Information Sciences 508 (2020) 50–63

Contents lists available at ScienceDirect

Information Sciences

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

# Surrogate-assisted classification-collaboration differential evolution for expensive constrained optimization problems

Zan Yang, Haobo Qiu, Liang Gao, Xiwen Cai, Chen Jiang, Liming Chen

*State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China*

ARTICLE INFO

*Article history:*
Received 21 April 2019
Revised 19 August 2019
Accepted 24 August 2019
Available online 26 August 2019

*Keywords:*
Expensive constrained optimization problems
Differential evolution
Global search
Classification-collaboration
Surrogate-assisted evolutionary algorithms

ABSTRACT

Expensive Constrained Optimization Problems (ECOPs) widely exist in various scientific and industrial applications. Surrogate-Assisted Evolutionary Algorithms (SAEAs) have recently exhibited great ability in solving these expensive optimization problems. This paper proposes a Surrogate-Assisted Classification-Collaboration Differential Evolution (SACCDE) algorithm for ECOPs with inequality constraints. In SACCDE, the current population is classified into two subpopulations based on certain feasibility rules, and a classification-collaboration mutation operation is designed to generate multiple promising mutant solutions by not only using promising information in good solutions but also fully exploiting potential information hidden in bad solutions. Afterwards, the surrogate is utilized to identify the most promising offspring solution for accelerating the convergence speed. Furthermore, considering that the population diversity may decrease due to the excessive incorporation of greedy information brought by the classified solutions, a global search framework that can adaptively adjust the classification-collaboration mutation operation based on the iterative information is introduced for achieving an effective global search. Therefore, the proposed algorithm can strike a well balance between local and global search. The experimental results of SACCDE and other state-of-the-art algorithms demonstrate that the performance of SACCDE is highly competitive.



## 1. Introduction

Many optimization problems in science and engineering are subject to different types of constraints. The presence of constraints divides the search space into feasible and infeasible regions, which further complicates the search process. General Constrained Optimization Problems (COPs) with inequality constraints can be formulated as follows [42]:

$$\begin{aligned} &\text{minimize } f(\vec{x}), \qquad \vec{x} \in S \\ &\text{subject to } g_j(\vec{x}) \le 0, \ j = 1, ..., q \end{aligned} \tag{1}$$

where $f(\vec{x})$ is the objective function. $\vec{x} = (x_1, x_2, ..., x_n)$ denotes an $n$-dimensional vector of decision variables. These variables are defined by the lower and upper bounds $L = (l_1, l_2, ..., l_n)$, $U = (u_1, u_2, ..., u_n)$, respectively. $S$ is the search space or design space. $q$ is the number of inequality constraints. $g_j(\vec{x})$ is the $j$th inequality constraint. Any solution $\vec{x}$ which satisfies all

*E-mail addresses:* priase@hust.edu.cn (Z. Yang), hobbyqiu@mail.hust.edu.cn (H. Qiu), gaoliang@mail.hust.edu.cn (L. Gao), 864533724@qq.com (X. Cai), chenjiang@hust.edu.cn (C. Jiang), liming_chen@hust.edu.cn (L. Chen).

https://doi.org/10.1016/j.ins.2019.08.054
0020-0255/

the constraints $g_j(\vec{x})$ is called a feasible solution; otherwise, $\vec{x}$ is called an infeasible solution. Moreover, the inequality constraints that satisfy $g_j(\vec{x}) = 0$ at the global optimal solution are called active constraints. Generally, the degree of constraint violation of $\vec{x}$ on the $j$th constraint can be formulated as follows:

$$G_j(\vec{x}) = \max\left\{g_j(\vec{x}), 0\right\}, j = 1, ..., q \tag{2}$$

Then, the overall constraint violation function for an infeasible solution $\vec{x}$ is expressed as

$$G(\vec{x}) = \sum_{j=1}^{q} G_j(\vec{x}) \tag{3}$$

Therefore, the objective is to minimize the fitness function $f(\vec{x})$ while the obtained optimal solution satisfies the overall constraint violation function ($G(\vec{x}) = 0$).

Evolutionary Algorithms (EAs), such as Differential Evolution (DE) [36], Evolutionary Programming (EP) [18], Particle Swarm Optimization (PSO) [8,14,27], Genetic Algorithm (GA) [29] and Evolution Strategy (ES) [2] have been widely used to solve different types of optimization problems [1,4,7]. However, they are originally unconstrained search methods and lack an explicit mechanism to bias the search towards the constrained search space [22]. As a result, appropriate constraint handling methods should be proposed to assist EAs to solve COPs.

Penalty function methods [6,10,39] are the most common constraint handling strategies for solving COPs. In these methods, each infeasible solution has a fitness function that includes the value of the overall constraint violation so that it is less likely to survive into the next generation than any feasible solution. Static penalty function methods have been applied to many EAs to solve COPs due to their simplicity, but they usually require the fine tuning of penalty factors. Moreover, the penalty factors are usually problem-dependent parameters [33], thus the generalization ability of these methods is relatively weak. In order to overcome these limitations, the adaptive penalty function methods [10,44], in which the information gathered from the search space is used to tune the penalty factors, have been proposed. At the meantime, methods based on Superiority of Feasible solutions (SF) also do not require users to define parameter values explicitly [40]. In addition, another constraint handling method, which is called Stochastic Ranking (SR), was introduced by Runarsson and Yao [33] to achieve a balance between the objective and the overall constraint violation. Furthermore, multi-objective optimization methods have also been employed to handle constraints [40–41,43,50] in a more comprehensive way.

As we all know, no matter what kind of constraint handling method is adopted by an EA to deal with COPs, a large number of function evaluations are needed to obtain an acceptable optimal solution. When the values of the objective and constraint functions at a given input are only available after running expensive simulations (for example, one simulation of a typical computational electromagnetics may take 20 min [46]), the computational cost of applying EAs to solve such kind of COPs (also named as Expensive Constrained Optimization Problems, ECOPs) is unaffordable. Therefore, surrogates (also known as meta-models and approximation models), which are computationally cheap models, are used to assist the EAs to obtain a satisfactory solution under an acceptable computational budget when solving ECOPs. Over recent years, the most commonly used surrogate models include Artificial Neural Networks (ANNs) [15,35], Support Vector Machines (SVMs) [12], Radial Basis Function (RBF) [26,31,32,49], and Gaussian Process (GP, also referred to as Kriging) [20]. The surrogates can be used to identify promising offspring solutions and the expensive simulations are implemented only on these promising solutions during the iterative process of EA [31], thus the computational cost can be greatly reduced. These Surrogate-Assisted Evolutionary Algorithms (SAEAs) have been used by some researchers to deal with ECOPs. Runarsson [34] used two nearest-neighbor regression models including penalty functions to assist the stochastic ranking evolution strategy for solving general nonlinear programming problems. In [35], Shi and Rasheed proposed an adaptive fitness approximation GA (ASAGA) which can adaptively adjust the model complexity and the frequency of model usage. Miranda-Varela and Mezura-Montes [24] used a penalty-based differential evolution and a nearest neighbor regression model to solve constrained optimization problems. Ong et al. [26] proposed a parallel evolutionary optimization algorithm, in which the RBF surrogates were utilized to approximate the objective and constraint functions. Regis [31] proposed a surrogate-assisted evolutionary programming algorithm in which the RBF surrogates of objective and constraint functions were utilized to pre-screen the offspring solutions. Wang et al. [46] proposed a novel global and local surrogate-assisted differential evolution algorithm, in which the global GRNNs were used to pre-screen the offspring solutions generated by two DE mutation operations and the local RBFs were utilized to further refine each solution. Jiao et al. [16] proposed a new Expected Improvement (EI) criterion for constrained optimization that can work properly even when no feasible solution is available in the current population. A very recent empirical study on surrogate-assisted evolutionary algorithms dealing with ECOPs can be seen in Miranda-Varela and Mezura-Montes [25].

As we can see, the existing SAEAs for ECOPs do not explicitly consider the constraint satisfaction differences among current solutions when the algorithm mutates. However, in the process of mutation, the appropriate usage of such difference information can lead the algorithm to search along the promising directions where the constraint violations are relieved, and thus the convergence speed of optimization can be accelerated. Moreover, the iterative information of SAEAs that can indicate the current search status of the algorithm is also not fully utilized. Therefore, how to generate promising offspring solutions by identifying and using the constraint differences among current solutions and how to utilize the iterative information of the algorithm to adaptively adjust the mutation operation have not been well studied for these SAEAs. The main motivation of this paper comes from the above considerations, which is to use the specific mutation operation of

the algorithm to achieve an efficient classification-collaboration between different solution subgroups and further adjust the mutation operation based on the iterative information to achieve different search goals. In this paper, a surrogate-assisted classification-collaboration differential evolution named SACCDE is proposed to solve ECOPs with inequality constraints.

The main contributions of this paper are summarized as follows.

(1) The current population is classified into two subpopulations based on certain feasibility rules that is used to identify the differences of constraint satisfaction among solutions. Then, the collaboration between these two classified subpopulations is achieved by a classification-collaboration mutation operation. It is worth noting that the classification-collaboration mutation operation can not only guide the mutant vectors towards the promising regions, but also prevent the mutant vectors from moving towards the unpromising regions. However, when confronted with a problem that is multimodal in the feasible region or has disjoint feasible region, this mutation operation is likely to lead the search into a local optimum. Therefore, a global search framework is further introduced that can use the iterative information of the algorithm to adaptively adjust this mutation operation promptly when the algorithm has the tendency to fall into local optimum.
(2) The selection operation used in this paper is different from that used in the classical differential evolution. The parent and offspring populations are mixed and sorted by following the feasibility rules explained in Section 3.2, and the next parent population can be formed by selecting in the top half of the mixed population. This strategy will be more effective in guiding solutions towards the feasible region when there is no or a few feasible solutions in the population.
(3) Multiple offspring solutions are generated after mutation and crossover. The RBF surrogate constructed by using all the samples in database can pre-screen these offspring solutions effectively so as to obtain the most potential offspring solution. Afterwards, the real objective and constraint functions are only called on this selected most potential offspring solution. This greatly reduces the number of expensive function evaluations, and thus accelerates the convergence speed of the proposed method.
(4) Experiments on benchmark problems such as IEEE CEC2006 [19], IEEE CEC2010 [21], and IEEE CEC 2017 [47] have been conducted to compare the proposed SACCDE with several state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 briefly introduces the related methods including Differential Evolution and RBF surrogate. In Section 3, the proposed SACCDE is put forward in detail. The experimental studies are presented and discussed in Section 4. Finally, Section 5 concludes this paper.

