Hindawi
Journal of Mathematics
Volume 2023, Article ID 8973912, 17 pages
https://doi.org/10.1155/2023/8973912

Research Article

Differential Evolution without the Scale Factor and the Crossover Probability

Xiaowei Zhang

School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China

Correspondence should be addressed to Xiaowei Zhang; x.w.zhang@126.com

Received 12 December 2021; Revised 10 June 2022; Accepted 24 November 2022; Published 19 April 2023

Academic Editor: A. M. Nagy

Differential evolution has made great achievements in various fields such as computational sciences, engineering optimization, and operations management in the past decades. It is well known that the control parameter setting plays a very important role in terms of the performance improvement of differential evolution. In this paper, a differential evolution without the scale factor and the crossover probability is presented, which eliminates almost all control parameters except for the population size. The proposed algorithm looks upon each individual as a charged particle to decide on the shift of the individual in the direction of the difference based on the attraction-repulsion mechanism in Coulomb's Law. Moreover, Taguchi's parameter design method with the two-level orthogonal array is merged into the crossover operation in order to obtain better individuals in the next generation by means of better combination of factor levels. What is more, a new ratio of the signal-to-noise is proposed for the purpose of fair comparison of the numerical experiment for the tested functions which have an optimal value with 0. Numerical experiments show that the proposed algorithm outperforms the other 5 compared algorithms for the 10 benchmark functions.

1. Introduction

With its efficiency and effectiveness, differential evolution (for short, DE) proposed by Storn and Price has been successfully applied in many different engineering fields [1, 2]. In order to keep improving the performance of DE, various efforts have been devoted over the past decades.

The researchers proposed three discrete DEs for the scheduling problems in the permutation flow shop environment [3]. These approaches focus on converting vectors of the continuous domain into permutation vectors of the discrete domain and self-adjusting the control parameters of these algorithms based on JADE [4] and SADE [5]. The results show that these proposed approaches are promising for scheduling problems.

For the parameter identification of solar cells, the original FSDE in reference [6] was improved, which is the hybridization between free search and DE with opposition-based learning by using a simple greedy strategy instead of a Gaussian noise update in the process of the potential solution generation for the proposed best solution update strategy [7]. Reference [8] also employed a DE with opposition-based learning for estimating optimum hourly energy generation scheduling of a hydro-thermal system.

The authors emphasized the population initialization on increasing the accuracy and convergence speed of DE and designed a new DE variant with a modified initialization scheme by combining the strengths of both chaotic maps and oppositional-based learning strategy in order to generate the initial population with a good quality of mean fitness and diversity of the solutions. Extensive simulation studies on benchmark functions show that the proposed algorithm outperforms its peers [9].

A cultural DE algorithm using a measure of population diversity was proposed as an alternative method for solving the economic load dispatch problems of thermal generators [10]. Based on the cultural algorithm technique using normative and situational knowledge sources, the proposed algorithm is able to balance well the trade-off between the exploration and the exploitation of the search space.

The scale factor F and the crossover probability Cr are two vital parameters in DE, which usually greatly improve the performance. Various strategies for parameter setting have been researched.

The values of $F = 0.5$ and $Cr = 0.9$ were suggested by Storn and Price [1]. The F was set to the normal distribution rand number with expectation 0 and standard deviation 1 for multiobjective optimization in reference [11].

Qin and Suganthan considered F and Cr as the random numbers following normal distribution $F \sim N(0.5, 0.3)$ and $Cr \sim N(Crm, 0.1)$ according to the learning experience, where the parameter Crm is set at 0.5 and updated once every 25 generations [5].

Kim et al. proposed that the scale factor F is calculated by the formula $F = a + b \cdot \text{rand}(0,1)$, where $a + b < 1$ and $a, b > 0$ [12].

Ali and törn empirically obtained an optimal value $Cr = 0.5$ and calculated automatically the scale factor F using the maximum and the minimum for focusing on the exploration at early generation and the exploitation at latter generation [13].

The parameters F and Cr were given, respectively, following α-stable distribution $S_\alpha(0, 0.1, \text{mean}(S_F))$ and $S_\alpha(0, 0.1, \text{mean}(S_{Cr}))$, where S_F and S_{Cr} denote the successfully evolved individuals' F and Cr based on some feedbacks from the optimization process [14].

The scale factor F was set using the Tsallis distribution in economic view for the optimization model in shell-and-tube heat exchangers [15]. F is fist initialized with uniform random values between 0.8 and 1.1, and then is determined by $F = F_{mu} + F_\sigma^2 \cdot P_F$ at each generation, where P_F obeys a q-Gaussian distribution or Tsallis distribution with the means F_{mu} and the variance F_σ^2, the parameter q is linked to the type of distribution that assumes values from 1 to 3.

A self-adaptive scaling factor $F = S \cdot \sqrt{\text{rand}(0,1)^2 \cdot d - b}$ was utilized in reference [16] for maximizing the profit of the distribution company with the several constraints based on the basic idea of the penalty function approach for solving optimal planning of energy storage systems in order to improve the rate of convergence of DE, where S, d, and b are an acceleration factor, a linear decreasing factor, and a deceleration factor, respectively.

Based on the different setups created by a simple orthogonal experimental design method, the paper [17] revealed that DE/best/1/bin with $F = 0.5$ and $Cr = 0.2 + 0.6 * rand(0,1)$ is promising to optimize the vector Jiles–Atherton vector hysteresis model from a workbench containing a rotational single sheet tester. Similarly, the self-adapting parameter strategy was used in reference [18].

Some researchers designed the novel selection operator or employed the classical derivative-free methods in DE or analyzed the search behavior in theory for improving the performance of DE [19–22].

These versions of DE do improve the algorithm performance. However, each of them only is superior to the other in some special aspects. The best setting for the control parameters can be different for different problems. Even though the self-adapting parameter strategies seem to be able to overcome the problem of parameter setting, some new control parameters are used. Several references reported that choosing the proper control parameters for DE is more difficult than expected. How to set reasonably these parameters is a nuisance [2, 23, 24].

A differential evolution without the scale factor and the crossover probability is presented in the paper. The algorithm calculates dynamically the scale factor F using the attraction-repulsion mechanism in Coulomb's Law and executes the crossover operation using Taguchi's parameter design method based on the orthogonal array. The proposed algorithm avoids the parameter settings. Numerical experiments show that the performance of the proposed algorithm is superior to that of the other compared algorithms.

The paper is the extended version which has been further researched based on "almost-parameter-free differential evolution" proposed by Zhang and Liu [24]. There are four different points between them. Firstly, this paper describes in detail the idea and particulars of the proposed algorithm. Secondly, we regard the scale factor F in the mutant equations (13) and (14) in Section 4 as the two different charges for the purpose of a better interpretation of the algorithms' idea and a better numerical experiment results. Thirdly, the vital shortcoming of the original definition of the ratio of the signal to noise (SNR) is analyzed in Section 4 and reveals the fact that it has thought of the optimal value of the tested problem before being solved as 0, then presents a modified definition of SNR for the sake of fairness. Finally, a brief convergence analysis is given under two assumptions.

The main contributions of this paper, which distinguish from the related literatures, are summarized as follows:

(i) Use the electromagnetism-like mechanism to decide on the step length in the direction of the difference for the mutation operation;

(ii) Employ Taguchi's parameter design with a two-level orthogonal array based on a new ratio of the signal to noise that is proposed for the crossover operation;

(iii) Eliminate almost all the control parameters of DE except for the population size.

The remainder of the paper is organized as follows. In Section 2, differential evolution algorithm is briefly introduced. Taguchi's parameter design method is described in the next section. In Section 4, a DE without the parameters is proposed and the convergence in probability is analyzed. In Section 5, the results of numerical experiments are given. Finally, we conclude this paper and consider the further research issues.

2. Differential Evolution

Like other evolutionary algorithms (EAs), DE starts with an initial population individual, followed by the successive operations of mutation, crossover, and selection. However, there are two main differences between them. (i) Mutation is caused not by the small changes of the genes in EAs, but by adding the weighted difference of two randomly selected individuals to a third randomly selected one in DE. The direction information from the current population is used to guide the search process. (ii) New individual is generated by adopting a greedy selection scheme in DE, which is only accepted if it improves on the fitness of the parent individual.

Storn and Price proposed several different mutation strategies [1]:

DE/Rand/1: $V = X_{r1} + F \cdot (X_{r2} - X_{r3})$

DE/Rand/2: $V = X_{r1} + F \cdot (X_{r2} - X_{r3} + X_{r4} - X_{r5})$

DE/Best/2: $V = X_{best} + F \cdot (X_{r2} - X_{r3} + X_{r4} - X_{r5})$

In the above, $r1 \neq r2 \neq r3 \neq r4 \neq r5$, and they are the random numbers distributing uniformly in $[1, NP]$, where NP is denoted by the population size. For the strategy DE/x/y, x represents the individual being perturbed and y is the number of difference vectors used to disturb x. Take DE/rand/1 as an example, it means that the target individual is randomly selected, and only one difference vector is used.

Although there are several variants of DE, a common variant, which is known as DE/rand/1, or "classic DE," is the most widely used in practice. Hence, this DE is described as follows:

(i) Initialization: like other EAs, classic DE initializes an initial population that distributes uniformly in the feasible domain.

(ii) Mutation: for each parent vector X_i, a mutant vector V_i is generated according to (1) where the random indexes $r1$, $r2$, and $r3$ are mutually distinct integers following uniform distribution in $[1, NP]$ and also are different from the current index i. The scale factor F is used to control the amplification of the differential variation.

$$V_i = X_{r1} + F \cdot (X_{r2} - X_{r3}). \tag{1}$$

(iii) Crossover: the trial individual W_i is generated using the parent and mutant individuals as follows:

$$W_i^j = \begin{cases} V_i^j, & \text{if } r(j) \leq Cr \text{ or } j = \text{rand}n(i), \\ X_i^j, & \text{else.} \end{cases} \tag{2}$$

In the above formula, j is denoted by the j-th component of the individual, $r(j)$ represents a random number with uniform distribution in [0, 1] for each j, the crossover probability Cr is set to a given number in $(0, 1)$, and the integer $randn(i)$ is randomly chosen in $[1, n]$, where n denotes the dimension of the tested problem. The trial individual is a stochastic combination of the parent and mutant individuals. When Cr is equal to 0, at least one of the components of the trial individual will differ from the parent X_i because of the condition $j = randn(i)$.

(iv) Selection: DE implements a very simple selection procedure. The offspring is generated only if the fitness of the offspring is better than that of the parent. Due to the greedy selection scheme, all the individuals of the next generation are as good as or better than their counterparts in the current generation.

$$X_{i+1} = \begin{cases} W_i, & \text{if } f(W_i) < f(X_i), \\ X_i, & \text{otherwise.} \end{cases} \tag{3}$$

The above process ii–iv repeats until the number of function evaluations or the number of the iterations reaches a given constant, namely, the termination criteria are satisfied. Further detailed descriptions about DE can be found in references [1, 23].

3. Taguchi's Parameter Design

Taguchi method [25] is a parameter design approach in the production and process conditions optimization. It can make high-quality products using less development and manufacturing costs. Two major tools used in the Taguchi method are the orthogonal array [26] and the signal-noise-ratio (SNR), which are briefly described as follows.

