A Combination of Specialized Differential Evolution Variants for Constrained Optimization

Luis Alfredo-Gordián-Rivera and Efrén Mezura-Montes

Laboratorio Nacional de Informática Avanzada (LANIA) A.C.
Rébsamen 80, Centro, Xalapa, Veracruz, 91000, MÉXICO
iie.alfredogordian@gmail.com, emezura@lania.mx

Abstract. A novel approach based on three differential evolution variants to solve numerical constrained optimization problems is presented. Each variant competes to get more vectors for reproduction from the population. Such competition is based on two performance measures for convergence and solution improvement. Two of the variants adopted in this work were precisely proposed to deal with constrained search spaces. Two experiments are carried out: one to analyze the behavior of each variant with respect to the features of the problem solved and another to compare the performance of the proposed approach with respect to state-of-the-art multi-operator algorithms. The results obtained show that the specialized variants are more useful in the search, either combined or just using one of them. Finally, the final results of the proposed approach were highly competitive, and better in some cases, with respect to those of the algorithms used in the comparison.

Keywords: Evolutionary Algorithms, Differential Evolution, Constrained Optimization

1 Introduction

Nowadays, the usage of metaheuristic algorithms to solve complex optimization problems has become very popular. Among such algorithms, evolutionary computing has been successfully applied, particularly, in numerical optimization problems [1]. However, evolutionary algorithms (EAs) in their original versions were designed to deal with unconstrained search spaces. On the other hand, several optimization problems have constraints in their definitions. This has motivated the design of techniques to incorporate feasibility information within an EA to sample constrained search spaces [2].

This work precisely focuses on the constrained numerical optimization problem (CNOP), which, without loss of generality can be formulated as follows:
Find $\boldsymbol{x}$ which optimizes $f(\boldsymbol{x})$
subject to: $g_j(\boldsymbol{x}) \leq 0, j = 1, \ldots, m$ and $h_k(\boldsymbol{x}) = 0, k = 1, \ldots, p$
where $\boldsymbol{x} \in \mathbb{R}^n$ is the vector of solutions $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$ and each x_i, $i = 1, \ldots, n$ is limited by lower and upper limits $L_i \leq x_i \leq U_i$. If the feasible region is $\mathcal{F}$ and the whole search space is $\mathcal{S}$, then $\mathcal{F} \subseteq \mathcal{S}$. For an inequality

constraint that satisfies $g_j(\boldsymbol{x}) = 0$, the constraint is active at $\boldsymbol{x}$. All equality constraints are active at all solutions of $\mathcal{F}$. To handle equality constraints they are transformed into inequality constraints as follows $|h_k(\boldsymbol{x})| - \epsilon \leq 0$, where ϵ is the tolerance value allowed(a very small value).

It is well-known by the No Free Lunch Theorems [3] that there is not an algorithm which ensures to outperform other algorithms in all search problems, i.e. an algorithm will not present the same good performance in all problem classes.

This feature has motivated the development of new ways to improve the performance of an algorithm in a wider set of problems. Regarding CNOPs, there are proposals based on multiple operators within the same algorithm [4, 5] and even multiple algorithms with multiple operators in a single approach [6]. However, based on a recent review of the state-of-the-art such efforts in constrained numerical optimization are still scarce [2]. Moreover, to the best of the authors' knowledge, the aforementioned proposals mostly report final results, while the behavior of the multiple operators has been overlooked. Furthermore, in those multiple-operator EAs for constrained optimization, no operators adapted for CNOPs have been considered.

On the other hand, among those EAs usually adopted to solve CNOPs, differential evolution (DE) has been the most preferred due to its better performance in such type of optimization problems [2].

All the aforementioned is the main motivation of this research, where three DE variants, two of them specifically adapted to solve CNOPs, are incorporated into a single algorithm. Two performance measures are used as criteria to adaptively give those more successful variants more chances to generate offspring. Therefore, a competition of operators is promoted and analyzed. The aim is, besides giving empirical evidence of the performance of the proposed approach, analyzing the behavior of the approach so as to detect which DE variants are preferred depending of the features of the CNOP being solved.

This paper is organized as follows. Section 2 details the DE variants adopted. Section 3 describes the adaptive mechanism to control those DE variants. The experimental design and results are presented in Section 4. Finally, some conclusions and the future work are shown in Section 5.

