Soft Comput (2019) 23:2409–2427
https://doi.org/10.1007/s00500-017-2936-5

METHODOLOGIES AND APPLICATION

An effective improved differential evolution algorithm to solve constrained optimization problems

Xiaobing Yu[1,2,3,4] · **Yiqun Lu**[4] · **Xuming Wang**[5] · **Xiang Luo**[4] · **Mei Cai**[4]

Published online: 25 November 2017

Abstract An effective extended differential evolution algorithm is proposed to deal with constrained optimization problems. The proposed algorithm adopts a new mechanism to cope with constrained problems by transforming the equality into inequality first. Then, two kinds of offspring generation approaches are applied to balance the diversity and the convergence speed of the population during evolution, and seven criteria are designed to compare feasible solution over infeasible solution. The performance of the novel algorithm is evaluated on a set of well-known constrained problems from CEC2006. The experimental results are quite competitive when comparing the proposed algorithm against state-of-the-art optimization algorithms.

Keywords Differential evolution · Evolutionary algorithm · Mutation strategy · Constrained optimization problems

Communicated by V. Loia.

✉ Xiaobing Yu

[1] Research Institute for History of Science and Technology, Nanjing University of Information Science and Technology, Nanjing 210044, China

[2] Collaborative Innovation Center on Forecast and Evaluation of Meteorological Disasters, Nanjing University of Information Science and Technology, Nanjing 210044, China

[3] China Institute for Manufacture Developing, Nanjing University of Information Science and Technology, Nanjing 210044, China

[4] School of Economics and Management, Nanjing University of Information Science and Technology, Nanjing 210044, China

[5] School of Electronic and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China

1 Introduction

Many engineering and scientific researches involve optimization (Yi et al. 2016; Deb 2000). Some of them contain several constraints which the optimal solution must meet. Without a loss of generality, the mathematical formula of constrained optimization problems (COPs) can be expressed as follows (Sun et al. 2016; Yu et al. 2016; Garg 2016; Zhang et al. 2015; Chuang et al. 2016; Asafuddoula et al. 2015):

$$\begin{aligned} &\min \ f(\vec{x}) \\ &\text{s.t.} \begin{cases} g_j(\vec{x}) \le 0, \ j = 1, 2, \ldots, q \\ h_j(\vec{x}) = 0, \ j = q+1, \ldots, m \end{cases}, \end{aligned} \tag{1}$$

where $\vec{x} = (x_1, x_2, \ldots, x_n) \in \Omega$ is the decision vector generated in the decision space S. Ω denotes the feasible region. S is the n-dimensional search space by the boundary constraints, $L_i \le x_i \le U_i, i = 1, \ldots, n$, where U_i and L_i are the upper boundary and the lower boundary of x_i, respectively. $f(\vec{x})$ is the decision function. The constraints $g_j(\vec{x})$ and $h_j(\vec{x})$ denote the inequality and the equality, respectively.

To solve COPs, the equality constraints are generally converted into inequality constraints:

$$\left|h_j(\vec{x})\right| - \delta \le 0, \tag{2}$$

where $j = q+1, \ldots, m$ and δ is the tolerance value for the equality constraints. According to Liang et al. (2006b), $\delta = 0.0001$. The absolute value operator can be removed by transforming the Eq. (2) into inequality constraints:

$$-\delta \le h_j(\vec{x}) \le \delta \rightarrow \begin{cases} h_j(\vec{x}) \le \delta & j = q+1, \ldots, m \\ -h_j(\vec{x}) \le \delta & j = m+1, \ldots, 2m-q \end{cases} \tag{3}$$

So, the constraints of COPs can be presented as follows:

$$\begin{cases} g_j(\vec{x}) \le 0 & j = 1, \ldots, q \\ h_j(\vec{x}) \le \delta & j = q+1, \ldots, m \\ -h_j(\vec{x}) \le \delta & j = m+1, \ldots, 2m-q \end{cases} \tag{4}$$

Then, the total constraint violation (*CV*) can be expressed briefly as follows:

$$\begin{aligned} CV(\vec{x}) &= \sum_{j=1}^{2m-q} cv_j(\vec{x}), cv_j(\vec{x}) \\ &= \begin{cases} \max(g_j(\vec{x}), 0) & j = 1, \ldots, q \\ \max(h_j(\vec{x}) - \delta, 0) & j = q+1, \ldots, m \\ \max(-h_j(\vec{x}) - \delta, 0) & j = m+1, \ldots, 2m-q \end{cases} \end{aligned} \tag{5}$$

There are many different kinds of COPs. The main differences among them are variables and constraint properties. Some of them are integer, real; some of them are discrete (Elsayed et al. 2012). Many optimization algorithms cannot perform well as they are computationally expensive or easily get stuck at local optima (Wang and Cai 2011). On the other hand, many researches have indicated that evolutionary algorithms (EAs) have advantage when solving COPs, especially the genetic algorithm (GA) (Long 2014; Barbosa and Lemonge 2003; Garcia-Martinez et al. 2008), differential evolution (DE) algorithm (Yong et al. 2008; Mezura-Montes and Coello 2004; Price et al. 2005), evolutionary strategies (ES) (Wei et al. 2011; Hansen and Ostermeier 2001), PSO (Kennedy and Eberhart 1995; Liang et al. 2006a; Sun et al. 2011), ant colony-based method (Mahdavi and Shiri 2015; Long et al. 2016), grey wolf (Arora Mehak Kohliand 2017y), and artificial bee colony(ABC) (Liang et al. 2015; Li and Yin 2014; Karaboga and Akay 2011). In the past decades, many techniques have been proposed to solve COPs, such as feasibility rules, stochastic ranking, ε-constrained method, penalty function, multi-objective concept, and ensemble of constraint-handling techniques (Mezura-Montes and Coello 2011; Wang et al. 2016; Takahama and Sakai 2006; Runarsson and Yao 2000; Wang et al. 2007; Tessema and Yen 2006).

Differential evolution is one of the EAs created by Storn and Price (1997). The DE is very simple. The algorithm is efficient and effective.DE has attracted increasing attention. It is originally designed to solve continuous unconstrained problems (Fan and Yan 2016; Qin et al. 2009; Mallipeddi et al. 2011; Brest et al. 2006; Zhang and Sanderson 2009; Yu et al. 2015a, b). Recently, an improved differential evolution by differential vector archive and hybrid repair method is proposed to solve single optimization problem. The proposed algorithm has achieved good performance on IEEE CEC 2013 (Zhang and Zhang 2016). A prior knowledge-guided DE (called PKDE) is implemented, in which the macro-levels and micro-levels of prior knowledge are extracted to guide the search direction. To validate the performance of PKDE, two sets of benchmark test functions from IEEE CEC2005 and IEEE CEC2014 are used. The experimental results have indicated that PKDE is competitive (Fan et al. 2016). Besides, enhanced adaptive differential evolution (EADE) algorithm is presented to solve large-scale global optimization problems. A new mutation rule is introduced to use the information of good and bad vectors in the DE population of EADE (Mohamed 2017). There is no doubt that these newly proposed algorithms have made great contributions to the development of DE algorithm and EAs community.

Now DE algorithm is extended to solve COPs. The adaptive ranking mutation operator (ARMOR) for DE is proposed to solve COPs. The ARMOR is to make DE converge faster and achieves feasible solutions faster (Gong et al. 2015). Inspired by the fact that in modern society, a dual-population differential evolution (DPDE) with coevolution is designed for COPs (Gao et al. 2015). For a better coverage of the problem characteristics, a self-adaptive differential evolution algorithm is introduced (Elsayed et al. 2014). A ranking-based mutation operator and an improved dynamic diversity mechanism based on DE are proposed (Gong et al. 2014a). With the improved augmented Lagrangian approach, a cooperative coevolutionary DE algorithm is proposed (Ghasemishabankareh et al. 2016). Multi-objective optimization is combined with differential evolution (CMODE) to deal with COPs (Wang and Cai 2012a). A dynamic hybrid framework (DyHF) is designed for solving COPs (Wang and Cai 2012b). Objective function information is incorporated into the feasibility rule to form a new algorithm: FROFI (Wang et al. 2016). To cope with COPs, an improved DE and a novel archiving-based adaptive trade-off model are employed (Jia et al. 2013).

The above algorithms based on DE make great contributions to solve COPs. When using DE to solve COPs, it generates feasible individuals and infeasible individuals during the evolution. It also has to tackle the constraints. Two issues are of great importance for solving COPs when using DE. One is how to cope with the feasible and infeasible solutions. Feasibility rule emphasizes the fact that feasible solution is better than infeasible solution(Deb 2000). However, a lot of researches have indicated that infeasible solution is also very critical to find the global optima (Singh et al. 2008; Mezura-Montes and Coello 2005; Lin et al. 2014). So, how to extend three criteria to reconsider the infeasible solution should be observed. The second one is how to generate the offspring. The conventional DE uses just single evolution strategy during evolution. However, different problems need different strategies with different parameter values relying on the properties of problems. A single mutation strategy often cannot meet the requirement.

Based on discussed above, an effective DE algorithm is proposed by two efforts. First, the novel algorithm extends three criteria of feasibility rule to seven criteria. The effort is to compare feasible solution over infeasible solution. Then, two kinds of offspring generation methods are designed. The first one is DE/rand/1 as it is said to be the most widely employed and successful mutation strategy according to the current research. The second one is the integration of current-to-rand/1 and current-to-best/1 (Wang and Cai 2011). The idea is significant different from the above research. In order to validate the effectiveness of the proposed algorithm, benchmark functions from CEC2006 are adopted. The experimental results have demonstrated that the proposed algorithm is effective compared with other optimization algorithms. Then, the proposed algorithm is employed to solve the weight in AHP.

The paper is organized as follows. Literature review is given in Sect. 2. In Sect. 3, DE is briefly introduced and the proposed algorithm is presented. In Sect. 4, experiments are done based on standard benchmarks. The proposed method is employed to solve the real application in Sect. 5. The conclusions are made in Sect. 6.

2 Literature review

A comprehensive survey on COPs is introduced, in which following techniques are mainly discussed to solve COPs (Mezura-Montes and Coello 2011).

(1) Penalty function method

The approach introduces a penalty function or a coefficient into the original objective function (Michalewicz 1995; Coit et al. 1996; Tessema and Yen 2009). The aim is to penalize solutions which violate constraints. To effectively cope with constraint problems, Lin proposed a hybrid algorithm the rough penalty genetic algorithm (RPGA), which contained genetic algorithm and rough set theory. The aim of RPGA was to effectively resolve COPs and achieve robust solutions (Lin 2013). As it is difficult for static penalty approach to adjust penalty factors, the dynamic penalty approach is put forward. A novel optimization algorithm was designed by utilizing a penalty function in the objective function to treat violation (Homaifar et al. 1994). However, when using penalty function approach to solve COPs, it is still the most difficult to find appropriate penalty coefficients, which guide the search direction toward the optimum (Joines and Houck 1994).