The orthogonal array is a fractional factorial matrix, which assures a balanced comparison among the factors or its levels. A two-level orthogonal array is a matrix consisting of 1 or 2 arranged in rows and columns. Each row represents the combination of factor levels in each experiment, and each column represents the special level of each factor. Let the element 2 in the orthogonal array be −1, then all column vectors are orthogonal to each other, namely, the dot product is zero. Generally, a two-level orthogonal array is denoted by $L_m(2^{m-1})$, where m, which is equal to 2^k, represents the number of experiments; k is a positive integer; the number 2 shows that each factor has two levels: 1 and 2; $m - 1$ is the number of the factors or columns. The two-level orthogonal arrays are commonly used in practice: $L_4(2^3)$, $L_8(2^7)$, $L_{16}(2^{15})$, and $L_{32}(2^{31})$. For more clearness, the following table (see Table 1) shows the orthogonal array $L_8(2^7)$ with the canonical form.

There are 8 factors in the array $L_8(2^7)$. For each factors, it can choose either 1 or 2. In order to obtain the better or best the combination of factor levels, only 8 experiments are under considered in the two-level orthogonal array $L_8(2^7)$ instead of all combinations of the factors which can reach up to $2^7 = 128$ experiments. The notation E_i represents the i-th experiment or row, and C_j the j-th column vector or factor. For simplicity, the sign $C_{i,j}$ denotes the level of the j-th factor in the i-th experiment. For instance, $C_{3,4} = 1$, $C_{4,3} = 2$, and $C_6 = [1\,2\,2\,1\,1\,2\,2\,1]^T$. If each 2 in array $L_8(2^7)$ is thought of as −1, $C_{i \neq j} \cdot C_j = 0$ for all i and j from 1 to 7.

The conception of the SNR is originally introduced in communication and electronic engineering, which is defined as the ratio of the signal to noise and is used to evaluate the quality of communication. In 1957, Taguchi applied the SNR conception to the design of engineering experiments, hence, Taguchi parameter design method was proposed. This method utilizes the SNR to evaluate quality and applies the orthogonal array to arrange experiments. According to the type of characteristic, the SNR can be classified into smaller-the-better, larger-the-better and nominal-the-best. Given

Table 1: The orthogonal array $L_8(2^7)$ with the canonical form.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7
E_1	1	1	1	1	1	1	1
E_2	1	1	1	2	2	2	2
E_3	1	2	2	1	1	2	2
E_4	1	2	2	2	2	1	1
E_5	2	1	2	1	2	1	2
E_6	2	1	2	2	1	2	1
E_7	2	2	1	1	2	2	1
E_8	2	2	1	2	1	1	2

a set of characteristics $y_1, y_2, \ldots, y_n$, then in the case of smaller-the-better characteristic the SNR is as follows:

$$SNR = -10 \cdot \log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right). \tag{4}$$

4. Differential Evolution without F and Cr

After the brief description about classical DE and Taguchi's parameter design, the ideals and the advantage of eliminating the parameters in DE are described, respectively. Finally, the differential evolution without the scale factor and the crossover probability are proposed.

Besides the parameters F and Cr, classic DE has a control parameter NP which are closely related to the problem under consideration. The population size, NP, is typically larger than a threshold value in order to obtain a global optimum and improve the success rate of convergence. However, too large NP may increase the number of function evaluations. Generally, separable and unimodal functions require the smallest population sizes, while parameter-dependent multimodal functions require the largest populations. For simplicity, the parameter NP is set as a constant according to the dimension of the problem under consideration.

The parameter F determines the amplification of the difference. A high (low) value of F makes DE more exploratory (less exploratory). The parameter Cr controls the distribution of coordinate points in the trial individual. A high (low) value of Cr means that the coordinates of the mutant individual dominate the trial individual. Between the two parameters Cr and F, Cr is much more sensitive to the problem's properties and complexity such as the multimodality, while F is more related to the convergence speed.

Finding the optimal values for these parameters is a difficult task as these values are problem specific, especially when one wants to strike a balance between reliability and efficiency. Thus, the performance of DE depends on how these control parameters are selected. However, how to set well these parameters is generally based on trial and error. An optimal parameter setting can be found via the boring preliminary numerical experiments for a special problem, whereas it is not probably optimal for the other problems.

In order to overcome these contradictions, we eliminate the scale factor and the crossover probability with exception of the population size by using the modified attraction-repulsion mechanism and Taguchi method. In the following subsections, how to eliminate these parameters is described in detail.

4.1. Eliminating the Scale Factor F.

According to the attraction-repulsion mechanism in Coulomb's Law, electromagnetism-like (EM) algorithm [27, 28] first calculates the charge of each individual in terms of its objective function value and then determines the resultant force exerted on each particle by all other particles in the population. The charge of each particle determines its power of attraction or repulsion. The particles with better objective function values attract others while those with inferior function values repel.

Like the method of calculating the force, the electromagnetic force exerted on the particle by other particles is obtained by the vector addition following the parallelogram law. For example, the charge of X_1 is less than that of X_2 while is greater than that of X_3 in Figure 1. Thus $EF_{1,2}$ is a repulsive force and $EF_{1,3}$ is an attractive force acting on X_1 by X_2 and X_3, respectively. The resultant force EF_1 exerted on X_1 is $EF_{1,2} + EF_{1,3}$. In a similar way, the resultant forces exerted on X_2, and on X_3 can also be calculated.

The charge Q_i of each X_i is determined by the objective function value of itself relative to that of the current best particle X_{best}:

$$Q_i = \exp\left(-n \cdot \frac{f(X_i) - f(X_{best})}{\sum_{j=1}^{NP}\left(f(X_j) - f(X_{best})\right)}\right), \tag{5}$$

where n is the dimension of the problem. The force vector $EF_{i,j}$ exerted on X_i by X_j is then determined by

$$EF_{i,j} = \begin{cases} \left(X_j - X_i\right) \cdot \dfrac{Q_i Q_j}{\left\|X_j - X_i\right\|^2}, & \text{if } f\left(X_j\right) \le f\left(X_i\right), \\[2ex] \left(X_i - X_j\right) \cdot \dfrac{Q_i Q_j}{\left\|X_j - X_i\right\|^2}, & \text{if } f\left(X_j\right) > f\left(X_i\right). \end{cases} \tag{6}$$

From (6), the particles with the relatively good objective function values will attract the other particles in the population while the particles with the worse objective function values repel the others. The resultant force vector EF_i exerted on a particle X_i by other $NP - 1$ particles in the population is calculated as follows:

$$EF_i = \sum_{j=1, j\neq i}^{NP} EF_{i,j}. \tag{7}$$

However, each particle has only one particle exerting force on it in a version of EM proposed by Debels et al. [29].

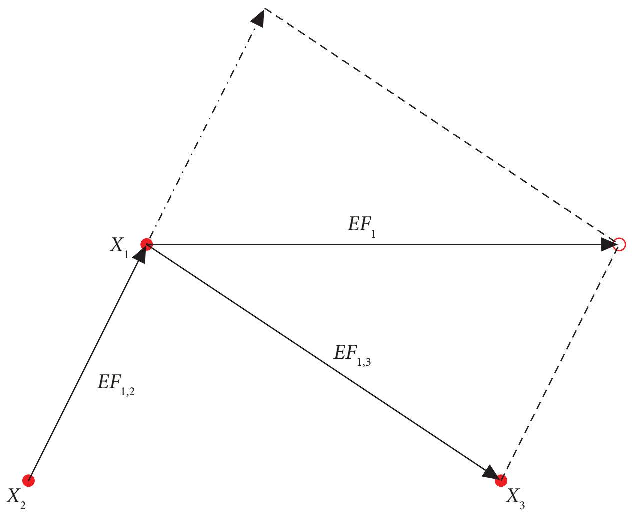


Figure 1: Exertion of forces on X_1 by X_2 and X_3.

In this approach, the charge Q_i^j of X_i is calculated based on the relative difference in the objective function values $f(X_i)$ and $f(X_j)$:

$$Q_i^j = \frac{f(X_i) - f(X_j)}{f(X_{worst}) - f(X_{best})}, \tag{8}$$

where X_{worst} and X_{best} denote, respectively, the worst and the best solutions, X_j is chosen randomly from the population. By the new definition of Q_i^j, obviously, a better(worst) particle X_j gives the higher(lower) Q_i^j value. Moreover, if $f(X_j) \le f(X_i)$, then Q_i^j is positive, otherwise, Q_i^j is negative. After calculating the charge Q_i^j of X_i, the particle X_i moves to the new particle $X_i + EF_{i,j}$, where

$$EF_{i,j} = Q_i^j \cdot (X_j - X_i). \tag{9}$$

It is obvious that when Q_i^j is positive (negative), X_j attracts(repels) X_i. This modified EM remains the basic ideal of EM, moreover, it is more simple and easier to utilize. Hence, for DE/Rand/1, the mutant individual $V = X_{r1} + F \cdot (X_{r2} - X_{r3})$ can be transformed to

$$\begin{aligned} V &= X_{r1} + F \cdot (X_{r2} - X_{r3}) \\ &= X_{r1} + F \cdot (X_{r2} - X_{r1}) + F \cdot (X_{r1} - X_{r3}) \\ &= X_{r1} + F \cdot (X_{r2} - X_{r1}) + F' \cdot (X_{r3} - X_{r1}), \end{aligned} \tag{10}$$

where $F = -F'$. If we regard the scale factor F and F' in equation (10) as the two different charges Q_i^j as shown in equation (8), viz.

$$F \triangleq Q_{r1}^{r2}, F' \triangleq Q_{r1}^{r3}, \tag{11}$$

then the equation (10) can be interpreted as the motion of the particle X_{r1} in the direction of the resultant force $F_{2,1} + F_{3,1}$. The magnitude of the motion is determined by the scale factors F and F'. Hence, the mutant individual is modified in our algorithm as follows:

$$\begin{aligned} V &= X_{r1} + Q_{r1}^{r2} \cdot (X_{r2} - X_{r1}) + Q_{r1}^{r3} \cdot (X_{r3} - X_{r1}) \\ &= X_{r1} + \left(\frac{f(X_{r1}) - f(X_{r2})}{f(X_{\text{worst}}) - f(X_{\text{best}})} \cdot (X_{r2} - X_{r1}) \right. \\ &\quad \left. + \frac{f(X_{r1}) - f(X_{r3})}{f(X_{\text{worst}}) - f(X_{\text{best}})} \cdot (X_{r3} - X_{r1}) \right). \end{aligned} \tag{12}$$

Similarly, we also have

$$\begin{aligned} V &= X_{r1} + Q_{r2}^{r3} \cdot (X_{r3} - X_{r2}) \\ &= X_{r1} + \frac{f(X_{r2}) - f(X_{r3})}{f(X_{\text{worst}}) - f(X_{\text{best}})} \cdot (X_{r3} - X_{r2}), \end{aligned} \tag{13}$$

or

$$\begin{aligned} V &= X_{r1} + Q_{r3}^{r2} \cdot (X_{r2} - X_{r3}) + Q_{r5}^{r4} \cdot (X_{r4} - X_{r5}) \\ &= X_{r1} + \left(\frac{f(X_{r3}) - f(X_{r2})}{f(X_{\text{worst}}) - f(X_{\text{best}})} \cdot (X_{r2} - X_{r3}) \right. \\ &\quad \left. + \frac{f(X_{r5}) - f(X_{r4})}{f(X_{\text{worst}}) - f(X_{\text{best}})} \cdot (X_{r4} - X_{r5}) \right). \end{aligned} \tag{14}$$

As described , equations (13) and (14) are easy to understand. The idea implied in equation (13) comes from *DE/rand*/1: the individual X_{r1} moves in the direction of $X_{r3} - X_{r2}$. The magnitude of the motion is not controlled artificially in DE/Rand/1, but is determined self-adaptively according to its charge obtained by the particle X_{r2}. The similar interpretation is also done for equation (14).

