PROBLEMS$(100D)$

P-values	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	*j*DE	SADE	DEGL	MDE_pBX
f_1	4.987e-21	6.384e-20	4.586e-17	3.384e-15	3.293e-16	4.927e-18	NA
f_2	3.875e-20	6.984e-21	1.093e-21	3.080e-11	7.804e-18	3.092e-20	NA
f_3	6.035e-18	5.904e-10	9.015e-06	3.036e-08	1.027e-08	9.067e-02	NA
f_4	5.036e-12	8.092e-10	9.036e-06	3.005e-07	4.018e-04	7.066e-16	NA
f_5	6.035e-18	7.018e-08	6.091e-06	8.917e-03	7.204e-05	5.028e-18	NA
f_6	3.304e-12	7.470e-08	1.120e-12	3.385e-10	4.011e-09	4.011e-11	NA
f_7	3.092e-21	7.459e-18	3.130e-06	4.381e-11	6.057e-06	3.305e-12	NA
f_8	7.463e-18	6.348e-16	2.018e-05	8.616e-09	4.605e-10	6.026e-02	NA
f_9	3.304e-19	5.406e-21	NA	1.365e-09	3.310e-18	4.306e-20	1.245e-22
f_{10}	7.069e-16	6.439e-15	5.016e-16	3.066e-16	1.015e-10	4.038e-16	NA
f_{11}	2.008e-18	5.105e-19	6.067e-16	2.092e-08	7.004e-19	7.005e-19	NA
f_{12}	7.086e-17	5.016e-17	4.016e-16	8.016e-19	NA	9.050e-05	4.014e-06
f_{13}	1.083e-12	1.180e-14	1.093e-15	1.018e-18	4.027e-08	6.036e-04	NA
f_{14}	5.018e-06	6.074e-06	7.014e-16	2.019e-06	6.016e-16	2.016e-18	NA

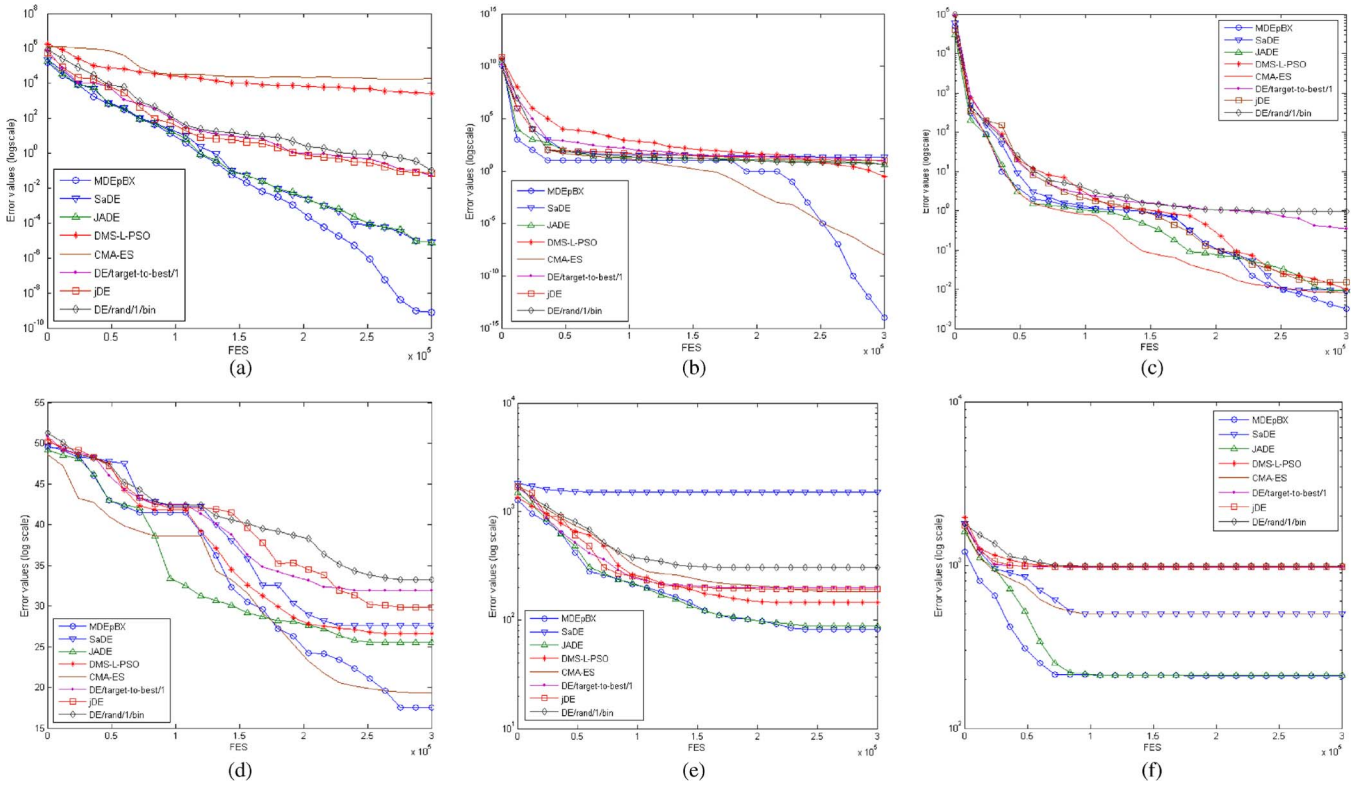

Fig. 1. Progress toward the optimum solution for median run of eight algorithms over six numerical benchmarks (in $30D$). (a) Shifted Schwefel's function f_4. (b) Shifted Rosenbrock's function f_6. (c) Shifted rotated Griewank's function f_7. (d) Shifted rotated Weierstrass function f_{11}. (e) Composition function f_{17} with noise. (f) Composition function f_{25}.

respectively. A thorough experimentation with all the test problems indicates that MDE_pBX provides reasonably accurate results, with high success rates over most of the benchmark problems covered here for the value of q set to 15% of Np (population size). From the aforementioned discussion and in Fig. 2, we perceive that MDE_pBX performs most efficiently when q is set to 15% of the population size.

F. Effectiveness of the Different Components of MDE_pBX

MDE_pBX modifies the basic DE algorithm in three stages: mutation, crossover, and parameter adaptation. In this section, we provide some experimental results to demonstrate the effectiveness of each of these modifications. For this purpose, we consider the following three variants of MDE_pBX:

1) MDE_pBX without parameter adaptation (i.e., we set $F_m = 0.8$ and $Cr_m = 0.9$ throughout the evolution process) referred to as nonadaptive MDE_pBX;
2) MDE_pBX where the p-best crossover is replaced by conventional binomial crossover denoted as MDE;
3) MDE_pBX with the mutation strategy DE/current-to-best/1 instead of DE/current-to-gr_best/1 named as ADE_pBX (ADE with p-best crossover).