(2) Multi-objective optimization techniques

The technique usually transforms the COPs into two objective function problems, the original objective function and the violation objective function. Then, the problems are handled by multi-objective optimization techniques. A generic optimization framework based on genetic algorithms was proposed to solve COPs (Venkatraman and Yen 2005). Wang and Cai proposed an effective evolutionary algorithm to optimize COPs according to multi-objective optimization principle (Cai and Wang 2006). Multi-objective optimization technique based on differential evolution algorithm was employed to tackle these problems (Qu and Suganthan 2011). Multi-objective optimization was combined with differential evolution to deal with COPs (Wang and Cai 2012c). However, solving multi-objective optimization problems themselves is still very complicated.

(3) Gradient method

An improved ε-constrained differential evolution algorithm was presented, which was integrated with pre-estimated comparison gradient (Yi et al. 2016). A novel distributed gradient algorithm was proposed for COPs (Yi et al. 2015). A new line search method as well as combining it with the spectral projected gradient approach was used to solve COPs (Yu 2008). A non-monotonic interior backtracking line search method combined with an affine scaling reduced preconditional conjugate gradient path approach was proposed (Zhu 2007). However, gradient method is applicable to two kinds of problems: a few variables or having convex feasible regions. The burden of computational cost will become increasingly heavier when the amount of variables increases (Deb 1995; Reklaitis et al. 1983).

(4) Ensemble of constraint-handling methods

Motivated by the no free lunch theorems, Mallipeddi and Suganthan proposed an ensemble of four constraint techniques (Mallipeddi and Suganthan 2010). Elsayed et al. employed two constraint-handling techniques to solve CMOPs (Ali and Kajee-Bagdadi 2009). Feasibility rule, ε-constrained, and an adaptive penalty function were combined to solve CMOPs (Tasgetiren et al. 2010).

(5) Feasibility rule

Let x_1 and x_2 be feasible solutions, and x_3 and x_4 are infeasible solutions.

1. Feasible solution is preferred to infeasible solutions without considering their objective value. So, x_1 and x_2 are preferred to x_3, x_4, respectively.
2. If $f(x_1)$ is smaller than $f(x_2)$, then x_1 is preferred to x_2.
3. If $CV(x_3) < CV(x_4)$, then x_3 is preferred to x_4.

There are also a number of other criteria, which are similar to the above. They are imposed to deal with COPs (Deb 2000; Richardson et al. 1989). These implementations employed different measures of constraint violations. By designing new comparison rules, the infeasible individuals with better objective function were made full use of during the evolution (Zheng et al. 2012). For instance, a novel diversity mechanism was designed (Mezura-Montes and Coello 2005). However, these criteria are too simple. They directly consider that infeasible solution is worse than feasible solution. In fact, infeasible solution is also of importance to maintain the diversity of the population.

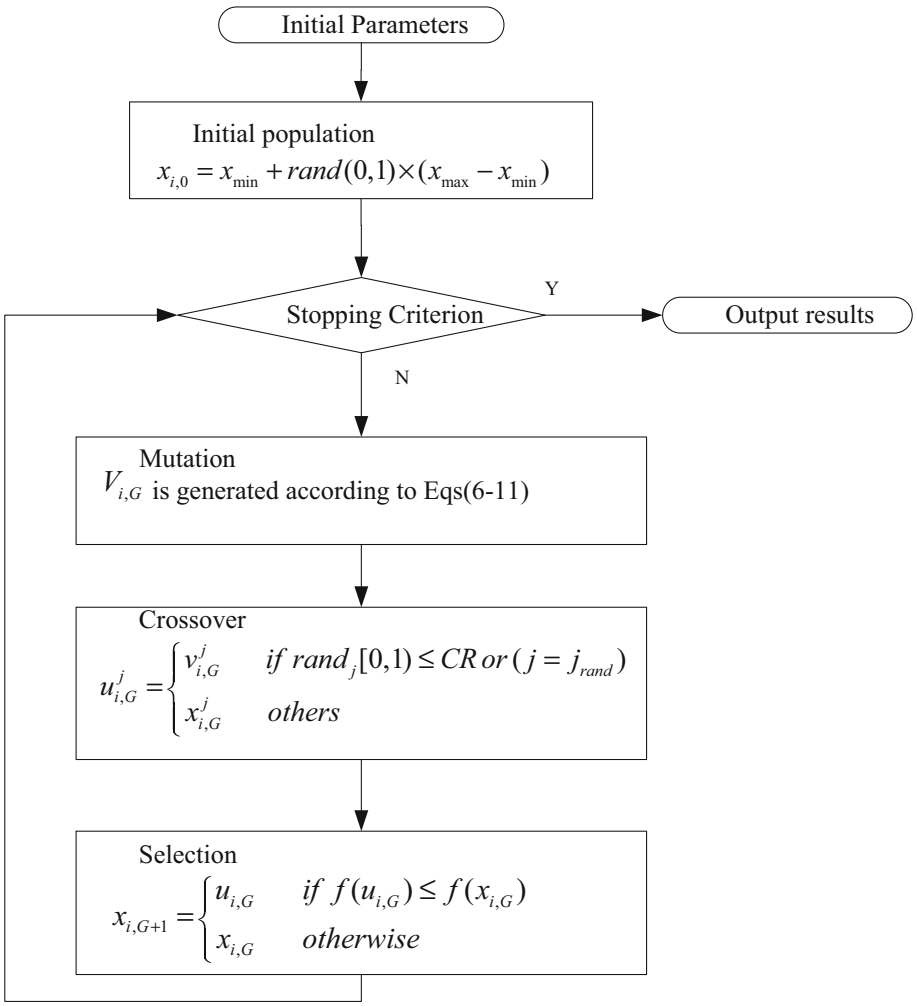

Fig. 1 The flowchart of conventional DE

3 DE

3.1 Conventional DE

In conventional DE, mutation, crossover, and selection are very critical. The main framework of DE is similar to EAs, which is exhibited in Fig. 1.

According to the main steps of DE, the trial vector $V_{i,G} = \{v_{i,G}^1, v_{i,G}^2, \ldots, v_{i,G}^D\}$ and $U_{i,G}^{=}\{u_{i,G}^1, u_{i,G}^2, \ldots, u_{i,G}^D\}$ are the trial vectors. The former is generated by mutation strategy in the mutation step. The latter is generated in crossover step.

Mutation strategy plays an important role during evolution. Price and Storn employed DE/rand/1/bin. It is widely used. The main strategies implemented in the DE are listed as follows (Sharma et al. 2012):

DE/best/1:

$$V_{i,G} = X_{best,G} + F.(X_{r_1^i,G} - X_{r_2^i,G}) \tag{6}$$

DE/best/2:

$$V_{i,G} = X_{best,G} + F.(X_{r_1^i,G} - X_{r_2^i,G}) + F.(X_{r_3^i,G} - X_{r_4^i,G}) \tag{7}$$

DE/current-to-best/1:

$$V_{i,G} = X_{i,G} + F.(X_{best,G} - X_{i,G}) + F.(X_{r_1^i,G} - X_{r_2^i,G}) \tag{8}$$

DE/rand/1:

$$V_{i,G} = X_{r_1^i,G} + F.(X_{r_2^i,G} - X_{r_3^i,G}) \tag{9}$$

DE/rand/2:

$$V_{i,G} = X_{r_1^i,G} + F.(X_{r_2^i,G} - X_{r_3^i,G}) + F.(X_{r_4^i,G} - X_{r_5^i,G}) \tag{10}$$

DE/current-to-rand/1 :

$$U_{i,G} = X_{i,G} + K.(X_{r_1^i,G} - X_{ri,G}) + F.(X_{r_2^i,G} - X_{r_3^i,G}). \tag{11}$$

Table 1 The pseudo codes of offspring generation methods

$V^1_{i,G} = X_{r^i_1,G} + F.(X_{r^i_2,G} - X_{r^i_3,G})$
if *rand*. ϕ **then**
  $V^2_{i,G} = X_{r^i_1,G} + K.(X_{r^i_1,G} - X_{r_i,G}) + F.(X_{r^i_2,G} - X_{r^i_3,G})$//current-to-rand
  else
  $V^2_{i,G} = X_{i,G} + F.(X_{best,G} - X_{i,G}) + F.(X_{r^i_1,G} - X_{r^i_2,G})$//current-to-best
end

The indices r^i_1, r^i_2, r^i_3 are mutually exclusive integers randomly generated within the range [0, 1], which are also different from the index i. F and K are the mutation scale factors, which are used in controlling the amplification of the differential variation.

3.2 The proposed algorithm

3.2.1 Mutation strategies

The first mutation strategy DE/rand/1 was developed for DE (Price et al. 2005) and is one of the most widely and successful employed strategies (Babu and Jehan 2003). But DE/best/2 may have some superiority to DE/rand/1 (Gamperle et al. 2002; Pahner and Hameyer 2000). Besides, it is beneficial to incorporate best solution information during evolution (Mezura-Montes et al. 2006). In contrast to DE/rand, algorithm can get benefits by holding best solution information. However, the behavior may also cause problems such as premature due as population diversity is reduced (Zhang and Sanderson 2009). Besides, it has been proved that current-to-best/1 strategy performs well when solving un-modal problems as it has the best individual information (Iorio and Li 2004). However, when solving multimodal problems, it easily traps in a local optimum. From the above discussion, it can be found that only one strategy cannot perform well during evolution. Here, two generation methods are used. The first one is DE/rand/1 as it is said to be the most wide and successful strategy (Babu and Jehan 2003). The second one is the integration of current-to-best/1 and current-to-rand/1 (Mezura-Montes et al. 2006).

Based on above discussion, the implementation of the mutation process is given in Table 1, where parameter $rand \in [0, 1]$ is a uniformly random number and ϕ is the feasible solution proportion of the last population. At the early stage, there are little feasible solutions and ϕ is very small. Thus, condition of current-to-rand can be met more frequently. At the middle and later stage, more and more feasible solutions are generated. Mutation strategy current-to-best will be employed more frequently, in which best solution is incorporated. By balancing the two strategies with the help of parameter φ, the algorithm can realize global and local search.

The crossover operation is the same as conventional DE. So, the two trial vectors $U^1_{i,G}, U^2_{i,G}$ are generated. Now, the key is how to compare the three vectors $X_{i,G}, U^1_{i,G}, U^2_{i,G}$.