## 2. Background methods and related studies

### 2.1. Differential Evolution (DE)

DE, proposed by Storn and Price [36], has exhibited remarkable performance for many complex optimization problems in diverse fields. In addition, DE is also a population-based optimization algorithm [48]. In DE, there are three main operations, i.e., mutation, crossover and selection. The initial population is generated randomly with uniform distribution. Then the mutation and crossover operations are applied to generate the trial vectors. Finally, the selection operation chooses the better one between target and trial vectors.

Suppose that the population of DE consists of $N$ solutions with $n-dimensional$ parameter vectors

$$\mathrm{x}_{i,g} = \left(\mathrm{x}_{i,1,g}, \ldots, \mathrm{x}_{i,n,g}\right) \tag{4}$$

The mutant vector $v_{i,g} = (v_{i,1,g}, \ldots, v_{i,n,g})$ is generated by a mutation operation for each target vector $\mathrm{x}_{i,g}$ in the current population. The commonly used mutation operations are listed as follows:

(1) DE/rand/1

$$v_{i,g} = \mathrm{x}_{r1,g} + F \cdot \left(\mathrm{x}_{r2,g} - \mathrm{x}_{r3,g}\right) \tag{5}$$

(2) DE/best/1

$$v_{i,g} = \mathrm{x}_{best,g} + F \cdot \left(\mathrm{x}_{r1,g} - \mathrm{x}_{r2,g}\right) \tag{6}$$

(3) DE/current-to-best/1

$$v_{i,g} = \mathrm{x}_{i,g} + F \cdot \left(\mathrm{x}_{best,g} - \mathrm{x}_{i,g}\right) + F \cdot \left(\mathrm{x}_{r1,g} - \mathrm{x}_{r2,g}\right) \tag{7}$$

(4) DE/best/2

$$v_{i,g} = \mathrm{x}_{best,g} + F \cdot \left(\mathrm{x}_{r1,g} - \mathrm{x}_{r2,g}\right) + F \cdot \left(\mathrm{x}_{r3,g} - \mathrm{x}_{r4,g}\right) \tag{8}$$

In the above equations, $r1$, $r2$, $r3$ and $r4$ are distinct integers randomly selected from range [1, $N$] and different from $i$, $F$ is a scaling factor. Although the above four mutation operations have different effects on the convergence rate of DE, the principles of generating the mutant vectors are similar. Here, we choose DE/rand/1 as an example to elaborate the process of mutation operation. Suppose the range of $F$ is [0.5, 1]. Fig. 1 illustrates how to generate the mutant vector $v_{i,g}$ in a two-dimensional parameter space. Firstly, when the base vector $\mathrm{x}_{r0,g}$ and difference vector $\mathrm{x}_{r1,g} - \mathrm{x}_{r2,g}$ are fixed, the position of

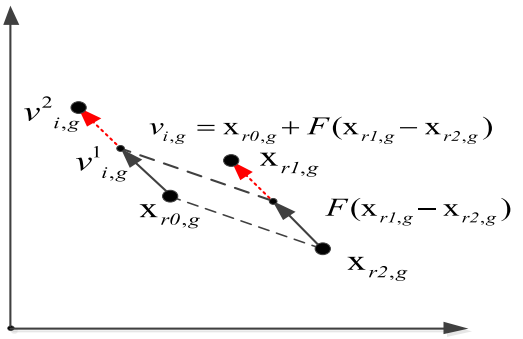


**Fig. 1.** The schematic diagram to illustrate the mutation operation.

the mutant vector $v_{i,g}$ varies with the change of the value of $F$ (in Fig. 1, the position of the mutant vector $v_{i,g}$ can be in any position on the red dotted line between $v^1{}_{i,g}$ and $v^2{}_{i,g}$), but the mutation direction of the target vector $x_{i,g}$ is always the same. In other words, the mutation direction of the target vector $x_{i,g}$ is determined by the base vector $x_{r0,g}$ and difference vector $x_{r1,g} - x_{r2,g}$. Therefore, if we adaptively select the base vector and the difference vector of the mutation operation according to the information carried by all the solutions in the current population, we can guide the algorithm to keep on searching in the promising direction during the whole iteration process. The proposed SACCDE can achieve this adaptive selection of specific solutions by population classification and collaboration.

After the mutation operation, the crossover operation is applied to the target vector $x_{i,g}$ and mutant vector $v_{i,g}$ to generate the trial vector $u_{i,g} = (u_{i,1,g}, ..., u_{i,n,g})$. And the formula of binomial crossover is expressed as

$$u_{i,j,g} = \begin{cases} v_{i,j,g} & \text{if } \text{rand}_j \le CR \text{ or } j = j_{\text{rand}} \\ x_{i,j,g} & \text{otherwise} \end{cases}, \ j = 1, ..., n \tag{9}$$

where $j_{\text{rand}}$ is a randomly chosen integer from {1, ..., $n$}, $\text{rand}_j$ is a random number on the interval [0, 1], and $CR \in [0, 1]$ is the crossover control parameter.

Finally, the selection operation is implemented to select the better one between $x_{i,g}$ and $u_{i,g}$ for the next generation.

### 2.2. Radial basis function (RBF)

RBF stems from [28]. It is a kind of interpolation model and uses a weighted sum of simple basis functions in an attempt to emulate complicated design landscape.

Given $m$ distinct points $x_1, ..., x_m \in R^n$ and their function values $f(x_1), ..., f(x_m)$, the RBF approximation can be expressed as

$$\hat{f}(x) = w^T \varphi = \sum_{i=1}^{m} \omega_i \phi(\|x - x_i\|) \tag{10}$$

where $\|\cdot\|$ is the Euclidean norm, $\omega_i \in R$ is the $i$th weight, $x_i$ denotes the $i$th of $m$ basis function centers, and $\varphi$ is an $m \times 1$ vector containing the values of the basis function $\phi(\cdot)$ which are evaluated at the Euclidean distances between the prediction site x and the centers $x_i$. The cubic form $\phi(r) = r^3$ is used in this paper because it was successfully used in various surrogate-assisted optimization algorithms [31,46]. And the unknown weight vector $w = (\omega_1, ..., \omega_m)^T$ can be calculated as

$$w = \left(\Phi^T \Phi\right)^{-1} \Phi^T y \tag{11}$$

where $y = (f(x_1), ..., f(x_m))^T$, $\Phi$ denotes the Gram matrix which is defined as

$$\Phi = \begin{bmatrix} \phi(\|x_1 - x_1\|) & \cdots & \phi(\|x_1 - x_m\|) \\ \vdots & \ddots & \vdots \\ \phi(\|x_m - x_1\|) & \cdots & \phi(\|x_m - x_m\|) \end{bmatrix} \tag{12}$$

## 3. The proposed algorithm

### 3.1. SACCDE method

As mentioned in Section 1, SACCDE is a method based on the idea of classification-collaboration mutation. More specifically, the population is classified into two subpopulations, and the different solutions selected from these subpopulations

are fully utilized as a group to collaboratively generate many promising mutant vectors for each target vector in the current population. Then, the surrogate is used to select the most promising solution for each target vector so as to locate the promising region. Afterwards, the global search framework can use the iterative information to adaptively adjust the classification-collaboration mutation operation in order to strengthen the global search ability. Therefore, SACCDE attempts to strike a well balance between the local and global search.

**Algorithm 1** SACCDE.

1. Initialize the population size $NP$ and the population $P_g = \{x_{1,g}, x_{2,g}, ..., x_{NP,g}\}$ by LHS. // $g = 1$.
2. Evaluate the values of exact objective function $f(x_{i,g})$ and constraints $g_j(x_{i,g})$ of each solution $x_{i,g}$. // $i = 1, 2, ..., NP$ and $j = 1, ..., Ncon$. $Ncon$ denotes the number of constraints.
3. Archive all solutions into database $B = \{(x_{i,g}, f(x_{i,g}), g_1(x_{i,g}), ..., g_{Ncon}(x_{i,g}))\}$ // $i = 1, 2, ..., NP$.
4. Determine the current best solution $x_{best,g}$ of the population.
5. Initialize the value of $T\_fail$: $T\_fail = 0$. // $T\_fail$ denotes the number of times that the best solution is updated unsuccessfully during the successive iterations.
6. $FEs = NP$. // $FEs$ denotes the number of fitness evaluations.
7. **While** $FEs < MaxFEs$
8. **For** each solution $x_{i,g}$ in $P_g$
9. Construct RBFs by using all the samples in database $B$ to approximate the exact objective and constraint functions respectively.
10. Generate $\beta$ offspring solutions $Subpop_{i,g} = \{x'_{1,g}, ..., x'_{\beta,g}\}$ via the multiple offspring solutions generation operation as elaborated in Section 3.2.
11. Select the most promising solution $xp_{i,g}$ from the $\beta$ offspring solutions via the surrogate assisted pre-screening operation as explained in Section 3.3.
12. Evaluate the values of the exact objective function $f(xp_{i,g})$ and constraints $g_j(xp_{i,g})$ of the selected most promising solution $xp_{i,g}$.
13. $FEs = FEs + 1$.
14. **If** $xp_{i,g}$ is better than $x_{best,g}$ based on the feasibility rules as elaborated in Section 3.3.
15. $x_{best,g} = xp_{i,g}$ and $T\_fail = 0$
16. **Else**
17. $x_{best,g} = x_{best,g-1}$ and $T\_fail = T\_fail + 1$
18. **End If**
19 Update the database: $B = B \cup \{xp_{i,g}, f(xp_{i,g}), g_1(xp_{i,g}), ..., g_{Ncon}(xp_{i,g})\}$
20. **End For**
21. Select the next parent population $P_{g+1}$ from the merged population $Totalpop_g = P_g \cup \{xp_{1,g}, xp_{2,g}, ..., xp_{NP,g}\}$ by the modified selection operation as explained in Section 3.4.
22. $g = g + 1$.
23. **End While**

Algorithm 1 shows the outline of the proposed SACCDE. The initial population $P_g$ is generated by using the Latin hypercube sampling (LHS) [11] that can sample uniformly in the design space. All of the initial solutions are evaluated by using the exact objective and constraint functions and archived into database $B$. Other parameters such as $T\_fail$, $FEs$ and so on are also initialized. The algorithm then loops through generations until the termination criterion is satisfied. At each generation, firstly, RBF model is constructed by using all the samples in database $B$ as the global surrogate to approximate the exact objective and constraint functions respectively. Secondly, $\beta$ offspring solutions are generated for each parent solution $x_{i,g}$ in $P_g$ through the multiple offspring solutions generation operation as explained in Section 3.2. Then the values of the objective and constraint functions of the $\beta$ offspring solutions are approximated by RBF surrogates, and the most promising solution $xp_{i,g}$ is selected based on the feasibility rules as explained in Section 3.3. Thirdly, the selected most promising solution $xp_{i,g}$ is evaluated by using the exact objective and constraint functions and compared with the current best solution $x_{best,g}$ based on the feasibility rules to decide whether or not to replace $x_{best,g}$. Then, the information of $xp_{i,g}$ is stored into $B$ and the value of $T\_fail$ is updated. Finally, the modified selection operation as elaborated in Section 3.4 is used to obtain the next parent population $P_{g+1}$.