Besides the self-adaptation of F and the simplicity of calculation, preliminary numerical experiments show that the modified equations (12)–(14) can generally improve the performance of DE, and equation (12) might avoid DE(DE/Ra nd/1) searching wrongly in the direction of "up hill." The detailed description is as follows.

For six hump camel back function (see $F0$ in Appendix), it is well known that the optimal value is $f^*(\{[-0.08984, 0.71265], [0.08984, -0.71265]\}) = -1.031628$. Given $X_{\text{best}} = [-0.07781, -0.73245]$ and $X_{\text{worst}} = [0.97667, -0.0033774]$, then two cases are given.

CASE 1 Let $X_{r1} = [-0.39, -0.91221]$, $X_{r2} = [-0.15301, 0.28698]$, and $X_{r3} = [0.13566, -0.58573]$. Thus, $V = [-0.12891, -0.54275]$ can be obtained by equation (12) (see Figure 2).

CASE 2 Let $X_{r1} = [0.15961, 0.48913]$, $X_{r2} = [0.28105, 0.86676]$, and $X_{r3} = [0.94169, -0.23207]$. Then, $V = [-0.4222, 0.97067]$, see Figure 3.

Figures 2 and 3 show the contour of SHCB on $[-1, 1]^2$ with the corresponding function value marked. The stars denote the optimal solutions; the circle denotes the individual X_{r1}; two outer squares 10 represent X_{r2} and X_{r3}, respectively; Two outer real line denote the shift of X_{r1} in direction of the force $EF_{r1,r2}$ and $EF_{r1,r3}$, respectively. The mutant individual V obtained by equation (12) is denoted by the diamond. The inner real line represents the shift of X_{r1} in

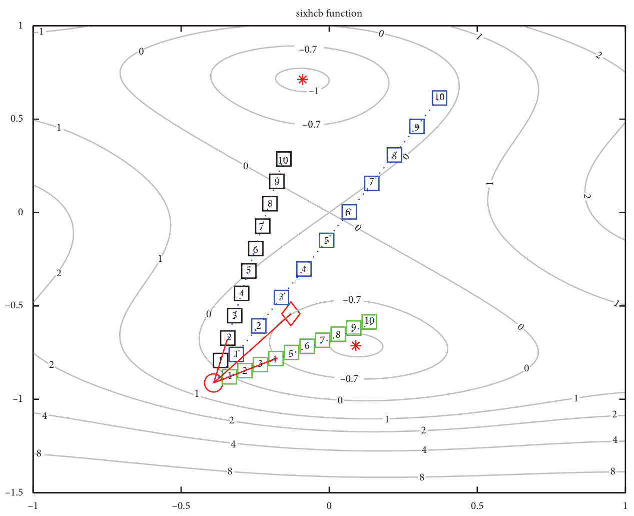


Figure 2: Case 1.

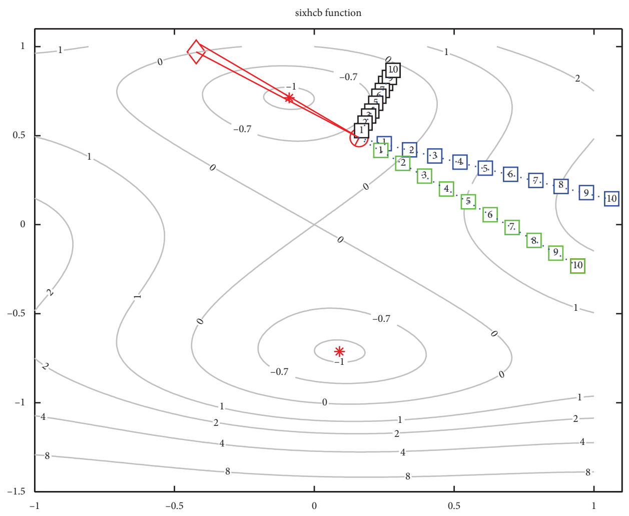


Figure 3: Case 2.

direction of the resultant force EF_{r1}. Two bunches of squares locating in outer dashed line denote the motions of the individual X_{r1} in directions of $F \cdot (X_{r2} - X_{r1})$ and $F \cdot (X_{r3} - X_{r1})$ with the different scale factor F, respectively. The scale factor F is chosen orderly from the set $\{0.1, 0.2, \ldots, 0.9, 1\}$, the corresponding results are shown in Figures 2 and 3 by the squares with the different number marked. A bunch of squares between outer squares gives the different mutant individual V (see equation (15)). All squares can be matched by the numbers locating in them.

$$\begin{aligned} V &= X_{r1} + F \cdot \left[\left(X_{r2} - X_{r1}\right) + \left(X_{r3} - X_{r1}\right) \right] \\ &= X_{r1} + F \cdot \left(X_{r2} - X_{r1}\right) + F \cdot \left(X_{r3} - X_{r1}\right). \end{aligned} \tag{15}$$

It is worth noting that equation (15) is different from $V = X_{r1} + F \cdot (X_{r2} - X_{r3} + X_{r4} - X_{r5})$. Five different mutually random individuals are selected in DE/Rand/2 while three individuals in equation (15). However, If $X_{r2} - X_{r1}$ and $X_{r1} - X_{r3}$ are thought of as two new individual, then equation (15) is the same as DE/rand/1 in essence. Thus a comparison is done between equation (12) and equation (15). The two formulas have the similar structure and is easier to distinguish in the figures if some dissimilarities appear in.

From Figure 2, only if $F = \{0.2,\ 0.3,\ 0.4\}$, the mutant individual obtained by equation (15) is better, whereas that obtained by equation (12) is closer to the global optimal solution. In Figure 3, it is very clear that equation (12) is superior to equation (15). Though the function value of the individual obtained by equation (15) for $F = 0.2$ is almost same as that of obtained by equation (12), it moves uphill wrong.

4.2. Avoiding the Crossover Probability Cr.

Taguchi method can obtain the better combination of the factor level with less cost. In the paper, a two-level orthogonal array $L_m(2^{m-1})$ is used. Since the number of factors (or variables) is $2^k - 1$, where k is an integer greater than 1, the number of experiments m is dependent on the dimension n of the problem. In our paper, m is given as follows:

$$m = \min\left\{2^k \mid k > 1, k \in \mathbb{Z}, 2^k - 1 \geq n\right\}. \tag{16}$$

For instance, if $n = 4$, then $m \geq 3$; if $n = 8$, then $m \geq 4$. In equation (16), the minimal value m subjecting to $m > n$ is chosen for avoiding the possible repeating experiments.

In what follows, a simple algorithm generating the two-level orthogonal array $L_m(2^{m-1})$ is described. The algorithm forms the array by using 2×2 Hadamard matrix H_2.

Definition 1. if any two columns in a matrix H_m consisting of 1 or −1 are orthogonal, then the matrix is called Hadamard matrix [30].

In the above definition, m denotes the order of the Hadamard matrix H_m. There are several operations on Hadamard matrices which preserve the Hadamard property:

(i) Permuting rows (columns)

(ii) Changing the sign of some rows (columns)

(iii) The Kronecker product

If H_m and H_n are known, then $H_{m\cdot n}$ can be obtained by their Kronecker product, namely, by replacing all 1s in H_m by H_n and all -1s by $-H_n$.

Example 1. If $H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, then

$$\begin{aligned} H_2 \otimes H_2 &= \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ &= \left(\begin{array}{c|c} 1\cdot\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 1\cdot\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ \hline 1\cdot\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & -1\cdot\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{array}\right) \\ \Rightarrow H_4 &= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \end{aligned} \tag{17}$$

$$H_2 \otimes H_4 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \Rightarrow$$

$$H_8 = \left(\begin{array}{cccccccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & E_1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & E_5 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & E_3 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & E_7 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & E_2 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & E_6 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & E_4 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & E_8 \end{array}\right)$$

where ⊗ denotes the Kronecker product. Hadamard matrix of high order can be similarly generated from that of lower order: $H_2 \otimes H_4 = H_8$, $H_2 \otimes H_8 = H_{16}$, $H_2 \otimes H_{16} = H_{32}$, $H_2 \otimes H_{32} = H_{64}$, etc.

After a Hadamard matrix H_m is obtained, a two-level orthogonal array $L_m(2^{m-1})$ can be given by discarding the all-one column and changing −1 s to 2 s in H_m. However, this obtained array is not generally canonical form. Therefore, the simple exchange of rows can fix it for consistency (see Table 1 and the gray part in H_8).

Recall the notations about the orthogonal array in Section 3: E_i is denoted by the i-th experiment, C_j by the j-th factor and $C_{i,j}$ by the level of the j-th factor in the i-th experiment. The effects of the factors can be defined as follows:

$$E_{C_j,\text{level}} = \sum_{\substack{1\le i\le m \\ C_{i,j}=\text{level}}} SNR_i. \tag{18}$$

For simplicity, the *SNR* is calculated as follows:

$$\text{SNR} = \sum_{i=1}^{n} \frac{1}{f_i^2}, \tag{19}$$

where level = {1, 2}, $1 \le i \le m$, and $1 \le j \le m-1$. This conception is used here to evaluate the level of the factor. If $E_{C_j,1} > E_{C_j,2}$, the optimal level of the factor C_j is 1, otherwise, the optimal level is 2. When each $E_{C_j,\text{level}}$ is determined, a new individual (an optimal or near-optimal combination) is generated.

Example 2. An example $\min f(X) = \|X\|_1$ is shown to illustrate this process of Taguchi parameter design method acting on two individuals, where $X \in R^7$. Without loss of generality, let $V = [0, 8, 1, 0, -72, 0, 0]$ and $X = [0, 0, -28, 35, 0, 32, 0]$. This problem has 7 variables (factors), thus according to equation (16):

$m-1=2^3-1=7\geq n$, the orthogonal array $L_8(2^7)$ is chosen(See Table 1). If $C_{i,j}$ is equal to 1 in Table 1, then the corresponding $C_{i,j}$ in Table 2 is the j-th component V^j of the mutant individual V, otherwise, the corresponding $C_{i,j}$ is X^j, see bold in Table 2.