3.2.2 Selection mechanism

The comparison among the three vectors $X_{i,G}, U^1_{i,G}, U^2_{i,G}$ is to select the one which is more suitable for the next evolution. There are two kinds: feasible solution and infeasible solution. If all the solutions are feasible, the one having lower objective function value is chosen. The main problems are to compare the feasible solution over infeasible solution and infeasible solution over infeasible solution. As the infeasible solution is also very important to keep the diversity of population, the algorithm should make good use of information from infeasible solution. The ε-constrained method introduces a relaxation of the limit ε to consider a solution as feasible (Takahama et al. 2005).

Let $f(x_1)$, $f(x_2)$ and $CV(x_1)$, $CV(x_2)$ be the fitness values and the constraint violation. Then, ε level comparison between $f(x_1)$ and $f(x_2)$ is defined as follows:

$$(f(x_1), CV(x_1)) <_\varepsilon (f(x_2), CV(x_2)) \Leftrightarrow \begin{cases} f(x_1) < f(x_2) & if\ CV(x_1), CV(x_2) \le \varepsilon \\ f(x_1) < f(x_2) & if\ CV(x_1) = CV(x_2) \\ CV(x_1) < CV(x_2) & otherwise \end{cases} \quad (12)$$

However, when the mechanism deals with complicated COPs, the global search ability is also limited. Combined with ε-constrained method and feasibility rule, the novel criteria are proposed in Table 2. In the procedure, there are two parameters e and P. The parameter e is the tolerance parameter, which is used to compare with total constraints CV in Eq. (5). The function of parameter e is similar to ε in Eq. (12). The selection probability parameter $P$is the probability parameter, which is between 0.9 and 1.

For the first criteria, the lower objective function value is preferred, which is the same to the feasibility rule (Deb 2000). In the second criterion, if the total constraints of two vectors are both less than e, the lower objective function value is also chosen. This rule is similar to ε-constrained (Takahama et al. 2005). The total constraints of two vectors are both more than e in the third criterion, and the lower total constraints vector is

selected with probability P. The rule is similar to feasibility rule (Deb 2000). If $rand > P$, the vector with lower fitness value is selected to keep the diversity of population in the fourth criterion. If $U_{i,G}^1$ is infeasible solution and $U_{i,G}^2$ is feasible solution, $U_{i,G}^2$ can be accepted in the fifth and sixth criteria. For the last criterion, the total constraints of $U_{i,G}^2$ are less than e and it is selected to maintain the diversity of population.

3.2.3 The procedure of the proposed algorithm e-DE

According to the discussion above, the main procedure of the novel algorithm *e*-DE is presented as follows:

Step 1 Initialize parameters. Maximum number of function evaluation evaluations (*Max_FES*); number of population *(NP)*; tolerance parameter *e;* probability *P.*

Step 2 Set *G=1* and randomly generate *NP* individuals $P_G = \{X_{1,G},...,X_{NP,G}\}$ with $X_{i,G} = \{x_{i,G}^1,...,x_{i,G}^D\}$, $i = 1,...,NP$ uniformly distributed in the range $[X_{\min}, X_{\max}]$, where $X_{\min} = \{x_{\min}^1, x_{\min}^2,..., x_{\min}^D\}$ and $X_{\max} = \{x_{\max}^1, x_{\max}^2,..., x_{\max}^D\}$.

Step 3 Calculate the fitness value, total constraint violations *CV*, and feasibility solution proportion ϕ.

Step 4 *FES= NP+FES*

Step 5 While *stopping criterion is not met*

Step 5.1 If (*e*<=10^-6) *e=0*; end

Step 5.2 Generate two vectors $V_{i,G}^1, V_{i,G}^2$ according to procedure in Table 1.

Step 5.3 Generate two vectors $U_{i,G}^1, U_{i,G}^2$ according to DE crossover and $V_{i,G}^1, V_{i,G}^2$.

Step 5.4 If the trial vectors $U_{i,G}^1, U_{i,G}^2$ are outside boundary; randomly generate them within the search space

Step 5.5 Selection

Calculate the fitness value, total constraint violations *CV* and feasibility solution proportion *p* of $U_{i,G}^1, U_{i,G}^2$ respectively.

$U_{i,G}^1$ = CompareVector($U_{i,G}^1, U_{i,G}^2$) by calling procedure in Table 2;

$X_{i,G}$ = CompareVector ($X_{i,G}, U_{i,G}^1$) by calling procedure in Table 2;

Step 5.6 Set *FES= NP+2× FES*;

Step 5.7 *e* decreases linearly;

Step 6 End while

4 Experiments

4.1 Test functions

For the purpose of validating the performance of the algorithm *e*-DE, twenty-two functions are selected from CEC2006 (Liang et al. 2006b) in Table 3.These well-known benchmark functions include linear, nonlinear, polynomial, quadratic, and so on, where $f(x^*)$ is the objective function value for optimal solution x^*.

For these test functions, 25 independent runs were performed. Parameters setting: *NP=50,CR=0.9, F=0.8, e=1,* selection probability parameter $P \in [0.9, 1]$. These parameters were maintained in all runs.

Table 2 The pseudo codes of selection mechanism for feasible solution and infeasible solution

Function CompareVector $(U_{i,G}^1, U_{i,G}^2)\{$
if$(CV(U_{i,G}^1) = CV(U_{i,G}^2) = 0 \& f(U_{i,G}^1) > f(U_{i,G}^2))$
return $U_{i,G}^2$ if//the above is the first criterion
if$(CV(U_{i,G}^1) \neq 0 \& CV(U_{i,G}^2) \neq 0 \& CV(U_{i,G}^1) \leq e \& CV(U_{i,G}^2) \leq e \& f(U_{i,G}^2) < f(U_{i,G}^1))$
return $U_{i,G}^2$if//the above is the second criterion
if$(CV(U_{i,G}^1) \geq e \& CV(U_{i,G}^2) \geq e \& CV(U_{i,G}^2) < CV(U_{i,G}^1) \& rand \leq P))$
return $U_{i,G}^2$//the above is the third criterion
if$(CV(U_{i,G}^1) \geq e \& CV(U_{i,G}^2) \geq e \& f(U_{i,G}^2) < f(U_{i,G}^1) \& rand \geq P))$
return $U_{i,G}^2$//the above is the fourth criterion
if$(CV(U_{i,G}^1) \neq 0 \& CV(U_{i,G}^2) = 0 \& CV(U_{i,G}^1) \leq e \& f(U_{i,G}^2) < f(U_{i,G}^1))$
return $U_{i,G}^2$//the above is the fifth criterion
if$(CV(U_{i,G}^1) \geq e \& CV(U_{i,G}^2) = 0)$
return $U_{i,G}^2$//the above is the sixth criterion
if$(CV(U_{i,G}^1) = 0 \& CV(U_{i,G}^2) \neq 0 \& CV(U_{i,G}^2) \leq e \& f(U_{i,G}^2) < f(U_{i,G}^1))$
return $U_{i,G}^2$the above is the seventh criterion
If above criteria are not met, return $U_{i,G}^1$

Table 3 Benchmark test functions

Test function	n	Objective function	ρ (%)	LI	NI	LE	NE	α	$f(x^*)$
g01	13	Quadratic	0.0111	9	0	0	0	6	− 15.0000000000
g02	20	Nonlinear	99.9971	0	2	0	0	1	− 0.8036191042
g03	10	Polynomial	0.0000	0	0	0	1	1	− 1.0005001000
g04	5	Quadratic	51.1230	0	6	0	0	2	− 30665.5386717834
g05	4	Cubic	0.0000	2	0	0	3	3	5126.4967140071
g06	2	Cubic	0.0066	0	2	0	0	2	− 6961.8138755802
g07	10	Quadratic	0.0003	3	5	0	0	6	24.3062090681
g08	2	Nonlinear	0.8560	0	2	0	0	0	− 0.0958250415
g09	7	Polynomial	0.5121	0	4	0	0	2	680.6300573745
g10	8	Linear	0.0010	3	3	0	0	0	7049.2480205286
g11	2	Quadratic	0.0000	0	0	0	1	1	0.7499000000
g12	3	Quadratic	4.7713	0	1	0	0	0	− 1.0000000000
g13	5	Nonlinear	0.0000	0	0	0	3	3	0.0539415140
g14	10	Nonlinear	0.0000	0	0	3	0	3	− 47.7648884595
g15	3	Quadratic	0.0000	0	0	1	1	2	961.7150222899
g16	5	Nonlinear	0.0204	4	34	0	0	4	− 1.9051552586
g17	6	Nonlinear	0.0000	0	0	0	4	4	8853.5338748065
g18	9	Quadratic	0.0000	0	13	0	0	0	− 0.8660254038
g19	15	Nonlinear	33.4761	0	5	0	0	0	32.6555929502
g21	7	Linear	0.0000	0	1	0	5	6	193.7245100700
g23	9	Linear	0.0000	0	2	3	1	6	− 400.0551000000
g24	2	Linear	79.6556	0	2	0	0	2	− 5.5080132716

4.2 Experimental results

The results of e-DE are presented in Table 4 and Table 5.On the basis of the reference (Liang et al. 2006b), the best, the worst, median, mean, and standard deviation (Std) of the function error value ($f(x)$-$f(x^*)$) of the acquired best results with 5×10^3, 5×10^4, and 5×10^5 FES for each benchmark function are presented, where x^* is the best known solution and $f(x^*)$ is the best function value.