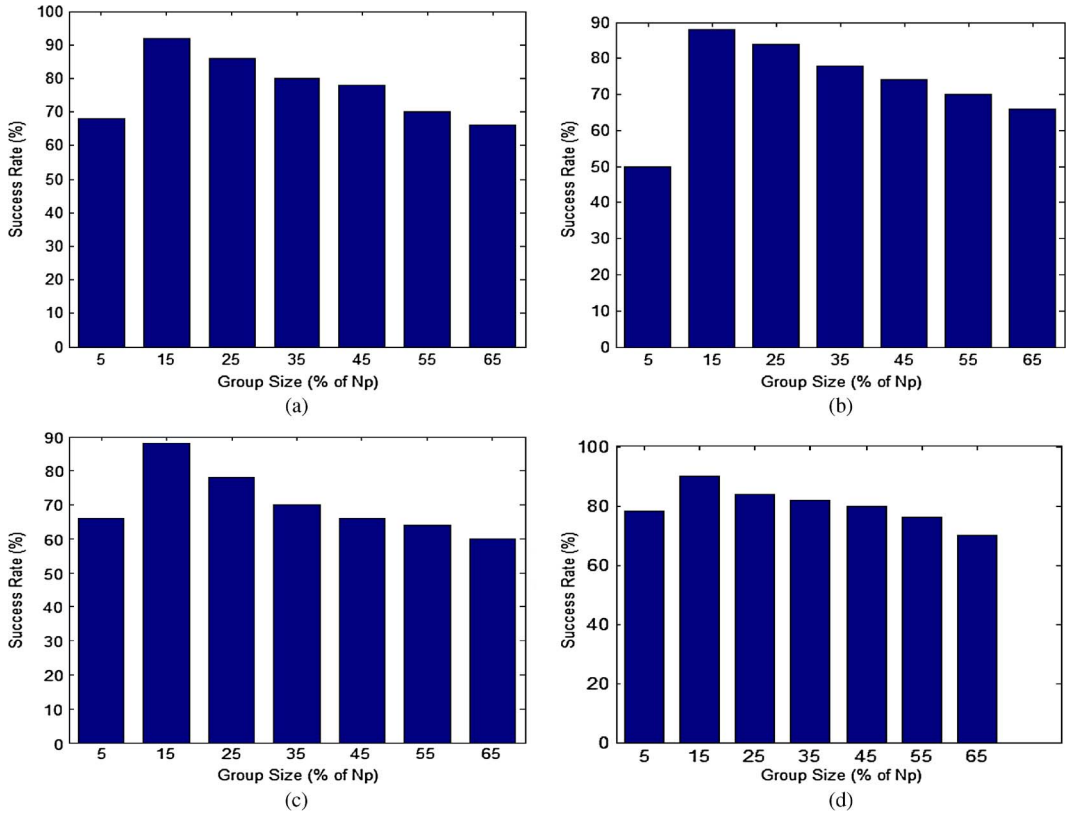

Fig. 2. Variation of the overall success rate of MDE_pBX with increasing value of the group size in DE/current-to-gr_best mutation scheme. (a) Shifted Rosenbrock's function f_6. (b) Shifted rotated Griewank's function f_7 without bounds. (c) Shifted rotated Weierstrass function f_{11}. (d) Shifted rotated expanded Schaffer's F6 function f_{14}.

Apart from the aforementioned three variants, in order to show that the parameter adaptation schemes of MDE_pBX are superior than that of JADE, we also present here the results of the MDE_pBX algorithm where the original parameter adaptation schemes of MDE_pBX are replaced with that of JADE. Furthermore, in order to demonstrate the effectiveness of the small random perturbations applied to the weight terms w_F and w_{Cr} in adaptation of F_m and Cr_m in MDE_pBX, we consider another variant of MDE_pBX, i.e., MDE_pBX, with w_F and w_{Cr} kept constant at 0.8 and 0.9, respectively. Parameter p in the novel p-best crossover scheme is linearly decreased with generations, as shown in (7). In order to clearly show the effectiveness of this linear decrement of p with generations, the following two variants of MDE_pBX are considered:

1) MDE_pBX with the value of p kept constant at 10% of the population size;
2) MDE_pBX with the value of p linearly increasing with generations from 1 to $Np/2$.

The mean and the standard deviation (within parentheses) of the best-of-run errors for 50 independent runs of MDE_pBX and all the variants of MDE_pBX considered in this section are presented in Table IX(a) for test functions $f_1 - f_{10}$, f_{16}, and f_{25} in 30 dimensions. To obtain the results shown in Table IX(a), all algorithms were run from the same initial population in every run so that any difference of their performances may be attributed to their internal search operators. It is to be noted that, for linearly increasing p, premature convergence may occur as parameter p takes on small values, causing huge exploitation instead of exploration at the initial stages of the search. Again, for a constant value of p, MDE_pBX is unable to keep a balanced ratio of exploration and exploitation throughout the search process. Hence, for both cases, the performance of MDE_pBX will deteriorate to some extent, as clearly demonstrated in Table IX(a). Our experiments on 25 numerical benchmarks indicate that, in majority of the cases, the combination of the DE/current-to-gr_best/1 mutation scheme, the p-best crossover with linearly decreasing p, and the proposed adaptation schemes for F and Cr yields the best results.

Although we provide the results for 12 functions in Table IX(a) for saving space, the trend remains the same for the entire 25 benchmark functions, where the performance of the combination of the DE/current-to-gr_best/1 mutation scheme, the p-best crossover with linearly decreasing p, and the proposed adaptation schemes for F and Cr, i.e., the MDE_pBX algorithm is the best, and performance deteriorates whenever any component of the MDE_pBX is removed or any parameter being tuned in an alternative fashion.

In order to further demonstrate the fact whether each component of MDE_pBX is really effective, we integrate each component with DE/current-to-best/1/bin and verify whether the performance of DE/current-to-best/1/bin integrated with each of these components is improved. Our experimental results on 12 numerical benchmarks, as shown in Table IX(b), indicate that, in majority of the cases, the performance of DE/

TABLE IX
(a) MEAN AND STANDARD DEVIATION OF THE BEST-OF-RUN SOLUTIONS OF 50 INDEPENDENT RUNS ON f_1-f_{10}, f_{16}, AND f_{25} IN $30D$ (THE BEST ENTRIES ARE MARKED IN BOLD) FOR TESTING THE EFFECTIVENESS OF DIFFERENT COMPONENTS OF MDE $-$ pBX. (b) MEAN AND STANDARD DEVIATION OF THE BEST-OF-RUN SOLUTIONS FOR 50 INDEPENDENT RUNS ON f_1-f_{10}, f_{16}, AND f_{25} IN $30D$ (THE BEST ENTRIES ARE MARKED IN BOLD) FOR TESTING THE EFFECTIVENESS OF DIFFERENT COMPONENTS OF MDE $-$ pBX

(a)

