Information Sciences 194 (2012) 171–208

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Constrained optimization based on modified differential evolution algorithm

Ali Wagdy Mohamed [a,b,*], Hegazy Zaher Sabry [c]

[a] Statistics Department, Faculty of Sciences, King Abdulaziz University, P.O. Box 80203, Jeddah 21589, Saudi Arabia
[b] Operations Research Department, Institute of Statistical Studies and Research, Cairo University, Giza, Egypt
[c] Mathematical Statistics Department, Institute of Statistical Studies and Research, Cairo University, Giza, Egypt

ARTICLE INFO

Article history:
Received 20 October 2010
Received in revised form 25 September 2011
Accepted 9 January 2012
Available online 17 January 2012

Keywords:
Differential evolution
Constrained optimization
Directed mutation
Dynamic non-linear crossover
Dynamic tolerance

ABSTRACT

This paper presents a novel Constrained Optimization based on Modified Differential Evolution algorithm (COMDE). In the new algorithm, a new directed mutation rule, based on the weighted difference vector between the best and the worst individuals at a particular generation, is introduced. The new directed mutation rule is combined with the modified basic mutation strategy DE/rand/1/bin, where only one of the two mutation rules is applied with the probability of 0.5. The proposed mutation rule is shown to enhance the local search ability of the basic Differential Evolution (DE) and to get a better trade-off between convergence rate and robustness. Two new scaling factors are introduced as uniform random variables to improve the diversity of the population and to bias the search direction. Additionally, a dynamic non-linear increased crossover probability is utilized to balance the global exploration and local exploitation. COMDE also includes a modified constraint handling technique based on feasibility and the sum of constraints violations. A new dynamic tolerance technique to handle equality constraints is also adopted. The effectiveness and benefits of the new directed mutation strategy and modified basic strategy used in COMDE has been experimentally investigated. The effect of the parameters of the crossover probability function and the parameters of the dynamic tolerance equation on the performance of COMDE have been analyzed and evaluated by different experiments. Numerical experiments on 13 well-known benchmark test functions and five engineering design problems have shown that the new approach is efficient, effective and robust. The comparison results between the COMDE and the other 28 state-of-the-art evolutionary algorithms indicate that the proposed COMDE algorithm is competitive with, and in some cases superior to, other existing algorithms in terms of the quality, efficiency, convergence rate, and robustness of the final solution.

1. Introduction

In real world problems and applications, most of the optimization problems involve different types of constraints. These problems are called constrained optimization problems (COPs). The optimization of the constrained optimization problems is considered a challenging task since the optimum solution(s) must be feasible. In their original design, evolutionary algorithms (EAs) are able to solve unconstrained optimization problems effectively. As a result, in the past decade, many researchers have developed a variety of constraint handling techniques, incorporated into (EAs) designs, to counter this deficiency. Differential Evolution (DE) algorithm, similar to (EAs), lacks a mechanism to incorporate feasibility information into

* Corresponding author. Tel.: +966 556269723.
E-mail addresses: aliwagdy@gmail.com (A.W. Mohamed), hgsabry@yahoo.com (H.Z. Sabry).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2012.01.008

the fitness value of a given solution. In other words, it lacks a mechanism to guide the search process towards the constrained search space (i.e. feasible region). Thus, the selection of an adequate constraint-handling technique for a given EA (DE in this case) is an open research area. Coello [3] has classified the constraint handling methods into five categories: (1) penalty functions, (2) special representations and operators, (3) repair algorithms, (4) separation of objectives and constraints and (5) hybrid methods. Penalty functions methods are considered the most common constraint handling techniques due to their simplicity and ease of implementation. The main idea of these methods is to transform a constrained optimization problem into an unconstrained one by adding (or subtracting) a penalty term to (or from) the objective function value based on the amount of constraint violation present in a certain solution. The shortcoming of the penalty function methods is the fine tuning of their parameters that is done by trial and error in order to get competitive results; otherwise the search procedure may converge to an infeasible solution. Additionally, the appropriate parameter values are problem-dependent and need to be adjusted with each particular problem. Unlike the penalty functions methods, other novel techniques have been developed and utilized to handle constraints; for instance the usage of special representations and operators for problems for which it is extremely difficult to locate at least a single feasible solution. However, the generalization of such operators to other (even similar) problems is by no means obvious [3]. Repair algorithms represent another technique to transform an infeasible solution into a feasible one. Repair algorithms are however problem-dependent [37] and repair operators may introduce a strong bias in the search by harming the evolutionary process itself. The separation of constraints and objectives are other promising constraint handling techniques that handle constraints and objectives separately (i.e. without combining the amount of constraint violation and the objective function value). The key concept of these techniques is to bias feasible over infeasible solutions. Where the ratio between the feasible region and the whole search space is too small (e.g. when constraints are very difficult to satisfy), the separation technique will fail unless a feasible point is introduced in the initial population [30]. Finally, hybrid methods merge different algorithms, mechanisms and mathematical programming techniques (e.g. fuzzy logic, lagrangian multipliers) with evolutionary computation techniques for numerical optimization problems. This combination of methods is still a new and open research path. Furthermore, over the past twenty years, many approaches have been proposed to solve constrained optimization problems using (EAs). We have decided to compare the proposed approach and the obtained results against three different types of approaches, from the literature review: (1) the state of the art EA approaches known to date, (2) the most competitive approaches based on hybridization of Differential Evolution (DE) with other EA, (3) approaches based on Differential Evolution (DE). Genetic algorithms (GA) are considered the most well-known branch of EAs and have been successfully applied to solve both unconstrained and constrained optimization problems. Koziel and Michalewicz [15] proposed GA which depends on a homomorphous mapping between an *n*dimensional cube and the feasible part of the search space. However, its drawback is that it requires an initial feasible solution. Deb [6] proposed a new and efficient feasibility-based rule to handle constraint for genetic algorithm where pair-wise solutions are compared using the following criteria:

- Between two feasible solutions, the one with the highest fitness values wins.
- If one solution is feasible and the other one is infeasible, the feasible solution wins.
- If both solutions are infeasible, the one with the lowest sum of constraint violation is preferred.

Chootinan and Chen [2] used the gradient information derived from the constraint set to systematically repair infeasible solutions. The proposed repair procedure embedded into a simple GA as a special operator solves only a set of 11 benchmark problems. Elfeky et al. [7] used GA with a new ranking, selection, and triangular crossover methods to solve a set of nine problems (all with inequality constraints). Similarly, Tessema and Yen [41] introduced a self adaptive penalty function using GA. The introduced method aimed to encourage infeasible individuals with low objective function value and low constraint violation. The approach was tested on a set of 13 benchmark problems and the results showed that the approach was able to find very good solutions in comparison to other state-of-the-art techniques. Later, a new agent-based memetic algorithm based on GA was applied successfully to solve optimization problems in [42]. The performance of the proposed algorithm is tested on 13 well-known benchmark problems and the experimental results showed convincing performance. Evolutionary Strategies (ES) have been widely used to solve several types of optimization problems. Mezura-Montes and Coello [24] proposed a simple multi membered evolution strategy to solve constrained optimization problems. The proposed approach uses a simple diversity mechanism based on allowing infeasible solutions to be selected to the next population. The approach tested on the benchmark problems and the results obtained were very competitive when compared with other techniques. Runasson and Yao [36] introduced an improved evolutionary algorithm based on evolutionary strategy and differential variation in conjunction with a multi-objective approach to handle constraints instead of using penalty function method. In fact, this approach is considered one of the major of the state of the art constrained optimization techniques due to its very good performance on well-known benchmark problems. Takahama and Sakai [40] proposed the αconstrained method. In this method, some improvements, including mutations for searching the boundaries and multiple simplexes instead of one simplex, are added to modify non-linear simplex method. This method can convert an algorithm for unconstrained optimization problems into an algorithm for constrained optimization problems by the αlevel. Taking advantage of multi-objective optimization techniques, Wang et al. [46] presented a multi-objective optimization based on a hybrid evolutionary algorithm to solve constrained optimization problems. This method consists of two main parts: the global search model and the local search model. In the global search model, a niching genetic algorithm is proposed. The local search model performs a parallel local search operation based on clustering partition of the population. In this method, the comparison

of individuals is based on multi-objective optimization technique. Recently, Wang et al. [47] developed a novel approach which incorporates a hybrid evolutionary algorithm and an adaptive constraint-handling technique. The hybrid EA uses simplex crossover and two mutation operators to generate the offspring population, and the adaptive constraint handling technique consists of three main situations, i.e. infeasible situation, semi-feasible situation, and feasible situation. Meanwhile, one constraint-handling mechanism is designed according to the characteristic of the current situation. EAs encompass three strongly related but independently developed branches: evolution strategies (ES), genetic algorithms (GAs), and evolutionary programming (EP). Nevertheless, during the past few years, another five branches have been added to the basic EAs algorithms, such as Genetic Programming (GP), Differential Evolution (DE), Cultural Evolution (CE), Co-evolution and Swarm Intelligence which included Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). As a result, the newly created techniques enrich the EAs community. Therefore, new hybridizations have been developed in order to solve unconstrained and constrained optimization problems and their application in different fields of sciences. Becerra and Coello [18] proposed a cultural algorithm with a differential evolution population for solving COPs. In this method, differential evolution is used as the population space of a cultural algorithm instead of using an evolutionary programming algorithm. Meanwhile, this culture algorithm has used different knowledge sources to influence the variation operator of the differential evolution algorithm in order to reduce the total number of function evaluations required to achieve competitive results. Another type of hybridization based on direct search technique known as the Nelder-Mead Method (NMM), named (HDESMCO), is proposed by Mezura-Montes and Coello [20]. The core idea behind this technique was to incorporate the NMM as an additional operator to the basic differential evolution in order to improve its convergence rate. The main role of DE is to approach a promising region of the search space quickly, and then the NNM or simplex operator can exploit it in order to reach the optimum solution within that region. Due to its success in solving unconstrained optimization problems, PSO has gained much attention in solving COPs in the last few years. Wang and Cai [45] introduced a hybrid multi-swarm particle swarm optimization to deal with COPs, named HMPSO. This method utilizes a parallel search operator in which the current swarm is portioned into several sub-swarms and PSO is served as the search engine for each sub-swarm. In addition, the personal best of each particle are updated by DE so as to further enhance the global search ability of PSO. In order to avoid the premature convergence into local optimum, a new neighborhood structure for PSO called Singly-Linked Rings which has innovative neighbors' selection is implemented by Muñoz et al. [32]. Additionally, a special technique to handle equality constraints with low side effects on the diversity is suggested. Two perturbation operators based on differential evolution and random mutation are used to improve the exploration, applying these modifications only in the particle best population. Similarly, Wang et al. [21] introduced hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, named PSO-DE. In this hybrid algorithm, in order to avoid the stagnation, DE is incorporated to update the previous best positions of particles to force PSO jump out of stagnation as well as to speed up the convergence. Even though DE was originally designed to solve unconstrained optimization problems like the recent ones reported in [10,14,31], many modifications and developments have been done to DE itself to deal with COPs. Mezura-Montes et al. [25] proposed an approach based on DE whose selection criterion is based on Deb's comparison criteria. In this approach, the DE algorithm was modified in a way that the newly chosen individual could be re-inserted in the current population (and not only inserted in the population for the next generation like in the original DE algorithm). The aim is to allow the newly-improved solutions to be selected as random parents for other offspring in the current generation and to accelerate convergence. The proposed approach provided good quality results, but it was not consistent (not all runs reached either the best known or the global optimum solution). However, they concluded that their approach as well as Deb's and Lampinen's algorithms [17] lack a mechanism to maintain diversity (to have both feasible and infeasible solutions in the population during all the evolutionary process), which is one of the most important aspects to consider when designing a competitive constraint-handling approach [23]. Hence, Mezura-Montes et al. [27], in order to keep diversity in the population, used Storn's idea of allowing the generation of more than one offspring for each individual, but now it was combined with a mechanism to allow infeasible solutions with a good value of the objective function to remain in the population. Nonetheless, Mezura-Montes and Palomeque-Ortiz [29] extended the previous work in [27] by proposing two parameter control mechanisms to keep the user from defining the values of four (out of six) parameters. Three parameters are self-adapted and the other one uses deterministic control. Similarly, stochastic ranking is applied to a multimember DE framework [49] that is different from the most current DE frameworks for constrained optimization. After experimental analysis, they concluded that the promising infeasible solutions should be paid more attention in the early stage of evolution and the same for the feasible solutions in the late stage. Thus, a novel dynamic stochastic selection is proposed to keep the feasible solutions as well as the promising infeasible solution during the evolution process. Recently, inspired by 'fuzzy control theory', Wang and Li [44] introduced a simple but effective differential evolution with level comparison, named (DELC), for constrained engineering design problems by applying the level comparison to convert the constrained optimization problem into an unconstrained one and using the differential evolution (DE) to perform a global search over the solution space. In addition, the mutation factor of DE is set to be a random number to enrich the search behavior, and the satisfaction level increases monotonously to gradually stress the feasibility. Motivated by the recent success of diverse approaches based on DE to solve (COPs), Mezura-Montes et al. [26] combined two variants: DE/rand/1/bin and DE/best/1/bin into one single approach, called (DECV). In this paper, the proposed Constrained Optimization based on Modified Differential Evolution (COMDE) algorithm is used to solve single objective problems with linear or non-linear functions and subject to both equality and inequality constraints which can be linear and non-linear. It is applied to various engineering design optimization problems with mixed- type variables. The proposed algorithm shows a superior and competitive performance to other recent constrained optimization

algorithms. The remainder of this paper is organized as follows. Section 2 presents the problem of our interest. Section 3 introduces the DE algorithm. Section 4 presents a detailed description of COMDE algorithm. Section 5 reports on the computational results of testing benchmark functions and some engineering optimization problems. Besides, the comparison with other techniques is discussed. Section 6 discusses the effectiveness of the proposed modifications and the effect of the parameter settings that significantly lead to the perfect performance of the proposed approach. Finally, Section 7 presents the study conclusions and future research.

2. Problem statement

In general, constrained optimization problem can be expressed as follows (without loss of generality minimization is considered here):

$$\text{minimize } f(\vec{x}), \quad \vec{x} = (x_1, x_2, \ldots, x_n) \in \Re^n \tag{1}$$

$$\text{subject to } g_j(\vec{x}) \leqslant 0, \quad j = 1, \ldots, q \tag{2}$$

$$h_j(\vec{x}) = 0, \quad j = q+1, \ldots, m \tag{3}$$

where $\vec{x} \in \Omega \subseteq S, \Omega$ is the feasible region, and S is an n-dimensional rectangular space in $\Re^n$ defined by the parametric constraints $l_i \leqslant x_i \leqslant u_i$, $1 \leqslant i \leqslant n$, where l_i and u_i are lower and upper bounds for a decision variable x_i, respectively. For an inequality constraint that satisfies $g_j(\vec{x}) = 0$ $(j \in 1, \ldots, q)$ at any point $\vec{x} \in \Omega$, we say it is active at $\vec{x}$. All equality constraints $h_j(\vec{x})$ $(j = q+1, \ldots, m)$ are considered active at all points of Ω. Most constraint-handling approaches used with EAs tend to deal only with inequality constraints. Therefore equality constraints are transformed into inequality constraints of the form $|h_j(\vec{x})| - \varepsilon \leqslant 0$ where ε is the tolerance allowed (a very small value).