Next, calculate the function value $f(X)$ and the SNR of each combination of the factor level in Table 2, respectively. All results appear in the two most right hand columns. Then the effect of each factor is determined in terms of equation (18) (take C_1 and C_7 as an example).

$$\begin{aligned}E_{C_1,1} &= \sum_{i=1,2,3,4} SNR_i\\ &= 0.00015+0.00017+0.00006+0.00025\approx 0.0006,\\ E_{C_1,2} &= \sum_{i=5,6,7,8} SNR_i\\ &= 0.00077+0.00003+0.00092+0.00009\approx 0.0018,\\ E_{C_7,1} &= \sum_{i=1,4,6,7} SNR_i\\ &= 0.00015+0.00025+0.00003+0.00092\approx 0.0014,\\ E_{C_7,2} &= \sum_{i=2,3,5,8} SNR_i\\ &= 0.00017+0.00006+0.00077+0.00009\approx 0.0011.\end{aligned}\tag{20}$$

Finally, we obtain the new individual or the trial vector W. The optimal level of the factor is decided by its effect. Since $E_{C_1,1}<E_{C_1,2}$, 2 is the optimal level of the factor C_1; $E_{C_2,1}>E_{C_2,2}$, therefore the optimal level of the factor C_2 is 2. The optimal levels of the other factors can be determined in a similar way. The component W^j of the new individual W consists of either V^j or X^j for all j, which is dependent on the optimal level of the factor C_j. If the optimal level is 1, then the corresponding component of the new individual is that of the individual V, otherwise, it is equal to that of the individual X.

Obviously, Taguchi parameter design method executes only 8 experiments instead of all 2^7 combinations of factor levels for obtaining a new individual W with the lower function value 1 (see the last row in Table 2). It is necessary to mention that only the first n columns is used in orthogonal array while the other columns are ignored if $n<m-1$.

In reference [31], the hybrid Taguchi-genetic algorithm (HTGA) is proposed for global numerical optimization with the continuous variables, which uses the systematic reasoning ability of Taguchi parameter design to gain the better genes in the crossover operation. The comparison results between HTGA and OGA/Q [32] show that HTGA can find the optimal or the near-optimal solutions with less function evaluations and better average values. However, this superiority is not very obvious for the tested function with nonzeros optimal values. Let we recall the original definition of SNR in the case of smaller-the-better characteristic, which is described in equation (4), and change it to

$$\mathrm{SNR}=-10\cdot\log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{(y_i-0)^2}\right).\tag{21}$$

In Taguchi method, the item $1/n\sum_{i=1}^{n}1/(y_i-s)^2$ represents the average loss of quality, where s denotes the ideal signal in the case of smaller-the-better characteristic. Therefore, equation (21) shows that HTGA has thought of the optimal value of the tested problem as 0 before this problem is solved. This is unfair and unreasonable. As described above, we found that the superiority of HTGA is not very obvious for those function with nonzero optimal value from Tables IV and V on page 273 and 275 in the reference [31], Hence, SNR is modified as follows:

$$\mathrm{SNR}=\sum_{i=1}^{n}\frac{1}{(f_i-f_t^*)^2},\tag{22}$$

where f_t^* is defined as the current optimal value after the t-th iteration.

In what follows, the differential evolutions without the scale factor and the crossover probability (for short, DE∖FCr) are proposed. For the sake of clarity, the flowcharts of DE and DE∖FCr(take DE∖FCr2 as an example) are also given in Figure 4, where cross() represents the crossover operation in equation (2) and Taguchi() denotes Taguchi parameter design method in equation (13) (see Algorithm 1).

In Step 2, a termination criterion $|f(X_{\text{worst}})-f(X_{\text{best}})|<\varepsilon$ is given since $f(X_{\text{worst}})-f(X_{\text{best}})$ is the denominator in Eq.(8). When this difference approximates to zero, the numerical stability of the algorithm will lose.

Let $X^{(t)}$ be the population at the t-th generation. Through Step 4-5 in *DE\FCr*, $X^{(t)}$ transforms into the next population $X^{(t+1)}$. Since the limitation of numerical calculation accuracy and $X^{(t+1)}$ relies only on the state of $X^{(t)}$, the population sequence $\{X^{(t)}\}_{t\geq 0}$ generated by DE\FCr can be described as finite-state Markov stochastic process.

Suppose (i) the objection function $f(X)$ has a unique global optimal solution. Let S be the state space of the stochastic process $X^{(t)}$, S^* be the state space of the global optimal solution, and f^* be the global optimal value.

Because of the limitation of state space or search space, the probability that the algorithm can find the optimal solution at the next generation is greater than 0 if it cannot find at the t-th generation, hence, suppose (ii) $P\left\{X^{(t+1)}=s_j\,|\,X^{(t)}=s_i\right\}>\rho>0$ for $s_i\notin S^*$ and $s_j\in S^*$, where $\rho\in(0,1)$.

Now, we consider the probability $\sum_{s_j\notin S^*}P\left\{X^{(t+1)}=s_j\right\}$ that the proposed algorithm can not find the global optimum at the $t+1$ generation.

```
**Step 1**: Initialization: population $P$, population size $NP$, maximal generation $T$, current generation $t = 1$, and $\varepsilon = 10^{-100}$, $i = 1$.
**Step 2**: If $t > T$ or $|f(X_{\text{worst}}) - f(X_{\text{best}})| < \varepsilon$, then output the current optimal value $f_t^*$.
**Step 3**: Mutation. For each $X_i \in P_t$ in the population, calculate the mutant individual according to equations (12) or (13) or equation (14). The corresponding algorithm is denoted by DE\FCr1, DE\FCr2 and DE\FCr3, respectively.
**Step 4**: Crossover. Execute Taguchi parameter design method with the $SNR$ denoted as equation (22) for the individual $X_i$ and the mutant individual $V_i$, so the trail individual $W_i$ is generated.
**Step 5**: Selection. If $f(W_i) < f(X_i)$, then $X_i = W_i$ and $i = i + 1$.
**Step 6**: If $i <= NP$, goto Step 3; otherwise, $t = t + 1$, goto Step 2.
```

Algorithm 1: (DE\FCr).

$$\begin{aligned}
&\sum_{s_j \notin S^*} P\left\{X^{(t+1)} = s_j\right\} \\
&= \sum_{s_j \notin S^*} \sum_{s_i \in S} P\left\{X^{(t+1)} = s_j, X^{(t)} = s_i\right\} \\
&= \sum_{s_j \notin S^*} \sum_{s_i \in S} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} \\
&= \sum_{s_j \notin S^*} \sum_{s_i \notin S^*} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} \\
&\quad + \sum_{s_j \notin S^*} \sum_{s_i \in S^*} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} \\
&= \sum_{s_j \notin S^*} \sum_{s_i \notin S^*} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} \\
&= \sum_{s_i \notin S^*} P\left\{X^{(t)} = s_i\right\} \\
&\quad - \sum_{s_j \in S^*} \sum_{s_i \notin S^*} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} \\
&< (1-\rho) \sum_{s_i \notin S^*} P\left\{X^{(t)} = s_i\right\}.
\end{aligned} \tag{23}$$

It is very obvious that the current known optimal solution still can be retained in the next generation from Step 5. Once $DE\backslash FCr$ finds the optimal solution, the $X^{(t+1)}$ will hold the current state S^*. Hence, in the equation (23),

$$\sum_{s_j \notin S^*} \sum_{s_i \in S^*} P\left\{X^{(t+1)} = s_j \mid X^{(t)} = s_i\right\} P\left\{X^{(t)} = s_i\right\} = 0. \tag{24}$$

Summarizing the result of equation (23), we have

$$0 \le \sum_{s \notin S^*} P\left\{X^{(t+1)} = s_j\right\} < (1-\rho) \sum_{s \notin S^*} P\left\{X^{(t)} = s_i\right\}. \tag{25}$$

Because the sequence $\sum_{s \notin S^*} P\{X^{(t)} = s_i\}$ is strictly monotonic decreasing as $t \longrightarrow \infty$, so

$$\lim_{t \longrightarrow \infty} \sum_{s \notin S^*} P\left\{X^{(t)} = s_i\right\} = 0. \tag{26}$$

Therefore,

$$\lim_{t \longrightarrow \infty} P\{f_t^* = f^*\} = 1 - \lim_{t \longrightarrow \infty} \sum_{s \notin S^*} P\left\{X^{(t)} = s_i\right\} = 1. \tag{27}$$

Equation (26) shows that the population sequence generated by $DE\backslash FCr$ can convergence in probability to the global optimum.

5. Numerical Experiments

The proposed Algorithms are executed in Matlab R2017 for the known numerical benchmark functions listed in Appendix with the default parameters $NP = 30$ and $T = 202$. Based on this parameter setting, each $DE\backslash FCr$ needs $30 \times (m+1)$ function evaluations at each iteration for 30 dimensional tested functions. *DEs* with the four strategies below are compared with our algorithms, respectively.

Strategy 1(DE): DE/Rand/1. $F = 0.5$, $Cr = 0.9$. This is a recommend parameters setting for DE/Rand/1 in most of the references [1–13];

Strategy 2(DEG): $F \sim N(0,1)$, $C_r = 0.9$ [11];

Strategy 3(DE0.4): $F = 0.4 + 0.4 \cdot rand(0,1)$, $C_r = 0.9$ [12]

Strategy 4(DEM): $C_r = 0.5$ and F is calculated by the following formula [13]:

$$F = \begin{cases} \max\left\{0.4, 1 - \left|\dfrac{f(X_{\text{worst}})}{f(X_{\text{best}})}\right|\right\}, & \text{if} \left|\dfrac{f(X_{\text{worst}})}{f(X_{\text{best}})}\right| < 1, \\[2ex] \max\left\{0.4, 1 - \left|\dfrac{f(X_{\text{best}})}{f(X_{\text{worst}})}\right|\right\}, & \text{otherwise.} \end{cases} \tag{28}$$

For Strategy 1–4, the population size NP and the maximal generation T are set as 100 and 2000, respectively. All algorithms are performed with 10 independent runs for each tested function with 30 variables. According to these settings, our algorithm has the almost same function evaluations as *DEs* with Strategy 1–4, that is, $100 + 100 \times 2000 = 2000100$ for *DEs* and $30 + 990 \times 202 = 200010$ for $DE\backslash FCr$. Hence, the results listed in Tables 3 and 4 are obtained under the assumption of the not same but different function evaluations. Obviously, the proposed algorithm evaluates 90 function values less than *DEs*. The average values of the

Table 2: The process of Taguchi parameter design method acting on the individuals V and X.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	$f(X)$	SNR
E_1	0	8	1	0	−72	0	0	81	0.00015
E_2	0	8	1	35	0	32	0	76	0.00017
E_3	0	0	−28	0	−72	32	0	132	0.00006
E_4	0	0	−28	35	0	0	0	63	0.00025
E_5	0	8	−28	0	0	0	0	36	0.00077
E_6	0	8	−28	35	−72	32	0	175	0.00003
E_7	0	0	1	0	0	32	0	33	0.00092
E_8	0	0	1	35	−72	0	0	108	0.00009
$E_{C,1}$	0.0006	0.0011	0.0013	0.0019	0.0003	0.0013	0.0014		
$E_{C,2}$	0.0018	0.0013	0.0011	0.0005	0.0021	0.0012	0.0011		
W	0	0	1	0	0	0	0	1	

Table 3: Result comparisons among 7 algorithms for $F1$–$F5$.