2 DE variants adopted

DE works with a population P with NP vectors, i.e, potential solutions of an optimization problem [7]. The population evolves by using two variation operators: recombination (usually discrete) and differential mutation based on differences between vectors chosen at random from P. At each iteration t, each vector $\boldsymbol{x}_{i,t}$ in $P(t)$, called parent vector at this time, uses the differential mutation to generate a mutation vector $\boldsymbol{v}_{i,t}$. After that, by using the recombination operator between the parent and mutation vectors, an offspring vector $\boldsymbol{u}_{i,t}$ is generated (see Eq. 1).

$$u_{j,i,t} = \begin{cases} v_{j,i,t} \text{ if } rand_j[0,1) \leq CR \text{ or } j = j_{rand} \\ x_{j,i,t} \text{ otherwise} \end{cases} \tag{1}$$

where $rand_j[0,1)$ is a uniform random number distributed between 0 and 1, j_{rand} is randomly chosen index from $\{1,2,\ldots,n\}$, and $0 \leq CR \leq 1$ determines the similarity of the offspring with respect to the mutation vector. Such offspring then competes with its parent $\boldsymbol{x}_{i,t}$ in a greedy selection to choose the one that will be part of the next population $P(t+1)$.

Three differential mutation variants are mixed in a competition scheme within the same algorithm to solve CNOPs in this current work. They are the following:

1. **Original mutation operator**: This mutation operator is used in the most popular DE variant called DE/rand/1/bin [7], where the mutation vector $\boldsymbol{v}_{i,t}$ is generated by three vectors chosen at random. The scaled difference (by a user-defined parameter F) between two of them ($\boldsymbol{x}_{r1,t}$ and $\boldsymbol{x}_{r2,t}$) defines the search direction and the third one, $\boldsymbol{x}_{r3,t}$, is used as a base vector (see Eq. 2).

$$\boldsymbol{v}_{i,t} = \boldsymbol{x}_{r3,t} + F(\boldsymbol{x}_{r1,t} - \boldsymbol{x}_{r2,t}) \tag{2}$$

2. **Modified differential evolution (MDE)**: Proposed in [8] to deal with CNOPs, where two differences are computed: (1) between the best vector in the current population $\boldsymbol{x}_{best,t}$ and a randomly chosen vector $\boldsymbol{x}_{r2,t}$, and (2) between the parent vector $\boldsymbol{x}_{i,t}$ and another randomly chosen vector $\boldsymbol{x}_{r1,t}$. Both differences are scaled by two different factors F_α and F_β, respectively. Finally, a third randomly chosen vector $\boldsymbol{x}_{r3,t}$ is used as a base vector (see Eq. 3).

$$\boldsymbol{v}_{i,t} = \boldsymbol{x}_{r_3,t} + F_\alpha(\boldsymbol{x}_{best,t} - \boldsymbol{x}_{r2,t}) + F_\beta(\boldsymbol{x}_{i,t} - \boldsymbol{x}_{r1,t}) \tag{3}$$

   It is important to note that MDE generates n_o offspring per parent vector and the best one is the one which competes against its parent.
3. **Adaptative Differential Evolution (ADE)**: Proposed in [9] to solve CNOPs and based on a multi-parent schema where, for each individual $\boldsymbol{x}_{i,t}$ in $P(t)$, the mutation vector $\boldsymbol{v}_{i,t}$ is generated as in Eq. 4:

$$\boldsymbol{v}_{i,t} = \boldsymbol{x}_{i,t} + \sum_{k=1}^{K} w_k * (\boldsymbol{x}_{r_k,t} - \boldsymbol{x}_{r_{k+1},t}) \tag{4}$$

   where, $r_1, r_2, \ldots, r_K \in \{1,2,\ldots,NP\}$, are K integers different among each other and different from i (the index of the parent vector), and $\boldsymbol{x}_{r_{K+1},t} = \boldsymbol{x}_{r_1,t}$. The weighted values w_k are calculated as indicated in Eq. 5:

$$\xi = randn(1,K), \boldsymbol{w} = \xi / sum(\xi) \tag{5}$$

   where, $randn(1,K)$ is a vector with K normally distributed random numbers, $sum(\xi)$ is the sum of all components of the vector ξ, and $\boldsymbol{w} = [w_1,\ldots,w_K]$. Eqs. 4 and 5 are used K times to generate K vectors by recombination. The best of them is chosen as the offspring to compete against its parent.