3. Differential evolution

Differential Evolution (DE) is a stochastic population-based search method, proposed by Storn and Price [33]. DE is relatively recent EAs for solving real-parameter optimization problems [39]. DE has many advantages including simplicity of implementation, reliability, robustness, and in general is considered as an effective global optimization algorithm [33]. In this paper, we use the scheme which can be classified using the notation as DE/rand/1/bin strategy [38,1]. This strategy is the most often used in practice. A set of D optimization parameters is called an individual, which is represented by a D-dimensional parameter vector. A population consists of NP parameter vectors $x_i^G,\ i = 1, 2, \ldots, NP$. G denotes one generation. NP is the number of members in a population. It does not change during the evolution process. The initial population is chosen randomly with uniform distribution in the search space. DE has three operators: mutation, crossover and selection. The crucial idea behind DE is a scheme for generating trial vectors. Mutation and crossover operators are used to generate trial vectors, and the selection operator then determines which of the vectors will survive into the next generation [1].

3.1. Initialization

In order to establish a starting point for the optimization process, an initial population must be created. Typically, each decision parameter in every vector of the initial population is assigned a randomly chosen value from the boundary constraints:

$$x_{ij}^0 = l_j + rand_j^*(u_j - l_j) \tag{4}$$

where $rand_j$ denotes a uniformly distributed number between [0, 1], generating a new value for each decision parameter. l_i and u_i are the lower and upper bounds for the jth decision parameter, respectively [39].

3.2. Mutation

For each target vector x_i^G, a mutant vector v is generated according to the following:

$$v_i^{G+1} = x_{r_1}^G + F * \left(x_{r_2}^G - x_{r_3}^G\right), \quad r_1 \neq r_2 \neq r_3 \neq i \tag{5}$$

with randomly chosen indices and $r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$.

Note that these indices have to be different from each other and from the running index i so that NP must be at least four. F is a real number to control the amplification of the difference vector $\left(x_{r_2}^G - x_{r_3}^G\right)$. According to [39], the range of F is in [0, 2]. If a component of a mutant vector goes off the search space, and then the new value of this component is generated using (4).

3.3. Crossover

The target vector is mixed with the mutated vector, using the following scheme, to yield the trial vector u.

$$u_{ij}^{G+1} = \begin{cases} x_{ij}^{G+1}, rand(j) \leqslant CR \quad \text{or} \quad j = randn(i) \\ v_{ij}^{G}, rand(j) > CR \quad \text{and} \quad j \neq randn(i) \end{cases} \tag{6}$$

where $j = 1, 2, \ldots, D$, $rand(j) \in [0, 1]$ is the jth evaluation of a uniform random generator number. $CR \in [0, 1]$ is the crossover probability constant, which has to be determined by the user. $randn(i) \in \{1, 2, \ldots, D\}$ is a randomly chosen index which ensures that u_i^{G+1} gets at least one element from v_i^{G+1}.

3.4. Selection

DE adapts a greedy selection strategy. If and only if the trial vector u_i^{G+1} yields a better fitness function value than x_i^G, then u_i^{G+1} is set to x_i^{G+1}. Otherwise, the old value x_i^G is retained. The selection scheme is as follows (for a minimization problem):

$$x_i^{G+1} = \begin{cases} u_i^{G+1}, & f(u_i^{G+1}) < f(x_i^G) \\ x_i^G, & f(u_i^{G+1}) \geqslant f(x_i^G) \end{cases} \tag{7}$$

4. Constrained Optimization based on Modified Differential Evolution algorithm (COMDE)

All evolutionary algorithms, including DE, are stochastic population-based search methods. Accordingly, there is no guarantee that the global optimal solution will be reached consistently. Furthermore, they are not originally designed to solve constrained optimization problems. Nonetheless, adjusting control parameters such as the scaling factor, the crossover rate and the population size, alongside with developing an appropriate mutation scheme and coupling with suitable and effective constraint-handling techniques can considerably improve the search capability of DE algorithms. Moreover, the possibility of achieving promising and successful results in complex numerical and engineering constrained optimization problems increases. Therefore, in this paper, five modifications are introduced in order to significantly enhance the overall performance of the standard DE algorithm.

4.1. Modification of mutations

The success of the population-based search algorithms is based on balancing two contradictory aspects: global exploration and local exploitation [5]. Moreover, the mutation scheme plays a vital role in the DE search capability and the convergence rate. However, even though the DE algorithm has good global exploration ability, it suffers from weak local exploitation ability as well as its convergence velocity being too low as the region of the optimal solution is reached [8]. Obviously, from the mutation Eq. (5), it can be observed that three vectors are chosen at random for mutation and the base vector is then selected at random among the three. Consequently, the basic mutation strategy DE/rand/1/bin is able to maintain population diversity and global search capability, but it slows down the convergence of DE algorithms. Hence, in order to enhance the local search ability and to accelerate the convergence speed of DE techniques, a new directed mutation scheme is proposed based on the weighted difference vector between the best and the worst individual at a particular generation. The modified mutation scheme is as follows:

$$v_i^{G+1} = x_r^G + F_l * (x_b^G - x_w^G) \tag{8}$$

where x_r^G is a randomly chosen vector and x_b^G and x_w^G are the best and worst vectors in the entire population, respectively. This modification is intended to keep the random base vector $x_{r_1}^G$ in the mutation Eq. (5) as it is and the remaining two vectors are replaced by the best and worst vectors in the entire population to yield the difference vector. In fact, the global solution can be easily reached if all vectors follow the same direction of the best vector. Besides, they also follow the opposite direction of the worst vector. Thus, the proposed directed mutation favors exploitation since all vectors of population are biased by the same direction but are perturbed by the different weights as discussed later on. As a result, the new mutation rule has better local search ability and faster convergence rate. It is worth mentioning that the proposed mutation is inspired from nature and human behavior. Briefly, although all people in a society are different in many ways such as aims, cultures, thoughts and so on, all of them try to significantly improve themselves by following the same direction of the other successful and superior people and similarly they tend to avoid the direction of failure in whatever field by competition and/or co-operation with others. The new mutation strategy is embedded into the DE algorithm and it is combined with the basic mutation strategy DE/rand/1/bin as follows:

$$\text{If } (rand[0, 1] \leqslant 0.5) \tag{9}$$

$$\text{Then } v_i^{G+1} = x_r^G + F_l * (x_b^G - x_w^G) \tag{10}$$

$$\text{Else } v_i^{G+1} = x_{r1}^G + F_g * (x_{r2}^G - x_{r3}^G) \tag{11}$$

where F_l and F_g are two uniform random variables, rand [0, 1] is a function that returns a real number between 0 and 1, G is the current generation number. From the above scheme, it can be realized that for each vector, only one of the two strategies is used for generating the current trial vector, depending on a uniformly distributed random value within the range [0, 1]. For

each vector, if the random value is smaller than or equal 0.5, the proposed one is performed. Otherwise, the basic mutation is applied. In other words, practically, no vector is subject to both mutations in the same generation, and only one of the above two mutations can be applied with equal probability of (0.5). However, both mutations can be performed in the same generation with two different vectors. Therefore, at any particular generation, the proposed algorithm has the equal chance to improve the exploration and exploitation abilities.

4.2. Modification of scaling factor

In the mutation Eq. (5), the constant of differentiation F is a scaling factor of the difference vector. It is an important parameter that controls the evolving rate of the population. Regarding global unconstrained optimization, in the original DE algorithm in [39], the constant of differentiation F was chosen to be a value in $[0, 2]$. The value of F has a considerable influence on exploration: small values of F lead to premature convergence. Besides, it can lead to speed up the convergence, and high values slow down the search [9]. However, to the best of our knowledge, there is no optimal value of F that has been derived based on theoretical and/or systematic study using all complex benchmark problems. In this paper, two scaling factors F_l and F_g are proposed for the two different mutation rules. Where F_l and F_g indicate scaling factor for the local mutation scheme and the scaling factor for global mutation scheme, respectively. For the difference vector in the mutation Eq. (10), it can be seen that it is a directed difference vector from the worst to the best vectors in the entire population. Hence, F_l must be a positive value in order to bias the search direction for all trial vectors in the same direction. Therefore, F_l is introduced as a uniform random variable within $(0, 1]$. Instead of keeping F constant during the search process, F_l is set as a random variable for each trial vector so as to perturb the random base vector by different directed weights. Therefore, the new directed mutation resembles the concept of gradient as the difference vector is oriented from the worst to the best vectors [9]. On the other hand, for the difference vector in the mutation Eq. (11), it can be observed that it is a pure random difference as the objective function values are not used. Accordingly, the best direction that can lead to a good exploration is unknown. Therefore, in order to advance the exploration and to cover the whole search space F_g is introduced as a uniform random variable in the interval $(-1, 0) \cup (0, 1)$, unlike keeping it as a constant in the range $[0, 2]$. Therefore, the new enlarger random variable can perturb the random base vector by different random weights with opposite directions. Hence, F_g is set to be random for each trial vector. As a result, the proposed evolutionary algorithm is still a random search that can enhance the global exploration performance as well as ensure the local search ability.

4.3. Modification of the crossover rate

The crossover operator, as in Eq. (6), shows that the constant crossover (CR) reflects the probability with which the trial individual inherits the actual individual's genes [9]. The constant crossover (CR) practically controls the diversity of the population. As mentioned in the above subsection, regarding global unconstrained optimization, if the CR value is relatively high, this will increase the population diversity and improve the convergence speed. Nevertheless, the convergence rate may decrease and/or the population may prematurely converge. On the other hand, small values of CR increase the possibility of stagnation and slow down the search process. Additionally, at the early stage of the search, the diversity of the population is large because the vectors in the population are completely different from each other and the variance of the whole population is large. Therefore, the CR must take a suitable value in order to avoid the exceeding level of diversity that may result in premature convergence or slow convergence rate. Then, through generations, the variance of the population will decrease as the vectors in the population become similar. Thus, in order to advance diversity and increase the convergence speed, the CR must be a large value. However, regarding constrained optimization, the presence of constraints may cause loss of the diversity of the population by preferring the feasible solution to the infeasible one. Based on the above analysis and discussion, and in order to balance between the diversity and the convergence rate or between global exploration ability and local exploitation tendency in constrained optimization, a dynamic non-linear increased crossover probability scheme is proposed as follows:

$$CR = CR_{\max} + (CR_{\min} - CR_{\max}) * (1 - G/GEN)^k \tag{12}$$

where G is the current generation number, GEN is the maximum number of generations, $CR_{\min}$ and $CR_{\max}$ denote the maximum and minimum value of the CR, respectively, and k is a positive number. The optimal settings for these parameters are $CR_{\min} = 0.5$, $CR_{\max} = 0.95$ and $k = 4$. The algorithm starts at $G = 0$ with $CR_{\min} = 0.5$ but as G increases toward GEN, the CR exponentially increases to reach $CR_{\max} = 0.95$. As can be seen from Eq. (12), $CR_{\min} = 0.5$ and $CR_{\max} = 0.95$ are considered as a good initial value and a reasonable maximum value of crossover that can balance between exploration and exploitation in constrained optimization. $CR_{\min} = 0.5$ means that the information provided by the mutant vector and target vector to create the trial vector are equal, i.e. it has the same contribution in the early stage of the search process. Additionally, in order to increase diversity through generations and get more feasible solutions, $CR_{\max} = 0.95$ is the maximum value of crossover that can balance between exploration and exploitation. However, beyond this value, the mutation vector v_i^{G+1} will be completely inherited to the trial vector u_i^{G+1}. Consequently, the target vector x_i^G drastically disappeared so that it destroys the individual's genes with better function values. On the other hand, k balances the crossover rate which results in changing the CR from small value to large value in dramatic curve. k was set to its mean value as it was observed that if it is

approximately less than or equal to 1 or 2 then the diversity of the population deteriorated for some functions and it might have cause stagnation. On the other hand, if it is nearly greater than 6 or 7, it could cause premature convergence as the diversity sharply increases. The mean value of 4 was thus selected for all numerical and engineering benchmark problems. The settings of these parameters will be discussed later on.

4.4. A modified constraint handling approach

As previously mentioned, Deb [6] has introduced the superiority of feasible solutions selection procedure based on the idea that any individual in a constrained search space must first comply with the constraints and then with the objective function. Therefore, in order to avoid stagnation that may occur through evolution process, a modified constraint handling approach based on Deb's approach is proposed. Practically, Deb's selection criterion does not need to fine-tune any parameter. Consequently, the researchers modify and incorporate this constraint handling technique as follows.

At each generation, during the selection process, each trial vector is compared with its corresponding target vector in the current population considering both the fitness value and the constraints. The trial vector will replace the target vector and enter the population of the next generation if any of the following conditions is true:

1. The trial vector is feasible and the target vector is not.
2. The trial vector and target vector are both feasible, but the trial vector has smaller than or equal fitness value than the corresponding target vector.
3. The trial vector and target vector are both infeasible, but the trial vector has smaller than or equal overall constraint violation than the corresponding target vector.

Obviously, the above selection procedure is relatively different from Deb's procedure since it also allows the trial vector to be entered in the new population if it has the same amount of constraint violation or objective function value as the target vector. Therefore, this simple modification can help the algorithm to spread out and pass through the search space, so the algorithm can escape from stagnation. In other words, this modification is useful because it can reduce the probability of stagnation even though it does not have a guarantee to succeed in all cases with all constrained problems. Typically, from the problem formulation in section 2, there are m constraints and hence the amount of constraint violation for an individual is represented by a vector of m dimensions. Using a tolerance (ε) allowed for equality constraints, the constraint violation of a decision vector or an individual $\vec{x}$ on the jth constraint is calculated by

$$cv_j(\vec{x}) = \begin{cases} \max(0, g_j(\vec{x})), & j = 1, \ldots, q \\ \max(0, |h_j(\vec{x})| - \varepsilon), & j = q+1, \ldots, m \end{cases} \tag{13}$$

If a decision vector or an individual $\vec{x}$ satisfies the jth constraint, $cv_j(\vec{x})$ is set to zero, it is greater than zero. As discussed [43], in order to consider all the constraints at the same time or to treat each constraint equally, each constraint violation is then normalized by dividing it by the largest violation of that constraint in the population. Thus, the maximum violation of each constraint in the population is given by

$$cv^j_{\max} = \max_{\vec{x} \in S} cv_j(\vec{x}) \tag{14}$$

These maximum constraint violation values are used to normalize each constraint violation. The normalized constraint violations are added together to produce a scalar constraint violation $cv(\vec{x})$ for that individual which takes a value between 0 and 1

$$cv(\vec{x}) = \frac{1}{m} \sum_{j=1}^{m} \frac{cv_j(\vec{x})}{cv^j_{\max}} \tag{15}$$

4.5. Equality constraint approach

For any feasible region limited by a set of constraints, if one of these constraints is represented by equality, the feasible region becomes a line segment. Furthermore, if all these constraints are equality constraints, then the feasible region is reduced to a point, i.e. the intersection point of all constraints. Therefore, although the equality constraints are transformed into inequality constraints of the form $|h_j(\vec{x})| - \varepsilon \leqslant 0$ where ε is the tolerance allowed (a very small value), it still makes the feasible region very small compared with the search space, which exacerbates the optimization process using any evolutionary algorithm to find the feasible solutions. As a result, the larger the tolerance value, the more reliable feasible solutions found. In other words, the temporary increase of the feasible space during the initial generations of the search process there will be some feasible individuals from the beginning of the optimization process. Then, by reducing the search space after some generations by decreasing the tolerance value (ε), the algorithm will improve the previous feasible solutions to fit into the new feasible region and satisfy the new equality constraints. Hence, in order to deal with equality constraints, the

researchers introduce the new approach that dynamically changes the value of (ε) through generations. The tolerance value (ε) is adapted and decreased through generations by using the following mathematical expression:

$$Factor = \begin{cases} F_{final} + (F_{initial} - F_{final}) * (1 - G/GEN)^k & ,0 < (G/GEN) \leqslant R \\ F_{final} & ,(G/GEN) > R \end{cases} \quad (16)$$

where $\varepsilon(t) = 10^{-Factor}$, $F_{initial} = -\log_{10}(a)$, a represents the initial tolerance, G is the current generation number, GEN is the maximum number of generations, ($F_{initial}$) and (F_{final}) denote the maximum and minimum tolerance value for $Factor$, respectively. The settings of $F_{initial}$ and F_{final} changes from problem to another. R is a predetermined positive ratio between 0 and 1 and is determined using this formula $R = 1 - (1/F_{final})$, and k is a positive number. The algorithm starts at $G = 0$ with $10^{-(F_{initial})}$, i.e. with the value of a, but as G/GEN increases towards R, the tolerance value $\varepsilon(t)$ decreases to reach $10^{-(F_{final})}$ and stays fixed in the later generations. As can be easily derived from the above expression, the parameter k influences the evolvement between $t = 0$ and $t = (G/GEN)$, $Factor$ decreases linearly with $k = 1$. However, for larger k, the tolerance value $\varepsilon(t)$ rapidly decreased to reach the final tolerance. In other words, as k increases, the widening feasible region considerably shrinks to the true feasible that may lead to converge to local optimal or it may also significantly deteriorate the search process as will be discussed later on. Indeed, the initial tolerance has also a significant impact in creating the suitable initial feasible region that can attract too many infeasible points which can be improved through generations to be feasible as will be discussed later on.