Func / Algorithms	f_1 Mean(Std)	f_2 Mean(Std)	f_3 Mean(Std)	f_4 Mean(Std)	f_5 Mean(Std)	f_6 Mean(Std)
MDE_pBX	**1.3429e-62 (2.4352e-61)**	**1.9981e-26 (2.4429e-26)**	**2.0977e+03 (1.2699e+03)**	**6.9268e-08 (8.9742e-08)**	**2.2057e+02 (1.6754e+02)**	**3.9870e-01 (1.0815e+00)**
Non-adaptive MDE_pBX	2.4576e-35 (3.4756e-35)	5.6478e-12 (8.3645e-12)	7.8364e+04 (4.5366e+04)	4.5634e-03 (1.0242e-03)	3.0582e+02 (1.0746e+02)	1.0265e+01 (7.7254e+00)
MDE	2.4576e-35 (3.4756e-35)	4.6734e-14 (2.2846e-14)	5.2435e+04 (3.8745e+04)	3.7645e-05 (3.3453e-05)	2.6423e+02 (1.2401e+02)	4.0345e+00 (1.0352e+01)
ADE_pBX	4.5764e-42 (2.6394e-42)	4.3654e-11 (7.4563e-11)	9.0374e+04 (1.4538e+04)	5.3645e-06 (9.4727e-06)	2.8329e+02 (1.8267e+02)	5.0972e+00 (1.0202e+01)
MDE_pBX with p constant	3.8583e-39 (6.7484e-39)	3.475e-18 (6.485e-18)	4.7366e+04 (3.8364e+04)	8.475e-07 (2.394e-07)	2.374e+02 (1.364e+02)	1.0747e+00 (5.756e+00)
MDE_pBX with constant w_F and w_{Cr}	7.846e-49 (5.687e-49)	6.574e-08 (8.475e-08)	7.384e+04 (4.856e+04)	6.573e-04 (9.374e-04)	3.465e+02 (1.475e+02)	1.352e+01 (1.836e+01)
MDE_pBX with parameter adaptation schemes of JADE	5.873e-36 (9.586e-37)	6.983e-16 (7.874e-16)	4.9465e+04 (2.865e+04)	7.946e-07 (1.783e-07)	2.574e+02 (1.743e+02)	1.1624e+00 (6.283e+00)
Func / Algorithms	f_7 Mean(Std)	f_8 Mean(Std)	f_9 Mean(Std)	f_{10} Mean(Std)	f_{16} Mean(Std)	f_{25} Mean(Std)
MDE_pBX	**6.6472e-03 (9.0313e-03)**	**2.0000e+01 (6.7185e-07)**	**1.0342e-09 (3.2346e-10)**	**1.4890e+01 (8.9159e-01)**	**5.2307e+01 (3.8872e+00)**	**2.0962e+02 (3.6271e+00)**
Non-adaptive MDE_pBX	1.2037e-02 (1.035e-02)	2.0345e+01 (5.9856e-02)	1.9273e-03 (3.4948e-06)	4.4756e+01 (1.3784e+01)	1.1253e+02 (5.2243e+01)	9.4352e+02 (1.0967e+01)
MDE	9.2634e-03 (8.0393e-03)	2.0643e+01 (4.8934e-02)	4.5858e-05 (6.3940e-07)	3.8735e+01 (5.8264e+00)	9.9132e+01 (4.8163e+01)	5.0076e +02 (7.9805e-02)
ADE_pBX	1.3892e-02 (9.2736e-03)	2.0536e+02 (6.9273e-02)	5.3846e-04 (7.2938e-04)	3.9829e+01 (9.2838e+00)	9.1253e+01 (4.5243e+01)	2.1245e+02 (3.0785e+00)
MDE_pBX with p constant	8.379e-03 (9.028e-03)	2.0267e+01 (1.748e-02)	3.9273e-06 (4.3948e-06)	3.637e+01 (7.475e+00)	9.2754e+01 (3.957e+01)	2.1145e+02 (1.0456e+00)
MDE_pBX with constant w_F and w_{Cr}	1.354e-02 (1.025e-02)	2.0946e+01 (5.938e-02)	3.2839e-02 (9.3948e-03)	4.928e+01 (1.047e+01)	1.1834e+02 (6.394e+01)	9.8567e+02 (4.0986e+01)
MDE_pBX with parameter adaptation schemes of JADE	1.017e-02 (8.263e-03)	2.0252e+01 (2.364e-02)	6.2930e-07 (8.9384e-07)	2.947e+01 (7.756e+00)	8.7529e+01 (3.936e+01)	2.1354e+02 (2.0875e+00)

current-to-best/1/bin is improved when each of the components of MDE_pBX is incorporated in it. For saving space, we provide the mean and the standard deviation (within parentheses) of the best-of-run errors for 50 independent runs of DE/current-to-best/1/bin integrated with each of the three components in Table IX(b) for test functions f_1-f_{10}, f_{16}, and f_{25} only in 30 dimensions. Note that, for obtaining the results shown in Table IX(b), all of the algorithms were run from the same initial population in every run, so any difference of their performances may be attributed to their internal search operators.

G. Spacecraft Trajectory Optimization With MDE_pBX

It is well known that DE has been applied to solve a plethora of real-world cybernetic problems; an interested reader can see articles like [6] for a detailed account. Here, the performance of MDE_pBX has been tested on two instantiations of a real-world spacecraft trajectory optimization problem that is also included in IEEE CEC 2011 competition on testing evolutionary algorithms on real-world problems [22]. In mathematical terms, it is a finite-dimensional global optimization problem with nonlinear constraints. It can be used to locate the best possible trajectory that an interplanetary probe equipped with a chemical propulsion engine may take to go from the Earth to another planet or asteroid. The spacecraft is constrained to thrust only at planetary encounters. We consider two instantiations of the problem—*The Messenger* and *The Cassini 2*—both of which are multigravity assist (MGA) trajectory design problems with deep space maneuvers (DSMs) [23]. A detailed description of the problem may be found in [22], [23], and [38].

The constraint on the spacecraft thrusting only at planetary encounters is often unacceptable as it may result in trajectories that are not realistic or that use more propellant than necessary. The MGA-1DSM problem removes most of these limitations. It represents an interplanetary trajectory of a spacecraft equipped with chemical propulsion, able to thrust its engine once at any time during each trajectory leg. Thus, the solutions to this problem are suitable to perform preliminary quantitative calculation for real space missions. This comes with the price of having to solve an optimization problem of larger dimensions. The "Messenger" trajectory optimization problem represents a rendezvous mission to Mercury modeled as an MGA-1DSM

TABLE IX

(Continued). (a) MEAN AND STANDARD DEVIATION OF THE BEST-OF-RUN SOLUTIONS OF 50 INDEPENDENT RUNS ON f_1-f_{10}, f_{16}, AND f_{25} IN $30D$ (THE BEST ENTRIES ARE MARKED IN BOLD) FOR TESTING THE EFFECTIVENESS OF DIFFERENT COMPONENTS OF MDE $-$ pBX. (b) MEAN AND STANDARD DEVIATION OF THE BEST-OF-RUN SOLUTIONS FOR 50 INDEPENDENT RUNS ON f_1-f_{10}, f_{16}, AND f_{25} IN $30D$ (THE BEST ENTRIES ARE MARKED IN BOLD) FOR TESTING THE EFFECTIVENESS OF DIFFERENT COMPONENTS OF MDE $-$ pBX

(b)