3 Proposed approach

As it was mentioned in Section 1, the three DE mutation variants will compete to generate offspring in a single population at each generation. Such competition will be based on two performance measures, one for measuring convergence and another for measuring the improvement of the best solution at each generation:

- **Convergence difference (CD)**: it is based on the convergence measure P_m proposed in [10], where the Euclidean distance between the center of the population and the individual farthest from it is computed. The center of the population $\boldsymbol{O_p}$ is calculated as the average position of all individuals in P as detailed in Eq. 6:

$$\boldsymbol{O_p} = \frac{\sum_{i=1}^{NP} \boldsymbol{x}_i}{NP} \tag{6}$$

  Then, P_m is defined in Eq. 7

$$P_m = max \parallel \boldsymbol{x}_i - O_p \parallel_E, i = 1, ..., NP. \tag{7}$$

  Thus, the convergence difference of two populations is presented in Eq. 8:

$$CD = P_m(P(t-1)) - P_m(P(t)) \tag{8}$$

  Where $P(t-1)$ is the population before the application of the DE variant and $P(t)$ is the population after the application of the DE variant.
- **Progress ratio (PR)**: originally proposed in [11] to measure improvement within the feasible region of the search space. In this work this measure is slightly modified in such a way that it is used to evaluate the ability on an algorithm to improve its best feasible solution at each generation as indicated in Eq. 9:

$$PR = \left| \ln \sqrt{\frac{f_{min}(t-1)}{f_{min}(t)}} \right| \tag{9}$$

  where $f_{min}(t-1)$ is the best feasible objective function value in $P(t-1)$ and $f_{min}(t)$ is the best feasible objective function value in $P(t)$.

The proposed algorithm works in a similar way as DE does it. However, unlike traditional DE where only one DE variant is used to generate the offspring of each vector in the population, the three variants described in Section 2 compete to get more vectors to reproduce. The initial population $P(0)$ of NP vectors is divided in three sub-populations ($P_1(0)$, $P_2(0)$, and $P_3(0)$) of equal size ($s_1(0) = s_2(0) = s_3(0)$). Each sub-population is assigned to each one of the three DE variants. Moreover, each DE variant is assigned a value which determines its strength. The initial strength value for each DE variant (at generation 0) is 1 ($ST_{v1}(0) = ST_{v2}(0) = ST_{v3}(0) = 1$) and if for some reason it falls below 1 it is reset to 1.

Each DE variant then generates the offspring for each vector in its sub-population. After all offspring are generated by each DE variant in its corresponding sub-population, the two performance measures are computed for all three sub-populations, considering in each case the current sub-population $P_{sp}(t)$ and the set of offspring recently generated $P^{'}_{sp}(t+1)$, $sp = 1, 2, 3$.

The process to update the strength values is through comparisons among the values of the performance measures obtained by each DE variant as follows: (1) if one DE variant outperforms other variant in both performance measures, a value of 1 is added to the strength of the best one and a value of 1 is subtracted to the strength of the outperformed DE variant, (2) if no DE variant is better than the other two, a value of 1 is added to the strength value of all of them, (3) if two DE variants have a tie in one measure value, the value of the remaining measure breaks the tie.

Based on the strength values updated at each generation of the approach, the size of each sub-population is also updated accordingly by using a so-called *portion* value for each sub-population, which is calculated in Eq. 10:

$$portion_i(t+1) = \frac{ST_{vi}(t)}{\sum_{j=1}^{3} ST_{vj}(t)}, i = 1, 2, 3 \tag{10}$$

Thereby, for each DE variant its sub-population size for the next generation can be calculated as indicated in Eq. 11:

$$s_i(t+1) = s_i(t) * portion_i(t+1); i = 1, 2, 3 \tag{11}$$

If $s_i(t+1) < 5$, vectors from other sub-populations are taken to maintain a minimum size of 5 vectors. At the end of the generation, the population is randomly merged to allow each DE variant to work with different vectors at each generation.