Fun	Alg	Best	Worst	Mean	Std	#Elav
	DE	6.445542E − 023	4.436277E − 022	2.290635E − 022	1.108992E − 022	200100
	DEG	**1.046763E − 040**	**4.175516E − 039**	**1.128546E − 039**	**1.247818E − 039**	200100
	*DE*0.4	1.348390E − 013	1.781579E − 012	6.963543E − 013	5.084985E − 013	200100
$F1$	*DEM*	1.212964E − 011	8.731799E − 011	3.082813E − 011	2.407010E − 011	200100
	*DE\FCr*1	3.228810E − 022	2.671220E − 019	3.358173E − 020	8.252692E − 020	**200010**
	*DE\FCr*2	2.611086E − 024	1.537412E + 001	1.546806E + 000	4.858513E + 000	**200010**
	*DE\FCr*3	1.808723E − 025	2.760420E − 024	8.956028E − 025	9.805783E − 025	**200010**
	DE	2.342126E − 012	1.023626E − 011	5.329959E − 012	2.673148E − 012	200100
	DEG	**4.440892E − 015**	**7.993606E − 015**	**7.638334E − 015**	**1.123467E − 015**	200100
	*DE*0.4	1.512876E − 007	4.988239E − 007	3.000771E − 007	1.363652E − 007	200100
$F2$	*DEM*	4.440892E − 015	7.993606E − 015	7.283063E − 015	1.497956E − 015	200100
	*DE\FCr*1	3.135270E − 013	2.617373E − 011	7.357492E − 012	8.539221E − 012	**200010**
	*DE\FCr*2	3.996803E − 014	1.434325E − 008	1.434625E − 009	4.535629E − 009	**200010**
	*DE\FCr*3	1.509903E − 014	1.927347E − 013	6.483702E − 014	5.207955E − 014	**200010**
	DE	4.262079E − 003	1.641628E − 002	8.883045E − 003	3.323616E − 003	200100
	DEG	**3.490604E − 003**	**1.002710E − 002**	**5.873780E − 003**	**1.893076E − 003**	200100
	*DE*0.4	7.746132E − 003	1.612982E − 002	1.284244E − 002	2.895560E − 003	200100
$F3$	*DEM*	5.340829E − 002	8.776692E − 002	7.115915E − 002	9.972597E − 003	200100
	*DE\FCr*1	1.040566E − 002	2.893510E − 002	1.649259E − 002	5.547522E − 003	**200010**
	*DE\FCr*2	1.087789E − 002	3.538599E − 002	2.176316E − 002	7.961170E − 003	**200010**
	*DE\FCr*3	2.047691E − 002	3.617698E − 002	2.779340E − 002	4.572319E − 003	**200010**
	DE	4.869042E − 024	1.608507E − 022	4.278891E − 023	5.275054E − 023	200100
	DEG	**1.570545E − 032**	4.146719E − 001	4.146719E − 002	1.311308E − 001	200100
	*DE*0.4	2.196374E − 014	1.812240E − 013	7.778539E − 014	5.733436E − 014	200100
$F4$	*DEM*	1.962642E + 004	1.637335E + 005	8.747525E + 004	4.588788E + 004	200100
	*DE\FCr*1	5.576314E − 024	1.036690E − 001	1.036690E − 002	3.278302E − 002	**200010**
	*DE\FCr*2	1.176714E − E − 025	3.090863E − 001	4.789975E − 002	9.877257E − 002	**200010**
	*DE\FCr*3	7.336740E − 027	**8.506532E − 025**	**1.761171E − 025**	**2.553095E − 025**	**200010**
	DE	3.538508E − 023	2.902433E − 022	1.590376E − 022	8.749484E − 023	200100
	DEG	**1.349784E − 032**	1.098737E − 002	1.098737E − 003	3.474510E − 003	200100
	*DE*0.4	1.032194E − 013	1.261348E − 012	5.065837E − 013	4.532350E − 013	200100
$F5$	*DEM*	2.220360E + 004	1.152671E + 005	6.319184E + 004	3.036898E + 004	200100
	*DE\FCr*1	1.155559E − 021	1.691974E − 015	1.692275E − 016	5.350387E − 016	**200010**
	*DE\FCr*2	3.452407E − 023	1.247897E + 000	2.167329E − 001	4.586437E − 001	**200010**
	*DE\FCr*3	6.397232E − 026	**2.198487E − 024**	**1.088356E − 024**	**6.533660E − 025**	**200010**

The best results in the table are bolded.

obtained results are given in Tables 3 and 4. The number of $f(x)$ evaluations(#EVALU.), the best function value(BEST), the worst function value(WORST), the mean of function values(MEAN) and the standard deviation of the best function values(STD.) are used for the comparisons among these algorithms.

Table 3 summarizes the results obtained by the 7 algorithms for $F1$-$F5$. For $F1$, DEG obtains the best mean of

Table 4: Result comparisons among 7 algorithms for $F6$–$F10$.

Fun	Alg	Best	Worst	Mean	Std	#Elav
	DE	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	200100
	DEG	**0.000000E + 000**	1.969000E − 002	5.666035E − 003	6.872102E − 003	200100
	*DE*0.4	7.412959E − 013	3.983369E − 012	1.929468E − 012	9.211849E − 013	200100
*F*6	*DEM*	8.237855E − 014	2.578935E − 010	2.624131E − 011	8.139491E − 011	200100
	*DE**FCr*1	**0.000000E + 000**	7.396040E − 003	7.396040E − 004	2.338833E − 003	167650
	*DE**FCr*2	**0.000000E + 000**	1.477241E − 002	2.216845E − 003	4.986442E − 003	170175
	*DE**FCr*3	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	**148553**
	DE	1.346933E + 002	1.815693E + 002	1.604947E + 002	1.647296E + 001	200100
	DEG	5.969754E + 000	2.089413E + 001	1.492438E + 001	4.115703E + 000	200100
	*DE*0.4	1.447778E + 002	1.987496E + 002	1.736384E + 002	1.838650E + 001	200100
*F*7	*DEM*	8.895676E + 001	1.067928E + 002	1.005014E + 002	6.041157E + 000	200100
	*DE**FCr*1	3.979836E + 000	8.954632E + 000	5.870258E + 000	1.654945E + 000	200010
	*DE**FCr*2	**0.000000E + 000**	2.984877E + 000	6.964713E − 001	1.153657E + 000	152929
	*DE**FCr*3	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	**0.000000E + 000**	**173891**
	DE	4.734026E + 000	**9.125210E + 000**	**7.317988E + 000**	1.118800E + 000	200100
	DEG	**3.129173E + 000**	7.336176E + 001	2.244019E + 001	2.658720E + 001	200100
	*DE*0.4	9.489415E + 000	1.273116E + 001	1.136916E + 001	**9.699009E − 001**	200100
*F*8	*DEM*	2.646006E + 001	4.035420E + 002	9.863638E + 001	1.151125E + 002	200100
	*DE**FCr*1	2.198234E + 001	1.319067E + 002	6.198206E + 001	3.617924E + 001	**200010**
	*DE**FCr*2	1.491798E + 001	1.011474E + 002	3.406901E + 001	2.894100E + 001	**200010**
	*DE**FCr*3	1.130096E + 001	7.634987E + 001	2.707867E + 001	1.774072E + 001	**200010**
	DE	6.212269E − 011	2.410512E − 010	1.249117E − 010	5.339068E − 011	200100
	DEG	**5.493251E − 023**	**3.940883E − 022**	**1.728275E − 022**	**1.114004E − 022**	200100
	*DE*0.4	1.482390E − 006	6.510805E − 006	3.673915E − 006	1.530019E − 006	200100
*F*9	*DEM*	6.282339E + 000	7.764649E + 001	6.079609E + 001	2.075941E + 001	200100
	*DE**FCr*1	8.632854E − 016	4.274906E − 014	9.826062E − 015	1.315808E − 014	**200010**
	*DE**FCr*2	1.792127E − 016	2.172822E − 001	2.702991E − 002	6.889298E − 002	**200010**
	*DE**FCr*3	1.435442E − 017	4.834137E − 016	9.552152E − 017	1.422569E − 016	**200010**
	DE	**6.521876E − 005**	**4.212407E − 003**	8.509549E − 004	1.555632E − 003	200100
	DEG	7.185030E + 000	2.254554E + 001	1.543480E + 001	5.184946E + 000	200100
	*DE*0.4	8.266525E − 003	5.784312E − 001	8.974530E − 002	1.759532E − 001	200100
*F*10	*DEM*	1.028353E − 004	2.981224E − 004	**2.063294E − 004**	**5.460973E − 005**	200100
	*DE**FCr*1	9.550667E + 000	2.092655E + 001	1.313933E + 001	3.863092E + 000	**200010**
	*DE**FCr*2	9.407884E + 000	2.291669E + 001	1.598827E + 001	4.119512E + 000	**200010**
	*DE**FCr*3	2.517749E − 002	3.977652E − 001	1.628583E − 001	1.193423E − 001	**200010**

The best results in the table are bolded.

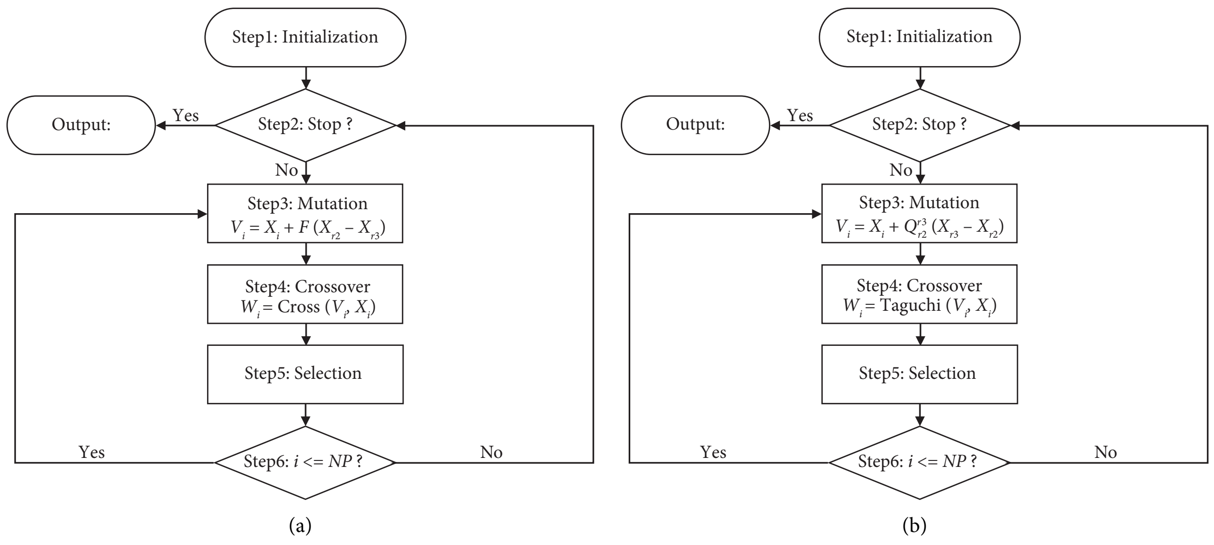
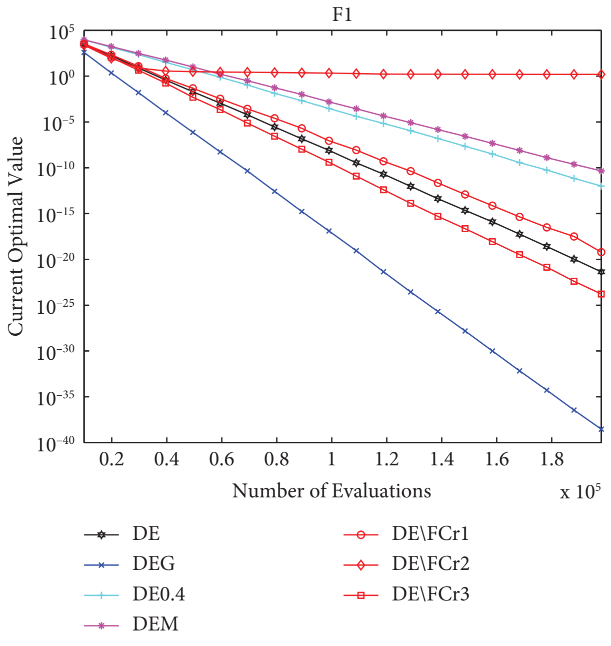


Figure 4: Flowcharts of DE and DE\FCr2. (a) DE. (b) DE\FCr2.