Func / Algorithms	f_1 Mean(Std)	f_2 Mean(Std)	f_3 Mean(Std)	f_4 Mean(Std)	f_5 Mean(Std)	f_6 Mean(Std)
DE/current-to-best/1/bin	2.4536e-25 (3.4756e-25)	6.2953e-08 (1.2927e-07)	1.8918e+05 (1.0392e+05)	1.0714e-02 (2.7821e-02)	5.7628e+02 (1.2476e+02)	9.2736e+00 (8.2835e+00)
DE/current-to-gr_best/1/bin (without parameter adaptation)	3.4657e-35 (6.3848e-35)	8.3747e-15 (1.3948e-14)	4.3848e+04 (2.4859e+04)	5.4273e-05 (9.2373e-05)	3.1273e+02 (1.0278e+02)	4.3746e+00 (8.3848e+00)
DE/current-to-best/1/bin with p-best crossover	5.4756e-31 (3.2923e-31)	4.3847e-11 (3.6938e-11)	8.3929e+04 (5.2484e+04)	4.3949e-04 (2.3847e-04)	**2.9834e+02 (1.4354e+02)**	8.3929e+00 (7.2535e+00)
DE/current-to-best/1/bin with parameter adaptation	**4.2838e-42 (5.3984e-42)**	**6.2737e-18 (8.2939e-18)**	**1.0928e+04 (1.3672e+04)**	**7.4843e-06 (3.2039e-06)**	3.0465e+02 (1.1723e+02)	**1.0235e+00 (9.2746e-01)**
Func / Algorithms	f_7 Mean(Std)	f_8 Mean(Std)	f_9 Mean(Std)	f_{10} Mean(Std)	f_{16} Mean(Std)	f_{25} Mean(Std)
DE/current-to-best/1/bin	8.4253e-01 (9.1442e-02)	2.0942e+01 (5.0063e-02)	2.3465e+01 (8.6395e+00)	1.9617e+02 (2.1683e+01)	2.2282e+02 (1.6392e+02)	9.6754e+02 (1.306e+01)
DE/current-to-gr_best/1/bin (without parameter adaptation)	4.0273e-02 (6.9327e-02)	2.0637e+02 (2.1928e-02)	4.3045e-02 (9.0143e-03)	6.8734e+01 (8.3647e+00)	1.0234e+02 (1.8934e+01)	9.0362e+02 (9.8075e+00)
DE/current-to-best/1/bin with p-best crossover	2.0354e-01 (1.2435e-01)	**2.0056e+01 (1.2536e-02)**	2.5643e+00 (5.0987e+00)	4.8253e+01 (6.2383e+00)	9.8354e+01 (1.0354e+01)	5.0072e+02 (1.0254e-02)
DE/current-to-best/1/bin with parameter adaptation	**1.2038e-02 (9.2635e-03)**	2.0104e+01 (3.2837e-02)	**5.7643e-05 (6.0986e-05)**	**3.8465e+01 (4.2038e+00)**	**9.2534e+02 (2.3948e+01)**	**5.0014e+02 (5.0086e-02)**
Func / Algorithms	f_1 Mean(Std)	f_2 Mean(Std)	f_3 Mean(Std)	f_4 Mean(Std)	f_5 Mean(Std)	f_6 Mean(Std)
DE/current-to-best/1/bin	2.4536e-25 (3.4756e-25)	6.2953e-08 (1.2927e-07)	1.8918e+05 (1.0392e+05)	1.0714e-02 (2.7821e-02)	5.7628e+02 (1.2476e+02)	9.2736e+00 (8.2835e+00)
DE/current-to-gr_best/1/bin (without parameter adaptation)	3.4657e-35 (6.3848e-35)	8.3747e-15 (1.3948e-14)	4.3848e+04 (2.4859e+04)	5.4273e-05 (9.2373e-05)	3.1273e+02 (1.0278e+02)	4.3746e+00 (8.3848e+00)
DE/current-to-best/1/bin with p-best crossover	5.4756e-31 (3.2923e-31)	4.3847e-11 (3.6938e-11)	8.3929e+04 (5.2484e+04)	4.3949e-04 (2.3847e-04)	**2.9834e+02 (1.4354e+02)**	8.3929e+00 (7.2535e+00)
DE/current-to-best/1/bin with parameter adaptation	**4.2838e-42 (5.3984e-42)**	**6.2737e-18 (8.2939e-18)**	**1.0928e+04 (1.3672e+04)**	**7.4843e-06 (3.2039e-06)**	3.0465e+02 (1.1723e+02)	**1.0235e+00 (9.2746e-01)**
Func / Algorithms	f_7 Mean(Std)	f_8 Mean(Std)	f_9 Mean(Std)	f_{10} Mean(Std)	f_{16} Mean(Std)	f_{25} Mean(Std)
DE/current-to-best/1/bin	8.4253e-01 (9.1442e-02)	2.0942e+01 (5.0063e-02)	2.3465e+01 (8.6395e+00)	1.9617e+02 (2.1683e+01)	2.2282e+02 (1.6392e+02)	9.6754e+02 (1.306e+01)
DE/current-to-gr_best/1/bin (without parameter adaptation)	4.0273e-02 (6.9327e-02)	2.0637e+02 (2.1928e-02)	4.3045e-02 (9.0143e-03)	6.8734e+01 (8.3647e+00)	1.0234e+02 (1.8934e+01)	9.0362e+02 (9.8075e+00)
DE/current-to-best/1/bin with p-best crossover	2.0354e-01 (1.2435e-01)	**2.0056e+01 (1.2536e-02)**	2.5643e+00 (5.0987e+00)	4.8253e+01 (6.2383e+00)	9.8354e+01 (1.0354e+01)	5.0072e+02 (1.0254e-02)
DE/current-to-best/1/bin with parameter adaptation	**1.2038e-02 (9.2635e-03)**	2.0104e+01 (3.2837e-02)	**5.7643e-05 (6.0986e-05)**	**3.8465e+01 (4.2038e+00)**	**9.2534e+02 (2.3948e+01)**	**5.0014e+02 (5.0086e-02)**

problem. The selected fly-by sequence and other parameters are compatible with the currently flying Messenger mission. With respect to the problem "Messenger," the fly-by sequence is more complex and allows for resonant fly-bys at Mercury to lower the arrival DV (the Delta V maneuver). For the 26-D global optimization problem, the detailed ranges of each variable, MATLAB codes, etc., can be found from the URL: http://www.esa.int/gsp/ACT/inf/op/globopt/MessengerFull.html.

Cassini 2 is another spacecraft trajectory problem, where the objective function (unconstrained) evaluates the DV required to reach Saturn using an Earth–Venus, Venus–Earth, and Jupiter–Saturn fly-by sequence with DSMs. Here, we will consider an alternative model for the Cassini trajectory: DSMs are allowed between each one of the planets. This leads to a higher dimensional problem with a much higher complexity. We also consider, in the objective FE, a rendezvous problem rather than an orbital insertion as in the MGA model of the Cassini mission. This is the main cause for the higher objective function values reached. For the 22-D state vector, the bounds, the MATLAB code for evaluating the objective functions, etc., can be found in the following URL: http://www.esa.int/gsp/ACT/inf/op/globopt/edvdvdedjds.htm.