The constraint-handling technique adopted in this work is the set of feasibility rules proposed by Deb [12]. They are used as criteria to choose between the parent vector and its child as follows:

- Between 2 feasible vectors, the one with the highest fitness value is preferred.
- If one vector is feasible and the other one is infeasible, the feasible vector is preferred.
- If both vectors are infeasible, the one with the lowest sum of constraint violation $(\sum_{j=1}^{m} max(0, g_j(\boldsymbol{x})) + \sum_{k=1}^{p} max(0, |h_k(\boldsymbol{x}) - \epsilon|))$ is preferred.

The complete details of the proposed approach are in Algorithm 1.

4 Experiments and results

To evaluate the performance of the proposed approach two experiments were designed: (1) an analysis of the behavior of each DE variant depending of two features of the CNOP solved, and (2) a comparison of final results with respect to state-of-the-art EAs, two with multiple operators to solve CNOPs; they are

Algorithm 1 Pseudocode of the proposed approach

Require: number of iterations CH, population size NP

Generate $P(0)$ of size NP at random and evaluate each vector in $P(0)$
Divide $P(0)$ in three sub-populations $P_1(0)$, $P_2(0)$, and $P_3(0)$ with equal size $s_1(0) = s_2(0) = s_3(0)$
$ST_{v1}(0) = ST_{v2}(0) = ST_{v3}(0) = 1$
$t = 0$
while $t \leq CH$ **do**
  Apply DE variants ($v1$, $v2$, and $v3$) to their corresponding sub-population $P_1(t)$, $P_2(t)$, and $P_3(t)$ to generate offspring $P_1^{'}(t+1)$, $P_2^{'}(t+1)$, and $P_3^{'}(t+1)$ and evaluate each new vector generated
  Calculate CD (Eq. 8) and PR (Eq. 9) for each sub-population sp based on $P_{sp}(t)$ and $P_{sp}^{'}(t+1)$, $sp = 1, 2, 3$
  Update the strength values for each DE variant $ST_{v1}(t)$, $ST_{v2}(t)$, and $ST_{v3}(t)$
  Calculate $portion_i(t+1)$, $i = 1, 2, 3$ (Eq.10)
  Update the sub-population sizes $s_i(t+1)$, $i = 1, 2, 3$ (Eq.11)
  Generate the population for the next generation $P(t+1)$ by selecting, based on Deb's rules, the best vector between parent and child for each sub-population $P_{sp}(t)$ and $P_{sp}^{'}(t+1)$, $sp = 1, 2, 3$
  Shuffle the new population $P(t+1)$
  Distribute the vectors in $P(t+1)$ based on the updated sizes for the sub-populations $s_i(t+1)$, $i = 1, 2, 3$, to each sub-population $P_{sp}(t+1)$, $sp = 1, 2, 3$
  $t = t + 1$
end while

the self-adaptive multi-operator genetic algorithm (SAMO-GA) and the self-adaptive multi-operator differential evolution (SAMO-DE) [6] and finally the elitist teaching-learning-based optimization (TLBO) algorithm [13]. Thirteen well-known test problems with different features were used in both experiments and their descriptions can be found in [6]. A summary of their main features is presented in Table 1.

The proposed approach used the following parameter values: NP=21, CH = 5000 (260,000 evaluations were carried out per independent run). For each DE variant the parameter values, as suggested in their references, were the following for DE: $CR = 0.7$ and $F = 0.9$, ADE: $CR = 0.5$, $K = 3$, MDE: $CR = 0.7$, $F_\alpha = 0.9$, $F_\beta = 0.4$, $N_o = 3$

For the first experiment, the average sub-population sizes per iteration on 30 independent runs for representative test problems were plotted. The aim is to analyze the adaptive behavior the proposed approach has depending on the features of the CNOP solved.