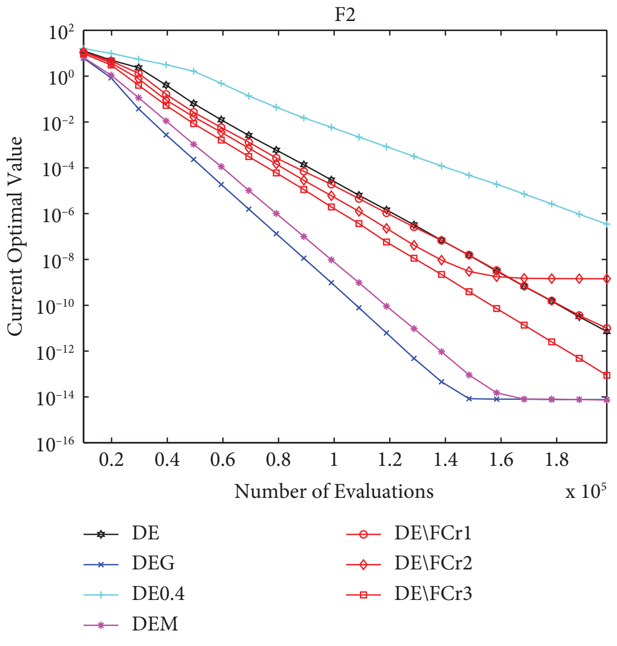
Table 5: Result comparisons between *DE**FCr*3 and *DENSO* for *F*1–*F*10.

Fun	Alg	Best	Worst	Mean	Std	#Elav
*F*1	*DE**FCr*3	**1.808723E − 25**	**2.760420E − 24**	**8.956028E − 25**	**9.805783E − 25**	**200010**
	DENSO	8.585195*E* − 06	1.446810*E* − 05	1.210160*E* − 05	1.972164*E* − 06	200100
*F*2	*DE**FCr*3	**1.509903E − 14**	**1.927347E − 13**	**6.483702E − 14**	**5.207955E − 14**	**200010**
	DENSO	7.522790*E* − 04	1.429218*E* − 03	9.793750*E* − 04	2.043630*E* − 04	200100
*F*3	*DE**FCr*3	2.047691*E* − 02	3.617698*E* − 02	2.779340*E* − 02	4.572319*E* − 03	**200010**
	DENSO	**8.027579E − 03**	**2.113126E − 02**	**1.426177E − 02**	**3.372486E − 03**	200100
*F*4	*DE**FCr*3	**7.336740E-27**	**8.506532E-25**	**1.761171E-25**	**2.553095E-25**	**200010**
	DENSO	9.835739*E* − 07	5.804517*E* − 06	2.661288*E* − 06	1.493261*E* − 06	200100
*F*5	*DE**FCr*3	**6.397232E − 26**	**2.198487E − 24**	**1.088356E − 24**	**6.533660E − 25**	**200010**
	DENSO	1.551095*E* − 05	8.342158*E* − 05	3.712426*E* − 05	2.019380*E* − 05	200100
*F*6	*DE**FCr*3	**0.000000E + 00**	**0.000000E + 00**	**0.000000E + 00**	**0.000000E + 00**	**148553**
	DENSO	2.194028*E* − 05	7.591791*E* − 03	8.146725*E* − 04	2.381668*E* − 03	200100
*F*7	*DE**FCr*3	**0.000000E + 00**	**0.000000E + 00**	**0.000000E + 00**	**0.000000E + 00**	**173891**
	DENSO	1.245423*E* + 01	1.903578*E* + 01	1.599377*E* + 01	2.169709*E* + 00	200100
*F*8	*DE**FCr*3	**1.130096E + 01**	7.634987*E* + 01	2.707867*E* + 01	1.774072*E* + 01	**200010**
	DENSO	2.430580*E* + 01	**2.530343E + 01**	**2.468209E + 01**	**3.004402E − 01**	200100
*F*9	*DE**FCr*3	**1.435442E + 17**	**4.834137E − 16**	**9.552152E − 17**	**1.422569E − 16**	**200010**
	DENSO	2.975740*E* + 04	7.035782*E* + 04	5.295819*E* + 04	1.148206*E* + 04	200100
*F*10	*DE**FCr*3	**2.517749E − 02**	**3.977652E − 01**	**1.628583E − 01**	**1.193423E − 01**	**200010**
	DENSO	2.013176*E* − 01	9.015901*E* − 01	4.872225*E* − 01	2.103323*E* − 01	200100

The best results in the table are bolded.

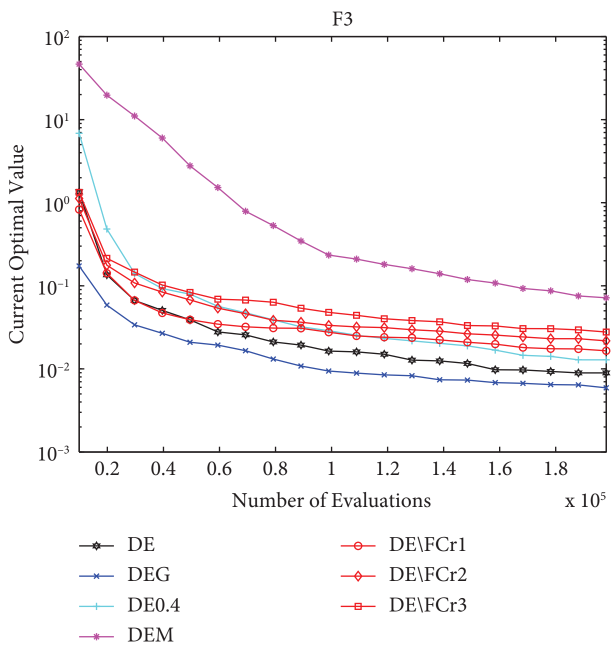


(a)

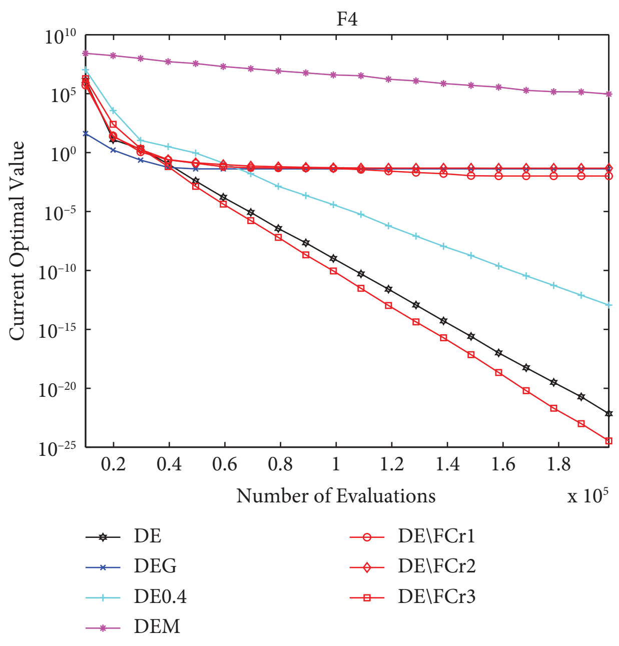


(b)

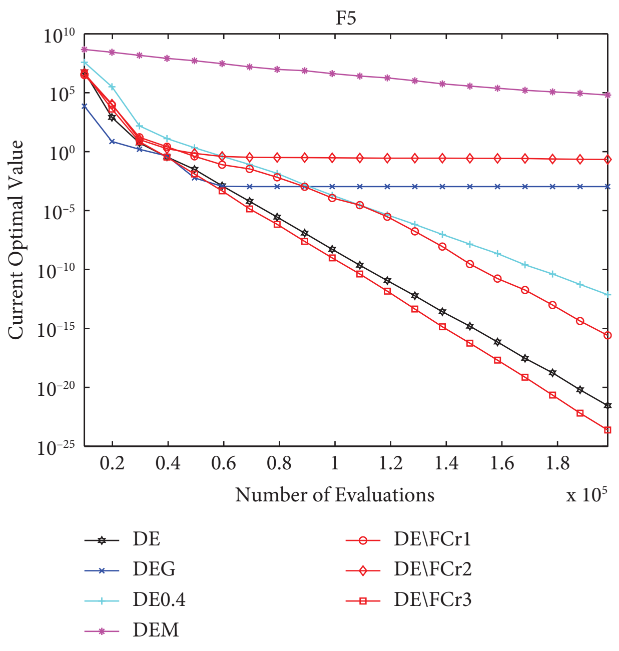
Figure 5: Continued.