5. Numerical experiments and comparisons

In order to evaluate the performance and show the efficiency and superiority of the proposed algorithm (COMDE), 13 well-known benchmark test functions mentioned in [35] are used. The objective of this section is to evaluate and compare the proposed COMDE algorithm with twenty state-of-the-art evolutionary algorithms. To the best of our knowledge, this is the first study that uses all these different types of approaches to carry out evaluation and comparisons. As a matter of fact, this study aims to prove that COMDE is a competitive and an efficient approach as well as being superior to the most recent techniques in the field of constrained optimization. Then, the proposed algorithm is also applied to solve five engineering design problems that are widely used in the field of constrained optimization, and its performance is studied by comparing it with other many existing techniques in the literature. The experiments were carried out on an Intel Pentium core 2 due processor 2200MHZ and 2 GB-RAM. COMDE algorithm is coded and realized in MATLAB. For each problem, 30 independent runs are performed and statistical results are provided including the best, median, mean, worst results and the standard deviation. The performance of different algorithms is statistically compared with COMDE by statistical t-test with a significance level of 0.05. Numerical values -1, 0, 1 represent that the COMDE is inferior to, equal to and superior to the algorithm with which it is compared, respectively. However, (#) means that the statistical comparison has not been done on this benchmark problem because the mean and/or standard deviation values of the compared algorithm are not available. Furthermore, the statistical comparison has not been done on the benchmark problems where both approaches have obtained the optimum in all runs and they are considered equal.

5.1. Simulation on benchmark test functions

5.1.1. Benchmark test functions and parameter settings

The main characteristics of 13 benchmark functions are reported in Table 1. Obviously, from Table 1, it can be observed that these benchmark functions contain several types of objective function shapes (e.g. linear, non-linear, and quadratic) as well as different types of constraints (e.g. linear inequality (Li), non-linear equality (NE), and non-linear-inequality (NI)),

Table 1
Main characteristics of 13 benchmark problems.

Problem	n	Type of f	ρ (%)	Li	NE	NI	a
g01	13	Quadratic	0.0003	9	0	0	6
g02	20	Non-linear	99.9975	1	0	1	1
g03	10	Nonlinear	0.0000	0	1	0	1
g04	5	Quadratic	26.9356	0	0	6	2
g05	4	Non-linear	0.0000	2	3	0	3
g06	2	Non-linear	0.0064	0	0	2	2
g07	10	Quadratic	0.0003	3	0	5	6
g08	2	Non-linear	0.8640	0	0	2	0
g09	7	Non-linear	0.5256	0	0	4	2
g10	8	Linear	0.0005	3	0	3	3
g11	2	Quadratic	0.0000	0	1	0	1
g12	3	Quadratic	0.0197	0	0	9^3	0
g13	5	Non-linear	0.0000	0	3	0	3

where n is the number of decision variables, Li denotes the number of linear inequalities, NE denotes the number of non-linear equation, NI denotes the number of non-linear inequalities, a represents the number of active constraints at the global optimum solution, and ρ is the feasibility ratio between the feasible region and whole search space. Generally, the metric ρ is determined as follows:

$$\rho = |F|/|S| \tag{17}$$

where $|F|$ is the number of feasible solutions in $|S|$ = 1,000,000 randomly generated solutions. It is to be noted that all test functions are minimization problems but the test functions g02; g03, g08, and g12 are maximization problems. However, in this study, the maximization problems are transformed into minimization using $-f(\vec{x})$. Additionally, only test functions g03, g05, g11, and g13 contains equality constraints. Factually, from Table 1, it can be concluded that, each benchmark problem has almost its own characteristics, such as the feasibility ratio, the shape of the objective function, the number and the shape of the constraints, the location of the optimum solution as well as the number of decision variables. On the other hand, besides the aforementioned discussion in the previous section about the crossover rate and the scaling factor, the population size plays a vital role in the convergence rate of the DE algorithm. A smaller population may cause a poor searching performance and/or it may lead to premature convergence. However, a larger population reduces the risk of stagnation but increases the possibility for finding optimal solutions at the expense of a large number of function evaluations [16]. The optimal selection of this parameter is that it offers good searching performance with a minimum number of individuals. As a result, the population size should increase with the dimension of the problem as suggested in [39]. They recommended a population size of 5–10 times the dimension of the problem. Therefore, inspired by [44,39], the population size NP is set as the following proposed simple rule except for g10 and g13 with 100 and 75, respectively.

$$NP = \begin{cases} 20n & 2 \leqslant n < 5 \\ 10n & 5 \leqslant n \leqslant 10 \\ 5n & n > 10 \end{cases} \tag{18}$$

Typically, this simple population size rule provides to the algorithm enough members for searching the solution space in each problem, with a minimum computational requirement. The required population size, max generation ($G_{\max}$), and the total number of function evaluation (TNFE) for each test function are listed in Table 2. Furthermore, as mentioned in the previous section, due to the challenging features of the constrained benchmark problems as well as in order to make a balance of global exploration ability and local exploitation tendency, the optimal parameter settings for the crossover rate are $CR_{\min}$ = 0.5, $CR_{\max}$ = 0.95 and k = 4. Moreover, from Table 1, due to the presence of constraints, it can be realized that there are different metric values, several characteristics with different problems that may burden the local exploitation process. Therefore, F_l should be randomly generated within [0.4, 0.6] instead of $(0, 1)$ where the range is determined empirically and will be discussed later on. The proposed new range is established as a tolerant average of the original one so as to avoid

Table 2
The required population size, maximum generation and number of fitness function evaluations for all benchmark problems.

Problem	n	N	$G_{\max}$	TNFE
g01	13	65	2000	130,000
g02	20	100	2000	200,000
g03	10	100	1500	150,000
g04	5	50	1000	50,000
g05	4	80	2500	200,000
g06	2	40	300	12,000
g07	10	100	2000	200,000
g08	2	40	100	4000
g09	7	70	1000	70,000
g10	8	100	2000	200,000
g11	2	40	1250	50,000
g12	3	60	100	6000
g13	5	75	2000	150,000

Table 3
The parameter settings of tolerance equation for constrained problems.

Problem	a	(F_{final})	R
g03	1	8	0.8750
g05	1	8	0.8750
g11	1	12	0.9167
g13	2	4	0.7500

the large step size with small or very small feasible regions such as g1, g7 and g10 that may slow down the search or increases the probability of stagnation by generating many trial infeasible solutions. It further tends to avoid the small step size with large or intermediate feasible region such as g2, g4 and g8 that may increase the possibility of premature convergence. For equality constrained problems, in order to increase the number of infeasible solutions to be improved through generations and become feasible with true feasible region, the factor equation decreases linearly with $k = 1$ and the other parameter setting for each constrained problem is listed in Table 3. These settings will be discussed later on. In fact, to the best of our knowledge, this is the first study that introduces the tolerance value greater than 1.0E−10, so it shows the effectiveness of the proposed new equality constrained approach. All the above settings for local scaling factor, crossover rate equation and dynamic tolerance equation will be discussed in Section 6. The description of the COMDE algorithm is demonstrated in Fig. 1.

5.1.2. Results of the proposed approach (COMDE)

The statistical results of the COMDE on the benchmarks are summarized in Table 4. It includes the known optimal solution for each test problem and the obtained best, median, mean, worst values and the standard deviations. As described in Table 4, COMDE is able to find the global optimal solution consistently in 12 test functions over 30 runs with the exception of test function g02. With respect to test function g02, although the optimal solutions are not consistently found, the best result achieved is very close to the global optimal solution which can be verified by the very small standard deviation. Additionally, the standard deviation for test functions g04, g05, g06, g11, and g12 is equal to 0 and it is also relatively small in all other functions. It is worth to mentioning that COMDE does not need to evaluate the objective function for infeasible solutions, as the objective function and constraints are handled separately which make COMDE more efficient. For all equality constrained problems, the results provided by COMDE are almost equal to the global optimal or best known solution because the tolerance value is extremely small at the end search process. Ultimately, the number of fitness function evaluations used is almost very small in all test functions with high quality solution and extreme robustness. Therefore, the proposed COMDE algorithm is proven to be an effective and powerful approach to solve constrained global optimization problems. In addition to that, it is able to provide the competitive results. COMDE is also able to find new solutions to test functions g03 and g09 which are better than the best known solutions from [19]. The previous best known solutions, the new best known solutions and the parameters that yielded the respective results are displayed in Table 5. The optimal solution of G03 is (-1), so the improvement is surely due to allowed tolerance value to reach 10E−08 in equality constraints. Therefore, the new solution is closer to the optimum solution than the former solution reported, although the former is less than the new solution.

5.1.3. COMDE and dynamic tolerance for handling equality constraints

In order to verify the efficiency of the proposed dynamic tolerance rule to handle equality constraints, COMDE solves the benchmark problems g03, g05, 511 and g13 which contain equality constraints without using the proposed rule, applying a constant tolerance value corresponding to each problem (1.00E−08, 1.00E−08, 1.00E−12, and 1.00E−04) along the whole search process. Note that each problem has its own tolerance value. For each problem, 30 independent runs are performed and statistical results are provided, including the best, median, mean and worst results and the standard deviation. Table 6 presents the results of COMDE without new dynamic tolerance rule using the same population size and (TNFE) as Table 2. The presented results in Table 6 show that COMDE without the proposed dynamic tolerance was only able to find the best solution which is very close to the optimal solution in g05 and g11,but it was not able with g03 and g13, i.e. 2 out of 4 test problems. Moreover, the median, the mean, and the worst are far away from the optimal region in all test problems. Consequently, it can be deduced that the proposed dynamic tolerance rule has the main role in solving constrained problems. Moreover, the search process without proposed rule is completely deteriorated in all constrained problems with equality constraints.

5.1.4. Comparisons with evolutionary algorithms based on genetic algorithms

Firstly, in order to show the effectiveness and superiority of COMDE, it is compared with five state-of-the-art GA based approaches. These approaches are: the homomorphous mapping by Kozal and Michalewwicz's (denoted as HM) [15], GA with gradient-based repair by Chootinan and Chen's [2] (abbreviated as CC), GA with triangular crossover by Elfeky et al.'s (denoted as TC) [7], GA with a self adaptive penalty function by Tessema and Yen (abbreviated as SAPF) [41], and a new agent based memetic algorithm by Barkat Ullah et al. (denoted as AMA) [42]. The TNFE, the best, the median, the mean, the worst, and the standard deviation are listed in Table 7. The independent runs performed by [15,2,7,41,42] algorithms are 20,20,30,50 and 30, respectively. The results provided by these approaches were directly taken from the original references for each approach. From Table 7, for g01, it is clear that all the algorithms can find the optimal results consistently except SAPF and HM algorithms. For g02, AMA has lower standard deviation than COMDE. However, COMDE can find the optimal solution, but the other algorithms were unable to do it. For g03, COMDE and AMA reached the optimal solution in all runs performed. SAPF reached the optimal in its best case while HM could not reach the optimal. On the other hand, TC did not solve all constrained problems. For g04 and g05, COMDE and CC reached the optimal in all runs performed, but AMA and SAPF were unable to do it in all runs. For g06, COMDE and CC had a better performance than AMA and TC with 0 standard deviations, but HM was not able to reach the optimal solution in all runs. For g08, COMDE, AMA and TC had almost the better performance than SAPF and CC, but HM could not reach the optimal. Similarly, for g08, COMDE, AMA and TC had almost the better performance with 0 standard deviations than SAPF, but CC and MH did not solve it. Finally, for g07, g09,

01. Begin
02. G=0
03. Create a random initial population $\vec{x}_i^G \ \forall i, i = 1,...,NP$
04. Evaluate $f(\vec{x}_i^G), cv(\vec{x}_i^G), \forall i, i = 1,...,NP$
05. Determine x_{best}^G and x_{worst}^G based on $f(\vec{x}_i^G)$ and $cv(\vec{x}_i^G), \forall i, i = 1,...,NP$
06. **For** G=1 to GEN **Do**
07. CR=0.95+(0.5-0.95)·(1-G/GEN)^4
08. $Factor = \begin{cases} F_{final} + (F_{initial} - F_{final}) * (1 - G/GEN) \wedge k & , 0 < (G/GEN) \le R \\ F_{final} & , (G/GEN) > R \end{cases}$
09. **For** i=1 to NP **Do**
10. **If** (rand[0,1]<=0.5) **Then** (Use New Directed Mutation Scheme)
11. Select randomly $r1 \neq best \neq worst \neq i \in [1, NP]$
12. F_l =rand[0.4,0.6]
13. j_{rand}= randint(1,D)
14. **For** j=1 to D **Do**
15. **If** ($rand_i$[0,1]< CR **or** j= j_{rand}) **Then**
16. $u_{ij}^{G+1} = x_{r1}^G + F_l * (x_{best}^G - x_{worst}^G)$
17. **Else**
18. $u_{ij}^{G+1} = x_{ij}^G$
19. **End If**
20. **End For**
21. **Else** (Use Modified Basic Mutation Scheme)
22. Select randomly $r1 \neq r2 \neq r3 \neq i \in [1, NP]$
23. F_g =rand(-1,0) $\bigcup$ rand(0,1)
24. j_{rand}= randint(1,D)
25. **For** j=1 to D **Do**
26. **If** ($rand_i$[0,1]< CR **or** j= j_{rand}) **Then**
27. $u_{ij}^{G+1} = x_{r1}^G + F_g * (x_{r2}^G - x_{r3}^G)$
28. **Else**
29. $u_{ij}^{G+1} = x_{ij}^G$
30. **End If**
31. **End For**
32. **End If**
33. **If** ($\vec{u}_i^{G+1}$ is better than $\vec{x}_i^G$ (based on the three selection criteria)) **Then**
34. $\vec{x}_i^{G+1} = \vec{u}_i^{G+1}$
35. **Else**
36. $\vec{x}_i^{G+1} = \vec{x}_i^G$
37. **End If**
38. Determine x_{best}^{G+1} and x_{worst}^{G+1} based on $f(\vec{x}_i^{G+1})$ and $cv(\vec{x}_i^{G+1}), \forall i, i = 1,...,NP$
39. **End For**
40. G=G+1
41. **End For**
42. **End**

Fig. 1. Description of **COMDE** algorithm.