Figure 1 shows the sub-population sizes for problems with different estimated sizes of the feasible region. As it can be seen, for large feasible regions, even if the three variants start with different sizes (i.e., different strengths), they tend to converge to a similar value before the first half of the search. The opposite is observed for small feasible regions (problem g10) where one DE variant (ADE in this case) dominates the other two since the beginning. A behavior “in the mid-

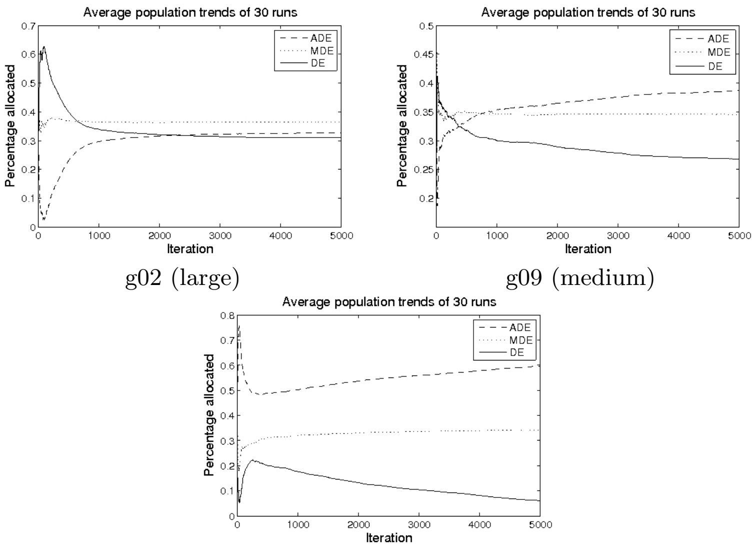
Table 1. Features of test problems. "n": dimensions, "LI", "NI", "LE", "NE": linear inequality, nonlinear inequality, linear equality and nonlinear equality constraints, respectively, "a": active constraints, and "ρ": estimated feasible region size

Function	n	Type of function	ρ	LI	NI	LE	NE	a
g01	13	quadratic	0.0003%	9	0	0	0	6
g02	20	nonlinear	99.9973%	0	2	0	0	1
g03	10	nonlinear	0.0026%	0	0	0	1	1
g04	5	quadratic	27.0079%	0	6	0	0	2
g05	4	nonlinear	0.0000%	2	0	0	3	3
g06	2	nonlinear	0.0057%	0	2	0	0	2
g07	10	quadratic	0.0000%	3	5	0	0	6
g08	2	nonlinear	0.8581%	0	2	0	0	0
g09	7	nonlinear	0.5199%	0	4	0	0	2
g10	8	linear	0.0020%	3	3	0	0	6
g11	2	quadratic	0.0973%	0	0	0	1	1
g12	3	quadratic	4.7697%	0	1	0	0	0
g13	5	nonlinear	0.0000%	0	0	0	3	3

dle" can be observed for medium sized feasible regions, where ADE dominates, but its dominance is lower with respect to MDE. In all three cases, the original DE was the least used variant.

Figure 2 presents the sub-population sizes for problems with different dimensionalities. For a high dimensionality test problem the two specialized DE variants to solve CNOPs, have the same strength after iteration 3000 while the traditional DE variants is almost unused. For a test problem with a medium dimensionality an interesting difference in strengths is observed between 1000 and 2000 iterations. After that, the three DE variants are used with almost the same sub-populations sizes. For a low dimensionality problem ADE is almost the only one used after 2000 iterations.

The comparison of the second experiment is summarized in Table 2. 95%-confidence two-sample t-tests were applied to the results in order to find significant differences among the algorithms. In the last column in Table 2 "$\surd$" and "X" mean significant and no significant difference, respectively, between our approach and the three compared algorithms (SAMO-GA, SAMO-DE and TLBO). Our approach was able to find the same results with respect to the compared algorithms in seven test problems (g01, g04, g06, g08, g09, g11 and g12). Moreover, the t-test results confirmed that our approach outperformed the other three algorithms in test problem g10, while outperforming SAMO-GA and TLBO in problem g05 (no significant difference was obtained with SAMO-DE). In problem g07, based on the t-test results, our approach outperformed SAMO-GA and no significant difference was observed with respect to SAMO-DE and TLBO. In problem g13, our approach outperformed TLBO but was outperformed by SAMO-GA and SAMO-DE. Finally, our approach was outperformed in problems g02 and g03 (problems with a nonlinear objective function, a high-dimensionality and nonlinear constraints). The overall results of the comparison against state-of-the-art approaches suggest a competitive performance of our approach and even better with respect to other multi-operator EAs as in problems g07 and g10.