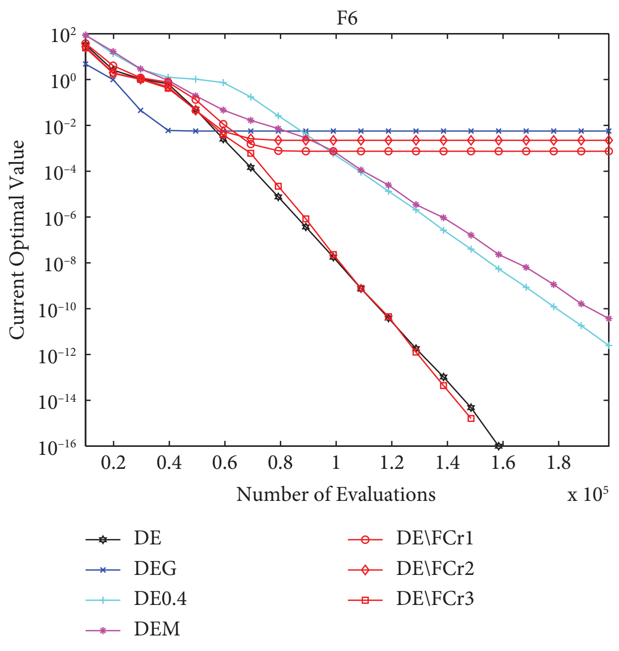
(c)



(d)



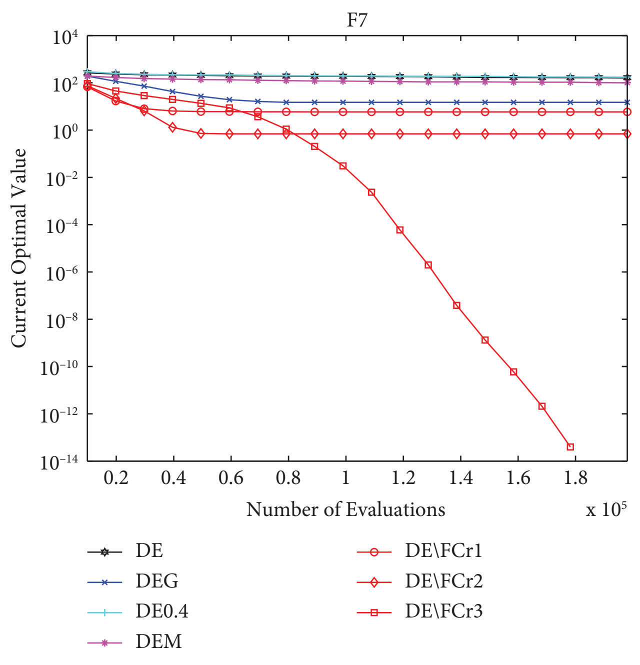
(e)



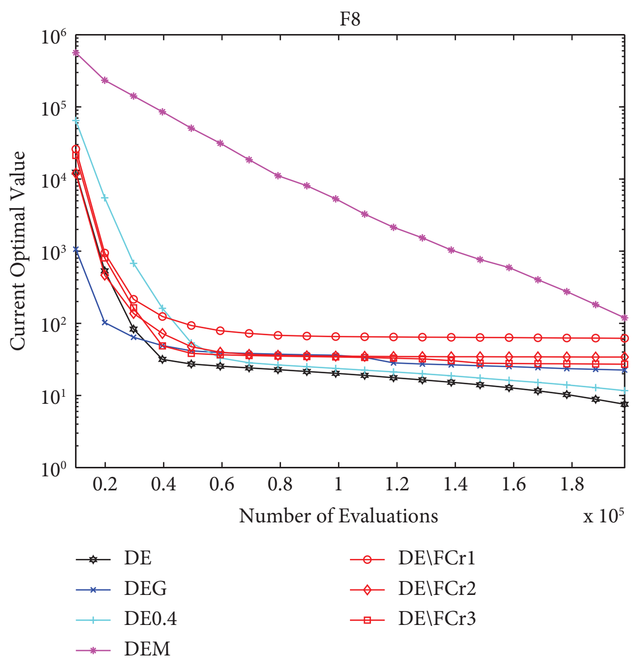
(f)

Figure 5: The mean of the current optimal values obtained by 7 algorithms with the number of function evaluations for $F1$–$F6$. (a) $F1$. (b) $F2$. (c) $F3$. (d) $F4$. (e) $F5$. (f) $F6$.

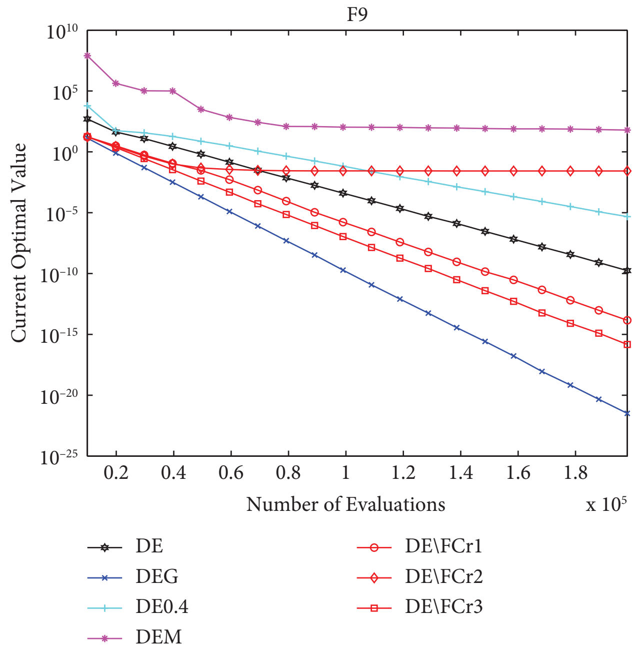
function values and the smallest standard deviation; $DE\backslash FCr1$, $DE\backslash FCr3$ and DE find the better results; The means and standard deviations given by both $DE0.4$ and DEM are worst among all algorithms; After 20 independent runs $DE\backslash FCr2$ finds a better function value, however, it obtains the worst standard deviation because it encounters twice the worst function value 15.37. For $F2$, DEG and DEM find the best function values with the almost same precision $E-15$; The precision given by $DE\backslash FCr3$ is $E-14$; DE and $DE\backslash FCr1$ give the precision of $E-12$; However, the lower precisions provided by $DE\backslash FCr2$ and $DE0.4$ are $E-9$ and $E-7$ respectively. All of algorithms obtain the best function values with the almost same precision for tested function $F3$. For $F4$, the results given by $DE\backslash FCr3$ are best, and those provided by DE are a little bit less promising; $DE0.4$ find the less promising optimum with the precision $E-14$, while the precisions provided by DEG and $DE\backslash FCr2$ reach only $E-2$; DEM fails in finding the optima of $F4$ among 20 independent runs, and traps into the local optima. Both $DE\backslash FCr1$, $DE\backslash FCr3$, DE and $DE0.4$ solve efficiently $F5$



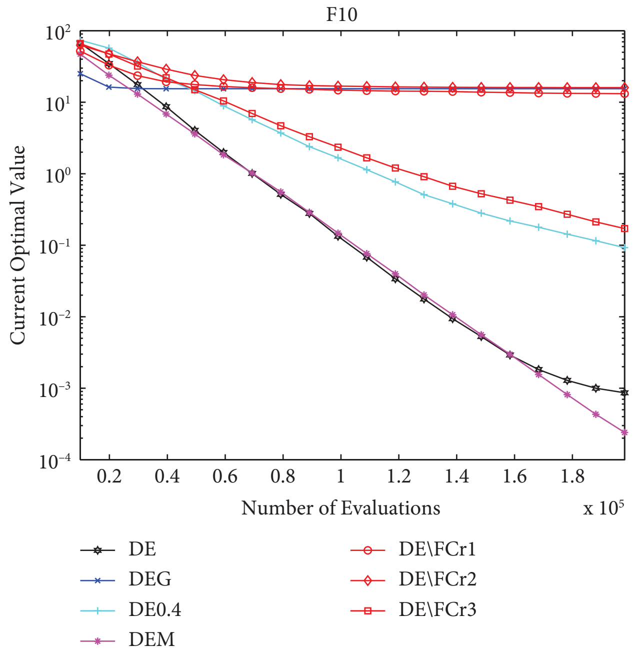
(a)



(b)



(c)



(d)

FIGURE 6: The mean of the current optimal values obtained by 7 algorithms with the number of function evaluations for *F*7–*F*10. (a) *F*7. (b) *F*8. (c) *F*9. (d) *F*10.

with the precision $E-24$, $E-22$, $E-16$ and $E-13$ respectively; The results given by DEG and *DE**FCr*2 are worse; Similarly, *DEM* fails to solve *F*5.

In Table 4, Both *DE**FCr*3 and *DE* find the optima of *F*6, however the fewer number of function evaluations 148533 is used by *DE*\ *FCr*3; The results obtained by *DE*0.4 and *DEM* are sightly worse than those obtained by *DE**FCr*3 and *DE*; However, *DE**FCr*1, *DE**FCr*2 and DEG have the lower precisions of about $E-3$. *DE**FCr*3 finds the optimum of tested function *F*7 with highest precision and smallest STD and the fewer number of function evaluations than the other algorithms; *DE**FCr*1 and *DE**FCr*2 reach the a little bit worse precision, and the anther algorithms fail to find the optimum of *F*7 with 200100 function evaluations. For 30 dimensional Rosenbrock tested function *F*8, none of all algorithms is obviously superior to the other one, namely, all algorithms can not find a satisfactory optima. For *F*9, DEG produces a best results, while $DE\backslash FCr1$ and $DE\backslash FCr3$ provides the sightly worse results; The precisions of the mean function values given by DE and $DE0.4$ reach $E-10$ and $E-6$, respectively; DEM can not find the reasonable result, and only the mean value 60.796 is presented. DE and DEM find the best means of function values of $F10$ with the precision of $E-4$; The precisions given by $DE0.4$ and $DE\backslash FCr3$ are $E-2$ and $E-1$ respectively; DEG, $DE\backslash FCr1$ and $DE\backslash FCr2$ give the almost same results and fail to find a satisfactory optima of $F10$.

In order to show further the efficiency of $DE\backslash FCr$, the means of the current optimal values obtained by 7 algorithms with the almost same number of function evaluations for each tested functions are respectively given in Figures 5 and 6. As mentioned in previous section, $|f(X_{\text{worst}}) - f(X_{\text{best}})| < \varepsilon$ is used for the numerical stability, hence each $DE\backslash FCr$ stops probably before the maximal generation is reached. For convenience to draw the following figures, the current optimal value is recorded repeatedly in succeeding generations if the algorithm stops in advance since we think that the algorithm cannot be improved greatly in succeeding running.

Since the proposed algorithm is not same as the compared algorithms in the number of function evaluations at each iteration, it is inconvenient to draw the evolution curves describing the variations of $MEAN$ with the number of function evaluations in a figure window for the reasonable comparison among all algorithms. Therefore, the current optimal values with the number of function evaluations which is denoted by $t \cdot \text{lcm}\{990, 100\}$ for $t = 1, 2, \ldots, 20$ respectively are drawn in Figures 5 and 6 without considering the number of function evaluations costed by initialization. In fact, the sequence $t \cdot \text{lcm}\{990, 100\}, t = 1, \ldots, 20$ is an arithmetic sequence with the initial term 9900 and the common difference 9900, where lcm denotes the least common multiple. 990 and 100 represent the number of function evaluations of $DE\backslash FCr$ and those of DEs at each iteration, respectively. It needs to be emphasized that each proposed algorithm evaluates 70(=100−30) less than the compared algorithm at each given iteration as above. Consequently, the current optimal value obtained by each compared algorithm under the given number of function evaluations is just recorded at certain generation which is $10 + (t-1) \cdot 10$ for each $DE\backslash FCr$ and $99 + (t-1) \cdot 99$ for DEs. Hence, only according to the recorded current optimal value at each generation can the figures below be given easily.

From each figure(see Figures 5 and 6), DEG outperforms the other algorithms for $F1$, $F2$, and $F9$, whereas $DE\backslash FCr3$ surpasses the other ones for $F4$–F7 and $F10$. For $F1$, $F2$, and $F9$, $DE\backslash FCr3$ is on the top three of 7 algorithms in terms of the performance. However, For $F4$–F7 and $F10$, DEG drops out of the top three almost into the last three. It is worth noting that both $F3$ and $F8$ don't be considered because all algorithms, especially DEG and $DE\backslash FCr3$, obtain the almost same results. We also find that DEG can obtain the optimum with the higher precision at earlier generation than the other algorithms and enters easily into the local optimum at latter generation for $F2$, $F4$–$F6$, and $F10$. $DE\backslash FCr3$ finds the satisfactory results of most of tested functions except $F8$ and also has not the tendency toward the local optimum with the default parameters.