The experimental results have demonstrated that the e-DE has the ability to succeed in finding feasible solutions, which are close to the best known solutions for g08 in 5×10^3

Table 4 Function errors acquired when 5×10^3, 5×10^4, and 5×10^5 for test functions g01–g12

FES		g01	g02	g03	g04	g05	g06
5000	Best	3.77E+0	4.13E−1	1.94E−1	1.11E+1	4.55E+1	6.14E−1
	Median	5.41E+0	4.74E−1	2.56E−1	3.30E+1	6.18E+1	4.37E+0
	Worst	6.57E+0	5.57E−1	3.11E−1	5.35E+1	7.10E+2	2.68E+1
	Mean	5.36E+0	4.77E−1	2.61E−1	3.36E+1	1.51E+2	6.84E+0
	Std	5.71E−1	1.28E−3	9.32E−4	1.36E+2	3.08E+4	4.49E+1
50,000	Best	5.42E−4	1.94E−1	3.11E−8	3.33E−9	5.37E−11	3.37E−11
	Median	2.07E−3	2.51E−1	9.11E−8	1.50E−8	9.97E+1	3.37E−11
	Worst	4.55E−3	307E−1	2.17E−7	6.30E−8	5.55E+2	3.37E−11
	Mean	2.06E−3	2.52E−1	1.02E−7	1.83E−8	1.47E+2	3.37E−11
	Std	9.06E−7	8.93E−4	2.00E−15	2.22E−16	2.36E+4	0
500,000	Best	0	1.16E−9	− 1.00e−11	7.64e−11	− 1.82e−12	3.37E−11
	Median	0	3.15E−8	− 1.00e−11	7.64e−11	− 1.82e−12	3.37E−11
	Worst	0	2.53E−2	− 1.00e−11	7.64e−11	− 1.82e−12	3.37E−11
	Mean	0	1.69E−3	− 1.00e−11	7.64e−11	− 1.82e−12	3.37E−11
	Std	0	3.58E−5	0	0	0	0
FES		g07	g08	g09	g10	g11	g12
5000	Best	2.94E+1	8.20E−11	1.17E+1	1.89E+3	5.50E−5	2.33E−5
	Median	5.93E+1	8.20E−11	2.76E+1	3.66E+3	3.31E−3	1.37E−4
	Worst	1.17E+2	8.21E−11	6.91E+1	6.60E+3	2.28E−1	5.00E−4
	Mean	6.17E+1	8.20E−11	2.81E+1	3.83E+3	3.81E−2	1.89E−4
	Std	4.21E+1	3.45E−27	1.39E+2	1.29E+6	4.30E−2	2.25E−8
50,000	Best	1.60E−2	8.19E−11	2.82E−9	2.54E+0	0	0
	Median	3.24E−2	8.19E−11	9.68E−9	5.82E+0	0	0
	Worst	6.67E−2	8.19E−11	2.77E−9	1.31E+1	1.52E−13	0
	Mean	3.42E−2	8.19E−11	1.17E−8	6.21E+0	6.10E−15	0
	Std	1.47E−4	0	4.89E−17	5.80E+0	0	0
500,000	Best	7.97E−11	8.20E−11	− 9.83E−11	6.18E−11	0	0
	Median	7.97E−11	8.20E−11	− 9.83E−11	6.28E−11	0	0
	Worst	7.97E−11	8.20E−11	− 9.83E−11	6.37E−11	0	0
	Mean	7.97E−11	8.20E−11	− 9.83E−11	6.28E−11	0	0
	Std	0	0	0	1.39E−25	0	0

function evaluations (FES) and for g03, g04, g06, g09, g11, g12, g16, and g24 in 5×10^4 FES. When FES is set to 5×10^5, the best solutions of experiments are close or equal to the optimal known solutions.

When FES is equal to 5×10^5, all the experimental results for functions g01, g11, and g12 are the same as the optimal values. The *e*-DE finds solutions close to the best solutions for the fourteen test functions (g04-g10, g14-g16, g18, g19, g23, g24). The acquired best solutions approximate the known optimal values with differences (10^{-10}). It has indicated that the *e*-DE can obtain the results, which are approximate to optimal solutions for these five test functions. The best results for the g03, g05, g09 problems are even beyond to its optimal solutions as equality constraints are converted into inequality constraints with a tolerance of 0.0001, revealing that the algorithm *e*-DE can consistently find the best solutions in 25 experiments.

Notably, the standard deviations of the *e*-DE are very small in fourteen test problems (g01, g03-g09, g11, g12, g14-g16, g24). This finding implies that the novel algorithm is robust when solving COPs.

4.3 The performance of success performance

The number of min, median, max, mean FES required in each run to find a feasible solution is shown in Table 6. Besides, the success rate, the feasible rate, and the success performance are also given in Table 6.

Once one feasible solution is found during the run, it is a feasible run. If the feasible solution $\vec{x}$ satisfies $f(\vec{x}) - f(\vec{x}^*) \leq 0.0001$, it is a successful run.

Table 5 Function errors achieved when 5×10^3, 5×10^4, and 5×10^5 for test functions g13–g19, g23, g24

FES		g13	g14	g15	g16	g17
5000	Best	1.55E−1	− 3.27E+1	9.07E−4	1.77E−2	− 2.09E+1
	Median	9.29E−1	− 8.56E+0	8.78E+0	3.78E−2	1.01E+2
	Worst	1.09E+1	4.78E+1	2.29E+0	5.80E−2	4.45E+2
	Mean	1.61E+0	− 3.09E+0	1.07E+0	3.64E−2	1.31E+2
	Std	6.78E+0	6.22E+2	7.29E+0	1.39E−4	1.33E+4
50,000	Best	1.96E−1	1.19E+0	6.09E−11	2.90E−10	1.99E+1
	Median	8.07E−1	5.39E+0	7.73E−1	5.47E−10	8.72E+1
	Worst	3.00E+0	7.16E+0	5.57E+0	1.22E−9	3.43E+2
	Mean	8.78E−1	5.17E+0	1.52E+0	5.74E−10	8.80E+1
	Std	2.48E−1	1.85E+0	3.01E+0	6.08E−20	3.39E+3
500,000	Best	4.19E−11	8.51E−12	6.08E−11	6.52E−11	1.82E−12
	Median	3.85E−6	8.51E−12	6.08E−11	6.52E−11	5.46E−8
	Worst	3.85E−5	8.51E−12	6.08E−11	6.52E−11	7.41E−1
	Mean	2.77E−6	8.51E−12	6.08E−11	6.52E−11	3.55E−1
	Std	3.11E−6	0	0	0	1.43E−2
FES		g18	g19	g21	g23	g24
5000	Best	4.60E−1	1.18E+2	1.21E+2	4.00E+2	6.80E−5
	Median	8.66E−1	1.99E+2	5.88E+2	4.00E+2	5.51E+0
	Worst	8.66E−1	3.14E+2	7.95E+2	4.00E+2	5.51E+0
	Mean	8.16E−1	2.07E+2	5.43E+2	4.00E+2	3.31E+0
	Std	1.42E−2	2.30E+3	4.00E+4	3.03E−26	7.58E+0
50,000	Best	6.40E−3	8.73E−1	9.23E−4	1.26E+2	4.67E−12
	Median	1.33E−2	1.54E+0	3.99E+1	3.20E+2	4.67E−12
	Worst	2.01E−1	2.11E+0	1.34E+2	7.10E+2	4.67E−12
	Mean	1.97E−2	1.51E+0	6.02E+1	3.12E+2	4.67E−12
	Std	1.44E−3	1.20E−1	3.93E+3	1.73E+4	0
500,000	Best	1.56E−11	4.69E−11	3.81E−10	− 1.71E−13	4.67E−12
	Median	1.60E−11	4.74E−11	3.00E−10	5.68E−14	4.67E−12
	Worst	2.08E−11	8.65E−11	7.31E−2	1.15E−11	4.67E−12
	Mean	1.65E−11	5.01E−11	4.19E−2	7.07E−13	4.67E−12
	Std	1.56E−24	6.41E−23	3.89E−3	5.54E−24	0

Feasible rate = (feasible runs)/(total runs)
Success rate = (successful runs)/(total runs)
Success performance = mean (FES for successful runs) ×(total runs)/(successful runs)

Table 6 exhibits the overall performance of *e*-DE. It can be noticed that the feasible rate is 100% for these functions. With the exception of test functions g02, g13, g17, and g21, the success rate of 100% has been achieved for the rest functions.

The *e*-DE needs fewer than 5000 FES for one function to reach the success condition and requires fewer than 50,000 FES to achieve success for ten functions. No more than 200,000 FES for 19 test functions and no more than 300,000 FES for 21 functions are demanded, respectively.

In practical terms, the proposed *e*-DE presents superior SP that has a lot of inequality constraints and disconnected feasible regions, for instance g16, g12, and g24. Moreover, it is observed that the proposed *e*-DE can successfully cope with great majority of test problems. Functions include g01, g07, g10, g16, and g18 with a larger amount of inequality constraints compared to the other COPs. The results have indicated that the *e*-DE with the dynamic control parameters is effective to deal with such constraints.

In addition, it is mentioned that some functions are considered as the comparatively more difficult functions among these test functions (Liang and Suganthan 2006; Huang et al. 2006). For instance, g02, g13, and g17 are the multimodal COPs. Similar to other EAs, the algorithm also has difficulty to tackle these complicated functions. However, the algorithm achieves success rates as high as 92, 44, and 44% for g02, g13, and g17, respectively. It is obviously better than majority of EAs.

Table 6 Number of FES to reach the success condition, success rate, feasible rate, and success performance

Function	Best	Median	Worst	Mean	Feasible rate (%)	Success rate (%)	Success performance
g01	57950	65350	68550	64274	100	100	64274
g02	155950	189650	253450	192297	100	92	209019
g03	29450	33350	36650	33066	100	100	33066
g04	42550	44050	45250	43942	100	100	43942
g05	34050	144550	265750	152110	100	100	152110
g06	41150	42150	42850	42098	100	100	42098
g07	90750	99050	111050	99614	100	100	99614
g08	4550	6050	9150	6254	100	100	6254
g09	27650	29550	31350	29446	100	100	29446
g10	128050	135850	142850	135934	100	100	135934
g11	9550	20750	54250	24566	100	100	24566
g12	650	1350	1850	1354	100	100	1354
g13	201850	303650	368750	303741	100	44	690320
g14	64150	80150	160850	92730	100	100	92730
g15	35250	73150	256050	95022	100	100	95022
g16	22750	24750	26150	24410	100	100	24410
g17	166450	253850	400650	264232	100	44	600527
g18	116150	137550	181650	140458	100	100	140458
g19	184650	205650	233450	207286	100	100	207286
g21	65750	74550	158250	84500	100	72	117361
g23	35350	43750	79750	46922	100	100	46922
g24	13150	20050	24850	19678	100	100	19678

Table 7 Mean number of FES to find the first feasible solution

Function	Mean	Function	Mean
g01	1418	g12	150
g02	210	g13	28394
g03	698	g14	45998
g04	150	g15	18450
g05	23430	g16	870
g06	742	g17	26286
g07	1774	g18	5706
g08	198	g19	150
g09	230	g21	25838
g10	3962	g23	41990
g11	8206	g24	150

In order to approve the velocity that *e*-DE finds the feasible region, Table 7 presents the mean number of FES to find feasible solutions over 25 runs. The proposed algorithm *e*-DE requires no more than 1000 FES for ten test problems, between 1000 and 10,000 FES for five test problems. For the rest test problems, the numbers of FES finding the first feasible solutions are between 10,000 and 50,000. The finding reveals that the algorithm has good convergence speed.