TABLE X
MEAN AND STANDARD DEVIATION OF THE OBJECTIVE FUNCTION VALUES FOR THE MESSENGER PROBLEM. THE BEST ENTRIES ARE MARKED IN BOLDFACE

FES	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	jDE	SaDE	DEGL	MDE_pBX
50000	1.9027e+01 (7.9273e-02)	1.8352e+01 (5.0026e-02)	1.5428e+01 (3.2936e-02)	1.6893e+01 (3.0293e-02)	1.6526e+01 (3.9264e-02)	1.7785e+01 (3.0167e-02)	**1.3972e+01 (2.3645e-02)**
100000	1.8129e+01 (5.2836e-02)	1.7362+01 (6.3627e-02)	1.3958e+01 (3.0738e-02)	1.5973e+01 (2.3273e-02)	1.5483e+01 (2.3946e-02)	1.6352e+01 (2.1453e-02)	**1.0251e+01 (1.6473e-02)**
150000	1.6814e+01 (4.3647e-02)	1.4624e+01 (5.8364e-02)	1.0434e+01 (2.9364e-02)	1.1935e+01 (2.2738e-02)	1.0853e+01 (2.0364e-02)	1.3381e+01 (1.9354e-02)	**9.7251e+00 (1.0538e-02)**

TABLE XI
MEAN AND STANDARD DEVIATION OF THE OBJECTIVE FUNCTION VALUES FOR THE CASSINI2 PROBLEM. THE BEST ENTRIES ARE MARKED IN BOLDFACE

FES	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	jDE	SaDE	DEGL	MDE_pBX
50000	2.19978e+01 (8.2737e-02)	2.15929e+01 (7.4938e-02)	1.94690e+01 (4.8837e-02)	2.04851e+01 (5.0384e-02)	1.98826e+01 (4.9374e-02)	2.09193e+01 (5.3949e-02)	**1.41815e+01 (3.9485e-02)**
100000	2.04402e+01 (6.6093e-02)	1.98935e+01 (6.7393e-02)	1.87155e+01 (4.0283e-02)	1.93957e+01 (4.3849e-02)	1.90914e+01 (4.3840e-02)	1.94901e+01 (4.7384e-02)	**1.12812e+01 (3.0182e-02)**
150000	1.86954e+01 (5.3839e-02)	1.83702e+01 (4.2937e-02)	1.68351e+01 (3.9475e-02)	1.79403e+01 (4.2901e-02)	1.75395e+01 (3.9274e-02)	1.78372e+01 (4.5901e-02)	**8.3986e+01 (2.6749e-02)**

TABLE XII
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST FOR THE MESSENGER AND CASSINI PROBLEMS, RESPECTIVELY, AT FINAL FES

FES	DE/rand/1/bin	DE/current-to-best/1/bin	JADE	jDE	SaDE	DEGL	MDE_pBX
Messenger	4.0574e-09	4.9472e-07	3.9362e-03	1.9453e-05	5.9453e-05	8.9073e-06	**NA**
Cassini2	5.0382e-09	8.9403e-08	2.9453-04	7.3619e-07	4.9352e-06	2.9035e-03	**NA**

Tables X and XI show the results for the Messenger and Cassini 2 problems, respectively, for the DE variants at three different numbers of FEs, and Table XII indicates that the MDE_pBX outperforms its competitors in a statistically significant way over both problems with respect to the final accuracies. All results are based on 50 independent runs for each algorithm. As before, in each run, the initial population was kept the same. The parameter of the contestant algorithms is kept as per Section IV-B.

H. Improving the Performances of the Existing DE Variants

In this section, we provide results of two sets of experiments conducted to investigate whether the proposed DE/current-to-gr_best/1 mutation, p-best crossover, and parameter adaptation schemes can also improve the performances of existing powerful DE variants like jDE [13] and JADE [20]. For the sake of space economy, we provide results for the first 14 functions in each case from the CEC 2005 benchmark set [34], [35]. Each function is tested for $D = 30$. Tables XIII and XIV provide the mean and the standard deviation of the best-of-run errors for 50 independent runs of five algorithmic variants of jDE and JADE, respectively, which consist of various modifications of the MDE_pBX. In these tables, the best entries are marked in boldface. The P-values obtained through Wilcoxon's rank-sum test results between the best algorithm and each of the remaining algorithms are also provided in Tables XV and XVI, respectively. Although we provide the results for the first 14 functions for the sake of space economy, our experiment is based on the entire set of 25 benchmark functions where the trend remains unaltered, i.e., the performance of both JADE and jDE is improved whenever one or more of the three proposed schemes are integrated with each of them.

A close inspection of Table XIII reveals that one or more jDE variants with one or more of the three proposed schemes integrated perform statically superior to the original jDE over all the 14 test functions. Although we achieve superior performance compared to jDE when we replace the parameter adaptation schemes of jDE with the proposed adaptation methods, the best results among the five algorithmic variants were obtained when the original parameter adaptation scheme of jDE was integrated with DE/current-to-gr_best/1 and p-best crossover schemes. Table XV indicates that, except for the scheme JADE with p-best crossover, the other three algorithmic variants of JADE are able to achieve better average error values for all the test functions, excluding only function f_{12}. The comparatively bleaker performance of JADE with p-best crossover can be attributed to the exploitative nature of both the p-best crossover and the DE/current-to-$p$best/1 mutation strategies that can lead to premature convergence at suboptimal basins especially for the multimodal functions.

V. CONCLUSION

We have proposed three algorithmic components, where one or more of which can be embedded in any DE variant to improve its search performance over complex fitness landscapes. DE/current-to-gr_best/1 adds a variation to the

TABLE XIII
AVERAGE AND STANDARD DEVIATION (IN PARENTHESES) OF THE BEST-OF-RUN ERROR FOR 50 INDEPENDENT RUNS OF FIVE ALGORITHMIC VARIANTS OF jDE TESTED IN FUNCTIONS f_1 TO f_{14} (IN $30D$). THE BEST ENTRIES ARE MARKED IN BOLDFACE