In a summary, $DE\backslash FCr3$ does rather well in terms of the performance, DEG and DE are a little bit less promising, $DE\backslash FCr1$ and $DE0.4$ are even less promising, and $DE\backslash FCr2$ and DEM are worst.

Furthermore, the numerical comparison experiments are done between $DE\backslash FCr3$ and $DENSO$ (see Table 5). $DENSO$ is proposed in reference [19], which employs three other candidate individuals to design a new selection operator for improving the ability to escape the local optimum. In Table 5, $DE\backslash FCr3$ find the optima of the tested functions $F1, F2, F4, F5, F9$ with higher precision. For $F3, F8, F10$, Both algorithms have the almost same precision, however, $DE\backslash FCr3$ reduces 90 function evaluations. Obviously, $DE\backslash FCr3$ give the global minimal value 0 with the fewer $\#ELAV$ for $F6, F7$.

6. Conclusion

For avoiding the settings of the parameters, the differential evolutions without F and Cr are presented. The proposed algorithms use the attraction-repulsion mechanism in Coulomb's Law and Taguchi parameter design method for the purpose of eliminating the scale factor and the crossover probability, respectively. Numerical experiments show that the proposed algorithm $DE\backslash FCr3$, which can balance well between exploration and exploitation, is superior to the compared algorithms with other strategies and can find quickly the optima or the near-optima of the problems. Although a smaller population size 30 is given in the proposed algorithms for all 30 dimensional tested functions, this small population maybe lead to the prematurity of algorithm such as $F8$. However a larger population will expend too many function evaluations because of using the two-level orthogonal array which is related with the dimension of the problems. Obviously, In our algorithms the number of function evaluations of each proposed algorithm at each generation is $(m+1) \cdot NP$. Therefore, as for future work, the following problems are going to be investigated: (i) decrease the function evaluations at each generation and increase the population size without the loss of the algorithmic performance; (ii) analyze the accelerated convergence behavior of the current optimal value f_t^* after the t-th iteration in equation (22).

Appendix

$$
\begin{aligned}
&F0\colon\ 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, x \in [-5,5]^n,\\
&F1\colon\ \sum_{i=1}^{n} x_i^2, x \in [-100,100]^n,\\
&F2\colon\ -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1), x \in [-32,32]^n,\\
&F3\colon\ \sum_{i=1}^{n} i\cdot x_i^4 + \mathrm{rand}[0,1], x \in [-1.28,1.28]^n,\\
&F4\colon\ \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right] + (y_n-1)^2\right\} + \sum_{i=1}^{n} u(x_i,10,100,4),\\
&y_i = 1 + \frac{x_i+1}{4}, i = 1,\ldots,n, u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m, x_i > a\\ 0, a \le x_i \le a\\ k(-x_i-a)^m, x_i < -a\end{cases}, x \in [-50,50]^n,\\
&F5\colon\ \frac{1}{10}\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right] + (x_n-1)^2\left[1+\sin^2(2\pi x_n)\right]\right\}\\
&\quad + \sum_{i=1}^{n} u(x_i,5,100,4), x \in [-50,50]^n,\\
&F6\colon\ \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\frac{x_i}{\sqrt{i}} + 1, x \in [-600,600]^n,\\
&F7\colon\ 10n + \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i)\right], x \in [-5.12,5.12]^n,\\
&F8\colon\ \sum_{i=1}^{n-1}\left[100\left(x_i^2 - x_{i+1}\right)^2 + (x_i-1)^2\right], x \in [-5,10]^n,\\
&F9\colon\ \sum_{i=1}^{n}|x_i| + \prod_{i=1}^{n}|x_i|, x \in [-100,100]^n\\
&F10\colon\ \max\left\{|x_i|, i = 1,2,\cdots,n\right\}, x \in [-100,100]^n.
\end{aligned}
\tag{A.1}
$$

Data Availability

All the data, MATLAB codes, and plots used in the paper can be found at https://github.com/zhang-xiaowei/Code.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the projects of UESTC under grant nos. JPKC20203-18 and JPKC20192-40.

References

[1] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[2] M. Bilal, M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Z. Abraham, “Differential evolution: a review of more than two decades of research,” *Engineering Applications of Artificial Intelligence*, vol. 90, pp. 103479–24, 2020.

[3] M. D. F. Morais, M. H. D. M. Ribeiro, R. G. da Silva, V. C. Mariani, and L. D. S. Coelho, “Discrete differential evolution metaheuristics for permutation flow shop

scheduling problems," *Computers and Industrial Engineering*, vol. 166, Article ID 107956, 2022.

[4] J. Zhang and A. C. J. A. D. E. Sanderson, "Self-adaptive differential evolution with fast and reliable convergence performance," in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, pp. 2251–2258, Singapore, September, 2007.

[5] A. Qin and P. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proceedings of the IEEE congress on evolutionary computation*, pp. 1785–1791, Edinburgh, UK, September, 2005.

[6] M. G. H. Omran and A. P. Engelbrecht, "Free search differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 110–117, Trondheim, Norway, May, 2009.

[7] H. V. Hultmann Ayala, L. D. S. Coelho, V. C. Mariani, and A. Askarzadeh, "An improved free search differential evolution algorithm: An improved free search differential evolution algorithm: A case study on parameters identification of one diode equivalent circuit of a solar cell module case study on parameters identification of one diode equivalent circuit of a solar cell module," *Energy*, vol. 93, pp. 1515–1522, 2015.

[8] C. Jena, P. Sinha, L. Nanda, A. Pradhan, B. S. Panda, and L. Nanda, "Optimal scheduling with opposition based differential evolution optimized fixed head hydro-thermal power system," *Materials Today Proceedings*, vol. 58, pp. 227–232, 2022.

[9] M. F. Ahmad, N. A. M. Isa, W. H. Lim, and K. M. Ang, "Differential evolution with modified initialization scheme using chaotic oppositional based learning strategy," *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 11835–11858, 2022.

[10] L. D. S. Coelho, R. C. T. Souza, and V. C. Mariani, "Improved differential evolution approach based on cultural algorithm and diversity measure applied to solve economic load dispatch problems," *Mathematics and Computers in Simulation*, vol. 79, no. 10, pp. 3136–3147, 2009.

[11] H. Abbass, "The self-adaptive Pareto differential evolution algorithm," in *Proceedings of the IEEE congress on evolutionary computation*, pp. 831–836, Honolulu, HI, USA, May, 2002.

[12] H. K. Kim, J. K. Chong, K. Y. Park, D. A. K. Lowther, J. K. Chong, and K. Y. Park, "Differential evolution strategy for constrained gLobal optimization and application to practical engineering problems," *IEEE Transactions on Magnetics*, vol. 43, no. 4, pp. 1565–1568, 2007.

[13] M. M. Ali and A. Torn, "Population set-based global optimization algorithms: some modifications and numerical studies," *Computers and Operations Research*, vol. 31, no. 10, pp. 1703–1725, 2004.

[14] T. J. Choi, C. W. J. C. Ahn, and C. W. Ahn, "Adaptive α-stable differential evolution in numerical optimization," *Natural Computing*, vol. 16, no. 4, pp. 637–657, 2017.

[15] E. H. D. Vasconcelos Segundo, A. L. Amoroso, V. C. Mariani, and L. D. S. C. Coelho, "Economic optimization design for shell-and-tube heat exchangers by a Tsallis differential evolution," *Applied Thermal Engineering*, vol. 111, pp. 143–151, 2017.

[16] C. Sasantia, A. Siritaratiwat, C. Surawanitkun, P. Khunkitti, and R. Chatthaworn, "Optimal planning of energy storage system using modified differential evolution algorithm," *Energy Procedia*, vol. 156, pp. 192–200, 2019.

[17] L. D. S. Coelho, V. C. Mariani, J. V. Leite, V. C. Mariani, and J. V. Leite, "Solution of Jiles–Atherton vector hysteresis parameters estimation by modified differential evolution approaches," *Expert Systems with Applications*, vol. 39, no. 2, pp. 2021–2025, 2012.

[18] M. Omran, A. Salman, and A. Engelbrecht, "Self-adaptive differential evolution," *Lecture notes in artificial intelligence*, vol. 3801, pp. 192–199, 2005.

[19] Z. Zeng, M. Zhang, T. Chen, Z. Z. Hong, and T. Chen, "A new selection operator for differential evolution algorithm," *Knowledge-Based Systems*, vol. 226, Article ID 107150, 2021.

[20] V. C. Mariani, L. G. Justi Luvizotto, F. A. Guerra, L. dos Santos Coelho, L. G. Justi Luvizotto, and F. Alessandro Guerra, "A hybrid shuffled complex evolution approach based on differential evolution for unconstrained optimization," *Applied Mathematics and Computation*, vol. 217, no. 12, pp. 5822–5829, 2011.

[21] Y. Diouane, S. Gratton, and L. N. Vicente, "Globally convergent evolution strategies," *Mathematical Programming*, vol. 152, no. 1-2, pp. 467–490, 2015.

[22] S. Ghosh, S. Das, A. V. Vasilakos, K. D. Suresh, and A. V. Vasilakos, "On convergence of Differential Evolution On Convergence of Differential Evolution Over a Class of Continuous Functions With Unique Global Optimumver a class of Continuous Functions with unique global optimum," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 1, pp. 107–124, 2012.

[23] V. Feoktstov, *Differential Evolution: In Search of Solutions*, Springer, Berlin, Germany, 2006.

[24] X. Zhang and S. Liu, "Almost-parameter-free Differntial evolution," in *Proceedings of the The Seventh Internation Conference on Nutural Computation*, pp. 1461–1465, Shanghai, China, July, 2011.

[25] K. L. Tsui and L. Tsui, "An overview of taguchi method and newly developed statistical methods for robust design," *IIE Transactions*, vol. 24, no. 5, pp. 44–57, 1992.

[26] P. D. Berger, R. E. Maurer, and G. B. Celli, *Experimental Design*, Springer, Berlin, Germany, 2018.

[27] S. I. Birbil and S. C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.

[28] S. I. Birbil, S. C. Fang, R L. C. Sheu, and R. L. Sheu, "On the Convergence of a Population-Based Global Optimization Algorithm," *Journal of Global Optimization*, vol. 30, no. 2-3, pp. 301–318, 2004.

[29] D. Debels, B. De Reyck, R. Leus, M. Vanhoucke, R. De, and R. Leus, "A hybrid scatter search/electromagnetism metaheuristic for project scheduling," *European Journal of Operational Research*, vol. 169, no. 2, pp. 638–653, 2006.

[30] R. K. Yarlagadda and J. E. Hershey, *Hadamard Matrix Analysis and Synthesis*, Kluwer, Boston, MA, USA, 1997.

[31] J. T. Tsai, T. K. Liu, J. H. Chou, L. Tung-Kuan, and C. Jyh-Horng, "Hybrid taguchi-genetic algorithm for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 4, pp. 365–377, 2004.

[32] Y. W. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.