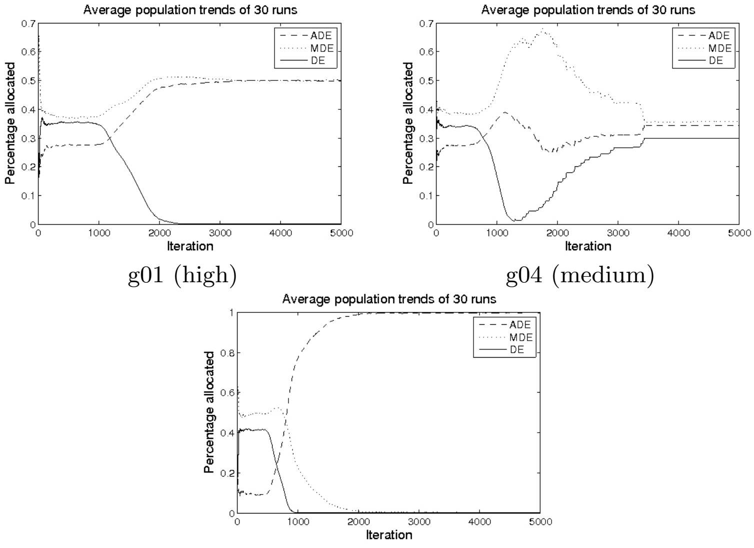
g02 (large) g09 (medium)

g10 (small)

Fig. 1. Average sub-population size per DE variant for representative test problems based on estimated size of the feasible region.

5 Conclusions and future work

This paper has presented the combination of three differential evolution variants, two of them designed for constrained search spaces, to solve CNOPs. Two performance measures, one for convergence and another to measure the improvement of the best solution so far, where used as criteria to assign more (or less) vectors to each variant. Two experiments were carried out: (1) one to analyze the behavior of each DE variant depending of the features of the CNOP and (2) another to compare the final results of the proposed approach with respect to three state-of-the-art EAs (two of them multi-operator-based). The results of the first experiment showed that for large or medium size feasible regions or medium dimensionalities, the three DE variants are used with a similar frequency, being the traditional DE/rand/1/bin the least used. For a small feasible region or a low dimensionality test problem, one DE variant (ADE) clearly dominated the other two. For a high-dimensionality test problem the two specialized DE variants to solve CNOPs (ADE and MDE) were clearly preferred. The results of the second experiment suggest a competitive performance of the proposed approach with respect to state-of-the-art algorithms, mostly in problems with a combination of linear and nonlinear inequality constraints (g07 and g10). The next paths of research include adding new performance measures, using new DE variants, and solving more benchmark and mechanical design problems.



g01 (high) g04 (medium)

g06 (low)

Fig. 2. Average sub-population size per DE variant for representative test problems based on dimensionality.

Acknowledgments: The authors acknowledge support from CONACyT through project 79809.

References

1. Eiben, A., Smith, J.E.: Introduction to Evolutionary Computing. Springer (2003)
2. Mezura-Montes, E., Coello Coello, C.A.: Constraint-handling in nature-inspired numerical optimization: Past, present and future. Swarm and Evolutionary Computation 1, 173–194 (2011)
3. Wolpert, D.H., Macready, W.G.: No Free Lunch Theorems for Optimization. IEEE Transactions on Evolutionary Computation 1, 67–82 (1997)
4. Mallipeddi, R., Suganthan, P., Pan, Q., Tasgetiren, M.: Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied Soft Computing 11, 1679–1696 (2011)
5. Wang, Y., Cai, Z., Zhang, Q.: Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evolutionary Computation 15, 55–66 (2011)
6. Elsayed, S.M., Sarker, R.A., Essam, D.L.: Multi-operator based evolutionary algorithms for solving constrained optimization problems. Computers and Operations Research 38, 1877–1896 (2011)
7. Price, K., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization. Springer (2005)

Table 2. Statistical comparison of the proposed approach against state-of-the-art EAs.