Functions	jDE	jDE with *p-best* Crossover	jDE with current-to-gr_best/1	jDE with proposed parameter adaptation schemes	jDE with current-to-gr_best/1 and *p*-best Crossover
f_1	3.8652e-29 (4.5732e-29)	8.8364e-31 (4.4845e-30)	5.5343e-35 (6.0809e-35)	**4.6297e-43 (5.0232e-43)**	1.6241e-39 (8.3551e-38)
f_2	7.5064e-06 (7.3804e-06)	7.3051e-10 (9.4523e-10)	8.9317e-13 (8.4326e-13)	2.2015e-15 (6.5132e-15)	**6.1935e-19 (3.7068e-19)**
f_3	2.2663e+05 (1.6085e+05)	2.0183e+05 (2.6743e+04)	1.3619e+05 (7.8254e+04)	8.6731e+04 (1.7685e+04)	**6.1038e+04 (1.5041e+04)**
f_4	2.7305e-01 (2.7305e-01)	3.7942e-02 (3.5763e-02)	2.0837e-03 (3.9757e-03)	9.1004e-06 (4.8865e-06)	**2.1572e-06 (6.0499e-06)**
f_5	1.1108e+03 (3.7238e+02)	1.0198e+03 (2.8646e+02)	8.9082e+02 (1.1675e+02)	5.6034e+02 (1.4823e+02)	**2.2418e+02 (1.5639e+02)**
f_6	1.1196e+01 (1.3987e+00)	2.1582e+01 (1.7023e+01)	9.0181e+00 (4.8452e+00)	6.2096e+00 (1.5365e+00)	**8.4293e-01 (1.0853e+00)**
f_7	9.8597e-03 (3.4824e-03)	8.1039e-03 (5.6786e-03)	7.0313e-03 (3.1412e-03)	7.4265e-03 (4.7276e-03)	**6.6017e-03 (4.5194e-03)**
f_8	2.0956e+01 (2.5067e-02)	2.0693e+01 (1.4479e-02)	2.0217e+01 (5.1331e-02)	2.0164e+01 (6.7483e-02)	**2.0049e+01 (2.8020e-03)**
f_9	8.3264e-12 (2.3645e-13)	1.8519e+01 (8.4351e+00)	6.6194e-14 (5.4378e-12)	**9.0483e-16 (1.9142e-15)**	1.2403e-14 (8.3812e-15)
f_{10}	5.2547e+01 (4.4660e+00)	4.3902e+01 (1.5871e+01)	3.6853e+01 (1.6710e+01)	3.1038e+01 (1.1672e+01)	**2.8413e+01 (9.4164e+00)**
f_{11}	3.1370e+01 (2.3952e+00)	3.0176e+01 (6.4387e+00)	2.9260e+01 (4.2376e+00)	2.2937e+01 (5.1250e+00)	**1.9543e+01 (6.5295e+00)**
f_{12}	3.8376e+04 (6.5374e+03)	3.1945e+04 (5.6834e+03)	2.8562e+04 (8.8211e+03)	2.1073e+04 (7.5823e+03)	**1.9274e+04 (6.2889e+03)**
f_{13}	1.6568e+00 (1.0313e-01)	1.4328e+00 (3.1523e-02)	1.8328e+00 (1.5345e-01)	1.2915e+00 (4.1854e-02)	**1.1279e+00 (2.6995e-01)**
f_{14}	1.3545e+01 (9.9402e-02)	1.3027e+01 (8.7456e-02)	1.2823e+01 (5.4946e-02)	1.2640e+01 (7.6054e-02)	**1.2584e+01 (6.1395e-02)**

TABLE XIV
AVERAGE AND STANDARD DEVIATION (IN PARENTHESES) OF THE BEST-OF-RUN ERROR FOR 50 INDEPENDENT RUNS OF FIVE ALGORITHMIC VARIANTS OF JADE TESTED IN FUNCTIONS f_1 TO f_{14} (IN $30D$). THE BEST ENTRIES ARE MARKED IN BOLDFACE

Functions	JADE	JADE with *p*-best Crossover	JADE with current-to-gr_best/1	JADE with modified adaptation	JADE with current-to-gr_best/1 ansd *p*-best Crossover
f_1	1.3258e-54 (9.2436e-54)	4.8578e-58 (6.4853e-58)	9.3273e-59 (4.8317e-58)	7.7487e-56 (1.4042e-55)	**6.0318e-61 (4.0938e-61)**
f_2	2.5146e-26 (3.4269e-26)	5.9132e-24 (2.8287e-24)	8.8793e-29 (3.3020e-29)	**1.8727e-29 (9.8705e-29)**	3.1204e-27 (4.8088e-27)
f_3	4.7421e+04 (1.6213e+04)	9.9817e+04 (2.1157e+04)	**3.0987e+04 (1.7433e+04)**	3.9726e+04 (8.1415e+03)	3.5271e+04 (9.4034e+03)
f_4	5.1159e-07 (4.0194e-07)	2.7132e-07 (1.4652e-07)	3.6713e-08 (4.4422e-08)	**7.8712e-08 (1.1281e-08)**	9.8722e-08 (7.8510e-08)
f_5	3.2792e+02 (1.8494e+02)	3.0178e+02 (2.0409e+02)	2.7918e+02 (1.5854e+02)	2.3916e+02 (1.3281e+02)	**2.0187e+02 (1.6563e+02)**
f_6	5.6094e+00 (1.9445e+01)	3.0981e+00 (1.8589e+01)	2.9816e+00 (7.9429e+00)	**2.5167e-01 (1.0934e+00)**	9.1027e-01 (1.1953e+00)
f_7	6.9598e-03 (4.4845e-03)	6.7890e-02 (1.1953e-02)	6.7018e-03 (1.3783e-03)	**5.0371e-03 (8.1031e-03)**	6.9102e-03 (2.1299e-03)
f_8	2.0929e+01 (2.6628e-02)	2.0677e+01 (2.0797e-02)	2.0414e+01 (3.2201e-02)	2.0310e+01 (2.1628e-02)	**2.0062e+01 (2.6441e-03)**
f_9	5.9272e-22 (8.9543e-22)	2.1743e+01 (9.8228e+00)	9.6713e-23 (4.6823e-23)	**1.6132e-24 (2.5709e-24)**	1.5492e+01 (8.9305e+00)
f_{10}	3.0313e+01 (8.3551e+00)	2.9045e+01 (9.6634e+00)	2.8028e+01 (5.7372e+00)	2.8981e+01 (4.6619e+00)	**2.7148e+01 (9.1327e+00)**
f_{11}	2.6456e+01 (1.9169e+01)	2.2191e+01 (8.5814e+00)	2.1075e+01 (3.3465+00)	2.0209e+01 (4.7951e+00)	**1.9098e+01 (6.7285e+00)**
f_{12}	**2.6978e+04 (6.8003e+03)**	6.6879e+04 (2.4712e+03)	5.3734e+04 (4.1953e+03)	4.8711e+04 (2.9718e+03)	5.1038e+04 (9.5372e+03)
f_{13}	1.6285e+00 (4.8739e-02)	1.5094e+00 (1.5704e-03)	1.3937e+00 (2.1097e-02)	1.2161e+00 (9.3937e-02)	**1.1402e+00 (1.3661e-02)**
f_{14}	1.2771e+01 (2.2057e-01)	1.2664e+01 (5.7039e-01)	**1.2572e+01 (4.7493e-01)**	1.2670e+01 (7.7393e-01)	1.2632e+01 (5.8373e-01)