4.4 Comparison against the latest DE variants

In the previous subsection, the efficacy of e-DE is verified through the benchmark functions. In this section, e-DE is compared with state-of-the-art EAs for the COPs. These algorithms are CMODE (Wang and Cai 2012a), DyHF (Wang and Cai 2012b), ICDE (Jia et al. 2013), rank-iMDDE (Gong et al. 2014a), and CCiALF (Ghasemishabankareh et al. 2016). The CMODE algorithm is proposed, which combines multi-objective optimization with differential evolution to deal with COPs. A dynamic hybrid framework, (DyHF) is designed for solving COPs. This framework consists of two major steps: global search model and local search model. To solve COPs, an improved DE and a novel archiving-based adaptive trade-off model are employed to form a new algorithm ICDE. An improved constrained differential evolution (rank-iMDDE) method is proposed, where a ranking-based mutation and an improved dynamic diversity mechanism are presented. A cooperative coevolutionary differential evolution algorithm based on the improved augmented Lagrangian approach (CCiALF) is proposed for solving COPs. We choose these five EAs for comparisons due to their good performance obtained and the same *Max_NFEs* (240,000) used. As it is fair to keep the same *Max_NFEs* for each algorithm,

Table 8 The best, mean, and standard variation of six algorithms for g01–g13

	CMODE	DyHF	ICDE	rank-iMDDE	CCiALF	e-DE
g01						
Best	− 14.99999999	− 15	− 15	− 15	− 15	− 15
Mean	− 14.9999999	**− 15**	**− 15**	**− 15**	**− 15**	**− 15**
Std	1.32E−8	0	0	0.00E+00	2.39E−08	0
g02						
Best	− 0.8036	− 0.803619	− 0.803611	− 0.80361905	− 0.8036176	− 0.803619
Mean	− 0.8036	− 0.803619	− 0.7976561	− 0.80202119	− 0.7930875	− 0.80196
Std	5.1E−6	5.7E−15	5.5E−5	4.57E−03	8.30E−03	1.4E−5
g03						
Best	− 1.0005	− 1.0005	− 1.0005	− 1.0005001	− 1.000501	− 1.0005
Mean	**− 1.0005**	**− 1.0005**	**− 1.0005**	**− 1.0005001**	**− 1.000501**	**− 1.0005**
Std	1.5E−16	2.3E−32	1.77E−31	0.00E+00	1.69E−08	2.8E−31
g04						
Best	30665.53867	− 30665.53867	− 30665.53867	− 30665.539	− 30665.539	− 30665.53867
Mean	**30665.53867**	**− 30665.53867**	**− 30665.53867**	− 30665.539	− 30665.539	**− 30665.53867**
Std	3.71E−23	1.4E−32	1.38E−23	0.00E+00	9.80E−06	1.38E−23
g05						
Best	5126.4967	5126.4967	5126.4967	5126.496714	5126.4967	5126.496714
Mean	**5126.4967**	**5126.4967**	**5126.4967**	**5126.496714**	**5126.497**	**5126.496714**
Std	7.75E−24	7.8E−24	7.75E−24	0.00E+00	9.17E−08	2.75E−12
g06						
Best	− 6961.813756	− 6961.81388	− 6961.81388	− 6961.81388	− 6961.814	− 69618.1388
Mean	− 6961.813756	**− 6961.81388**	**− 6961.81388**	**− 6961.81388**	− 6961.814	**− 6961.81388**
Std	0E+0	0E+0	0E+0	0.00E+00	5.19E−11	0E+0
g07						
Best	24.30621	24.30621	24.30621	24.30620907	24.3062	24.30621
Mean	**24.30621**	**24.30621**	**24.30621**	**24.30620907**	24.3062	**24.30621**
Std	1.1E−14	2.0E−23	9.89E−29	0.00E+00	6.82E−07	1.87E−23
g08						
Best	− 0.095825	− 0.095825	− 0.095825	− 0.09582504	− 0.09582505	− 0.095825
Mean	**− 0.095825**	**− 0.095825**	**− 0.095825**	**− 0.09582504**	**− 0.09582505**	**− 0.095825**
Std	3.0E−43	2.0E−34	2E−34	0.00E+00	1.07E−15	2.0E−34
g09						
Best	680.630057	680.630057	680.630057	680.6300574	680.6300	680.630057
Mean	**680.630057**	**680.630057**	**680.630057**	**680.6300574**	**680.6300**	**680.630057**
Std	2.7E−26	8.1E−26	1.13E−25	0.00E+00	5.43E−08	1.10E−25
g10						
Best	7049.24802	7049.24802	7049.24802	7049.248021	7049.248	7049.24802
Mean	**7049.24802**	**7049.24802**	**7049.24802**	**7049.248021**	**7049.248**	**7049.24802**
Std	1.3E−13	2.7E−14	4.76E−22	0.00E+00	6.04E−07	4.89E−19
g11						
Best	0.7499	0.7499	0.7499	0.7499	0.7498959	0.7499
Mean	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.7498984**	**0.7499**
Std	1.3E−32	1.3E−32	1.28E−32	0.00E+00	2.04944E−16	1.28E−32
g12						
Best	− 1	− 1	− 1	− 1	− 1	-1

Table 8 continued

	CMODE	DyHF	ICDE	rank-iMDDE	CCiALF	e-DE
Mean	**− 1**	**− 1**	**− 1**	**− 1**	**− 1**	**− 1**
Std	0	0	0	0.00E+00	0	0
g13						
Best	0.0539415	0.0539415	0.0539415	0.0539415	0.05394151	0.0539415
Mean	0.069335957	**0.0539415**	0.146308	**0.0539415**	0.05394261	**0.0539415**
Std	5.9E−3	3.4E−34	0E−0	0.00E+00	4.03E−06	1.27E−10

Table 9 The best, mean, and standard variation of six algorithms for g14–g19, g21, and g24

	CMODE	DyHF	ICDE	rank-iMDDE	CCiALF	e-DE
g14						
Best	− 47.764888	− 47.764888	− 47.764888	− 47.7648885	− 47.7649	− 47.764888
Mean	**− 47.764888**	**− 47.764888**	**− 47.764888**	− 47.7648885	− 47.7649	**− 47.764888**
Std	7.2E−19	8.1E−28	5.91E−28	0.00E+00	4.04E−08	5.47E−23
g15						
Best	961.715	961.715	961.715	961.7150	961.7150	961.715
Mean	**961.715**	**961.715**	**961.715**	**961.7150**	**961.7150**	**961.715**
Std	5.4E−25	3.4E−25	3.37E−25	0.00E+00	1.86E−08	2.41E−14
g16						
Best	− 1.905155	− 1.905155	− 1.905155	− 1.905155	− 1.905155	− 1.905155
Mean	**− 1.905155**	**− 1.905155**	**− 1.905155**	**− 1.905155**	**− 1.905155**	**− 1.905155**
Std	2.1E−32	5.0E−31	2.05E−31	0.00E+00	9.77E−09	2.05E−31
g17						
Best	8853.53387	8853.53387	8853.53387	8853.539675	8857.447	8853.53387
Mean	8853.53389	8854.64472	**8853.53387**	8853.539675	8916.856	8853.8165
Std	2.2E−9	6.8E+0	1.79E−24	0.00E+00	3.64E+01	1.7E−2
g18						
Best	− 0.866025	− 0.866025	− 0.866025	− 0.866025	− 0.8660255	− 0.866025
Mean	**− 0.866025**	**− 0.866025**	**− 0.866025**	**− 0.866025**	− 0.8660255	**− 0.866025**
Std	1.3E−17	9.5E−21	5.94E−29	0.00E+00	3.58E−07	1.14E−12
g19						
Best	32.6557	32.65559	32.65559	32.6559	32.65561	32.65559
Mean	32.6558	**32.65559**	**32.65559**	32.6556	32.66077	**32.65559**
Std	6.0E−9	1.6E−17	6.21E−17	8.30E−05	2.35E−04	3.97E−11
g21						
Best	193.7245	193.7245	193.7245	193.7245	193.7243	193.7245
Mean	230.398	200.9283	**193.7245**	**193.7245**	193.7352	193.7275
Std	3.60E+3	7.61E+2	1.73E−21	0.00E+00	1.20E−02	6.0E−10
g23						
Best	− 400.0551	− 400.0551	− 400.0551	− 400.0551	− 400.0551	− 400.0551
Mean	− 400.0545	**− 400.0551**	**− 400.0551**	− 398.180865	− 400.0536	**− 400.0551**
Std	3.2E−6	3.6E−11	2.47E−10	4.51E+00	5.00E−03	1.0E−6
g24						
Best	− 5.508013	− 5.508013	− 5.508013	− 5.508013	− 5.508013	− 5.508013
Mean	**− 5.508013**	**− 5.508013**	**− 5.508013**	**− 5.508013**	**− 5.508013**	**− 5.508013**
Std	8.2E−31	8.2E−31	8.2E−31	0.00E+00	1.30E−08	8.2E−31

Table 10 Average rankings of CMODE, DyHF, ICDE, rank-iMDDE, CCiALF, and e-DE by the Friedman test for the 22 functions in terms of the mean values

Algorithm	Ranking
CMODE	4.07
DyHF	3.48
ICDE	4.45
rank-iMDDE	3.09
CCiALF	4.73
e-DE	3.09

the *Max_NFEs* of e-DE is also set to 240,000. Otherwise, the comparisons are unfair and unfaithful.

In Tables 8 and 9, the best, mean, and standard deviation of the objective function values of the final solutions for each algorithms are shown. The overall best results among the six EAs are highlighted in boldface. Note that the results of rank-iMDDE and CCiALF are directly obtained from their corresponding literature.

From the Tables 8 and 9, it is very clear that the six algorithms have achieved the same performance in ten functions (g03, g07, g08, g09, g10, g11, g12, g15, g16, and g24), in which three equality functions and seven inequality functions are included. These results have indicated that the proposed algorithm has the same search ability to the five algorithms. CMODE has achieved the best performance in g14 and g18. DyHF and e-DE have the best performance in g01, g06, g13, g14, g18, g19, and g23. ICDE has the superior performance in g01, g06, g14, g17, g18, g19, g21, and g23. Rank-iMDDE has the superior performance in g01,g06,g13,g18, and g21.

According to the results shown in Tables 8 and 9, it can be clearly seen that e-DE consistently obtains highly competitive results in all functions compared with other five EAs. In terms of the best results, e-DE gets the best or similar values among the six EAs in all 22 functions. With respect to the mean results, only in two functions g17 and g21, e-DE is slightly worse. In the rest 20 functions, e-DE is able to obtain better or similar results compared with other five EAs.