Table 4
The results of COMDE on the benchmarks.

Problem	Optimal	Best	Median	Mean	Worst	SD
g01	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**1.97E−13**
g02	**−0.803619**	**−0.803619**	**−0.803616**	**−0.801238**	**−0.785265**	**5.0E−03**
g03	**−1.000000**	**−1.000000049**	**−1.000000039**	**−1.000000027**	**−0.99999994**	**3.026E−08**
g04	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**0.00E+00**
g05	**5126.4981**	**5126.498109**	**5126.498109**	**5126.4981094**	**5126.4981094**	**0.00E+00**
g06	**−6961.8138**	**−6961.813875**	**−6961.813875**	**−6961.813875**	**−6961.813875**	**0.00E+00**
g07	**24.306209**	**24.306209**	**24.306209**	**24.306209**	**24.306211**	**4.7E−07**
g08	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**9.00E−18**
g09	**680.630057**	**680.630057**	**680.630057**	**680.630057**	**680.630057**	**4.071E−13**
g10	**7049.248020**	**7049.248020**	**7049.248020**	**7049.248077**	**7049.248615**	**1.5E−04**
g11	**0.750000**	**0.749999**	**0.749999**	**0.749999**	**0.749999**	**0.00E+00**
g12	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**0.00E+00**
g13	**0.0539415**	**0.0539415**	**0.0539415**	**0.0539415**	**0.0539415**	**1.4E−17**

Table 5
Better solution than the given best known solutions from [19].

Problem	Former $f(x)$	New $f(x^*)$	Parameter values
g03	−1.00050010001000	−1.0000000494686891	3.1622801911602222E−01
			3.1622564827382660E−01
			3.1622610640734011E−01
			3.1622726591038042E−01
			3.1623210785879297E−01
			3.1622918966687991E−01
			3.1622425404331617E−01
			3.1622609840831090E−01
			3.1623063868655948E−01
			3.1622834752275863E−01
g09	680.630057374402	680.63005737440176	2.3304993488400036E+00
			1.9513723728859003E+00
			−4.7754140616656954E−01
			4.3657262397533270E+00
			−6.2448696534810577E−00
			1.0381310439580118E+00
			1.5942266975790407E+00

Table 6
The results of COMDE without using the proposed dynamic tolerance.

Problem	Optimal	Best	Median	Mean	Worst	SD
g03	**−1.0000**	**−0.4158768**	**−0.0386126**	**−0.0670364**	**−0.0003986**	**8.98E−02**
g05	**5126.4981**	**5126.7803869**	**5241.922519**	**5293.338419**	**5793.443985**	**1.7484E+02**
g11	**0.7499**	**0.7500132**	**0.9508141**	**0.9102824**	**0.9999854**	**9.119E−02**
g13	**0.0539415**	**0.4388026**	**0.76017337**	**0.7450305**	**0.8739415**	**1.062E−01**

g10, g11, and g13, the performance of COMDE was superior to all other five algorithms. However, CC and HM did not solve g13 which is very difficult to solve. From the *t*-test results, it can be observed that COMDE are superior to, equal to, inferior to compared algorithms in 29, 17 and 0 cases, respectively out of the total 46 cases. The *t*-test has not been done in the remaining 19 cases. Thus, the COMDE is always either superior or comparable. Additionally, from Fig. 2, it can be observed that the COMDE algorithm requires less computational effort than the other five algorithms. Therefore, it is considered the fastest one with the least (TNFE) in addition to its extreme efficiency and robustness with the smallest standard deviations in almost problems. Moreover, it is noteworthy that the total (TNFE) that is used by COMDE to execute a single run for all 13 test problems is equal to 1,422,000 function evaluations. These function evaluations are slightly more than the (TNFE) which is 1,400,000 to execute only one run for one test problem using HM algorithm as well as the remarkable difference of the performance of two algorithms.

5.1.5. Comparisons with other evolutionary algorithms

To further verify the performance of COMDE, comparisons are carried out with five competitive evolutionary algorithms based on evolutionary strategy as well as genetic algorithms. The first two algorithms are Evolutionary Strategy based approaches, which are Improved Stochastic Ranking Evolution Strategy (ISRES) by Runarsson and Yao [36] and the simple

Table 7
Statistical features of the results obtained by of COMDE, AMA, TC, SAPF, CC and HM.

Problem	Features	COMDE	AMA [42]	TC [7]	SAPF [41]	CC [2]	HM [15]
g01	Best	**−15.000000**	**−15.000**	**−15.000**	**−15.000**	**−15.000**	−14.7864
	Median	**−15.000000**	**−15.000**	NA	−14.966	NA	NA
	Mean	**−15.000000**	**−15.000**	**−15.000**	−14.552	**−15.000**	−14.7082
	Worst	**−15.000000**	**−15.000**	NA	−13.097	**−15.000**	−14.6154
	SD	1.97E−13	7.31E−09	0.00E+00	7.00E−01	**0.00E+00**	NA
	TNFE	130,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	0	0	1	0	#
g02	Best	**−0.803619**	−0.803549	−0.803616	−0.803202	−0.801119	−0.79953
	Median	**−0.803616**	−0.803488	NA	−0.789398	NA	NA
	Mean	−0.801238	**−0.803495**	−0.791345	−0.755798	−0.785746	−0.79671
	Worst	−0.785265	**−0.803465**	NA	−0.745712	−0.745329	−0.79119
	SD	5.0E−03	**2.56E−05**	9.42E−03	1.3321E−01	1.37E−02	NA
	TNFE	200,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	0	1	1	1	#
g03	Best	**−1.000000049**	**−1.000**	NA	**−1.000**	−0.99998	−0.9997
	Median	**−1.000000039**	**−1.000**	NA	−0.971	NA	NA
	Mean	**−1.000000027**	**−1.000**	NA	−0.964	−0.99992	−0.9989
	Worst	−0.99999994	**−1.000**	NA	−0.887	−0.99979	−0.9978
	SD	**3.026E−08**	3.14E−08	NA	3.01E−01	5.99E−05	NA
	TNFE	150,000	350,000	–	500,000	350,000	1,400,000
	h	–	1	#	1	1	#
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**	−30665.401	**−30665.539**	−30664.500
	Median	**−30665.539**	**−30665.539**	NA	−30663.921	NA	NA
	Mean	**−30665.539**	**−30665.539**	−30665.531	−30665.922	**−30665.539**	−30655.300
	Worst	**−30665.539**	−30665.538	NA	−30656.471	**−30665.539**	−30645.900
	SD	**0.00E+00**	1.64E−04	9.16E−03	2.043E+00	**0.00E+00**	NA
	TNFE	50,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	0	1	1	0	#
g05	Best	**5126.4981094**	5126.514	NA	5126.907	**5126.4981**	NA
	Median	**5126.4981094**	5134.349	NA	5208.897	NA	NA
	Mean	**5126.4981094**	5148.967	NA	5214.232	**5126.4981**	NA
	Worst	**5126.4981094**	5482.953	NA	5564.642	**5126.4981**	NA
	SD	**0.00E+00**	6.41E+01	NA	2.4747E+02	**0.00E+00**	NA
	TNFE	200,000	350,000	–	500,000	350,000	–
	h	–	1	#	1	0	#
g06	Best	**−6961.813875**	**−6961.814**	**−6961.814**	−6961.046	**−6961.8139**	−6942.100
	Median	**−6961.813875**	**−6961.814**	NA	−6953.823	NA	NA
	Mean	**−6961.813875**	**−6961.814**	**−6961.814**	−6953.061	**−6961.8139**	−6342.6000
	Worst	**−6961.813875**	**−6961.814**	NA	−6943.304	**−6961.8139**	−5473.90
	SD	**0.00E+00**	2.88E−08	3.70E−12	5.876E+00	**0.00E+00**	NA
	TNFE	12,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	0	0	1	0	#
g07	Best	**24.306209**	24.325	24.505	24.838	24.3294	24.620
	Median	**24.306209**	24.378	NA	25.415	NA	NA
	Mean	**24.306209**	24.392	25.057	27.328	24.4719	24.826
	Worst	24.306211	24.491	NA	33.095	24.8352	25.069
	SD	**4.7E−07**	5.18E−02	2.38E−01	2.172E+00	1.29E−01	NA
	TNFE	200,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	1	1	1	1	#
g08	Best	**−0.095825**	**−0.095825**	−0.095825	**−0.095825**	**−0.095825**	−0.0958250
	Median	**−0.095825**	**−0.095825**	NA	**−0.095825**	NA	NA
	Mean	**−0.095825**	**−0.095825**	−0.095825	−0.095635	**−0.095825**	−0.0991568
	Worst	**−0.095825**	**−0.095825**	NA	−0.092697	**−0.095825**	−0.0291438
	SD	**9.00E−18**	4.23E−17	4.23E−17	1.055E−03	2.70E−09	NA
	TNFE	4000	350,000	350,000	500,000	350,000	1,400,000
	h	–	0	0	1	0	#
g09	Best	**680.630057**	680.631	680.633	680.773	680.6303	680.91
	Median	**680.630057**	680.726	NA	681.235	NA	NA
	Mean	**680.630057**	680.721	680.659	681.246	680.6381	681.16
	Worst	**680.630057**	680.802	NA	682.081	680.6538	683.18
	SD	**4.071E−13**	5.26E−02	1.98E−02	3.22E−01	6.61E−03	NA
	TNFE	70,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	1	1	1	1	#

(*continued on next page*)

Table 7 (*continued*)

Problem	Features	COMDE	AMA [42]	TC [7]	SAPF [41]	CC [2]	HM [15]
g10	Best	**7049.248020**	7280.436	7049.474	7069.981	7049.2607	7147.9
	Median	**7049.248020**	7498.673	NA	7201.017	NA	NA
	Mean	**7049.248077**	7473.925	7493.719	7238.964	7049.5659	8163.6000
	Worst	**7049.248615**	7598.573	NA	7489.406	7051.6857	9659.3000
	SD	**1.5E−04**	9.71E+01	3.87E+02	1.3777E+02	5.70E−01	NA
	TNFE	200,000	350,000	350,000	500,000	350,000	1,400,000
	h	–	1	1	1	1	#
g11	Best	**0.749999999**	**0.75**	NA	**0.749**	**0.75**	**0.75**
	Median	**0.749999999**	**0.75**	NA	**0.750**	NA	NA
	Mean	**0.749999999**	**0.75**	NA	0.751	**0.75**	**0.75**
	Worst	**0.749999999**	**0.75**	NA	0.757	**0.75**	**0.75**
	SD	**0.00E+00**	8.37E−09	NA	2.00E−03	3.21E−08	NA
	TNFE	50,000	350,000	–	500,000	350,000	1,400,000
	h	–	0	#	1	0	#
g12	Best	**−1.000000**	**−1.000**	**−1.000**	**−1.000**	NA	NA
	Median	**−1.000000**	**−1.000**	NA	**−1.000**	NA	NA
	Mean	**−1.000000**	**−1.000**	**−1.000**	−0.999940	NA	NA
	Worst	**−1.000000**	**−1.000**	NA	−0.999548	NA	NA
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.41E−04	NA	NA
	TNFE	6,000	350,000	350,000	500,000	–	–
	h	–	0	0	1	#	#
g13	Best	**0.0539415**	0.05395	NA	**0.053941**	NA	NA
	Median	**0.0539415**	0.05413	NA	0.054713	NA	NA
	Mean	**0.0539415**	0.05413	NA	0.286270	NA	NA
	Worst	**0.0539415**	0.05434	NA	0.885276	NA	NA
	SD	**1.4E−17**	1.16E−04	NA	2.7546E−01	NA	NA
	TNFE	150,000	350,000	–	500,000	–	–
	h	–	1	#	1	#	#

A result in boldface indicates the best result or the global optimum. NA means not available.

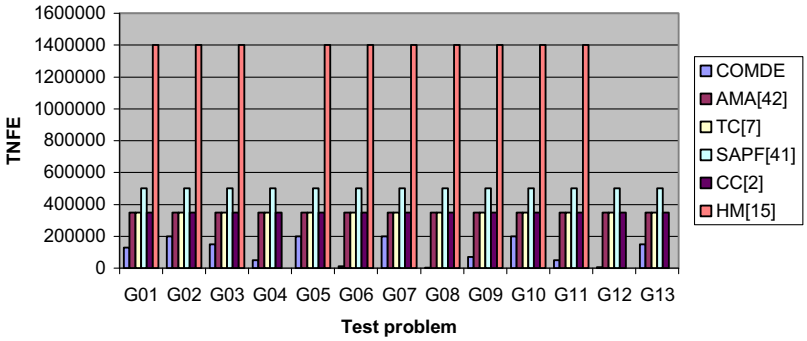

Fig. 2. Comparisons of COMDE with other competitive GA based algorithms in terms of (TNFE) used in all test problems.

multimember evolutionary strategy (SMES) by Mezura-Montes and Coello [24]. The other three competitive evolutionary approaches are α constrained simplex method (abbreviated to α Simplex) by Takahama and Sakai [40], the hybrid constrained optimization EA (abbreviated to HCOEA) by Wang et al.[46], and hybrid evolutionary algorithm with adaptive constraint-handling (denoted as HEA-ACT) by Wang et al.[47]. The TNFE, the best, the median, the mean, the worst, and the standard deviation are listed in Table 8. The independent runs performed by [36,24,40,46,47] algorithms are 100,30,30,30 and 30, respectively. The results provided by these approaches were directly taken from the original references for each approach. From Table 8, it can be obviously seen that almost all the six algorithms can find the optimal solution consistently for five test functions (g03, g04, g08, g11, g12), but the results provided by COMDE for g03 is closer to the optimal solution than all other algorithms, although they practically provided the better solution. Those results are more infeasible in theory than the proposed results. For g01, COMDE, ISRES,SMES, and HEA-ACT find the similar "best", "median", "mean", and "worst" results, but SMES was more robust with 0 standard deviation. However, HCOEA did not reach the optimal only in its worst case and α Simplex could not reach the optimum in all cases. For g02, COMDE, ISRES, and α Simplex find the optimal solution, but the other algorithms were not able to do it. However, the better "median" result was found by COMDE and it also has the lower standard deviation. For g05, G06, g07 and g09 almost all six algorithms can find the optimal results consistently except

Table 8
Statistical features of the results obtained by of COMDE, ISRES, SMES, α Simplex, HCOEA and HEA–ACT.