Problem/Optimal	Stat	Our approach	SAMO-GA	SAMO-DE	TLBO	t-test
g01	**Best**	**-15.0000**	**-15.0000**	**-15.0000**	**-15.0000**	
-15.0000	**Average**	**-15.0000**	**-15.0000**	**-15.0000**	**-15.0000**	X X X
	Std. dev	**0.00E+0**	**0.00E+0**	**0.00E+0**	**0.00E+00**	
g02	**Best**	-0.803603	-0.803591	**-0.803619**	-0.803619	
-0.803619	**Average**	-0.782710	-0.796048	-0.798735	**-0.803619**	√ √ √
	Std. dev	20.1E-03	5.8025E-03	8.8005E-03	**0.00E+00**	
g03	**Best**	**-1**	**-1**	**-1**	**-1**	
-1.	**Average**	-0.988	**-1**	**-1**	**-1**	√ √ √
	Std. dev	23.5E-03	**0.00E+00**	**0.00E+00**	1.40E-04	
g04	**Best**	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	
-30665.539	**Average**	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	X X X
	Std. dev	1.4138E-11	**0.00E+00**	**0.00E+00**	**0.00E+00**	
g05	**Best**	**5126.484**	5126.497	5126.497	**5126.484**	
5126.484	**Average**	5126.591	5127.976	**5126.497**	5168.7194	√ X √
	Std. dev	585.7E-03	1.1166E+00	**0.00E+00**	5.41E+01	
g06	**Best**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**	
-6961.814	**Average**	**-6961.814**	**-6961.814**	**-6961.813875**	**-6961.814**	X X X
	Std. dev	4.6252E-12	**0.00E+00**	**0.00E+00**	**0.00E+00**	
g07	**Best**	24.307	**24.306**	**24.306**	**24.306**	
24.306	**Average**	**24.309**	24.411	24.310	24.310	√ X X
	Std. dev	2E-03	4.5905E-02	**1.5888E-03**	7.11E-03	
g08	**Best**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	
-0.095825	**Average**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	X X X
	Std. dev	1.4115E-17	**0.00E+00**	**0.00E+00**	**0.00E+00**	
g09	**Best**	**680.630**	**680.630**	**680.630**	**680.630**	
680.630	**Average**	**680.630**	680.634	**680.630**	**680.630**	X X X
	Std. dev	1.9868E-07	1.4573E-03	1.1567E-05	**0.00E+00**	
g10	**Best**	7049.365	**7049.248**	**7049.248**	7052.488	
7049.28	**Average**	**7050.373**	7144.40311	7059.813	7143.45	√ √ √
	Std. dev	**771.8E-03**	6.7860E+01	7.856E+00	1.13E+02	
g11	**Best**	**0.75**	**0.75**	**0.75**	**0.75**	
0.75	**Average**	**0.75**	**0.75**	**0.75**	**0.75**	X X X
	Std. dev	**1.1292E-16**	0.00E+00	0.00E+00	7.06E-05	
g12	**Best**	**-1**	**-1**	**-1**	**-1**	
-1	**Average**	**-1**	**-1**	**-1**	**-1**	X X X
	Std. dev	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	
g13	**Best**	**0.05390**	0.05394	0.05394	0.13314	
0.05394	**Average**	0.33032	0.05403	**0.05394**	0.83851	X X √
	Std. dev	333.0E-03	5.9414E-05	**1.7541E-08**	2.26E-01	

8. Mezura-Montes, E., Velázquez-Reyes, J., Coello Coello, C.A.: Modified Differential Evolution for Constrained Optimization. In: 2006 IEEE Congress on Evolutionary Computation, pp. 332–339. IEEE Press, New York (2006)
9. Youyun, A., Hongqin, C.: An adaptive differential evolution algorithm to solve constrained optimization problems in engineering design. Engineering 2, 65–77 (2010)
10. Feoktistov, V.: Differential Evolution: In Search of Solutions. Springer (2006)
11. Mezura-Montes, E., Coello, C.A.C.: Identifying On-line Behavior and Some Sources of Difficulty in Two Competitive Approaches for Constrained Optimization. In: 2005 IEEE Congress on Evolutionary Computation, vol. 2, pp. 1477–1484. IEEE Press, New York (2005)
12. Deb, K.: An Efficient Constraint Handling Method for Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering 186, 311–338 (2000)
13. Rao, R.V., Patel, V.: An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. International Journal of Industrial Engineering Computations 3, 535–560 (2012)