### 3.2. Multiple offspring solutions generation operation

Multiple offspring solutions generation operation is utilized to produce multiple solutions as promising as possible for each parent in the current population so that the RBF surrogate can select the most promising solution by pre-screening these solutions. Different from other methods, the proposed SACCDE uses population classification and collaboration to achieve this goal. The outline of the multiple offspring solutions generation operation is shown in Algorithm 2. The four important parts of it are population classification (line 1), classification-collaboration mutation operation (lines 6–8), global search framework (lines 5–15), and crossover operation (lines 16–17). The first three parts are elaborated in detail as follows.

#### 3.2.1. Population classification

In SACCDE, the current population is classified into two subpopulations (If the population size is even, the two subpopulations have the same size. If the population size is odd, the size of one of the subpopulations is one more than that of the other) based on the feasibility rules. One subpopulation named as $G\_subpop_g$ contains the top half of the current population, and the other named as $B\_subpop_g$ contains the rest ones. More specifically, the position information carried by the solutions in $G\_subpop_g$ represents almost all of the promising regions where the current population is located, and

thus the corresponding mutation operation should appropriately select solutions from this subpopulation to guide the target vectors $x_{i,g}$ to mutate towards these promising regions. On the contrary, the position information carried by the solutions in $B\_subpop_g$ represents almost all of the unpromising regions, and thus the corresponding mutation operation should avoid the target vectors $x_{i,g}$ to mutate towards these unpromising regions. In addition, the feasibility rules used for classification are elaborated as follows.

(1) Between two feasible solutions, the one with better objective value wins.
(2) Any feasible solution is preferred to any infeasible solution.
(3) Between two infeasible solutions, the one with the fewer number of constraint violations wins.
(4) Between two infeasible solutions with the same number of constraints violations, the one with the lower degree of constraint violation wins.

To some extent, the feasibility rules can eliminate the misjudgment on the superiority of two infeasible solutions due to the inconsistency of the magnitudes of constraint violation for different constraint functions.

*3.2.2. Classification-collaboration mutation operation*

As mentioned above, after population classification, it is necessary to use a specific mutation operation to adaptively select solutions from the two subpopulations for mutation. Considering that the DE/best/2 incorporates the information of the current best solution when mutating, we use it as the main mutation operator to increase the speed of producing promising candidate solutions. As shown in lines 6–8 of Algorithm 2, the DE/best/2 not only appropriately uses the information of the good solutions in $G\_subpop_g$ to guide the algorithm to move towards promising regions, but also utilizes the information hidden in the bad solutions in $B\_subpop_g$ to avoid the unpromising regions. To some extent, this reflects the fact that the DE/best/2 mutates through the collaboration between good and bad solutions. Therefore, we refer to this mutation operation as the classification-collaboration mutation operation. In general, the algorithm needs to go through three stages when dealing with ECOPs:

**Algorithm 2** Multiple offspring solutions generation operation.

1. Classify the current population $P_g$ into two subpopulations (including subpopulation $G\_subpop_g$ and subpopulation $B\_subpop_g$) based on the feasibility rules. // The sizes of the two subpopulations are set to $[NP/2]$ and $NP-[NP/2]$ respectively.
2. **For** $j = 1 : \beta$
3. $F_1 = 0.5 \cdot rand + 0.5$ and $F_2 = 0.5 \cdot rand + 0.5$
5. **If** $T\_fail \le N_c$
6. Randomly select two integers $r1$ and $r3$ from $[1, i) \cup (i, [NP/2]]$. // $r1$ and $r3$ are selected from $G\_subpop_g$.
7. Randomly select two integers $r2$ and $r4$ from $[1, i) \cup (i, NP-[NP/2]]$. // $r2$ and $r4$ are selected from $B\_subpop_g$.
8. $v_{j,g} = x_{best,g} + F_1 \cdot (x^G{}_{r1,g} - x^B{}_{r2,g}) + F_2 \cdot (x^G{}_{r3,g} - x^B{}_{r4,g})$. // $x^G{}_{r1,g}$ and $x^G{}_{r3,g}$ are the $r1-th$ and $r3-th$ solutions of $G\_subpop_g$ respectively. $x^B{}_{r2,g}$ and $x^B{}_{r4,g}$ are the $r2-th$ and $r4-th$ solutions of $B\_subpop_g$ respectively.
9. **ElseIf** $N_c < T\_fail \le 2N_c$
10. Randomly select four integers $r1$, $r2$, $r3$, and $r4$ from $[1, i) \cup (i, NP]$. // $r1$, $r2$, $r3$, and $r4$ are selected from $P_g$.
11. $v_{j,g} = x_{best,g} + F_1 \cdot (x^t{}_{r1,g} - x^t{}_{r2,g}) + F_2 \cdot (x^t{}_{r3,g} - x^t{}_{r4,g})$. // $x^t{}_{r1,g}$, $x^t{}_{r2,g}$, $x^t{}_{r3,g}$, and $x^t{}_{r4,g}$ are the corresponding solutions of $P_g$ respectively.
12. **ElseIf** $T\_fail > 2N_c$
13. Randomly select five integers $r1$, $r2$, $r3$, $r4$, and $r5$ from $[1, i) \cup (i, NP]$. // $r1$, $r2$, $r3$, $r4$, and $r5$ are selected from $P_g$.
14. $v_{j,g} = x^t{}_{r1,g} + F_1 \cdot (x^t{}_{r2,g} - x^t{}_{r3,g}) + F_2 \cdot (x^t{}_{r4,g} - x^t{}_{r5,g})$.
15. **End If**
16. $CR = 0.5 \cdot rand + 0.5$.
17. Implement the binomial crossover of classical DE in Eq. (9) on $x_{i,g}$ and $v_{j,g}$ to generate offspring solutions (also can be named as trial vectors) $x'_{j,g}$.
18. **End For**
19. All these $\beta$ offspring solutions are archived into $Subpop_{i,g}$.

(1) Stage One: the current population only contains infeasible solutions.
(2) Stage Two: the current population not only contains feasible solutions, but also contains infeasible solutions.
(3) Stage Three: the current population only contains feasible solutions.

Because the principles of the movement of mutant vectors are similar during these different stages, we only choose stage one as an example to show the tendency of movement of mutant vectors as illustrated in Fig. 2. (Fig. 2(a) and (b) shows the tendency of the movement of the mutant vectors under the formula $w_{j,g} = x_{best,g} + F_1 \cdot (x_{r1,g} - x_{r2,g})$ and $v_{j,g} = w_{j,g} + F_2 \cdot (x_{r3,g} - x_{r4,g})$) respectively. The tendency of movement of mutant vectors at stage two and stage three are shown in Fig. A1 and Fig. A2 respectively in the Appendix.

In Fig. 2(a) and (b), *FR* represents the feasible region, *IFR* represents the infeasible region, $x_{r1,g}$ and $x_{r3,g}$ are two randomly selected solutions from $G\_subpop_g$, $x_{r2,g}$ and $x_{r4,g}$ are two randomly selected solutions from $B\_subpop_g$. The lines between $x_{r1,g}$ and $x_{r2,g}$ consists of a black solid line with an arrowhead and a red dotted line with an arrowhead. The black solid line represents the direction and length of $F_1 \cdot (x_{r1,g} - x_{r2,g})$. And the red dotted line represents all the possible cases that $F_1 \cdot (x_{r1,g} - x_{r2,g})$ can be extended due to the different values of parameter $F_1$. Similarly, The black solid line between $x_{r3,g}$ and $x_{r4,g}$ represents the direction and length of $F_2 \cdot (x_{r3,g} - x_{r4,g})$. And the red dotted line between $x_{r3,g}$ and $x_{r4,g}$ represents