Problem	Features	**COMDE**	ISRES [36]	SMES [24]	α Simplex [40]	HCOEA [46]	HEA–ACT [47]
g01	Best	**−15.000000**	**−15.000**	**−15.000**	−14.9999998	**−15.000**	**−15.000**
	Median	**−15.000000**	**−15.000**	**−15.000**	−14.9999990	**−15.000**	**−15.000**
	Mean	**−15.000000**	**−15.000**	**−15.000**	−14.9999951	**−15.000**	**−15.000**
	Worst	**−15.000000**	**−15.000**	**−15.000**	−14.9999765	−14.999998	**−15.000**
	SD	1.97E−13	5.8E−14	**0.00E+00**	6.4E−06	4.29E−07	7.7E−11
	TNFE	130,000	350,000	240,000	303,162	240,000	200,000
	h	–	0	0	0	0	0
g02	Best	**−0.803619**	**−0.803619**	−0.803601	**−0.8036191**	−0.803241	−0.803582
	Median	**−0.803616**	−0793082	−0792549	−0.7851627	−0.802556	−0.767844
	Mean	−0.801238	−0.782715	−0.785238	−0.7841868	**−0.801258**	−0.758182
	Worst	−0.785265	−0.723591	−0.751322	−0.7542585	**−0.792363**	−0.673096
	SD	5.0E−03	2.2E−02	1.67E−02	1.3E−02	**3.832E−03**	3.2E−02
	TNFE	200,000	349,600	240,000	328,931	240,000	200,000
	h	–	1	1	1	0	1
g03	Best	**−1.000000049**	**−1.001**	**−1.000**	**−1.0005001**	**−1.00000000**	**−1.000**
	Median	**−1.000000039**	**−1.001**	**−1.000**	**−1.0005001**	**−1.00000000**	**−1.000**
	Mean	**−1.000000027**	**−1.001**	**−1.000**	**−1.0005001**	**−1.00000000**	**−1.000**
	Worst	−0.99999994	**−1.001**	**−1.000**	**−1.0005001**	**−1.00000000**	**−1.000**
	SD	3.026E−08	8.2E−09	2.09E−04	8.5E−14	1.304E−12	**5.2E−15**
	TNFE	150,000	349,200	240,000	310,968	240,000	200,000
	h	–	−1	0	−1	1	1
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Median	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Mean	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Worst	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	SD	**0.00E+00**	1.1E−11	**0.00E+00**	4.2E−11	5.404E−07	7.4E−12
	TNFE	50,000	192,000	240,000	305,343	240,000	200,000
	h	–	0	0	0	0	0
g05	Best	**5126.4981094**	**5126.497**	5126.599	**5126.4971**	**5126.4981**	**5126.498**
	Median	**5126.4981094**	**5126.497**	5160.198	**5126.4971**	**5126.4981**	**5126.498**
	Mean	**5126.4981094**	**5126.497**	5147.492	**5126.4971**	**5126.4981**	**5126.498**
	Worst	**5126.4981094**	**5126.497**	5304.167	**5126.4971**	**5126.4981**	**5126.498**
	SD	**0.00E+00**	7.2E−13	50.06E+0	3.5E−11	1.727E−07	9.3E−13
	TNFE	200,000	195,600	240,000	308,389	240,000	200,000
	h	–	0	1	0	0	0
g06	Best	**−6961.813875**	**−6961.814**	**−6961.814**	**−6961.81387**	**−6961.8138**	**−6961.814**
	Median	**−6961.813875**	**−6961.814**	**−6961.814**	**−6961.81387**	**−6961.8138**	**−6961.814**
	Mean	**−6961.813875**	**−6961.814**	−6961.284	**−6961.81387**	**−6961.8138**	**−6961.814**
	Worst	**−6961.813875**	**−6961.814**	−6952.482	**−6961.81387**	**−6961.8138**	**−6961.814**
	SD	**0.00E+00**	1.9E−12	1.85E+0	1.3E−10	8.507E−12	4.6E−12
	TNFE	12,000	168,800	240,000	293,367	240,000	200,000
	h	–	0	1	0	0	0
g07	Best	**24.306209**	**24.306**	24.327	**24.3062096**	**24.3064582**	**24.306**
	Median	**24.306209**	**24.306**	24.426	**24.3062158**	24.3073055	**24.306**
	Mean	**24.306209**	**24.306**	24.475	**24.3062625**	24.3073989	**24.306**
	Worst	**24.306211**	**24.306**	24.843	**24.3068044**	24.3092401	**24.306**
	SD	**4.7E−07**	6.3E−05	1.32E−01	1.3E−04	7.118E−04	**1.9E−11**
	TNFE	200,000	350,000	240,000	317,587	240,000	200,000
	h	–	0	1	0	1	0
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Mean	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	SD	9.00E−18	2.7E−17	**0.00E+00**	3.8E−11	2.417E−17	2.8E−17
	TNFE	4000	160,000	240,000	306,248	240,000	200,000
	h	–	0	0	0	0	0
g09	Best	**680.6300574**	**680.630**	680.632	**680.6300574**	**680.6300574**	**680.630**
	Median	**680.6300574**	**680.630**	680.642	**680.6300574**	**680.6300574**	**680.630**
	Mean	**680.6300574**	**680.630**	680.643	**680.6300574**	**680.6300574**	**680.630**
	Worst	**680.6300574**	**680.630**	680.719	**680.6300574**	680.6300578	**680.630**
	SD	**4.071E−13**	3.2E−13	1.55E−02	2.9E−10	9.411E−8	5.8E−13
	TNFE	70,000	271,200	240,000	323,426	240,000	200,000
	h	–	0	1	0	0	0

(*continued on next page*)

Table 8 (*continued*)

Problem	Features	**COMDE**	ISRES [36]	SMES [24]	α Simplex [40]	HCOEA [46]	HEA–ACT [47]
g10	Best	**7049.248020**	**7049.248**	7051.903	**7049.2480207**	7049.286598	**7049.248**
	Median	**7049.248020**	**7049.248**	7253.603	**7049.2480208**	7049.486145	**7049.248**
	Mean	**7049.248077**	7049.250	7253.047	7049.2480217	7049.525438	**7049.248**
	Worst	**7049.248615**	7049.270	7638.366	7049.2480468	7049.984208	**7049.248**
	SD	1.5E−04	3.2E−03	136.02E+0	**4.7E−06**	1.502E−01	1.4E−05
	TNFE	200,000	348,800	240,000	316,037	240,000	200,000
	h	–	1	1	0	1	0
g11	Best	**0.749999999**	**0.75**	**0.75**	**0.7499000**	**0.75**	**0.75**
	Median	**0.749999999**	**0.75**	**0.75**	**0.7499000**	**0.75**	**0.75**
	Mean	**0.749999999**	**0.75**	**0.75**	**0.7499000**	**0.75**	**0.75**
	Worst	**0.749999999**	**0.75**	**0.75**	**0.7499000**	**0.75**	**0.75**
	SD	**0.00E+00**	1.1E−16	1.52E−02	4.9E−16	1.546E−12	3.4E−16
	TNFE	50,000	137,200	240,000	308,125	240,000	200,000
	h	–	1	1	1	1	1
g12	Best	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Median	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Mean	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Worst	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	SD	**0.00E+00**	1.2E−09	**0.00E+00**	3.9E−10	**0.00E+00**	**0.00E+00**
	TNFE	6,000	33,600	240,000	30,283	240,000	200,000
	h	–	0	0	0	0	0
g13	Best	**0.0539415**	**0.053942**	0.053986	**0.0539415**	**0.0539498**	**0.0539498**
	Median	**0.0539415**	**0.053942**	0.061873	**0.0539415**	**0.0539498**	**0.0539498**
	Mean	**0.0539415**	0.066770	0.166385	0.0667702	**0.0539498**	**0.0539498**
	Worst	**0.0539415**	0.438803	0.468294	0.4388026	**0.0539499**	**0.0539498**
	SD	**1.4E−17**	7.0E−02	1.77E−01	6.9E−02	8.678E−08	1.5E−15
	TNFE	150,000	223,600	240,000	311,144	240,000	200,000
	h	–	0	1	0	1	1

A result in boldface indicates the best result or the global optimum. NA means not available.

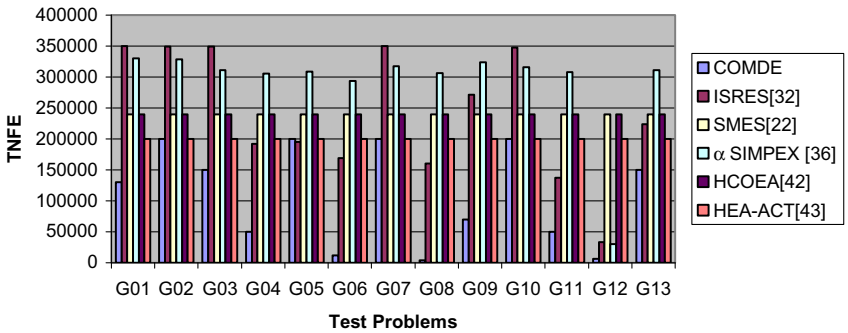

Fig. 3. Comparisons of COMDE with other competitive evolutionary algorithms in terms of (TNFE) used in all test problems.

SMES, but the lower variability with 0 standard deviation was provided by COMDE for g05 and g06 and it was also provided by HEA-ACT for g07. For test function g10, better "best", "median", "mean", "worst" results were reached by COMDE, ISRES, and HEA-ACT algorithms. α Simplex reached the optimum only in its best and median cases. HCOEA could not reach the optimum and SMES was far away from the feasible region almost in all cases. In the problem g13, COMDE, HCOEA, and HEA-ACT can find the optimal result consistently. However, ISRES, SMES, as well as α Simplex were sometimes prematurely converged. From the *t*-test results, we can observe that COMDE are superior to, equal to, inferior to compared algorithms in 22, 41 and 2 cases, respectively out of the total 65 cases. Thus, the COMDE is almost either better or equal. Furthermore, from Fig. 3, it can be observed that the COMDE algorithm requires less computational effort than the other five algorithms except HEA-ACT which had the same (TNFE) equal 200,000 for g02, g05, g07 and g10, so it still remained the fastest one in most problems.

5.1.6. Comparisons with hybrid approaches based on differential evolution

Indeed, in recent years, Differential Evolution has been embedded into most of the evolutionary algorithms such as Particle Swarm Optimization and Culture Evolution as well as traditional direct methods such as Simplex methods so as to increase the exploration capability of the designed algorithms. Therefore, the performance of the proposed COMDE algorithm

Table 9
Statistical features of the results obtained by of COMDE, CDE, HDESMCO, HMPSO, COPSO and PSO–DE.

Problem	Features	**COMDE**	CDE [18]	HDESMCO [20]	HMPSO [45]	COPSO [32]	PSO–DE [21]
g01	Best	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**
	Median	**−15.000000**	NA	NA	**−15.000000**	**−15.000000**	**−15.000000**
	Mean	**−15.000000**	−14.999996	−14.515000	**−15.000000**	**−15.000000**	**−15.000000**
	Worst	**−15.000000**	−14.999996	−9.000000	**−15.000000**	**−15.000000**	**−15.000000**
	SD	1.97E−13	2.0E−06	8.15E−01	**0.00E+00**	**0.00E+00**	2.1E−08
	TNFE	130,000	100,100	176,000	300,000	350,000	140,100
	h	–	1	1	0	0	0
g02	Best	**−0.803619**	**−0.803619**	−0744986	**−0.803619**	**−0.803619**	−0.8036145
	Median	**−0.803616**	NA	NA	−0.803617	**−0.803619**	−0.7620745
	Mean	**−0.801238**	−0.724886	−0514042	−0.798110	−0.799859	−0.7566775
	Worst	**−0.785265**	−0.590908	−0.362386	−0.776242	−0.787296	−0.63679947
	SD	**5.0E−03**	7.012E−02	4.738E−02	8.1E−03	5.016E−03	3.3E−02
	TNFE	**200,000**	100,100	176,000	300000	350,000	140,100
	h	–	1	1	0	0	1
g03	Best	**−1.000000049**	−0.995413	−1.0005	**−1.0005**	**−1.000005**	**−1.0050100**
	Median	**−1.000000039**	NA	NA	**−1.0005**	**−1.000005**	**−1.0050100**
	Mean	**−1.000000027**	−0.788635	−0.960465	**−1.0005**	**−1.000005**	**−1.0050100**
	Worst	−0.99999994	−0639920	0	**−1.0005**	**−1.000005**	**−1.0050100**
	SD	3.026E−08	1.152E−01	7.68E−02	3.8E−06	**0.00E+00**	3.8E−12
	TNFE	150,000	100,100	176,000	300,000	350,000	140,100
	h	–	1	1	−1	−1	−1
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Median	**−30665.539**	NA	NA	**−30665.539**	**−30665.539**	**−30665.539**
	Mean	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Worst	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.1E−11	**0.00E+00**	8.3E−10
	TNFE	50,000	100,100	176,000	300,000	350,000	70,100
	h	–	0	0	0	0	0
g05	Best	**5126.4981094**	**5126.57092**	**5126.497**	**5126.496714**	**5126.498096**	NA
	Median	**5126.4981094**	**NA**	**NA**	**5126.496714**	**5126.498096**	NA
	Mean	**5126.4981094**	**5207.41065**	**5126.497**	**5126.496714**	**5126.498096**	NA
	Worst	**5126.4981094**	**5327.39049**	**5126.497**	**5126.496714**	**5126.498096**	NA
	SD	**0.00E+00**	6.9225E+01	**0.00E+00**	1.1E−08	**0.00E+00**	NA
	TNFE	200,000	100,100	176,000	300,000	350,000	NA
	h	–	1	0	0	0	#
g06	Best	**−6961.813875**	**−6961.81387**	**−6961.81387**	**−6961.813875**	**−6961.813875**	**−6961.81387**
	Median	**−6961.813875**	NA	NA	**−6961.813875**	**−6961.813875**	**−6961.81387**
	Mean	**−6961.813875**	**6961.81387**	**−6961.81387**	**−6961.813875**	**−6961.813875**	**−6961.81387**
	Worst	**−6961.813875**	**6961.81387**	**−6961.81387**	**−6961.813875**	**−6961.813875**	**−6961.81387**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.9E−12	**0.00E+00**	2.3E−09
	TNFE	12,000	100,100	176,000	300,000	350,000	140,100
	h	–	0	0	0	0	0
g07	Best	**24.306209**	**24.306209**	**24.306**	**24.306209**	**24.306209**	**24.3062091**
	Median	**24.306209**	NA	NA	**24.306209**	**24.306209**	**24.3062096**
	Mean	**24.306209**	24.306210	**24.306**	**24.306209**	**24.306209**	24.3062100
	Worst	24.306211	24.306212	**24.306**	**24.306209**	24.306218	24.3062172
	SD	4.7E−07	1.0E−06	**0.00E+00**	9.7E−14	1.611E−06	1.3E−06
	TNFE	200,000	100,100	176,000	300,000	350,000	140,100
	h	–	0	−1	0	0	0
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	NA	NA	**−0.095825**	**−0.095825**	**−0.095825**
	Mean	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	SD	9.00E−18	**0.00E+00**	**0.00E+00**	2.4E−17	**0.00E+00**	1.3E−12
	TNFE	4000	100,100	176,000	300,000	350,000	10,600
	h	–	0	0	0	0	0
g09	Best	**680.630057**	**680.630057**	**680.630**	**680.630057**	**680.630057**	**680.630057**
	Median	**680.630057**	NA	NA	**680.630057**	**680.630057**	**680.630057**
	Mean	**680.630057**	**680.630057**	**680.630**	**680.630057**	**680.630057**	**680.630057**
	Worst	**680.630057**	**680.630057**	**680.630**	**680.630057**	**680.630057**	**680.630057**
	SD	4.071E−13	**0.00E+00**	**0.00E+00**	4.5E−13	**0.00E+00**	4.6E−13
	TNFE	70,000	100,100	176,000	300,000	350,000	140,100
	h	–	0	0	0	0	0