Friedman test has been used to rank these algorithms. It is a nonparametric statistical test .It is used to detect differences in treatments across multiple test attempts. Based on the mean values in Tables 8 and 9, the final rankings obtained by the Friedman test are shown in Table 10. With respect to the average rankings of all algorithms by the Friedman test, Table 10 shows that our proposed e-DE and rank-iMDDE get the first ranking among six algorithms, followed by DyHF, CMODE, ICDE, and CCiALF. It denotes that the e-DE can perform as well as or even superior to other algorithms. In other words, the advantages of the proposed e-DE method are stated and proven.

It is reported that the test function g17 has the new best known function value 8853.533875 (Brajevic 2015). The results also have been achieved for the four algorithms (CMODE, DyHF, ICDE, and e-DE). The finding also indicates that the four algorithms have good abilities to deal with COPs.

4.5 The parameter sensitivity analysis

The parameter e needs to be tested as it is one of the most important parameters during the evolution. In this section, the impact of this parameter on the algorithm is focused on. The e-DE runs 25 times on each function with ten different parameters of 1, 2, 3, 4, 5,6,7,8,9,10, random between 1 and 10, respectively.

Figures 2, 3, and 4 reveal the box plots of $\log(f(\vec{x}) - f(\vec{x}^*) + \exp(-10))$ that e-DE algorithm obtained in the 25 runs. $f(\vec{x}^*)$ is the objective function value for known optimal solution $\vec{x}^*$, and $f(\vec{x})$ is the optimal solution obtained by the algorithm in each run.

From Figs. 2, 3, and 4, it is found that the proposed algorithm is not sensitive to parameter value on g01, g03, g04, g06, g08, g11, g12, g15, g16, and g24,in which three equality constrains and seven inequality constrains are included. The parameter e has little influence on g02, g05, g07, g09, g10, g13, g18, and g19 and has some impact on g14, g17, g21, and g23.

In particular, test function g02 has a nonlinear objective function and 20 decision variables; test function g16 has thirty-four nonlinear inequality constraints and four linear inequality constraints. The feasible region is limited. With ten different parameters, the algorithm can obtain the known optimal values in 25 runs. The performance of e-DE algorithm is also stable to find the known optimal solutions. Therefore, the e-DE algorithm is less sensitive to the parameter e between 1 and 10 for most of test functions. Therefore, the algorithm is robust.

4.6 Effectiveness of the selection mechanism

Experiments were performed to validate the effectiveness of the algorithm for rand/1/bin and combined mutation strategies. The former one is the e-DE with the rand/1/bin (e-DE-1), and the latter one is the current-to-best/1 and current-to-rand/1 (e-DE-2). The results obtained from the above experiments are compared with the e-DE. The comparisons of four representative test functions g14, g17, g19, and g21 are listed in Table 11.

Test function g14 has ten decision variables, three equality constraints, and a nonlinear objective function. Note that e-DE-2 can obtain similar results with e-DE, which reveals that adding rand/1/bin cannot give negative effects on the diversity of population. The mean and STD results of e-DE and e-DE-2 are better than those of e-DE-1. With respect

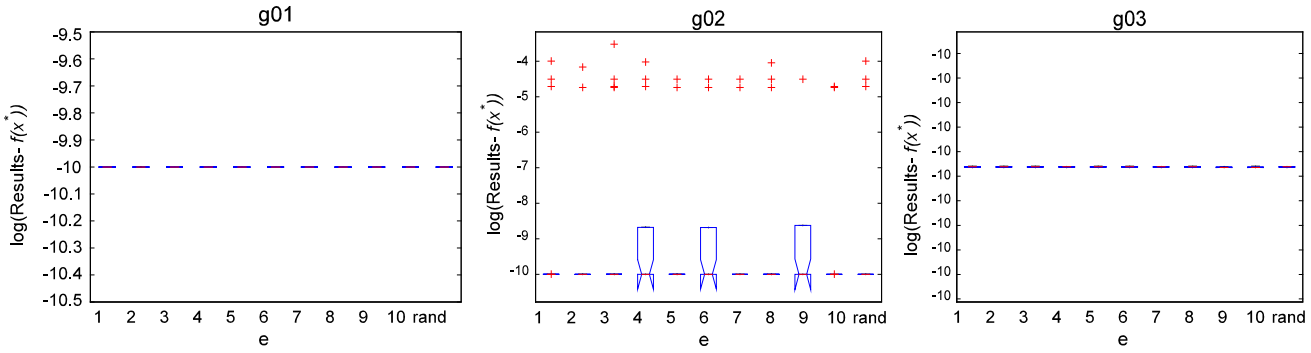

Fig. 2 *e*-DE with different parameters on g01–g03

to test function g17, *e*-DE-1 and *e*-DE-2 have not achieved any success during the evolution. The success rate of proposed algorithm is 44%, which is the highest among the three algorithms. The above phenomenon signifies that the algorithm including two strategies has higher convergence speed. Function g19 has a nonlinear objective function with fifteen decision variables. The results of *e*-DE and *e*-DE-2 are much better than those of *e*-DE-1. It manifests that rand/1/bin strategy deteriorates the convergence speed. However, the success rate of *e*-DE-1 is higher than *e*-DE-2 for function g21. Function g21 is a complex constrained problem. As the *e*-DE-2 has the information of the optimal individual, it degrades the diversity of the population.

According to the above discussion, *e*-DE-1 with rand/1/bin strategy can enhance the diversity of the population and deteriorate the convergence speed, *e*-DE-2 with current-to-best/1 and current-to-rand/1 can accelerate the convergence speed and degrade the diversity of the population, and *e*-DE with the above two strategies can trade off the diversity of the population and the convergence speed very well. The comparison above confirms that strategy selection is reasonable.

5 Solve weights in analytic hierarchy process (AHP)

AHP, developed by Saaty (1980), is used to tack multiple criteria decision making (MCDM) in real applications (Wu et al. 2016; Xiaobing et al. 2011; Gong et al. 2014b). MCDM is to screen, prioritize, rank, or select a set of alternatives under usually independent, incommensurate, or conflicting attributes. In AHP, multiple pair-wise comparisons are from a standardized comparison scale of nine levels shown in Table 12.

Suppose that $C = \{C_j | j = 1, 2...n\}$ be the set of criteria. Evaluation matrix Acan be gotten, in which every element a_{ij} $(i, j = 1, 2...n)$ represents the relative weights of the criteria illustrated:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots \\ \dots & \dots & a_{ii} & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \tag{13}$$

If matrix A is completely consistent, it has to comply with following condition:

$$\begin{aligned} a_{ii} &= \frac{w_i}{w_i} = 1 \\ a_{ji} &= \frac{w_j}{w_i} = \frac{1}{a_{ij}} \\ a_{ij}a_{jk} &= \frac{w_i}{w_j} \times \frac{w_j}{w_k} = \frac{w_i}{w_k} = a_{ik}. \end{aligned} \tag{14}$$

According to above properties, the following equations can be obtained:

$$\sum_{k=1}^{n}(a_{ik}w_k) = \sum_{k=1}^{n}(w_i/w_k)w_k = nw_i, i = 1, 2, \dots, n \tag{15}$$

$$\sum_{i=1}^{n}\left|\sum_{k=1}^{n}(a_{ik}w_k) - nw_i\right| = 0. \tag{16}$$

In other words, if the judgment matrix Ameets the Eq. (16), it is completely consistent. However, it is very difficult to achieve the condition in the real application. In fact, the matrix A has just to meet the satisfactory consistency. The Eq. (16) can be converted into the following format:

$$\min\ CIF(w) = \frac{\sum_{i=1}^{n}\left|\sum_{k=1}^{n}(a_{ik}w_k) - nw_i\right|}{n} \tag{17}$$

$$0 < w_k < 1, \sum_{k=1}^{n} w_k = 1 \tag{18}$$

The smaller the *CIF* is, the more consistent the matrix A is. So, the weight acquisition is converted into the single objective optimization with constraint. The objective is to minimize *CIF*, and the constraint is:$0 < w_k < 1, \sum_{k=1}^{n} w_k = 1$. Thus, it is a typical COP and can be solved