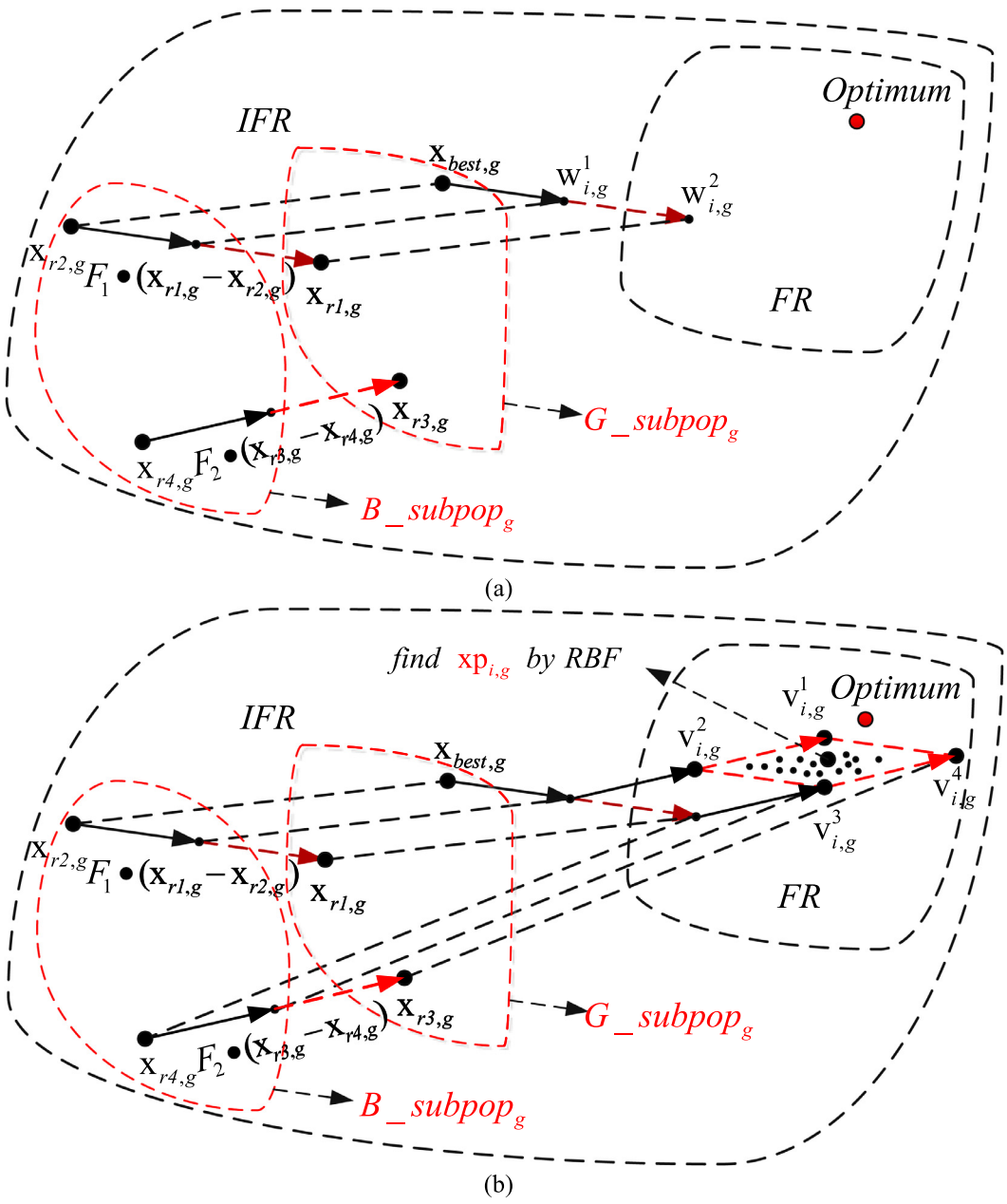


**Fig. 2.** The tendency of the movement of the mutant vectors at stage one.

all the possible cases that $F_2 \cdot (x_{r3,g} - x_{r4,g})$ can be extended due to the different values of parameter $F_2$. Therefore, all possible mutation solutions generated by using the classification-collaboration mutation operation are within the interior region of a parallelogram with vertices $v^1_{i,g}$, $v^2_{i,g}$, $v^3_{i,g}$, and $v^4_{i,g}$.

In Fig. 2(b), there is no feasible solution in the current population at stage one, but the solutions such as $x_{r1,g}$ and $x_{r3,g}$ in the $G\_subpop_g$ are to some extent closer to the feasible region than the solutions such as $x_{r2,g}$ and $x_{r4,g}$ in the $B\_subpop_g$. Therefore, $F_1 \cdot (x_{r1,g} - x_{r2,g})$ and $F_2 \cdot (x_{r3,g} - x_{r4,g})$ represent two directions towards the feasible region respectively, and the purpose of this classification-collaboration mutation operation $v_{j,g} = x_{best,g} + F_1 \cdot (x_{r1,g} - x_{r2,g}) + F_2 \cdot (x_{r3,g} - x_{r4,g})$ is to further guide the mutant vector $v_{j,g}$ to move towards the more promising feasible region from the promising region where the current best solution is located. Similarly, in Fig. A1, the mutant vectors are guided to enter into the feasible region by the efficient collaboration between feasible and infeasible solutions. In Fig. A2, the objective function values at mutant vectors can be further optimized by learning the different location information among feasible solutions.

*3.2.3. Global search framework*

The above classification-collaboration mutation operation is able to achieve fast convergence by effectively incorporating different solution information into the evolutionary search, but it also may guide the mutant vectors to move towards some local regions and make the population diversity limited. Hence, the algorithm may fall into premature convergence due to the insufficiency of global search ability. Therefore, in this subsection, a global search framework (lines 5–15 of Algorithm 2) is introduced to adaptively modify the classification-collaboration mutation operation so as to avoid trapping into a local optimum. In the global search framework, the iterative information $T\_fail$, which is defined as the number of times that the best solution is updated unsuccessfully during the successive iterations, is used to determine whether the current population has a tendency to fall into a local optimum. A threshold value $N_c$ is used to define this tendency. If the value of $T\_fail$ is between $N_c$ and $2N_c$, the classification-collaboration mutation operation is modified to $v_{j,g} = x_{best,g} + F_1 \cdot (x^t_{r1,g} - x^t_{r2,g}) + F_2 \cdot (x^t_{r3,g} - x^t_{r4,g})$ for generating mutant vectors with more diversity. On the other hand, the algorithm may be trapped into local optimum with a great probability if the value of $T\_fail$ is bigger than $2N_c$. This means that the search direction may be moved into a local optimum, and thus the mutation operation is changed to $v_{j,g} = x^t_{r1,g} + F_1 \cdot (x^t_{r2,g} - x^t_{r3,g}) + F_2 \cdot (x^t_{r4,g} - x^t_{r5,g})$ for jumping out of this local optimum.

### 3.3. Surrogate assisted pre-screening operation

For a problem whose fitness evaluations need computationally expensive simulations, a surrogate model may not be able to accurately predict the responses of unknown samples, but it can capture the general trend of the landscape of the problem to some extent. This means that the surrogate models can be used instead of the real functions to judge the quality of the offspring solutions, and thus the number of real function evaluations can be reduced. Therefore, the RBF surrogates are used in this paper to approximate the objective and constraint functions, and then a promising solution $xp_{i,g}$ can be obtained for any target vector $x_{i,g}$ shown in Fig 2. This process is called the surrogate assisted pre-screening operation as shown in Algorithm 3.

**Algorithm 3** Surrogate assisted pre-screening operation.

1. Approximate the values of objective function $\tilde{f}(x'_{k,g})$ and constraints $\tilde{g}_l(x'_{k,g})$ of each offspring solution by using RBF respectively. // $k = 1, 2, ..., \beta$ and $l = 1, ..., Ncon$.
2. Select the most promising solution $xp_{i,g}$ from $Subpop_{i,g}$ according to the predicted values of the objective and constraint functions based on the feasibility rules.

### 3.4. Modified selection operation

The selection operation in classical DE uses a one-to-one survivor selection strategy which selects the better one between $x_{i,g}$ and $u_{i,g}$ for the next generation. However, in this paper, the next parent population is selected as the best $NP$ solutions from $Totalpop_g$, while the $Totalpop_g$ includes the current parent population $P_g$ and all of the $xp_{i,g}$ obtained by surrogate assisted pre-screening. The modification of the selection operation (Algorithm 4) is used to accelerate the speed of entering into the feasible region.

**Algorithm 4** Modified selection operation.

1. Combine all of the pre-screened $xp_{i,g}$ for each target solution $x_{i,g}$ and the current population $P_g$ to form the merged population $Totalpop_g$. // $i = 1, 2, ..., NP$
2. Select the best $NP$ solutions from $Totalpop_g$ based on the feasibility rules to form the next parent population $P_{g+1}$

## 4. Experimental studies

### 4.1. Experimental settings

To investigate the effectiveness of the proposed SACCDE algorithm for solving ECOPs with only inequality constraints, empirical studies are conducted on thirteen widely used benchmark problems collected in CEC2006 [19], five 30-dimensional benchmark problems collected in CEC2010 [21], and seven 30-dimensional benchmark problems collected in CEC2017 [47]. The main characteristics of these test problems are listed in Table 1, Table A1 and Table A2 (Appendix) respectively. In order to record the number of real function evaluations conveniently, we assume that for any input $x \in [L, U] \subseteq R^n$, the values of $f(x), g_1(x), ..., g_{Ncon}(x)$ are obtained by running a simulator (a computer code) at the input x due to the fact that the computations of both real objective and constraint functions are expensive.

The parameters in SACCDE are set as follows: the population size $NP$ is set to 15, the number of trial vectors generated for each target vector is set as $\beta = \min(100 * n, 1000)$, the threshold for capturing iterative information in Algorithm 2 of Section 3.2 is set as $N_c = 5$. In order to make fair comparisons, the parameter settings of other state-of-the-art compared

**Table 1**
The main characteristics of the thirteen benchmark problems from CEC2006.

| Prob. | $n$ | Type of objective function | $\rho$ | $LI$ | $NI$ | $a$ |
|---|---|---|---|---|---|---|
| g01 | 13 | Quadratic | 0.0111% | 9 | 0 | 6 |
| g02 | 20 | Nonlinear | 99.9971% | 0 | 2 | 1 |
| g04 | 5 | Quadratic | 52.1230% | 0 | 6 | 2 |
| g06 | 2 | Cubic | 0.0066% | 0 | 2 | 2 |
| g07 | 10 | Quadratic | 0.0003% | 3 | 5 | 6 |
| g08 | 2 | Nonlinear | 0.8560% | 0 | 2 | 0 |
| g09 | 7 | Polynomial | 0.5121% | 0 | 4 | 2 |
| g10 | 8 | Linear | 0.0010% | 3 | 3 | 6 |
| g12 | 3 | Quadratic | 4.7713% | 0 | 1 | 0 |
| g16 | 5 | Nonlinear | 0.0204% | 4 | 34 | 4 |
| g18 | 9 | Quadratic | 0.0000% | 0 | 13 | 6 |
| g19 | 15 | Nonlinear | 33.4761% | 0 | 5 | 0 |
| g24 | 2 | Linear | 79.6556% | 0 | 2 | 2 |

*Note:* $n$ is the dimension of the test problem, $\rho=|F|/|S|$ is the estimated ratio between the feasible region and the search space, $LI$ is the number of linear inequality constraints, $NI$ is the number of nonlinear inequality constraints, $a$ is the number of active constraints at x.

methods follow their original literatures. The maximum number of fitness evaluations *MaxFEs* is set to 1000 for all the test problems. All experimental results are obtained over 25 independent runs in Matlab R2014a. In addition, two performance metrics are used to compare the efficiency and effectiveness of these compared methods. The first one is the number of real fitness evaluations required for the algorithm to enter into the feasible region (FEs_EF). And the other is the effective rate (ER), which represents the ratio of the number of effective runs to the total runs. A run is considered as effective if it can find at least one feasible solution within the given number of function evaluations (*MaxFEs*). Furthermore, in all the Tables below, "Mean", "Std", "Worst" and "Best" respectively represent the average, the standard deviation, the worst and the best of the minimum function values or minimum function errors obtained only in all effective runs.