(*continued on next page*)

Table 9 (*continued*)

Problem	Features	**COMDE**	CDE [18]	HDESMCO [20]	HMPSO [45]	COPSO [32]	PSO–DE [21]
g10	Best	**7049.248020**	**7049.24805**	**7049.248**	**7049.248020**	**7049.248020**	**7049.248021**
	Median	**7049.248020**	NA	NA	**7049.248020**	**7049.248020**	**7049.248028**
	Mean	**7049.248077**	**7049.24826**	**7049.248**	**7049.248020**	**7049.248074**	**7049.248038**
	Worst	**7049.248615**	**7049.24848**	**7049.248**	**7049.248020**	7049.249209	7049.249223
	SD	1.5E−04	1.67E−04	0.00E+00	8.1E−12	2.228E−04	3.0E−05
	TNFE	200,000	100,100	176,000	300,000	350,000	140,100
	h	–	1	0	0	0	0
g11	Best	**0.749999999**	**0.749900**	**0.75**	**0.749900**	**0.749999999**	**0.749999**
	Median	**0.749999999**	NA	NA	**0.749900**	**0.749999999**	**0.749999**
	Mean	**0.749999999**	0.757995	**0.75**	**0.749900**	**0.749999999**	**0.749999**
	Worst	**0.749999999**	0.796455	**0.75**	**0.749900**	**0.749999999**	0.750001
	SD	**0.00E+00**	1.713E−02	**0.00E+00**	1.1E−16	**0.00E+00**	2.5E−07
	TNFE	50,000	100,100	176,000	300,000	350,000	70,100
	h	–	0	0	0	0	0
g12	Best	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Median	**−1.000000**	NA	NA	**−1.000000**	**−1.000000**	**−1.000000**
	Mean	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Worst	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	6,000	100,100	176,000	300,000	350,000	17,600
	h	–	0	0	0	0	0
g13	Best	**0.0539415**	0.056180	0.053942	0.0539415	**0.053950**	NA
	Median	**0.0539415**	NA	NA	0.438803	**0.053950**	NA
	Mean	**0.0539415**	0.288324	0.259401	0.342101	**0.053950**	NA
	Worst	**0.0539415**	0.392100	0.438841	0.438960	**0.053950**	NA
	SD	1.4E−17	1.67E−01	1.901E−01	1.5E−1	**0.00E+00**	NA
	TNFE	150,000	100,100	176,000	300,000	350,000	NA
	h	–	1	1	1	0	#

A result in boldface indicates the best result or the global optimum. NA means not available.

is also compared with five competitive hybrid approaches based on the differential evolution algorithm. The competitive approaches are Cultured Differential Evolution (denoted as CDE) by Becerra and Coello [18], hybrid of Differential Evolution and the Nelder-Mead method (denoted as HDESMCO) by Menchaca-Mendez and Ceollo [20], A Hybrid multi-swarm Particle Swarm (denoted as HMPSO) by Wang and Cai [45], Continuous Optimization via Particle Swarm Optimization (denoted as COPSO) by Muñoz el al.[32], and PSO hybrid with DE (denoted as PSO-DE) by Wang et al.[21]. The TNFE, the best, the median, the mean, the worst, and the standard deviation are listed in Table 9. The independent runs performed by [18,20,45,32,21] algorithms are 30, 100, 30, 30 and 100, respectively. The results provided by these approaches were directly taken from the original references for each approach. As shown in Table 9, almost all the six algorithms can find the optimal solution consistently for six test functions (g04, g06, g07, g08, g09, and g12). For problem g01, COMDE, HMPSO, COPSO, and PSO-DE find the similar ''best'', ''median'', ''mean'', and ''worst'' results, but HMPSO and COPSO were more robust with 0 standard deviation. However, CDE and HDESMCO reach the optimal only in their best case. For g02, COMDE, CDE, HMPSO, and COPSO find the optimal solution, but the other algorithms were not able to do it. However, the better ''mean'' result has been found by COMDE and it also has the lower standard deviation. As previously mentioned, for g03, the results provided by COMDE for g03 is very closer to the optimal solution than all other algorithms, although they practically provided the better solution.

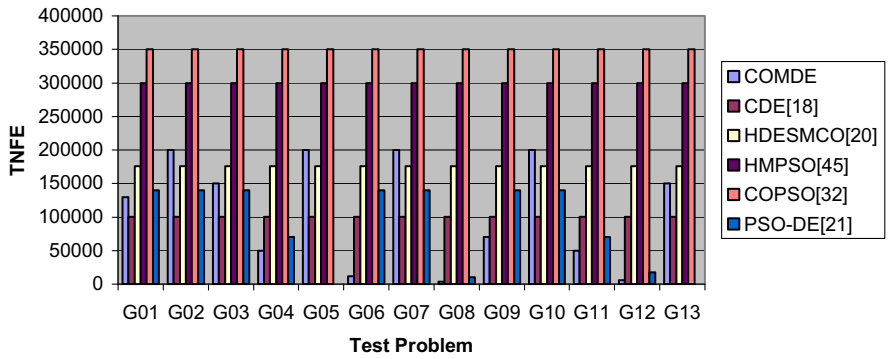

Fig. 4. Comparisons of COMDE with other competitive hybrid approaches based on differential evolution in terms of (TNFE) used in all test problems.

Table 10
Statistical features of the results obtained by of COMDE, CHDE, DDE, A–DDE, DSS–MDE, DECV and DELC.

Problem	Features	**COMDE**	CHDE [25]	DDE [27]	A–DDE [29]	DSS–MDE [49]	DECV [26]	DELC [44]
g01	Best	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**	**−15.000000**
	Median	**−15.000000**	**−15.000000**	NA	**−15.000000**	NA	NA	**−15.000000**
	Mean	**−15.000000**	−14.792134	**−15.000000**	**−15.000000**	**−15.000000**	**−14.855**	**−15.000000**
	Worst	**−15.000000**	−12.743044	**−15.000000**	**−15.000000**	**−15.000000**	**−13.000**	**−15.000000**
	SD	1.97E−13	4.01E−01	1.0E−09	7.00E−6	7.7E−11	4.59E−01	**6.5E−15**
	TNFE	130,000	348,000	225,000	180,000	225,000	240,000	150,000
	h	–	1	0	0	0	1	0
g02	Best	**−0.803619**	**−0.803619**	**−0.803619**	−0.803605	−0.803619	−0.704009	**−0.803619**
	Median	−0.803616	−0.800445	NA	−0.777368	NA	NA	**−0.803618**
	Mean	**−0.801238**	−0.746236	−0.798079	−0.771090	−0.786970	−0.569458	−0.798877
	Worst	**−0.785265**	−0.302179	−0.751742	−0.609853	−0.728531	−0.238203	−0.777443
	SD	**5.0E−03**	8.1E−02	1.02E−02	3.66E−02	3.2E−02	9.51E−02	7.7E−03
	TNFE	200,000	348,000	225,000	180,000	225,000	240,000	250,000
	h	–	1	0	1	1	1	0
g03	Best	**−1.000000049**	**−1.00000**	**−1.0000**	**−1.0000**	**−1.0005**	−0.461	**−1.0000**
	Median	**−1.000000039**	−0.702939	**NA**	**−1.0000**	**NA**	NA	**−1.0000**
	Mean	**−1.000000027**	−0.640326	**−1.0000**	**−1.0000**	**−1.0005**	−0.134	**−1.0000**
	Worst	−0.99999994	−0.029601	**−1.0000**	**−1.0000**	**−1.0005**	−0.002	**−1.0000**
	SD	3.026E−08	2.39E−01	**0.00E+00**	9.30E−12	1.9E−09	1.17E−01	2.2E−06
	TNFE	150,000	348,000	225,000	180,000	225,000	240,000	200,000
	h	–	1	0	1	0	1	0
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Median	**−30665.539**	**−30665.539**	NA	**−30665.539**	NA	NA	**−30665.539**
	Mean	**−30665.539**	−30592.154	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	Worst	**−30665.539**	−29986.214	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.539**
	SD	**0.00E+00**	1.0877E+02	**0.00E+00**	3.20E−13	2.7E−11	1.56E−06	1.0E−11
	TNFE	50,000	348,000	225,000	180,000	225,000	240,000	50,000
	h	–	1	0	0	0	0	0
g05	Best	**5126.4981094**	**5126.49671**	**5126.497**	**5126.497**	**5126.497**	**5126.497**	**5126.498**
	Median	**5126.4981094**	5231.55763	NA	**5126.497**	NA	NA	**5126.498**
	Mean	**5126.4981094**	5218.72911	**5126.497**	**5126.497**	**5126.497**	**5126.497**	**5126.498**
	Worst	**5126.4981094**	5502.41039	**5126.497**	**5126.497**	**5126.497**	**5126.497**	5126.500
	SD	**0.00E+00**	7.6422E+01	**0.00E+00**	2.10E−11	**0.00E+00**	**0.00E+00**	5.1E−04
	TNFE	**200,000**	348,000	225,000	180,000	225,000	240,000	200,000
	h	–	1	0	0	0	0	0
g06	Best	**−6961.813875**	**−6961.8138**	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**
	Median	**−6961.813875**	**−6961.8138**	NA	**−6961.814**	NA	NA	**−6961.814**
	Mean	**−6961.813875**	−6367.5754	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**
	Worst	**−6961.813875**	−2236.9503	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**
	SD	**0.00E+00**	7.7080E+02	**0.00E+00**	2.11E−12	**0.00E+00**	**0.00E+00**	7.3E−10
	TNFE	12,000	348,000	225,000	180,000	225,000	240,000	20,000
	h	–	1	0	0	0	0	0
g07	Best	**24.306209**	**24.306**	**24.306**	**24.306**	**24.306**	**24.306**	**24.306**
	Median	**24.306209**	24.482980	NA	**24.306**	NA	NA	**24.306**
	Mean	**24.306209**	104.599221	**24.306**	**24.306**	**24.306**	24.794	**24.306**
	Worst	**24.306211**	1120.54149	**24.306**	**24.306**	**24.306**	29.511	**24.306**
	SD	4.7E−07	1.7676E+02	**8.22E−09**	4.02E−05	7.5E−07	1.37E+00	1.5E−06
	TNFE	200,000	348,000	225,000	180,000	225,000	240,000	200,000
	h	–	1	0	0	0	1	0
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	NA	**−0.095825**	NA	NA	**−0.095825**
	Mean	**−0.095825**	**−0.091292**	**−0.095825**	**− 0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	−0.027188	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	SD	9.00E−18	1.2E−02	**0.00E+00**	9.10E−10	4.0E−17	4.23E−17	1.0E−17
	TNFE	4000	348,000	225,000	180,000	225,000	240,000	5,000
	h	–	1	0	0	0	0	0
g09	Best	**680.630057**	**680.6300**	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**
	Median	**680.630057**	680.639178	NA	**680.630**	NA	NA	**680.630**
	Mean	**680.630057**	692.472322	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**
	Worst	**680.630057**	839.782911	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**
	SD	4.071E−13	2.3575E+01	**0.00E+00**	1.15E−10	2.9E−13	3.45E−07	3.2E−12
	TNFE	70,000	348,000	225,000	180,000	225,000	240,000	80,000
	h	–	1	0	0	0	0	0

(*continued on next page*)

Table 10 (*continued*)

Problem	Features	**COMDE**	CHDE [25]	DDE [27]	A–DDE [29]	DSS–MDE [49]	DECV [26]	DELC [44]
g10	Best	**7049.248020**	**7049.24802**	**7049.248**	**7049.248**	**7049.248**	**7049.248**	**7049.248**
	Median	**7049.248020**	7137.41530	NA	**7049.248**	NA	NA	**7049.248**
	Mean	**7049.248077**	8442.65694	7049.266	**7049.248**	7049.249	7103.548	**7049.248**
	Worst	**7049.248615**	15580.3703	7049.617	**7049.248**	7049.225	7808.980	**7049.249**
	SD	1.5E−04	2186.5	4.45E−02	3.23E−04	1.4E−03	1.48E+02	**1.4E−04**
	TNFE	200,000	348,000	225,000	180,000	225,000	240,000	200,000
	h	–	1	1	0	1	1	0
g11	Best	**0.749999999**	0.749	0.75	0.75	0.7499	0.75	0.75
	Median	**0.749999999**	0.749	NA	0.75	NA	NA	0.75
	Mean	**0.749999999**	0.761823	0.75	0.75	0.7499	0.75	0.75
	Worst	**0.749999999**	0.870984	0.75	0.75	0.7499	0.75	0.75
	SD	**0.00E+00**	2.0E−02	0.00E+00	5.35E−15	0.00E+00	1.12E−16	0.00E+00
	TNFE	50,000	348,000	225,000	180,000	225,000	240,000	50,000
	h	–	1	0	0	0	0	0
g12	Best	**−1.000000**	**−1.000000**	**−1.000**	**−1.000**	**−1.000**	**−1.000**	**−1.000**
	Median	**−1.000000**	**−1.000000**	NA	**−1.000**	NA	NA	**−1.000**
	Mean	**−1.000000**	**−1.000000**	**−1.000**	**− 1.000**	**−1.000**	**−1.000**	**−1.000**
	Worst	**−1.000000**	**−1.000000**	**−1.000**	**− 1.000**	**−1.000**	**−1.000**	**−1.000**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.10E−9	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	6,000	348,000	225,000	180,000	225,000	240,000	5,000
	h	–	0	0	0	0	0	0
g13	Best	**0.0539415**	0.053866	**0.053941**	**0.053942**	**0.053942**	0.059798	**0.0539498**
	Median	**0.0539415**	0.980831	NA	**0.053942**	NA	NA	**0.0539498**
	Mean	**0.0539415**	0.747227	0.069336	0.079627	**0.053942**	0.382401	**0.0539498**
	Worst	**0.0539415**	2.259875	0.438803	0.438803	**0.053942**	0.999094	**0.0539499**
	SD	**1.4E−17**	3.13E−01	7.58E−02	9.60E−02	1.0E−13	2.68E−01	4.7E−09
	TNFE	150,000	348,000	225,000	180,000	225,000	240,000	200,000
	h	–	1	1	1	0	1	0

A result in boldface indicates the best result or the global optimum. NA means not available.