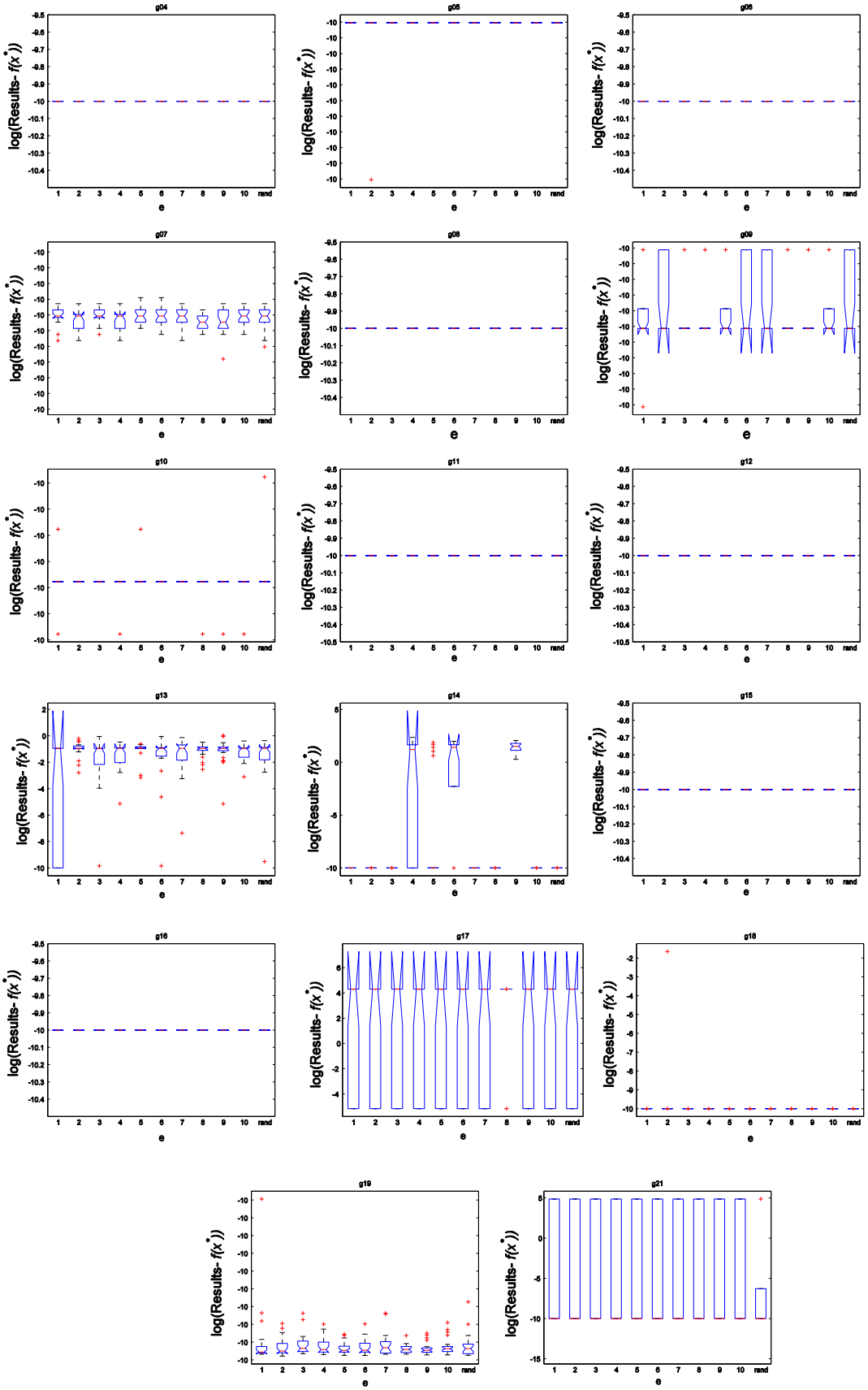

Fig. 3 *e*-DE with different parameters on g04–g19, g21

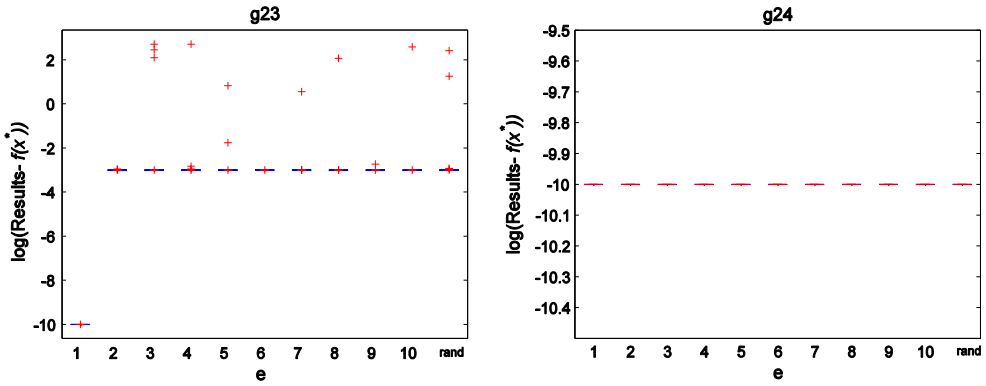

Fig. 4 *e*-DE with different parameters on g23 and g24

Table 11 Comparison of *e*-DE with *e*-DE-1 and *e*-DE-2

Function		*e*-DE	*e*-DE-1	*e*-DE-2
g14	Mean	8.51E−12	4.51E+0	8.51E−12
	Std	0	2.50E+0	0
	Success rate	100%	0	100%
g17	Mean	3.55E+1	5.25E+1	4.74E+1
	Std	1.43E+3	3.40E+1	1.32E+3
	Success rate	44%	0%	0%
g19	Mean	5.01E−11	9.58E−2	4.63E−11
	Std	6.41E−23	9.54E−2	2.39E−29
	Success rate	100%	0%	100%
g21	Mean	4.19E+1	7.33E+1	2.67E+1
	Std	3.89E+3	4.40E+3	2.45E+4
	Success rate	72%	44%	0%

by the proposed algorithm. For example,

$$A = \begin{bmatrix} 1 & 3 & 7 \\ 1/3 & 1 & 5 \\ 1/7 & 1/5 & 1 \end{bmatrix}$$

The min *CIF* is as follows:

$$\min CIF(w) = \frac{\sum_{i=1}^{3} \left| \sum_{k=1}^{3} (A(k,i) \times w_k) - n w_i \right|}{n}. \tag{19}$$

The *CIF* can be solved by the proposed algorithm, and convergence graph is presented in Fig. 5. To verify the rational results, both weights from the proposed algorithm and AHP theory are presented in Table 13, respectively. It can be observed that the weights are consistent with the AHP theory. Both weights have little difference. The results have indicated that the proposed algorithm has the ability to solve real-application COPs.

Table 12 Standardized comparison scale of nine levels

Definition	Value
Equal importance	1
Weak importance	3
Essential importance	5
Demonstrated importance	7
Extreme importance	9
Intermediate values	2, 4, 6, 8

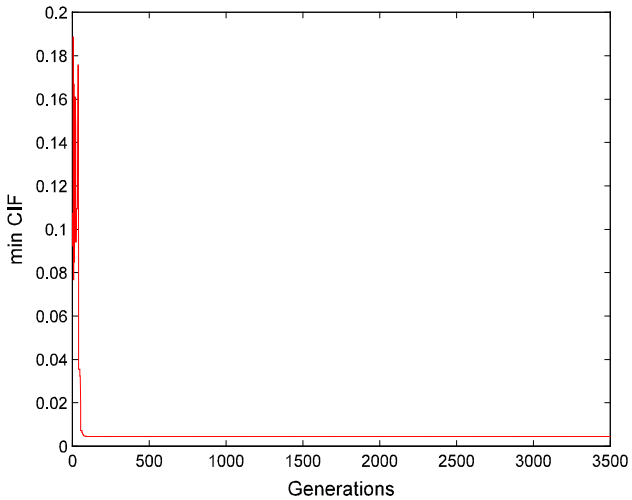

Fig. 5 Convergence graph for min *CIF*

Table 13 The weights comparison between algorithm and AHP theory

Method	W_1	W_2	W_3
Algorithm	0.6429	0.2831	0.0740
AHP theory	0.6433	0.2828	0.0739

6 Conclusions

An extended effective differential evolution algorithm is designed, named e-DE. The framework of e-DE is the same as the conventional DE. It is mainly composed of seven criteria and two offspring generation approaches. The seven criteria are used to compare the feasible solutions over infeasible solutions, which is quite different from traditional research. The two kinds of offspring generation approaches combine rand/1/bin, current-to-best/1, and current-to-rand/1. The experimental results based on CEC2006 have revealed the effectiveness of the proposed algorithm. The proposed algorithm can reach 100% feasible rate. In addition, compare to some state-of-the-art optimization algorithms, the algorithm is competitive. Besides, the proposed algorithm is used to solve the weights in AHP. The results are consistent with the weights from AHP theory.

In the following studies, we will use the algorithm to solve emergency management optimization problems, in which a lot of COPs are involved.

Acknowledgements The authors would like to thank the associate editor and the anonymous reviewers for their very helpful suggestions. This study was funded by China Natural Science Foundation (Grant Numbers 71503134, 91546117, 71373131, 71171116), Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD), Key Project of National Social and Scientific Fund Program (16ZDA047), Research Center for Prospering Jiangsu Province with Talents (Grant Number skrc201400-14), Natural Science Foundation of Higher Education of Jiangsu Province of China (16KJB120003), Philosophy and Social Sciences in Universities of Jiangsu (Grant Number 2016SJB630016), and Research Institute for History of Science and Technology (2017KJSKT005).

Compliance with ethical standards

Conflict of Interest The authors declare that they have no conflict of interest.

References

Ali MM, Kajee-Bagdadi Z (2009) A local exploration-based differential evolution algorithm for constrained global optimization. Appl Math Comput 208(1):31–48

Arora M, Kohliand S (2017) Chaotic grey wolf optimization algorithm for constrained optimization problems. J Comput Des Eng. https://doi.org/10.1016/j.jcde.2017.02.005

Asafuddoula M, Ray T, Sarker R (2015) An improved self-adaptive constraint sequencing approach for constrained optimization problems. Appl Math Comput 253:23–39

Babu BV, Jehan MML (2003) Differential evolution for multi-objective optimization. In: Proceedings of IEEE congress on evolutionary computing, pp 2696–2703

Barbosa HJC, Lemonge ACC (2003) A new adaptive penalty scheme for genetic algorithms. Inf Sci 156(3–4):215–251

Brajevic I (2015) Crossover-based artificial bee colony algorithm for constrained optimization problems. Neural Comput Appl 26(7):1587–1601

Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evolut Comput 10(2):646–657

Cai Z, Wang Y (2006) A multiobjective optimization-based evolutionary algorithm for constrained optimization. IEEE Trans Evolut Comput 10(6):658–675

Chuang Y-C, Chen C-T, Hwang C (2016) A simple and efficient real-coded genetic algorithm for constrained optimization. Appl Soft Comput 38:87–105

Coit DW, Smith AE, Tate DM (1996) Adaptive penalty methods for genetic optimization of constrained combinatorial problems. INFORMS J Comput 8(2):173–182

Deb K (1995) Optimization for engineering design: algorithms and examples. Prentice-Hall, New Delhi

Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(2–4):311–338

Elsayed SM, Sarker RA, Essam DL (2012) On an evolutionary approach for constrained optimization problem solving. Appl Soft Comput 12(10):3208–3227

Elsayed SM, Sarker RA, Essam DL (2014) A self-adaptive combined strategies algorithm for constrained optimization using differential evolution. Appl Math Comput 241:267–282

Fan QQ, Yan XF (2016) Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies. IEEE Trans Cybernet 46(1):219–232

Fan Q, Yan X, Xue Y (2016) Prior knowledge guided differential evolution. Soft Comput 21(22):6841–6858

Gamperle R, Muller SD, Koumoutsakos P (2002) A parameter study for differential evolution. In: Proceedings of advances intelligent system, fuzzy system, evolutionary computing, Crete, Greece, pp 293–298

Gao WF, Yen GG, Liu SY (2015) A dual-population differential evolution with coevolution for constrained optimization. IEEE Trans Cybernet 45(5):1108–1121

Garcia-Martinez C, Lozano M, Herrera F, Molina D, Sanchez AM (2008) Global and local real-coded genetic algorithms based on parent-centric crossover operators. Eur J Oper Res 185(3):1088–1113

Garg H (2016) A hybrid PSO-GA algorithm for constrained optimization problems. Appl Math Comput 274:292–305

Ghasemishabankareh B, Li X, Ozlen M (2016) Cooperative coevolutionary differential evolution with improved augmented Lagrangian to solve constrained optimisation problems. Inf Sci 369:441–456

Gong WY, Cai ZH, Liang DW (2014a) Engineering optimization by means of an improved constrained differential evolution. Comput Methods Appl Mech Eng 268:884–904

Gong ZW, Chen CQ, Ge XM (2014b) Risk prediction of low temperature in Nanjing city based on grey weighted Markov model. Natural Hazards 71:1159–1180

Gong W, Cai Z, Liang D (2015) Adaptive ranking mutation operator based differential evolution for constrained optimization. IEEE Trans Cybernet 45(4):716–727