### 4.2. Comparison with three state-of-the-art algorithms

The compared algorithms are CMODE [42], DPDE [13], FROFI [45]. These three algorithms are all recently proposed algorithms that do not use surrogate models. More specifically, CMODE is a differential evolution algorithm based on multi-objective optimization, and it has been proven to be capable of producing more competitive results than some other DE-based methods such as $\varepsilon DE$ [37], a variant of SaDE [3], MPDE [38], GDE [17], and MDE [23]. DPDE proposes an information-sharing strategy to exchange search information between different subpopulations. FROFI is a very recent algorithm which incorporates objective function information to the feasibility rules for balancing objective and constraint functions. The source codes of CMODE and FROFI come from their original literatures, and we recode the code of DPDE. In order to make fair comparisons, the initial populations of these three compared algorithms are generated by the same LHS. The experimental results (function values) of all the compared algorithms on the thirteen test problems from CEC2006 are listed in Table 2, and the comparison results on the test problems from CEC2010 and CEC2017 are listed in Table A3 and Table A4 in the Appendix. In addition, These three tables include the results of the t-tests calculated at a significant level of $\alpha = 0.05$, where"$-$", "$+$", and "$\approx$" respectively indicate that the performance of the corresponding algorithm is worse than, better than and similar to that of SACCDE.

Table 2 also records the Cohen's d effect size [5], which can be used to compare the performances of two algorithms. For this metric, the absolute value quantifies the performance difference, and the sign represents the superiority or inferiority of the algorithm. In general, we call a "small" effect if it is between 0.2 and 0.3, a "medium" effect if it is around 0.5, and a "large" effect if it is from 0.8 to infinity. For example, the effect size is $-114.078$ when DPDE is compared with SACCDE in F1. It means that the performance difference between DPDE and SACCDE is large and that the negative value shows DPDE performs much worse.

In Table 2, firstly, it can be seen that the proposed SACCDE is able to obtain a higher or comparative ER than DPDE, CMODE, and FROFI on all the test problems. Furthermore, for eight test problems (i.e., g01, g06, g07, g08, g09, g10, g16, g18) with smaller feasibility ratio, SACCDE is able to move the algorithm towards a feasible region with a 100% ER on all these problems. This means that SACCDE can effectively find at least one feasible solution in dealing with various types of problems. Secondly, compared with these three algorithms, SACCDE can move towards the feasible region with the smallest FEs_EF for nine test problems (i.e., g01, g06, g07, g08, g09, g10, g12, g16, g18). It means that SACCDE is capable of entering into the feasible region faster than the compared algorithms. The reason why the FEs_EFs of SACCDE on other four problems (i.e., g02, g04, g19, g24) are not the smallest ones may be attributed to that there are already several feasible solutions in the initial population when dealing with these problems. Thirdly, it can be seen that SACCDE has achieved significantly better or comparative results than DPDE, CMODE, and FROFI on all the test problems in terms of $t$-test and Cohen's d effect size. In addition, similar comparison results can be seen in other two suites of test problems from CEC2010 and CEC2017. In

**Table 2**
Experimental function values of SACCDE, DPDE, CMODE, and FROFI on the thirteen test problems from CEC2006.

| Prob. | Method | Best | Mean | Worst | Std | FEs_EF | ER | d | test |
|---|---|---|---|---|---|---|---|---|---|
| g01 | SACCDE | −1.50E+01 | −1.50E+01 | −1.50E+01 | 4.55E−14 | 3.10E+01 | 100% | | |
| | DPDE | −3.00E+00 | −2.90E+00 | −2.79E+00 | 1.50E−01 | 4.16E+02 | 8% | −1.14E+02 | – |
| | CMODE | −3.99E+00 | −3.32E+00 | −2.21E+00 | 8.25E−01 | 5.94E+02 | 16% | −2.00E+01 | – |
| | FROFI | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| g02 | SACCDE | −6.28E−01 | −4.66E-01 | −3.31E−01 | 8.32E−02 | 1.50E+01 | 100% | | |
| | DPDE | −2.62E−01 | −2.18E-01 | −1.88E−01 | 1.98E−02 | 1.02E+02 | 100% | −4.11E+00 | – |
| | CMODE | −3.00E−01 | −2.23E-01 | −1.76E−01 | 2.95E−02 | 1.00E+02 | 100% | −3.90E+00 | – |
| | FROFI | −2.39E−01 | −2.19E-01 | −1.96E−01 | 1.31E−02 | 8.00E+01 | 100% | −4.16E+00 | – |
| g04 | SACCDE | −3.07E+04 | −3.07E+04 | −3.07E+04 | 6.76E−07 | 1.50E+01 | 100% | | |
| | DPDE | −3.02E+04 | −2.99E+04 | −2.97E+04 | 1.54E+02 | 1.02E+02 | 100% | −6.85E+00 | – |
| | CMODE | −3.06E+04 | −3.05E+04 | −3.03E+04 | 8.65E+01 | 2.50E+01 | 100% | −2.89E+00 | – |
| | FROFI | −3.03E+04 | −3.01E+04 | −2.99E+04 | 1.01E+02 | 8.00E+01 | 100% | −7.74E+00 | – |
| g06 | SACCDE | −6.96E+03 | −6.96E+03 | −6.96E+03 | 1.02E-05 | 4.24E+01 | 100% | | |
| | DPDE | −6.47E+03 | −3.87E+03 | −2.01E+03 | 1.30E+03 | 7.50E+02 | 56% | −3.36E+00 | – |
| | CMODE | −6.96E+03 | −5.99E+03 | −1.76E+03 | 1.45E+03 | 2.03E+02 | 72% | −9.45E−01 | – |
| | FROFI | −6.09E+03 | −4.72E+03 | −2.08E+03 | 1.09E+03 | 7.32E+02 | 100% | −2.92E+00 | – |
| g07 | SACCDE | 2.43E+01 | 2.43E+01 | 2.43E+01 | 1.18E−04 | 7.02E+01 | 100% | | |
| | DPDE | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| | CMODE | 7.33E+01 | 1.50E+02 | 2.91E+02 | 5.99E+01 | 6.95E+02 | 80% | −2.96E+00 | – |
| | FROFI | 7.67E+02 | 1.00E+03 | 1.24E+03 | 3.35E+02 | 9.46E+02 | 8% | −4.13E+00 | – |
| g08 | SACCDE | −9.58E−02 | −9.58E−02 | −9.58E−02 | 3.89E−10 | 1.62E+01 | 100% | | |
| | DPDE | −9.48E−02 | −5.48E−02 | −1.52E−02 | 2.81E−02 | 1.39E+02 | 100% | −2.07E+00 | – |
| | CMODE | −9.58E−02 | −9.23E−02 | −5.46E−02 | 8.46E−03 | 6.20E+01 | 96% | −5.98E−01 | ≈ |
| | FROFI | −9.54E−02 | −9.15E−02 | −8.16E−02 | 4.11E−03 | 1.03E+02 | 100% | −1.50E+00 | – |
| g09 | SACCDE | 7.09E+02 | 7.45E+02 | 7.92E+02 | 2.52E+01 | 2.90E+01 | 100% | | |
| | DPDE | 8.27E+02 | 3.21E+03 | 9.90E+03 | 2.08E+03 | 1.84E+02 | 100% | −1.68E+00 | – |
| | CMODE | 7.03E+02 | 7.43E+02 | 7.89E+02 | 2.27E+01 | 1.21E+02 | 100% | 9.47E−02 | ≈ |
| | FROFI | 8.28E+02 | 1.23E+03 | 1.73E+03 | 2.20E+02 | 1.95E+02 | 100% | −3.12E+00 | – |
| g10 | SACCDE | 7.05E+03 | 7.05E+03 | 7.05E+03 | 2.43E-01 | 5.32E+01 | 100% | | |
| | DPDE | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| | CMODE | 1.07E+04 | 1.55E+04 | 2.09E+04 | 3.51E+03 | 8.39E+02 | 32% | −3.41E+00 | – |
| | FROFI | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| g12 | SACCDE | −1.00E+00 | −9.99E−01 | −9.88E−01 | 2.56E−03 | 2.05E+01 | 100% | | |
| | DPDE | −1.00E+00 | −9.84E−01 | −9.48E−01 | 1.34E−02 | 1.02E+02 | 100% | −1.57E+00 | – |
| | CMODE | −1.00E+00 | −9.98E−01 | −9.82E−01 | 4.91E−03 | 2.63E+01 | 100% | −2.35E−01 | ≈ |
| | FROFI | −1.00E+00 | −9.90E−01 | −9.76E−01 | 5.81E−03 | 8.25E+01 | 100% | −2.04E+00 | – |
| g16 | SACCDE | −1.91E+00 | −1.91E+00 | −1.91E+00 | 2.51E−09 | 4.09E+01 | 100% | | |
| | DPDE | −1.49E+00 | −1.25E+00 | −8.09E−01 | 1.82E−01 | 6.46E+02 | 68% | −5.09E+00 | – |
| | CMODE | −1.88E+00 | −1.77E+00 | −1.53E+00 | 8.34E−02 | 3.64E+02 | 96% | −2.23E+00 | – |
| | FROFI | −1.55E+00 | −1.29E+00 | −9.78E−01 | 1.55E−01 | 7.18E+02 | 80% | −5.60E+00 | – |
| g18 | SACCDE | −8.66E−01 | −8.47E−01 | −6.75E−01 | 5.88E−02 | 1.54E+02 | 100% | | |
| | DPDE | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| | CMODE | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| | FROFI | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| g19 | SACCDE | 3.27E+01 | 3.49E+01 | 4.18E+01 | 2.86E+00 | 1.50E+01 | 100% | | |
| | DPDE | 9.61E+02 | 1.99E+03 | 3.15E+03 | 6.67E+02 | 1.02E+02 | 100% | −4.15E+00 | – |
| | CMODE | 6.08E+02 | 8.62E+02 | 1.39E+03 | 2.30E+02 | 7.50E+01 | 100% | −5.08E+00 | – |
| | FROFI | 5.08E+02 | 9.88E+02 | 1.79E+03 | 2.90E+02 | 8.00E+01 | 100% | −1.85E+00 | – |
| g24 | SACCDE | −5.51E+00 | −5.51E+00 | −5.51E+00 | 3.63E−06 | 1.50E+01 | 100% | | |
| | DPDE | −5.45E+00 | −5.28E+00 | −5.07E+00 | 9.61E−02 | 1.02E+02 | 100% | −3.29E+00 | – |
| | CMODE | −5.51E+00 | −5.49E+00 | −5.38E+00 | 3.29E−02 | 1.00E+01 | 100% | −7.33E−01 | – |
| | FROFI | −5.47E+00 | −5.38E+00 | −5.25E+00 | 5.13E-02 | 8.00E+01 | 100% | −1.68E+00 | – |