TABLE XV
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST CORRESPONDING TO THE RESULTS OF TABLE XIII

Functions	jDE	jDE with *p*-best Crossover	jDE with current-to-gr_best/1	jDE with modified adaptation	jDE with current-to-gr_best/1 and *p*-best Crossover
f_1	5.7657e-18	7.6523e-19	6.0173e-20	NA	2.2771e-21
f_2	4.3802e-09	6.9183e-10	9.6142e-11	3.0183e-12	NA
f_3	5.7264e-03	4.1654e-02	2.7821e-03	7.9173e-02	NA
f_4	9.1583e-05	5.7816e-05	3.0037e-04	8.1441e-05	NA
f_5	8.3114e-03	2.1928e-03	7.9982e-03	6.6034e-03	NA
f_6	8.6353e-10	3.8443e-12	9.9273e-11	5.2096e-11	NA
f_7	1.0120e-11	9.1569e-11	4.0313e-11	8.4265e-11	NA
f_8	5.6161e-12	4.0982e-10	3.3528e-10	1.8273e-11	NA
f_9	1.8219e-20	8.0483e-17	5.7865e-19	NA	1.2403e-22
f_{10}	9.4656e-18	8.3902e-18	4.6853e-18	3.1038e-18	NA
f_{11}	6.6586e-07	3.0176e-07	2.9260e-07	2.293e-07	NA
f_{12}	8.6543e-17	3.1945e-18	2.8562e-18	2.1073e-18	NA
f_{13}	7.8328e-14	5.4328e-17	2.009e-15	3.5362e-17	NA
f_{14}	5.3129e-04	3.8172e-04	3.1130e-04	1.6940e-04	NA

TABLE XVI
P-VALUES CALCULATED FOR WILCOXON'S RANK-SUM TEST CORRESPONDING TO THE RESULTS OF TABLE XIV

Functions	JADE	JADE with *p-best* Crossover	JADE with current-to-gr_best/1	JADE with modified adaptation	JADE with current-to-gr_best/1 and *p*-best Crossover
f_1	1.8263e-23	3.9136e-23	8.8631e-22	6.9182e-22	NA
f_2	2.2054e-09	5 0349e-10	7.8793e-10	NA	2.2054e-09
f_3	7.5242e-04	1.5242e-04	4.1537e-04	2.0172e-04	NA
f_4	7.8345e-15	3.0183e-19	6.9827e-18	3.4756e-16	NA
f_5	6.0183e-19	4.6253e-20	3.1273e-20	2.0183e-20	NA
f_6	1.9754e-09	4.8127e-10	2.9846e-11	2.1342e-10	NA
f_7	8.4238e-13	4.3746e-13	NA	8.1735e-03	2.4238e-04
f_8	5.0351e-04	1.5180e-04	3.8409e-04	2.0543e-04	NA
f_9	2.2450e-17	2.2450e-25	7.1524e-23	6.4958e-18	NA
f_{10}	6.0661e-16	6.0661e-20	4.2558e-18	2.9436e-17	NA
f_{11}	8.8573e-18	8.1616e-18	3.1987e-17	1.9124e-19	NA
f_{12}	6.6185e-18	6.6776e-19	4.5342e-18	7.9173e-16	NA
f_{13}	3.3901e-15	5.5463e-16	4.3798e-18	1.2143e-17	NA
f_{14}	5.4466e-17	5.4716e-18	2.2898e-16	9.4857e-14	NA

classical and greedy DE/current-to-best/1 mutation scheme by perturbing the current target vector with the best member of a group of randomly selected population members. The group is reformed for each vector in a generation. This weakens the attraction toward any fixed point of the fitness landscape and helps in alleviating the tendency of premature convergence and/or stagnation. In p-best crossover operation, basically, a binomial crossover is performed between the current donor vector and any other individual from p top-ranked individuals in the present generation. The crossover operation is exploitative in nature and promotes the inclusion of genetic information from the elite class of individuals of current generation into the offspring. Our proposed adaptation schemes update the values of scale factor F and crossover rate Cr associated with each individual and in each generation, based on the success history of previously used F and Cr values. We have developed a DE variant, called MDE_pBX, by incorporating all the three components within the structure of the classical DE algorithm. An extensive performance comparison with six frequently used DE variants over 25 numerical benchmarks (in 30, 50, and 100 dimensions) and one real-world optimization problem indicated that the proposed approach enhances the ability of the basic DE to accurately locate solutions in the search space by a large extent. In addition, we have demonstrated that the proposed mutation, crossover, and parameter adaptation schemes, if integrated with state-of-the-art DE variants like jDE and JADE, can also significantly improve their performances.

The presented work can be extended in multiple directions. Future research may focus on an analytical investigation of the p-best crossover and DE/current-to-gr_best/1 mutation schemes in order to understand their effects on population diversity and convergence rate. The notion of dynamic grouping can also be used to develop mutation operators like DE/gr_best/1, DE/gr_best/2, etc., and their effectiveness can be tested on different kinds of functions. The group size may also be varied over generations, and the effect on the performance may be studied. The theoretical guidelines for setting the values of q and p should be investigated. MDE_pBX should also be adopted in the future for handling multiobjective and constrained optimization problems. One or more of the proposed algorithmic components can be integrated with the recently developed ensemble schemes for DE [39], [40], with an objective of enhancing the overall performance of the ensemble DE.

REFERENCES

[1] R. Storn and K. V. Price, Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces, ICSI, Berkeley, CA, Tech. Rep. TR-95-012. [Online]. Available: http://http.icsi.berkeley.edu/~storn/litera.html

[2] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[3] K. Price, R. Storn, and J. Lampinen, *Differential Evolution—A Practical Approach to Global Optimization*. Berlin, Germany: Springer-Verlag, 2005.

[4] R. Storn and K. V. Price, "Minimizing the real functions of the ICEC 1996 contest by differential evolution," in *Proc. IEEE Int. Conf. Evol. Comput.*, Nagoya, Japan, 1996, pp. 842–844.

[5] F. Neri and V. Tirronen, "Recent advances in differential evolution: A review and experimental analysis," *Artif. Intell. Rev.*, vol. 33, no. 1, pp. 61–106, Feb. 2010.

[6] S. Das and P. N. Suganthan, "Differential evolution—A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[7] J. Ronkkonen, S. Kukkonen, and K. V. Price, "Real parameter optimization with differential evolution," in *Proc. IEEE CEC*, 2005, vol. 1, pp. 506–513.

[8] N. Hansen and S. Kern, "Evaluating the CMA evolution strategy on multimodal test functions," in *Proc. 8th Int. Conf. PPSN*, Berlin, Germany, 2004, pp. 282–291.

[9] W. B. Langdon and R. Poli, "Evolving problems to learn about particle swarm optimizers and other search algorithms," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 561–578, Oct. 2007.

[10] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Proc. GECCO*, 2006, pp. 485–492.

[11] J. Liu and J. Lampinen, "On setting the control parameters of the differential evolution method," in *Proc. 8th MENDEL*, R. Matoušek and P. Ošmera, Ed.s, 2002, pp. 11–18.

[12] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[13] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[14] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput., Fusion Found. Methodologies Appl.*, vol. 9, no. 6, pp. 448–462, Jun. 2005.

[15] J. Liu and J. Lampinen, "Adaptive parameter control of differential evolution," in *Proc. 8th MENDEL*, R. Matoušek and P. Ošmera, Eds., 2002, pp 19–26.

[16] J. Rönkkönen and J. Lampinen, "On using normally distributed mutation step length for the differential evolution algorithm," in *Proc. 9th MENDEL*, Brno, Czech Republic, 2003, pp. 11–18.