Those results are more infeasible in theory than the proposed results. Nevertheless, HDESMCO reaches the optimal only in its best case and CDE could not reach the optimum in all cases. For g05, COMDE, HDESMCO, HMPSO and COPSO reached the optimal solution consistently, but the results obtained by COMDE were closer to the optimal solution than the other approaches. However, CDE was unable to find the optimum solution in all cases. Additionally, PSO-DE did not solve g05 and g13. For g10, almost all the six algorithms can find the optimal result consistently, but the lower variability with 0 standard deviation has been provided by HDESMCO. Finally, the performance of COMDE and COPSO are similar for g11 and g13 and they were able to find the optimal solutions in all 30 runs consistently and HDESMCO. Besides, HMPSO has had better results than CDE to g11. However, they sometimes could be trapped in local solution with g13. From the *t*-test results, it can be observed that COMDE are superior to, equal to, inferior to compared algorithms in 12, 47 and 4 cases, respectively out of the total 63 cases. The t-test has not been done in the remaining two cases. Thus, the COMDE is almost either better or equal. Furthermore, regarding the computation cost, from Fig. 4, it can be observed that the (TNFE) used in HDESMCO, PSO-DE and CDE was smaller in 4 and 6 out of 13 problems than the (TNFE) used in COMDE, HMPSO, and COPSO. However, the suggested approach still remains the fastest one in the remaining problems. On the other hand, regarding the quality of final solution and robustness, COMDE was able to find the optimal solutions in all 30 runs consistently in 12 test functions except test function g2, while CDE, HDESMCO, and PSO-DE did it in only 5, 8 and 10 test functions, respectively. It is also emphasized that, in most cases, COMDE is more robust than CDE, HDESMCO, and PSO-DE. All in all, COMDE is superior to CDE, HDESMCO, and PSO-DE as well as competitive with HMPSO, and COPSO with the smallest (TNFE) used.

5.1.7. Comparisons with competitive approaches based on differential evolution

In order to demonstrate the efficiency and superiority of the proposed COMDE algorithm, six competitive approaches based on DE algorithms are used for comparison. These approaches are Simple Feasibility Rule and Differential Evolution (denoted as CHDE) by Mezura-Montes [25], Differential Evolution in Constrained numerical Optimization: An Empirical Study (denoted as DECV) by Mezura-Montes et al. [26], Promising Infeasibility and Multiple Offspring Incorporated to Differential Evolution (denoted as DDE) by Mezura-Montes [27], Self-adaptive Differential Evolution (denoted as A-DDE) by Mezura-Montes [29], Differential Evolution with Dynamic Stochastic Selection (denoted as DEDS) [49], and Effective Differential Evolution with Level Comparison (denoted as DELC) by Wang et al. [45]. The TNFE, the best, the median, the mean, the worst, and the standard deviation are listed in Table 10. 30, 30, 100, 30, 100 and 30 independent runs are performed by [25–27,29,49,45] algorithms, respectively. The results provided by these approaches were directly taken from the original references for each approach. As shown in Table 10, it is obvious that COMDE, DDE, A-DDE, DSS-MDE, and DELC exhibit better high quality results for all benchmark problems than CHDE and DECV. They were able to reach the global optimal solution

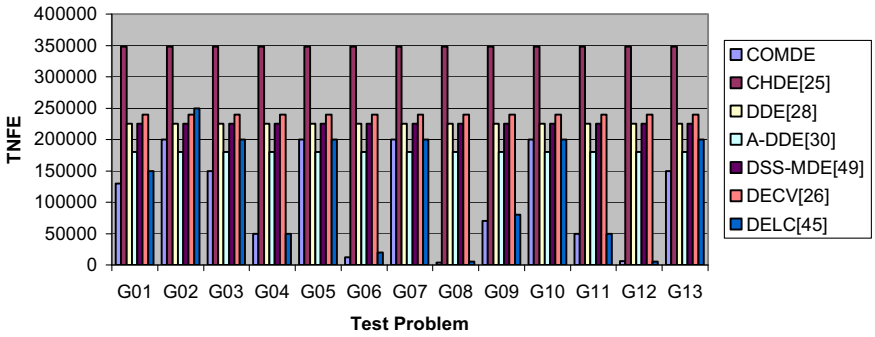

Fig. 5. Comparisons of COMDE with other competitive approaches based on differential evolution in terms of (TNFE) used in all test problems.

consistently in 10 test functions over 30 runs with the exception of test functions g02, g10, and g13, while CHDE was able to find the same results only with test function g12. Besides, DECV was able to find the same results in seven test functions over 30 runs with the exception of the test functions g01, g02, g03, g07, g10 and g13. For g02, all algorithms were able to find the optimum in their best cases, but A-DDE could not reach the optimum. Additionally, COMDE finds better "mean" and "worst" results. For g10, as g02, all algorithms have been able to find the optimum in their best cases, but the better "worst" result was provided by COMDE and A-DDE. CHDE and DECV show a poor performance in this test function. For g13, which is a very difficult problem to solve, COMDE surpass all other algorithms in terms of all performance metrics except DSS-MDE. Nevertheless, DDE and A-DDE could be prematurely converged. From the t-test results, it can be observed that COMDE are superior to, equal to, inferior to compared algorithms in 25, 53 and 0 cases, respectively out of the total 78 cases. Thus, the COMDE is almost either better or equal. Furthermore, from Fig. 5, it can be noticed that the COMDE algorithm requires less computational effort than CHDE, DDE, DECV and DSS-MDE. Thus, it still remains the fastest one in almost problems. A-DDE performed 180,000 (TNFE), 20,000 less than COMDE in four test functions which are g02, g05, g07 and g10, but it has not a significant difference for the comparison. Concretely, the (TNFE) used by COMDE, is less than or equal to the (TNFE) that is used by DELC except for g12 which is 1000 more than DELC. In the final analysis, the performance of the COMDE algorithm is superior and competitive to the state-of-the-art well-known Evolutionary algorithms in terms of final solution quality, convergence rate, efficiency and robustness. Moreover, it is easily implemented and a reliable approach to constrained optimization problem.

5.2. Simulation on engineering design problems

In order to study the performance of the proposed algorithm COMDE on real-world constrained optimization problems, five well-studied engineering design problems that are widely used in literature have been solved. For each problem, 30 independent runs are performed and statistical results are provided including the best, median, mean and worst results and the standard deviation. Moreover, the statistical results provided by COMDE were compared with some algorithms used in literature. The same settings of parameters are used and the required population size, max generation (G_{max}), and the total

Table 11
The required population size, maximum generation and number of fitness function evaluations for engineering design problems.

Problem	n	NP	G_{max}	TNFE
Welded beam	4	40	500	20,000
Spring design	3	60	400	24,000
Speed reducer	7	30	700	21,000
Three-bar truss	2	40	175	7000
Pressure vessel	4	100	300	30,000

Table 12
Experimental results obtained by COMDE with 30 independent runs on five engineering design problems.

Problem	Optimal	Best	Median	Mean	Worst	SD
Welded beam	**1.724852309**	**1.724852309**	**1.724852309**	**1.724852309**	**1.724852309**	1.60E−12
Spring design	**0.012665233**	**0.012665232**	**0.012665423**	**0.012667168**	**0.012676809**	3.09E−06
Speed reducer	**2994.4710661**	**2994.4710661**	**2994.4710661**	**2994.4710661**	**2994.4710661**	1.54E−12
Three-bar truss	**263.89584338**	**263.8958433**	**263.8958433**	**263.8958433**	**263.8958433**	5.34E−13
Pressure vessel	**6059.714355**	**6059.714335**	**6059.714335**	**6059.714335**	**6059.714335**	3.62E−10

A result in boldface indicates the best result or the global optimum. NA means not available.

number of function evaluation (TNFE) for each test function are listed in Table 11. To get good and satisfactory results with all problems, the simple population size rule has only been used with spring design and three-bar truss problems. The suitable population size for Welded beam, Speed reducer and Pressure vessel are 40, 30, and 100, respectively. The statistical results of all engineering design problems that measure the quality of results (best, median, mean, and worst) as well as the robustness of COMDE (standard deviation) are summarized in Table 12. From Table 12, it can be concluded that COMDE is able to consistently find the global optimal in all engineering problems with a very small standard deviation and with a very small (TNFE) which indicates that the proposed COMDE has a remarkable ability to solve constrained engineering design problems with a perfect performance in terms of high quality solution, rapid convergence speed, efficiency and robustness.

5.2.1. Welded beam design problem

This problem has been studied by several approaches in literature, including feasibility based Genetic Algorithm inspired by the multi-objective optimization (denoted as FGA) by Coello and Mezura-Montes [22], Culture Evolutionary algorithm (abbreviated as CEA) by Coello and Baccera [4], Co-evolutionary Particle Swarm optimization (denoted as CPSO) by He and Wang [11], Hybrid Particle Swarm optimization with feasibility rules (denoted as HPSO) by He and Wang [12], Co-evolutionary Differential Evolution (abbreviated by CDE) by Huang el al [13], and Differential Evolution with Level Comparison (denoted as DELC) by Wang and Li [44]. The statistical results of all algorithms and COMDE are listed in Table 13. The independent runs performed by [22,4,11–13,44] algorithms are 30, 10, 30, 30, 50 and 30, respectively. For the best solution provided by COMDE, the constraints are [−1.8190E−12, 0, 0, −3.4329837853, −0.0807296397, −0.2355403225, −3.638E−12], and the objective function value is 1.72485230859736480. As can be seen from Table 13 that, COMDE is able to find the optimal solution consistently and its statistical result is similar to DELC with the same used (TNFE) which is the smallest among all competitive algorithms. Therefore, from the t-test results, it can be deduced that COMDE is superior to all algorithms and is competitive with DELC with high quality final solution with lower standard deviation in solving the welded beam design problem.

5.2.2. Spring design problem

The approaches applied to this problem for the comparison include [44,4,12,13] besides DEDS [49], and HEA-ACT [47]. The statistical results of all algorithms and COMDE are listed in Table 14. The independent runs performed by [44,4,12,13,49,47] algorithms are 30, 10, 30, 50, 50 and 30, respectively. For the best results obtained by COMDE, the constraints are [−4.041E−12, −1.454E−12, −4.05378500049, −0.72772902084], and the objective function value is 0.01266523278840155100. From Table 14, it is clear that COMDE can reach the optimal solution as well as it outperformed CEA and CDE in terms of the "best", "mean", and "worst" and the standard deviation. Additionally, it further performs better than DEDS and HPSO in terms of "mean", "worst" besides the standard deviation. Although HEA-ACT produces the best statistical results, DELC used the smallest (TNFE) among all algorithms. Finally, from the t-test results, the results obtained by COMDE are equal to DELC and are superior to the results produced by CEA, HPSO and CDE. However, COMDE is inferior to DELC and HEA-ACT. Consequently, COMDE can solve spring design problem efficiently.

Table 13
Results of welded beam design problem.

Algorithm	Best	Median	Mean	Worst	SD	TNFE	*h*
COMDE	**1.724852**	**1.724852**	**1.724852**	**1.724852**	1.60E−12	20,000	–
DELC [44]	**1.724852**	**1.724852**	**1.724852**	**1.724852**	**4.10E−13**	20,000	0
FGA [22]	1.728226	NA	1.792654	1.993408	7.4E−02	80,000	1
CEA [4]	**1.724852**	NA	1.971809	3.179709	4.4E−01	50,020	1
CPSO [11]	1.728024	NA	1.748831	1.782143	1.3E−02	200,000	1
HPSO [12]	**1.724852**	NA	1.749040	1.814295	4.0E−02	81,000	1
CDE [13]	1.733461	NA	1.768158	1.824105	2.2E−02	240,000	1

A result in boldface indicates the best result or the global optimum. NA means not available.

Table 14
Results of tension/compression spring design problem.

Algorithm	Best	Median	Mean	Worst	SD	TNFE	*h*
COMDE	**0.012665233**	0.012665423	0.012667168	0.012676809	3.09E−06	24,000	–
DELC [44]	**0.012665233**	**0.012665233**	0.012665267	**0.012665575**	**1.3E−07**	20,000	−1
DEDS [49]	**0.012665233**	0.012665304	0.012669366	0.012738262	1.3E−05	24,000	0
HEA–ACT [47]	**0.012665233**	0.012665234	**0.012665234**	0.012665240	1.4E−09	24,000	−1
CEA [4]	0.0127210	NA	0.0135681	0.015116	8.4E−04	50,020	1
HPSO [12]	0.0126652	NA	0.0127072	0.0127191	1.6E−05	81,000	1
CDE [13]	0.0126702	NA	0.012703	0.012790	2.7E−05	240,000	1

A result in boldface indicates the best result or the global optimum. NA means not available.

Table 15
Results of speed reducer design problem.

Algorithm	Best	Median	Mean	Worst	SD	TNFE	*h*
COMDE	**2994.4710661**	**2994.4710661**	**2994.4710661**	**2994.4710661**	**1.54E−12**	21,000	–
DELC [44]	**2994.471066**	**2994.471066**	**2994.471066**	**2994.471066**	1.90E−12	30,000	0
DEDS [49]	**2994.471066**	**2994.471066**	**2994.471066**	**2994.471066**	3.60E−12	30,000	0
HEA–ACT [47]	2994.499107	2994.599748	2994.613368	2994.752311	7.0E−02	40,000	1
SBO [34]	2994.744241	3001.758264	3001.758264	3009.964736	4.0E+00	54,456	1
MDE [25]	2996.356689	NA	2996.367220	NA	8.20E−03	24,000	1

A result in boldface indicates the best result or the global optimum. NA means not available.

Table 16
Results of three-bar truss design problem.

Algorithm	Best	Median	Mean	Worst	SD	TNFE	*h*
COMDE	**263.8958434**	**263.8958434**	**263.8958434**	**263.8958434**	5.34E−13	7,000	–
DELC [44]	**263.8958434**	**263.8958434**	**263.8958434**	**263.8958434**	**4.34E−14**	10,000	0
DEDS [49]	263.8958433	263.8958433	263.8958436	263.8958498	9.7E−07	15,000	0
HEA–ACT [47]	263.895843	263.895848	263.895865	263.896099	4.9E−05	15,000	0
SBO [34]	263.8958466	263.8989	263.9033	263.96975	1.3E−02	17,610	1

A result in boldface indicates the best result or the global optimum. NA means not available.

Table 17
Results of pressure vessel design problem.

Algorithm	Best	Median	Mean	Worst	SD	TNFE	*h*
COMDE	**6059.714335**	**6059.714335**	**6059.714335**	**6059.714335**	3.62E−10	30,000	–
DELC [44]	**6059.7143**	**6059.7143**	**6059.7143**	**6059.7143**	**2.1E−11**	30,000	0
FGA [22]	6059.9463	NA	6177.2533	6469.3220	130.9	50,020	1
CPSO [11]	6061.0777	NA	6147.1332	6363.8041	86.5	200,000	1
HPSO [12]	6059.7143	NA	6099.9323	6288.6770	86.2	81,000	1
CDE [13]	6061.0777	NA	6085.2303	6371.0455	43.0	240,000	1

A result in boldface indicates the best result or the global optimum. NA means not available.

Table 18
Experimental comparisons between COMDE, COMDE_1, COMDE_2, and COMDE_3 on 13 benchmark problems over 30 independent runs.