*Note*: 'NaN' means that the corresponding method does not find a feasible solution in 25 independent runs.

a word, based on the above analyses, it can be concluded that compared with these three methods, SACCDE can not only enter the feasible regions more effectively and quickly, but also obtain a solution with a higher accuracy.

### *4.3. Comparison with $(\mu+\mu)$-CEP-RBF and GLoSADE*

In this subsection, two state-of-the-art surrogate-assisted evolutionary algorithms are used to make a comparison with SACCDE. GLoSADE is a novel global and local surrogate-assisted DE for solving ECOPs with inequality constraints. And it is the latest method which has demonstrated excellent performance. In addition, $(\mu+\mu)$-CEP-RBF is a RBF-assisted evolutionary programming algorithm for solving high-dimensional black-box ECOPs. And it has been proven to be more competitive than some other constrained optimization methods such as SRES [33], eSS [9] and ConstrLMSRBF-LHD [30]. The source code of GLoSADE comes from its original literature, and we recode the code of $(\mu+\mu)$-CEP-RBF. The initial populations of these compared algorithms are also generated by the same LHS. Table 3 lists the experimental results of all the compared algo-

**Table 3**
Experimental function errors of SACCDE, GLoSADE, and $(\mu+\mu)$-CEP-RBF on the thirteen test problems from CEC2006.

| Prob. | Method | Best | Mean | Worst | Std | FEs_EF | ER | d | test |
|---|---|---|---|---|---|---|---|---|---|
| g01 | SACCDE | 5.33E−14 | 1.12E−13 | 2.22E−13 | 4.55E−14 | 3.10E+01 | 100% | | |
| | GLoSADE | 1.64E−07 | 5.79E−07 | 1.83E−06 | 4.82E−07 | 1.61E+02 | 100% | −1.70E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | 3.95E−04 | 4.64E−01 | 2.00E+00 | 8.66E−01 | 2.08E+01 | 100% | −7.59E−01 | – |
| g02 | SACCDE | 1.76E−01 | 3.37E−01 | 4.72E−01 | 8.32E−02 | 1.50E+01 | 100% | | |
| | GLoSADE | 4.09E−01 | 4.98E−01 | 5.68E−01 | 4.01E−02 | 8.10E+01 | 100% | −2.46E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | 6.22E−01 | 6.46E−01 | 6.73E−01 | 1.64E−02 | 1.00E+01 | 100% | −5.16E+00 | – |
| g04 | SACCDE | 2.36E−07 | 9.37E−07 | 2.72E−06 | 6.76E−07 | 1.50E+01 | 100% | | |
| | GLoSADE | 7.80E−08 | 5.73E−07 | 2.73E−06 | 8.35E−07 | 8.38E+01 | 100% | 4.80E−01 | ≈ |
| | $(\mu+\mu)$-CEP-RBF | 9.80E+00 | 2.13E+01 | 3.37E+01 | 7.78E+00 | 1.00E+01 | 100% | −3.87E+00 | – |
| g06 | SACCDE | 5.30E−08 | 6.60E−06 | 4.43E−05 | 1.02E−05 | 4.24E+01 | 100% | | |
| | GLoSADE | 2.04E−08 | 3.52E−05 | 4.18E−04 | 9.33E−05 | 2.00E+02 | 100% | −4.31E−01 | – |
| | $(\mu+\mu)$-CEP-RBF | 1.38E+03 | 1.89E+03 | 2.45E+03 | 5.72E+02 | 3.57E+02 | 40% | −4.67E+00 | – |
| g07 | SACCDE | 4.38E−06 | 1.06E−04 | 4.20E−04 | 1.18E−04 | 7.02E+01 | 100% | | |
| | GLoSADE | 1.28E−07 | 9.62E−05 | 1.40E−03 | 3.23E−04 | 1.79E+02 | 100% | 4.03E−02 | ≈ |
| | $(\mu+\mu)$-CEP-RBF | 3.21E−02 | 5.81E−01 | 1.53E+00 | 5.23E−01 | 7.31E+01 | 100% | −1.57E+00 | – |
| g08 | SACCDE | 1.30E−12 | 1.80E−10 | 1.47E−09 | 3.89E−10 | 1.62E+01 | 100% | | |
| | GLoSADE | 8.70E−15 | 1.57E−11 | 2.72E−10 | 5.40E−11 | 1.01E+02 | 100% | 5.92E−01 | ≈ |
| | $(\mu+\mu)$-CEP-RBF | 0.00E+00 | 2.00E−02 | 6.67E−02 | 3.22E−02 | 1.69E+01 | 100% | −8.78E−01 | – |
| g09 | SACCDE | 2.86E+01 | 6.44E+01 | 1.12E+02 | 2.52E+01 | 2.90E+01 | 100% | | |
| | GLoSADE | 9.29E+01 | 2.05E+02 | 3.92E+02 | 7.72E+01 | 9.96E+01 | 100% | −2.44E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | 8.06E+00 | 1.25E+02 | 4.85E+02 | 1.48E+02 | 3.17E+01 | 100% | −5.66E−01 | – |
| g10 | SACCDE | 6.94E−04 | 6.07E−02 | 1.22E+00 | 2.43E−01 | 5.32E+01 | 100% | | |
| | GLoSADE | 8.53E−01 | 7.39E+01 | 4.20E+02 | 9.78E+01 | 1.68E+02 | 100% | −1.07E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | 4.77E+02 | 4.77E+02 | 4.77E+02 | 0.00E+00 | 1.90E+01 | 4% | −2.77E+03 | – |
| g12 | SACCDE | 4.51E−07 | 7.16E−04 | 1.19E−02 | 2.56E−03 | 2.05E+01 | 100% | | |
| | GLoSADE | 3.51E−13 | 9.07E−04 | 5.73E−03 | 2.12E−03 | 9.73E+01 | 100% | −8.15E−02 | ≈ |
| | $(\mu+\mu)$-CEP-RBF | 0.00E+00 | 2.81E−03 | 5.63E−03 | 2.96E−03 | 1.62E+01 | 100% | −7.57E−01 | – |
| g16 | SACCDE | 5.23E−12 | 1.65E−09 | 8.65E−09 | 2.51E−09 | 4.09E+01 | 100% | | |
| | GLoSADE | 8.79E−08 | 3.13E−02 | 1.12E−01 | 3.31E−02 | 1.66E+02 | 100% | −1.34E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | 3.16E−01 | 5.06E−01 | 6.09E−01 | 7.46E−02 | 1.47E+02 | 100% | −9.60E+00 | – |
| g18 | SACCDE | 1.08E−08 | 1.91E−02 | 1.91E−01 | 5.88E−02 | 1.54E+02 | 100% | | |
| | GLoSADE | 4.08E−05 | 9.91E−02 | 2.47E−01 | 9.26E−02 | 3.90E+02 | 100% | −1.03E+00 | – |
| | $(\mu+\mu)$-CEP-RBF | NaN | NaN | NaN | NaN | NaN | 0% | -inf | – |
| g19 | SACCDE | 4.64E−02 | 2.22E+00 | 9.15E+00 | 2.86E+00 | 1.50E+01 | 100% | | |
| | GLoSADE | 6.12E−02 | 5.65E+00 | 2.70E+01 | 6.24E+00 | 8.32E+01 | 100% | −7.07E−01 | – |
| | $(\mu+\mu)$-CEP-RBF | 9.95E+00 | 2.09E+01 | 4.62E+01 | 1.51E+01 | 1.00E+01 | 100% | −5.90E−01 | – |
| g24 | SACCDE | 1.77E−08 | 2.58E−06 | 1.55E−05 | 3.63E−06 | 1.50E+01 | 100% | | |
| | GLoSADE | 2.07E−08 | 2.04E−04 | 2.15E−03 | 5.84E−04 | 8.22E+01 | 100% | −4.86E−01 | – |
| | $(\mu+\mu)$-CEP-RBF | 6.09E−12 | 5.11E−04 | 1.03E−02 | 2.06E−03 | 1.20E+01 | 100% | −3.48E−01 | – |

*Note*: test means the Wilcoxon's rank sum test.

rithms on thirteen test problems from CEC2006, and the comparison results on test problems from CEC2010 and CEC2017 are also listed in Table A5 and Table A6 in the Appendix. In addition, these tables also include the results of the Wilcoxon's rank sum test calculated at a significant level of $\alpha = 0.05$ and the Cohen's d effect size.