[17] M. M. Ali and A. Törn, "Population set based global optimization algorithms: Some modifications and numerical studies," *Computers and Operations Research, Elsevier*, vol. 31, no. 10, pp. 1703–1725, Sep. 2004.

[18] W. Gong, Z. Cai, C. X. Ling, and H. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 2, pp. 397–413, Apr. 2011.

[19] M. Weber, F. Neri, and V. Tirronen, "Distributed differential evolution with explorative–exploitative population families," *Genetic Programm. Evol. Mach.*, vol. 10, no. 4, pp. 343–371, Dec. 2009.

[20] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[21] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.

[22] S. Das and P. N. Suganthan, "Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems," Jadavpur Univ., Nanyang Technol. Univ., Kolkata, India, Dec. 2010.

[23] D. Izzo, "Global optimization and space pruning for spacecraft trajectory design," in *Spacecraft Trajectory Optimization*, B. Conway, Ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[24] R. Gamperle, S. D. Muller, and A. Koumoutsakos, "Parameter study for differential evolution," in *Proc. WSEAS NNA-FSFS-EC*, Interlaken, Switzerland, Feb. 11–15, 2002.

[25] E. Mezura-Montes, J. Velazquez-Reyes, and C. A. Coello Coello, "Modified differential evolution for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, Jul. 2006, pp. 25–32.

[26] C. O. Imoru, "The power mean and the logarithmic mean," *Int. J. Math. Math. Sci.*, vol. 5, no. 2, pp. 337–343, 1982.

[27] D. Zaharie, "On the explorative power of differential evolution," in *Proc. 3rd Int. Workshop Symbol. Numer. Algorithms Sci. Comput.*, Timişoara, Romania, Oct. 2–4, 2001.

[28] S. Dasgupta, S. Das, A. Biswas, and A. Abraham, "On stability and convergence of the population-dynamics in differential evolution," *AI Commun.*, vol. 22, no. 1, pp. 1–20, 2009.

[29] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[30] A. Caponio, F. Neri, and V. Tirronen, "Super-fit control adaptation in memetic differential evolution frameworks," *Soft Comput., Fusion Found. Methodologies Appl.*, vol. 13, no. 8, pp. 811–831, Jul. 2009.

[31] V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava, and T. Rossi, "An enhanced memetic differential evolution in filter design for defect detection in paper production," *Evol. Comput. J.*, vol. 16, no. 4, pp. 529–555, Winter, 2008.

[32] M. Weber, V. Tirronen, and F. Neri, "Scale factor inheritance mechanism in distributed differential evolution," *Soft Comput., Fusion Found. Methodologies Appl.*, vol. 14, no. 11, pp. 1187–1207, Sep. 2010.

[33] M. Weber, F. Neri, and V. Tirronen, "A study on scale factor in distributed differential evolution," *Inf. Sci.*, vol. 181, no. 12, pp. 2488–2511, Jun. 2011.

[34] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technol. Univ., Singapore, May 2005.

[35] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," IIT, Kanpur, India, KanGAL Rep. 2005005, May 2005.

[36] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[37] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[38] M. Vasile, E. Minisci, and M. Locatelli, "Analysis of some global optimization algorithms for space trajectory design," *AIAA J. Spacecraft Rockets*, vol. 47, no. 2, pp. 334–344, 2010.

[39] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, Mar. 2011.

[40] R. Mallipeddi and P. N. Suganthan, "Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies," in *Proc. Swarm Evol. Memetic Comput. Conf.*, Chennai, India, 2010, pp. 71–78.

Sk. Minhazul Islam was born in West Bengal, India, in 1991. He is currently working toward the B.S. degree in the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata, India.

He has published research articles in peer-reviewed journals and international conference proceedings. He has acted as a Reviewer for journals like *Information Sciences* and *Swarm and Evolutionary Computation*. His current research interests include digital signal processing, wireless communication, digital image processing, pattern recognition, evolutionary computing, and solving real-world optimization problems using evolutionary algorithms.

Swagatam Das (M'10) received the B.E.Tel.E., M.E.Tel.E. (control engineering specialization), and Ph.D. degrees from Jadavpur University, Kolkata, India, in 2003, 2005, and 2009, respectively.

He was an Assistant Professor with the Department of Electronics and Telecommunication Engineering, Jadavpur University, from 2006 to 2011. He is currently a Visiting Assistant Professor with the Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata. He has published more than 120 research articles in peer-reviewed journals and international conferences. He coauthored a research monograph on metaheuristic clustering techniques from Springer in 2009. He is the founding Coeditor-in-Chief of *Swarm and Evolutionary Computation*, an international journal from Elsevier. He serves as an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS (PART—A) and *Information Sciences* (Elsevier). He is an editorial board member of the *International Journal of Artificial Intelligence and Soft Computing* and *International Journal of Autonomous and Adaptive Communications Systems*. He has been acting as a regular reviewer for journals like *Pattern Recognition*, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE TRANSACTIONS ON SMC PART A, PART B, AND PART C. He has acted as a Guest Editor for special issues in journals like IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and IEEE TRANSACTIONS ON SMC, PART C. His current research interests include evolutionary computing, pattern recognition, multiagent systems, and wireless communication.

Saurav Ghosh was born in West Bengal, India, in 1989. He is currently working toward the B.S. degree in the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata, India.

He has published five research articles in peer-reviewed journals and international conference proceedings. His current research interests include evolutionary computing and its application to real-world optimization problems arising from the domains of digital signal processing, digital image processing, and wireless communication using evolutionary algorithms.

Subhrajit Roy was born in West Bengal, India, in 1989. He is currently working toward the B.S. degree in the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata, India.

He has published research articles in peer-reviewed journals and international conference proceedings. He has the experience of reviewing papers for journals like *Swarm and Evolutionary Computation*, IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS, PART-C, and *International Conference on Swarm, Evolutionary, and Memetic Computing* (SEMCCO 2011). His research interest includes computational intelligence and its application to various real-world optimization problems arising from the domain of digital signal processing, digital image and video processing, and wireless communication.

Ponnuthurai Nagaratnam Suganthan (S'91–M'92–SM'00) received the B.A. degree, postgraduate certificate, and M.A. degree in electrical and information engineering from the University of Cambridge, Cambridge, U.K., in 1990, 1992, and 1994, respectively, and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, Singapore.

He was a Predoctoral Research Assistant with the Department of Electrical Engineering, University of Sydney, Sydney, N.S.W., Australia, in 1995–1996 and a Lecturer with the Department of Computer Science and Electrical Engineering, University of Queensland, Brisbane, Qld., Australia, in 1996–1999. Since 1999, he has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, where he was an Assistant Professor and where he is now an Associate Professor. He is an Associate Editor of the IEEE TRANS ON EVOLUTIONARY COMPUTATION, *Information Sciences*, *Pattern Recognition*, and *International Journal of Swarm Intelligence Research*. He is the Founding Coeditor-in-Chief of *Swarm and Evolutionary Computation*, an Elsevier journal. His coauthored article on SaDE (April 2009) won the "IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION" outstanding paper award. His research interests include evolutionary computation, pattern recognition, multiobjective evolutionary algorithms, bioinformatics, applications of evolutionary computation, and neural networks.