Hansen N, Ostermeier A (2001) Completely derandomized self adaptation in evolution strategies. Evolut Comput 9(2):159–195

Homaifar A, Qi C, Lai S (1994) Constrained optimization via genetic algorithms. Simulation 62(41):242–254

Huang VL, Qin AK, Suganthan PN (2006) Self-adaptive differential evolution algorithm for constrained real-parameter optimization. In: Proceedings of the congress on evolutionary computation (CEC'2006), Vancouver, BC, pp 17–24

Iorio A, Li X (2004) Solving rotated multi-objective optimization problems using differential evolution. In: Australian conference on artificial intelligence, Cairns, Australia, pp 861–872

Jia G, Wang Y, Cai Z, Jin Y (2013) An improved ($\mu+\lambda$)-constrained differential evolution for constrained optimization. Inf Sci 222:302–322

Joines JA, Houck CR (1994) On the use of nonstationary penalty functions to solve nonlinear constrained optimization problems with GA's. In: Proceedings of 1st IEEE conference on evolutionary computation, Orlando, FL, pp 579–584

Karaboga D, Akay B (2011) A modified artificial bee colony (ABC) algorithm for constrained optimization problems. Appl Soft Comput 11(3):3021–3031

Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of the 995 EEE international conference on neural networks, Perth, Australia, pp 1942–1948

Li X, Yin M (2014) Self-adaptive constrained artificial bee colony for constrained numerical optimization. Neural Comput Appl 24(3):723–734

Liang JJ, Suganthan PN (2006) Dynamic multi-swarm particle swarm optimizer with a novel constraint-handling mechanism. In: Proceedings of the congress on evolutionary computation (CEC'2006), Vancouver, BC, Canada, pp 9–16

Liang JJ, Qin AK, Suganthan PN, Baskar S (2006a) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evolut Comput 10(3):281–294

Liang JJ, Runarsson TP, Mezura-Montes E, Clerc M, Suganthan PN, Coello CAC, Deb K (2006b) Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. Technical report, Nanyang Technological University, Singapore

Liang Y, Wan Z, Fang D (2015) An improved artificial bee colony algorithm for solving constrained optimization problems. Int J Mach Learn Cybernet. https://doi.org/10.1007/s13042-015-0357-2

Lin C-H (2013) A rough penalty genetic algorithm for constrained optimization. Inf Sci 241:119–137

Lin HB, Fan Z, Cai XY, Li W, Wang S, Li J, Zhang CD (2014) Hybridizing infeasibility driven and constrained domination principle with moea/d for constrained multiobjective evolutionary optimization. In: TAAI, LNAI 8916, pp 249–261

Long Q (2014) A constraint handling technique for constrained multiobjective genetic algorithm. Swarm Evolut Comput 15:66–79

Long W, Liang X, Cai S (2016) A modified augmented Lagrangian with improved grey wolf optimization to constrained optimization problems. Neural Comput Appl. https://doi.org/10.1007/s00521-016-2357-x

Mahdavi A, Shiri ME (2015) An augmented Lagrangian ant colony based method for constrained optimization. Comput Optim Appl 60(1):263–276

Mallipeddi R, Suganthan PN (2010) Ensemble of constraint handling techniques. IEEE Trans Evolut Comput 14(4):561–579

Mallipeddi R, Suganthan PN, Pan QK, Tasgetiren MF (2011) Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl Soft Comput 11(2):1679–1696

Mezura-Montes E, Coello CAC (2004) An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J Gen Syst 37(4):443–473

Mezura-Montes E, Coello CAC (2005) A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans Evolut Comput 9(1):1–17

Mezura-Montes E, Coello CAC (2011) Constraint-handling in nature inspired numerical optimization: past, present and future. Swarm Evolut Comput 1:173–194

Mezura-Montes E, Velazquez-Reyes J, Coello CA Coello (2006) Modified differential evolution for constrained optimization. In: Proceedings of the congress on evolutionary computation (CEC'2006), Vancouver, BC, pp 25–32

Michalewicz Z (1995) A survey of constraint handling techniques in evolutionary computation methods, In: Proceedings of 4th annual conference on evolutionary programming, pp 135–155

Mohamed AW (2017) Solving large-scale global optimization problems using enhanced adaptive differential evolution algorithm. Complex Intell Syst. https://doi.org/10.1007/s40747-017-0041-0

Pahner U, Hameyer K (2000) Adaptive coupling of differential evolution and multiquadrics approximation for the tuning of the optimization process. IEEE Trans Magn 36(4):1047–1051

Price KV, Storn RM, Lampinen JA (2005) Differential evolution: a practical approach to global optimization. Springer, Berlin, Heidelberg

Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evolut Comput 13(2):398–417

Qu BY, Suganthan PN (2011) Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods. Eng Optim 43(4):403–416

Reklaitis GV, Ravindran A, Ragsdell KM (1983) Engineering optimization methods and applications. Wiley, New York

Richardson JT, Palmer MR, Liepins G, Hilliard M (1989) Some guidelines for genetic algorithms with penalty functions. In: Scha er JD (ed) Proceedings of the third international conference on genetic algorithms, Morgan Kau man, San Mateo, pp. 191–197

Runarsson TP, Yao X (2000) Stochastic ranking for constrained evolutionary optimization. IEEE Trans Evolut Comput 4:284–294

SaatyThe TL (1980) The analytic hierarchy process. McGraw-Hill, New York

Sharma H, Bansal JC, Arya KV (2012) Fitness based differential evolution. Memet Comput 4(4):303–316

Singh HK, Isaacs A, Ray T, Smith W (2008) Infeasibility driven evolutionary algorithm (IDEA) for engineering design optimization. In: Wobcke W, Zhang M (eds) AI 2008, LNAI 5360, pp 104–115

Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359

Sun C, Zeng J, Pan J (2011) An improved vector particle swarm optimization for constrained optimization problems. Inf Sci 181(6):1153–1163

Sun JY, Garibaldi JM, Zhang YQ, Al-Shawabkeh A (2016) A multicycled sequential memetic computing approach for constrained optimisation. Inf Sci 340–341:175–190

Takahama T, Sakai S (2006) Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites. In: Proceedings of IEEE congress on evolutionary computation, Vancouver, BC, Canada, pp 1–8

Takahama T, Sakai S, Iwane N (2005) Constrained optimization by the epsilon constrained hybrid algorithm of particle swarm optimization and genetic algorithm. In: AI 2005: Advances artificial intelligence, lecture notes in artificial intelligence, vol 3809. Springer, pp 389–400

Tasgetiren M, Suganthan P, Pan Q, Mallipeddi R, Sarman S (2010) An ensemble of differential evolution algorithm for constrained function optimization. In: 2010 Congress on evolutionary computation, IEEE Service Center, Barcelona, Spain, pp 967–975

Tessema B, Yen GG (2009) An adaptive penalty formulation for constrained evolutionary optimization. IEEE Trans Syst Man Cybernet Part A Syst Hum 39(3):565–578

Tessema B, Yen GG (2006) A self-adaptive penalty function based algorithm for constrained optimization. In: Proceedings of IEEE congress on evolutionary computation, Vancouver, BC, Canada, pp 246–253

Venkatraman S, Yen GG (2005) A generic framework for constrained optimization using genetic algorithms. IEEE Trans Evolut Comput 9(4):424–435

Wang Y, Cai ZX (2011) Constrained evolutionary optimization by means of $(\mu + \lambda)$-differential evolution and improved adaptive trade-off model. Evolut Comput 19(2):249–285

Wang Y, Cai Z (2012a) Combining multiobjective optimization with differential evolution to solve constrained optimization problems. IEEE Trans Evolut Comput 16(1):117–134

Wang Y, Cai Z (2012b) A dynamic hybrid framework for constrained evolutionary optimization. IEEE Trans Syst Man Cybernet Part B Cybernet 42(1):203–217

Wang Y, Cai Z (2012c) Combining multi-objective optimization with differential evolution to solve constrained optimization problems. IEEE Trans Evolut Comput 16(1):117–134

Wang Y, Cai Z, Guo G, Zhou Y (2007) Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems. IEEE Trans Syst Man Cybernet 37:560–575

Wang Y, Wang BC, Li HX, Yen Gary G (2016) Incorporating objective function information into the feasibility rule for constrained evolutionary optimization. IEEE Trans Cybernet 46:2938–2952

Wei L, Chen Z, Li J (2011) Evolution strategies based adaptive Lp LS-SVM. Inf Sci 181(14):3000–3016

Wu X, Wang Y, Yang L, Song S, Wei G, Guo J (2016) Impact of political dispute on international trade based on an international trade inoperability input-output model. J Int Trade Econ Dev 25(1): 1–24

Xiaobing Y, Guo S, Guo J, Huang X (2011) Rank B2C e-commerce websites in e-alliance based on AHP and fuzzy TOPSIS. Expert Syst Appl 38(4):3550–3557

Yi P, Hong YG, Liu F (2015) Distributed gradient algorithm for constrained optimization with application to load sharing in power systems. Syst Control Lett 83:45–52

Yi WC, Li XY, Gao L, Zhou YZ, Huang JD (2016) ε constrained differential evolution with pre-estimated comparison using gradient-based approximation for constrained optimization problems. Expert Syst Appl 44:37–49

Yong W, Zixing C, Yuren Z, Wei Z (2008) An adaptive tradeoff model for constrained evolutionary optimization. IEEE Trans Evolut Comput 12(1):80–92

Yu ZS (2008) Solving bound constrained optimization via a new nonmonotone spectral projected gradient method. Appl Numer Math 58(9):1340–1348

Yu XB, Wang XM, Cao J, Cai M (2015a) An ensemble differential evolution for numerical opimization. Int J Inf Technol Decis Mak 14(4):915–942

Yu XB, Cai M, Cao J (2015b) A novel mutation differential evolution for global optimization. J Intell Fuzzy Syst 28(3):1047–1060

Yu KJ, Wang X, Wang ZL (2016) Constrained optimization based on improved teaching-learning-based optimization algorithm, Inf Sci 352–353:61–78

Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evolut Comput 13(5):945–958

Zhang X, Zhang X (2016) Improving differential evolution by differential vector archive and hybrid repair method for global optimization. Soft Comput 21(23):7107–7116

Zhang CJ, Lin Q, Gao L, Li XY (2015) Backtracking Search Algorithm with three constraint handling methods for constrained optimization problems. Expert Syst Appl 42(21):7831–7845

Zheng JG, Wang X, Liu RH (2012) ε-Differential evolution algorithm for constrained optimization problems. J Softw 23(9):2374–2387

Zhu DT (2007) An affine scaling reduced preconditional conjugate gradient path method for linear constrained optimization. Appl Math Comput 184(2):181–198