In Table 3, it can be seen that both SACCDE and GLoSADE are capable of entering into the feasible region on all the thirteen test problems in terms of ER, which means that both these two algorithms can guide the current solutions to move towards the right search directions when all the solutions of the current population are infeasible. Afterwards, for functions g01, g06, g07, g10, g16, and g18, the values of FEs_EF achieved by SACCDE are much smaller than those achieved by GLoSADE. This may be because the classification-collaboration mutation operation can accelerate the speed of algorithm entering into the feasible region by fully utilizing the valuable information hidden in infeasible solutions. Moreover, the performance achieved by SACCDE is significantly better or comparative than those achieved by GLoSADE on all the test problems in terms of statistical tests. For $(\mu+\mu)$-CEP-RBF, it is worth noting that in the original literature [31], the initial population of $(\mu+\mu)$-CEP-RBF must be given a feasible solution so as to obtain better results. Therefore, $(\mu+\mu)$-CEP-RBF can't effectively guide the current infeasible solutions to move towards feasible regions when dealing with test problems such as g06, g10, and g18 that contains many nonlinear constraints or the feasibility ratio is very low. Furthermore, SACCDE is able to achieve significantly better results than $(\mu+\mu)$-CEP-RBF on all the thirteen test problems in terms of statistical tests.

### *4.4. Effectiveness of some strategies in SACCDE*

In this subsection, five typical test functions from IEEE CEC2006 such as g02, g06, g10, g12, and g18 are chosen to discuss the effectiveness of the two strategies proposed in this paper. The maximum number of fitness evaluations *MaxFEs* is set to 1000, and the rest parameters of SACCDE are set the same as those suggested in Section 4.1. Furthermore, both the *t*-test and Cohen's d effect size are also used to show the difference in performance between different methods.

**Table 4**
Experimental function errors of SACCDE and SACCDE_noCC on the five selected test problems from CEC2006.

| Prob. | Approach | Best | Mean | Worst | Std | FEs_EF | ER | d | *t*-test |
|---|---|---|---|---|---|---|---|---|---|
| g02 | SACCDE | 1.76E−01 | 3.37E−01 | 4.72E−01 | 8.32E−02 | 1.50E+01 | 100% | | |
| | SACCDE_noCC | 3.37E−01 | 4.46E−01 | 5.46E−01 | 6.63E−02 | 1.50E+01 | 100% | −1.45E+00 | – |
| g06 | SACCDE | 5.30E−08 | 6.60E−06 | 4.43E−05 | 1.02E−05 | 4.24E+01 | 100% | | |
| | SACCDE_noCC | 9.97E−05 | 1.84E−03 | 7.04E−03 | 2.07E−03 | 6.79E+01 | 100% | −1.25E+00 | – |
| g10 | SACCDE | 6.94E−04 | 6.07E−02 | 1.22E+00 | 2.43E−01 | 5.32E+01 | 100% | | |
| | SACCDE_noCC | 2.89E−01 | 9.99E−01 | 2.61E+00 | 8.66E−04 | 6.93E+01 | 100% | −5.46E+00 | – |
| g12 | SACCDE | 4.51E−07 | 7.16E−04 | 1.19E−02 | 2.56E−03 | 2.05E+01 | 100% | | |
| | SACCDE_noCC | 1.36E−02 | 1.42E−01 | 3.11E−01 | 1.02E−01 | 3.61E+01 | 100% | −1.96E+00 | – |
| g18 | SACCDE | 1.08E−08 | 1.91E−02 | 1.91E−01 | 5.88E−02 | 1.54E+02 | 100% | | |
| | SACCDE_noCC | 3.91E−04 | 2.08E−02 | 1.94E−01 | 6.04E−02 | 2.25E+02 | 100% | −2.82E−02 | ≈ |

**Table 5**
Experimental function errors of SACCDE and SACCDE_noGS on the five selected test problems from CEC2006.

| Prob. | Approach | Best | Mean | Worst | Std | FEs_EF | ER | d | *t*-test |
|---|---|---|---|---|---|---|---|---|---|
| g02 | SACCDE | 1.76E−01 | 3.37E−01 | 4.72E−01 | 8.32E−02 | 1.50E+01 | 100% | | |
| | SACCDE_noGS | 3.77E−01 | 4.61E−01 | 5.10E−01 | 4.66E−02 | 1.50E+01 | 100% | −1.83E+00 | – |
| g06 | SACCDE | 5.30E−08 | 6.60E−06 | 4.43E−05 | 1.02E−05 | 4.24E+01 | 100% | | |
| | SACCDE_noGS | 3.63E+00 | 2.51E+02 | 1.02E+03 | 2.96E+02 | 2.87E+01 | 100% | −1.20E+00 | – |
| g10 | SACCDE | 6.94E−04 | 6.07E−02 | 1.22E+00 | 2.43E−01 | 5.32E+01 | 100% | | |
| | SACCDE_noGS | 7.94E+00 | 3.88E+01 | 8.38E+01 | 2.74E+01 | 5.25E+01 | 100% | −2.00E+00 | – |
| g12 | SACCDE | 4.51E−07 | 7.16E−04 | 1.19E−02 | 2.56E−03 | 2.05E+01 | 100% | | |
| | SACCDE_noGS | 5.70E−03 | 6.01E−02 | 1.78E−01 | 5.05E−02 | 1.79E+01 | 100% | −1.66E+00 | – |
| g18 | SACCDE | 1.08E−08 | 1.91E−02 | 1.91E−01 | 5.88E−02 | 1.54E+02 | 100% | | |
| | SACCDE_noGS | 4.17E−06 | 3.82E−02 | 1.91E−01 | 8.05E−02 | 1.27E+02 | 100% | −2.71E−01 | ≈ |

#### *4.4.1. Effectiveness of classification-collaboration mutation operation*

In order to verify the effectiveness of the classification-collaboration operation proposed in this paper, a variant of SACCDE called as SACCDE_noCC is introduced for a detailed comparison. The SACCDE_noCC is a method that SACCDE adopts classical DE/best/2 operation elaborated in Eq. (8). The experimental results of both SACCDE and SACCDE_noCC on the five test problems are listed in Table 4.

From Table 4, SACCDE achieves much smaller FEs_EF than SACCDE_noCC on all the test problems except for g02. This means that the classification-collaboration operation is capable of fully digging out the valuable information hidden in the classified infeasible solutions to guide the search into feasible regions more quickly. For g02, the feasibility ratio is approximate to 1, thus the initial population of both SACCDE and SACCDE_noCC already contains feasible solutions, which leads to almost the same FEs_EF. Furthermore, SACCDE exhibits superior or comparative performance than SACCDE_noCC on all the problems in terms of statistical tests. This indicates that the classification-collaboration operation can utilize the information in feasible solutions effectively to acquire more accurate solutions.

#### *4.4.2. Effectiveness of the global search framework*

The global search framework is introduced in this paper to avoid SACCDE trapping into a local optimum. In order to verify the effectiveness of this framework, a variant of SACCDE called as SACCDE_noGS is constructed. The SACCDE_noGS is a method that the global search framework is removed from SACCDE. The experimental results of both SACCDE and SACCDE_noGS on the selected five test problems are listed in Table 5.

As shown in Table 5, it can be seen that SACCDE is able to obtain significantly better results than SACCDE_noGS on all these test problems in terms of accuracy. This may be mainly due to the fact that the global search framework can effectively increase the diversity of the population when the algorithm has a tendency to enter into a local promising region, and thus a more accurate solution can be obtained. Furthermore, the convergence curves for SACCDE and SACCDE_noGS on these test problems are plotted in Fig A3 in the Appendix. As shown in Fig A3, the SACCDE_noGS converges as fast as SACCDE in the early stage of the iterative process, but much slower in the later stages of the evolution. This further validates that the global search framework can achieve an effective global search.

### *4.5. Effect of the parameter settings*

To further investigate the effect of the parameter settings of the proposed SACCDE, three experiments were conducted on five test functions (i.e., g02, g06, g10, g12, g18) from CEC2006. When one of these parameters is analyzed, the rest parameters remain unchanged. The maximum number of fitness evaluations *MaxFEs* is set to 1000. All experimental results are obtained over 25 independent runs.

*4.5.1. Effect of the threshold of iterative information* $N_c$

As one of the most important parameters for the global search framework introduced in SACCDE, the parameter $N_c$ determines when to adjust the mutation operation of the algorithm during the iteration. If $N_c$ is set too large, the classification-collaboration mutation operation cannot be adjusted in time, which may make the algorithm trapped into local optimum with higher possibility. On the contrary, if the value of $N_c$ is set too small, the convergence speed may slow down due to the insufficient use of the classification-collaboration mutation. Therefore, we test the performance of SACCDE with varying $N_c$: 2, 3, 5, 8, 10, 15, 20, 30, 40, 50 in order to give a suggestion on the value of $N_c$. Tables A7–A11 show the performances of SACCDE with varying $N_c$ on these five test problems.

From Tables A7-A11, we can observe that SACCDE is actually sensitive to the parameter $N_c$ to some extent, and that $N_c$ can be chosen from a relatively small range to achieve competitive performance. Generally, the value of $N_c$ is suggested in the interval [5,15].

*4.5.2. Effect of the population size* NP

Too low population size may diminish the number of available moves and prevent convergence within the specified number of function evaluations. However, too big population size may waste function evaluations spent on unnecessary solutions. Therefore, we test the performance of SACCDE with different *NP*: 5, 10, 15, 20, 30, 40, 50, 60, 70, 80 to investigate the effect of *NP*. The experimental results are given in Tables A12-A16. The results reveal that a value between 10 and 30 is a suitable choice for this parameter.

*4.5.3. Effect of the number of trial vectors* $\beta$

$\beta$ is the number of trial vectors generated by mutating from target vectors. Eight different values of $\beta$ such as $\min(10{*}n, 100)$, $\min(30{*}n, 300)$, $\min(50{*}n, 500)$, $\min(80{*}n, 800)$, $\min(100{*}n, 1000)$, $\min(200{*}n, 2000)$, $\min(500{*}n, 5000)$, $\min(1000{*}n, 10000)$ are utilized to investigate the effect of this parameter. Tables A17-A21 show the experiment results.

From Tables A17–A21, we can observe that the performance tends to get worse when using a relatively smaller value for this parameter. In addition, although the performance of SACCDE is not sensitive to this parameter with a relatively larger value, the computational cost brought by larger $\beta$ is higher than that brought by smaller $\beta$. Therefore, the value of $\beta$ is recommended between $\min(50{*}n, 500)$ and $\min(200{*}n, 2000)$.

## 5. Conclusion

This paper introduces a surrogate-assisted classification-collaboration differential evolution algorithm named as SACCDE which is capable of solving expensive constrained optimization problems with inequality constraints. In the proposed method, the classification-collaboration mutation operation can achieve group collaboration between two classified subpopulations. At the same time, the global search framework is proposed to adaptively adjust this mutation operation during the iterative process for avoiding trapped into local optimum. Afterwards, the surrogate is utilized to identify the most promising trial solution, which is evaluated with real objective and constraint functions, making the number of fitness evaluations greatly reduced. In addition, a modified selection operation is introduced to accelerate the speed of entering into the feasible region. Experimental results on some typical optimization benchmark problems demonstrate the efficiency and effectiveness of the proposed SACCDE algorithm when compared with other state-of-the-art algorithms.

Currently, multi-objective or many objective expensive optimization problems are the hot topics in research. Therefore, as a future work, the feasibility of applying the proposed method to solve such complicated problems will be further studied.

## Declaration of Competing Interest

The authors declared that they have no conflicts of interest to this work. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

## Acknowledgement

This research is supported by the National Natural Science Foundation of China under Grant Nos. 51675198, 51721092, the National Natural Science Foundation for Distinguished Young Scholars of China under Grant No. 51825502, and the Program for HUST Academic Frontier Youth Team.

## Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ins.2019.08.054.

## References

[1] D.V. Arnold, Noisy Optimization With Evolution Strategies, Springer Science & Business Media, 2012.
[2] H.-G. Beyer, H.-P. Schwefel, Evolution strategies – a comprehensive introduction, Nat. Comput. 1 (2002) 3–52.

[3] J. Brest, V. Zumer, M.S. Maucec, Self-adaptive differential evolution algorithm in constrained real-parameter optimization, in: 2006 IEEE Int. Conf. Evol. Comput., 2006, pp. 215–222.
[4] C.A.C. Coello, G.B. Lamont, D.A. Van Veldhuizen, et al., Evolutionary Algorithms for Solving Multi-Objective Problems, Springer, 2007.
[5] J. Cohen, Statistical Power Analysis for the Behavioural Sciences, 1988.
[6] M. Daneshyari, G.G. Yen, Constrained multiple-swarm particle swarm optimization within a cultural framework, IEEE Trans. Syst. Man Cybern. A Syst. Hum. 42 (2011) 475–490.
[7] L. dos Santos Coelho, H.V.H. Ayala, V.C. Mariani, A self-adaptive chaotic differential evolution algorithm using gamma distribution for unconstrained global optimization, Appl. Math. Comput. 234 (2014) 452–459.
[8] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory. InMicro machine and human science, in: 1995. mhs '95., Proceedings of the Sixth International Symposium on, 1995.
[9] J.A. Egea, M. Rodríguez-Fernández, J.R. Banga, R. Martí, Scatter search for chemical and bio-process optimization, J. Glob. Optim. 37 (2007) 481–503.
[10] R. Farmani, J.A. Wright, Self-adaptive fitness formulation for constrained optimization, IEEE Trans. Evolut. Comput. 7 (2003) 445–455.
[11] A.I.J. Forrester, A.J. Keane, Recent advances in surrogate-based optimization, Prog. Aerosp. Sci. 45 (2009) 50–79.
[12] G. Gao, C. Sun, J. Zeng, S. Xue, A constraint approximation assisted pso for computationally expensive constrained problems, in: Intell. Control Autom. (WCICA), 2014 11th World Congr., IEEE, 2014, pp. 1354–1359.
[13] W.-F. Gao, G.G. Yen, S.-Y. Liu, A dual-population differential evolution with coevolution for constrained optimization, IEEE Trans. Cybern. 45 (2014) 1108–1121.
[14] N. Ben Guedria, Improved accelerated PSO algorithm for mechanical engineering optimization problems, Appl. Soft Comput. 40 (2016) 455–467.
[15] S.-C. Horng, S.-Y. Lin, Evolutionary algorithm assisted by surrogate model in the framework of ordinal optimization and optimal computing budget allocation, Inf. Sci. 233 (2013) 214–229.
[16] R. Jiao, S. Zeng, C. Li, Y. Jiang, Y. Jin, A complete expected improvement criterion for Gaussian process assisted highly constrained expensive optimization, Inf. Sci. 471 (2019) 80–96.
[17] S. Kukkonen, J. Lampinen, Constrained real-parameter optimization with generalized differential evolution, in: 2006 IEEE Int. Conf. Evol. Comput., 2006, pp. 207–214.
[18] C.-Y. Lee, X. Yao, Evolutionary programming using mutations based on the Lévy probability distribution, IEEE Trans. Evolut. Comput. 8 (2004) 1–13.
[19] J.J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A.C. Coello, K. Deb, Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization, J. Appl. Mech. 41 (2006) 8–31.
[20] B. Liu, Q. Zhang, G. Gielen, A surrogate-model-assisted evolutionary algorithm for computationally expensive design optimization problems with inequality constraints, in: Simulation-Driven Model. Optim., Springer, 2016, pp. 347–370.
[21] R. Mallipeddi, P.N. Suganthan, in: Problem Definitions and Evaluation Criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization, Singapore, Nanyang Technol. Univ., 2010, p. 24.
[22] E. Mezura-Montes, C.A.C. Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Trans. Evolut. Comput. 9 (2005) 1–17.
[23] E. Mezura-Montes, J. Velázquez-Reyes, C.A.C. Coello, Modified differential evolution for constrained optimization, in: 2006 IEEE Int. Conf. Evol. Comput., 2006, pp. 25–32.
[24] M.-E. Miranda-Varela, E. Mezura-Montes, Surrogate-assisted differential evolution with an adaptive evolution control based on feasibility to solve constrained optimization problems, in: Proc. Fifth Int. Conf. Soft Comput. Probl. Solving, Springer, 2016, pp. 809–822.
[25] M.-E. Miranda-Varela, E. Mezura-Montes, Constraint-handling techniques in surrogate-assisted evolutionary optimization. An empirical study, Appl. Soft Comput. 73 (2018) 215–229.
[26] Y.S. Ong, P.B. Nair, A.J. Keane, Evolutionary optimization of computationally expensive problems via surrogate modeling, AIAA J. 41 (2003) 687–696.
[27] R. Poli, J. Kennedy, T. Blackwell, in: Particle Swarm Optimization, 1, Swarm Intell, 2007, pp. 33–57.
[28] M.J.D. Powell, Restart procedures for the conjugate gradient method, Math. Program. 12 (1977) 241–254.
[29] Y. Rahmat-Samii, E. Michielssen, Electromagnetic optimization by genetic algorithms, Microw. J. 42 (1999) 232.
[30] R.G. Regis, Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions, Comput. Oper. Res. 38 (2011) 837–853.
[31] R.G. Regis, Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions, IEEE Trans. Evolut. Comput. 18 (2014) 326–347.
[32] R.G. Regis, Trust regions in surrogate-assisted evolutionary programming for constrained expensive black-box optimization, in: Evol. Constrained Optim., Springer, 2015, pp. 51–94.
[33] T.P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Trans. Evolut. Comput. 4 (2000) 284–294.
[34] T.P. Runarsson, Approximate evolution strategy using stochastic ranking, in: Evol. Comput. 2006. CEC 2006. IEEE Congr., IEEE, 2006, pp. 745–752.
[35] L. Shi, K. Rasheed, ASAGA: an adaptive surrogate-assisted genetic algorithm, in: Proc. 10th Annu. Conf. Genet. Evol. Comput., ACM, 2008, pp. 1049–1056.
[36] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359.
[37] T. Takahama, S. Sakai, Constrained optimization by the $\varepsilon$ constrained differential evolution with an archive and gradient-based mutation, in: Evol. Comput. (CEC), 2010 IEEE Congr., IEEE, 2010, pp. 1–9.
[38] M.F. Tasgetiren, P.N. Suganthan, A multi-populated differential evolution algorithm for solving constrained optimization problem, in: 2006 IEEE Int. Conf. Evol. Comput., 2006, pp. 33–40.
[39] B. Tessema, G.G. Yen, An adaptive penalty formulation for constrained evolutionary optimization, IEEE Trans. Syst. Man Cybern. A Syst. Hum. 39 (2009) 565–578.
[40] S. Venkatraman, G.G. Yen, A generic framework for constrained optimization using genetic algorithms, IEEE Trans. Evolut. Comput. 9 (2005) 424–435.
[41] Y. Wang, Z. Cai, A dynamic hybrid framework for constrained evolutionary optimization, IEEE Trans. Syst. Man Cybern. Part B 42 (2012) 203–217.
[42] Y. Wang, Z. Cai, Combining multiobjective optimization with differential evolution to solve constrained optimization problems, IEEE Trans. Evolut. Comput. 16 (2012) 117–134.
[43] Y. Wang, Z. Cai, G. Guo, Y. Zhou, Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems, IEEE Trans. Syst. Man Cybern. Part B 37 (2007) 560–575.
[44] Y. Wang, Z. Cai, Y. Zhou, W. Zeng, An adaptive tradeoff model for constrained evolutionary optimization, IEEE Trans. Evolut. Comput. 12 (2008) 80–92.
[45] Y. Wang, B.-C. Wang, H.-X. Li, G.G. Yen, Incorporating objective function information into the feasibility rule for constrained evolutionary optimization, IEEE Trans. Cybern. 46 (2016) 2938–2952.
[46] Y. Wang, D.-Q. Yin, S. Yang, G. Sun, Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints, IEEE Trans. Cybern. (2018) 1–15.
[47] G. Wu, R. Mallipeddi, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization, 2017 Natl. Univ. Def. Technol. Chang. Hunan, PR China Kyungpook Natl. Univ. Daegu, South Korea Nanyang Technol. Univ. Singapore, Tech. Rep.
[48] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, P.N. Suganthan, Ensemble of differential evolution variants, Inf. Sci. 423 (2018) 172–186.
[49] H. Yu, Y. Tan, J. Zeng, C. Sun, Y. Jin, Surrogate-assisted hierarchical particle swarm optimization, Inf. Sci. 454 (2018) 59–72.
[50] Y. Zhou, Y. Li, J. He, L. Kang, Multi-objective and MGG evolutionary algorithm for constrained optimization, in: Evol. Comput. 2003, IEEE, 2003, pp. 1–5. CEC'03. 2003 Congr.