Problem	Features	COMDE	COMDE_3	COMDE_2	COMDE_1
g01	Best	**−15.000000**	**−15.000000**	−14.99999999	−14.9999999
	Median	**−15.000000**	**−15.000000**	−14.99999999	−14.9999990
	Mean	**−15.000000**	**−15.000000**	−14.99999999	−14.9999804
	Worst	**−15.000000**	**−15.000000**	−14.99999999	−14.9998746
	SD	1.97E−13	**0.00E+00**	1.28E−08	3.95E−05
	TNFE	130,000	130,000	130,000	130,000
g02	Best	**−0.803619**	−0.803618	−0.803615	−0.8036091
	Median	**−0.803616**	−0.803609	−0.8035993	−0.7939353
	Mean	**−0.801238**	−0.799306	−0.7999478	−0.7926019
	Worst	**−0.785265**	−0.771747	−0.7782909	−0.7585091
	SD	**5.0E−03**	8.6E−03	6.0E−03	1.07E−02
	TNFE	200,000	200,000	200,000	200,000
g03	Best	**−1.000000049**	**−1.000000048**	−0.999996698	−0.99999910
	Median	**−1.000000039**	**−1.000000041**	−0.999967863	−0.999992437
	Mean	**−1.000000027**	**−1.000000029**	−0.999938594	−0.999990989
	Worst	−0.99999994	**−0.99999998**	−0.999752074	−0.999976201
	SD	**3.026E−08**	4.192E−08	6.98E−05	7.88E−06
	TNFE	150,000	150,000	150,000	150,000
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.53867**
	Median	**−30665.539**	**−30665.539**	**−30665.539**	−30665.53844
	Mean	**−30665.539**	**−30665.539**	**−30665.539**	−30665.52739
	Worst	**−30665.539**	**−30665.539**	**−30665.539**	−30665.29895
	SD	**0.00E+00**	**0.00E+00**	6.8E−09	4.37E−02
	TNFE	50,000	50,000	50,000	50,000
g05	Best	**5126.4981094**	**5126.4981094**	**5126.498109**	**5126.498109**
	Median	**5126.4981094**	**5126.4981094**	5136.204554	5126.498112
	Mean	**5126.4981094**	**5126.4981094**	5191.249586	5184.276828
	Worst	**5126.4981094**	**5126.4981094**	5387.658077	5552.440104

(*continued on next page*)

Table 18 (*continued*)

Problem	Features	COMDE	COMDE_3	COMDE_2	COMDE_1
	SD	**0.00E+00**	1.02E−013	8.6916E+01	1.2108E+02
	TNFE	200,000	200,000	200,000	200,000
g06	Best	**−6961.813875**	**−6961.813875**	**−6961.813875**	**−6961.813875**
	Median	**−6961.813875**	**−6961.813875**	**−6961.813875**	−6961.810116
	Mean	**−6961.813875**	**−6961.813875**	**−6961.813875**	−6961.721582
	Worst	**−6961.813875**	**−6961.813875**	−6961.813876	−6960.354454
	SD	**0.00E+00**	**0.00E+00**	1.4E−06	2.81E−01
	TNFE	12,000	12,000	12,000	12,000
g07	Best	**24.306209**	**24.306209**	24.306368	24.306812
	Median	**24.306209**	**24.306209**	24.310323	24.320316
	Mean	**24.306209**	24.306215	24.313842	24.328770
	Worst	**24.306211**	24.306339	24.353354	24.423982
	SD	**4.7E−07**	3.4E−06	1.059E−02	2.54E−02
	TNFE	200,000	200,000	200,000	200,000
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Mean	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	SD	**9.00E−18**	1.20E−17	3.90E−17	5.36E−16
	TNFE	4000	4000	4000	4000
g09	Best	**680.630057**	**680.630057**	680.630058	680.630083
	Median	**680.630057**	**680.630057**	680.630360	680.631610
	Mean	**680.630057**	**680.630057**	680.631136	680.635623
	Worst	**680.630057**	**680.630057**	680.636583	680.665080
	SD	**4.071E−13**	1.7E−08	1.69E−03	8.93E−03
	TNFE	70,000	70,000	70,000	70,000
g10	Best	**7049.248020**	**7049.248020**	7049.250587	7049.459986
	Median	**7049.248020**	7049.248092	7057.679949	7062.797670
	Mean	**7049.248077**	7049.248212	7061.410605	7070.536034
	Worst	**7049.248615**	7049.249390	7124.050941	7165.356012
	SD	**1.5E−04**	3.6E−04	1.57853E+01	2.555E+01
	TNFE	200,000	200,000	200,000	200,000
g11	Best	**0.749999999**	**0.749999999**	**0.749999999**	**0.749999999**
	Median	**0.749999999**	**0.749999999**	**0.749999999**	**0.749999999**
	Mean	**0.749999999**	**0.749999999**	**0.75**	**0.75**
	Worst	**0.749999999**	**0.750000000**	**0.75**	**0.75**
	SD	**0.00E+00**	4.0E−15	3.00E−12	7.00E−12
	TNFE	50,000	50,000	50,000	50,000
g12	Best	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Median	**−1.000000**	**−1.000000**	**−1.000000**	**−1.000000**
	Mean	**−1.000000**	**−1.000000**	**−0.99999999**	**−1.000000**
	Worst	**−1.000000**	**−1.000000**	**−0.99999999**	**−1.000000**
	SD	**0.00E+00**	**0.00E+00**	**1.61E−16**	**0.00E+00**
	TNFE	6,000	6,000	6,000	6,000
g13	Best	**0.0539415**	**0.0539415**	**0.0539415**	**0.0539415**
	Median	**0.0539415**	**0.0539415**	**0.0539415**	0.0539418
	Mean	**0.0539415**	**0.0539415**	0.0667709	0.1052579
	Worst	**0.0539415**	**0.0539415**	0.4388042	0.4388148
	SD	**1.4E−17**	2.00E−13	7.026E−02	1.33E−01
	TNFE	150,000	150,000	150,000	150,000

A result in boldface indicates the best result or the global optimum. NA means not available.

5.2.3. Speed reducer design problem

This problem has been solved by society and civilization algorithm (abbreviated to SBO) by Ray and Liew [34], by increasing successful offspring and diversity in differential mutation (denoted as MDE) by Mezura-Montes et al. [28] as well as [47,49,44]. The statistical results of all algorithms and COMDE are listed in Table 15. 50, 30, 30, 50 and 30 independent runs are performed by [34,28,47,49,44] algorithms, respectively. For the best results obtained by COMDE, the constraints are [−0.07391528039, −0.19799852714, −0.49917224810, −0.90464390455, −1.332E−15,0, −0.7025000000,0, −0.7958333333, −0.05132575354, −5.56E−16], and the objective function value is 2994.47106614682020. It can be seen from Table 15 that COMDE, DELC, and DEDS are able to find the optimal solution consistently in all runs but COMDE has the lower standard deviation. Moreover, COMDE has the better "best", "median", "mean", "worst" and standard deviation results than those obtained by HEA-ACT and SBO. In fact, the worst result provided by COMDE is better than the best result obtained by MDE. From the *t*-test results, it is obvious that the performance of COMDE is superior to SBO and MDE algorithms

and it is equal to HEA-ACT, DELC, and DEDS algorithms. It is to be noted that the (TNFE) of COMDE is less than half of the (TNFE) used in HEA-ACT and SBO and it only used 70% of the (TNFE) used in DELC and DEDS. Therefore, COMDE is considered the most efficient with the smallest (TNFE).

5.2.4. Three-bar truss design problem

This problem has been solved by [47,49,44,34]. The statistical results of all algorithms and COMDE are listed in Table 16. 30, 50, 30 and 50 independent runs are performed by [47,49,44,34] algorithms, respectively. For the best results obtained by

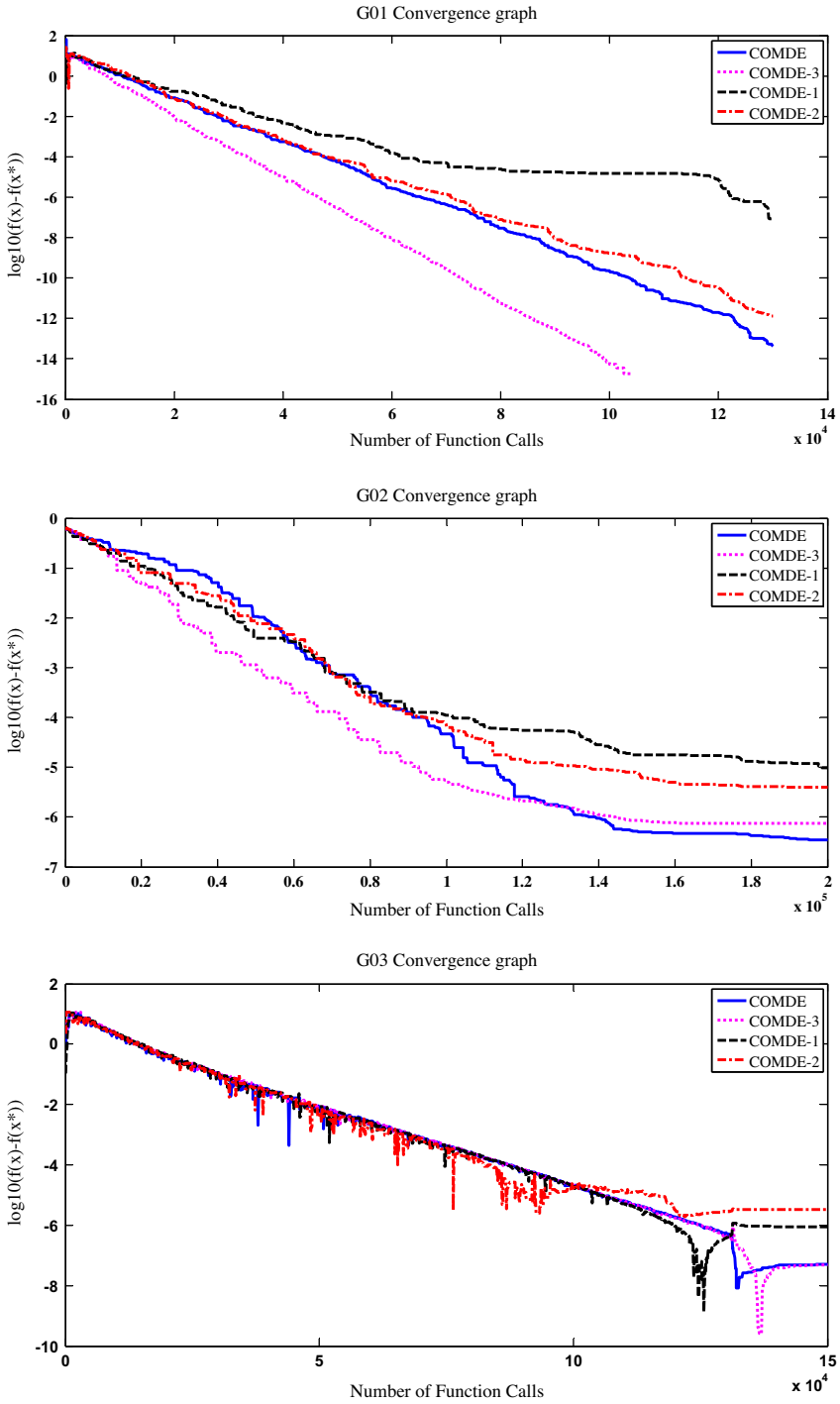

Fig. 6. The convergence graph of COMDE, COMDE_1, COMDE_2 and COMDE_3 algorithms for all 13 benchmark problems.

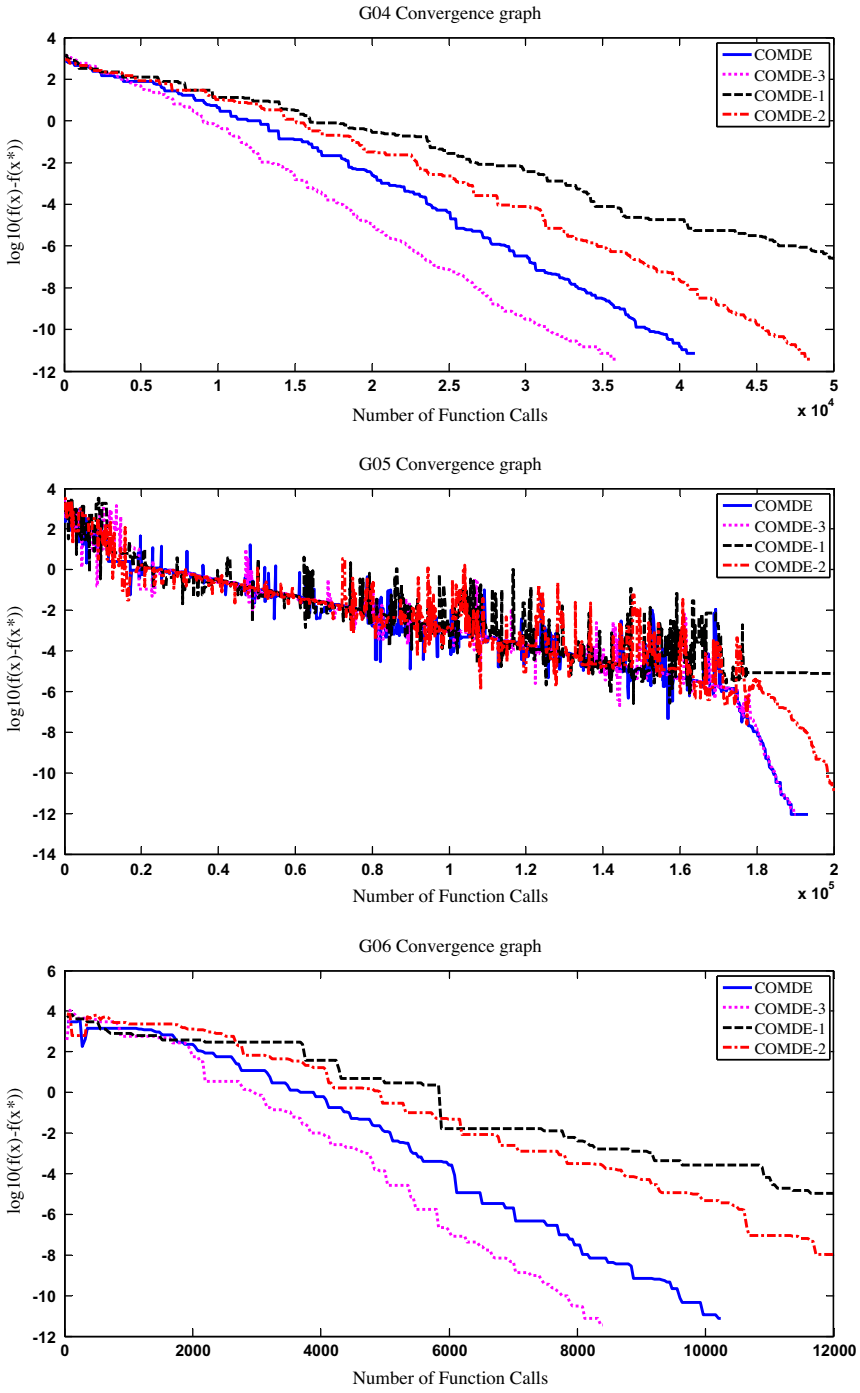

Fig. 6 (*continued*)

COMDE, the constraints are [−8.14E−13, −1.464101628960505, −0.535898371040309], and the objective function value is 263.895843376468. It can be seen from Table 16 that COMDE and DELC are able to find the optimal solution consistently in all the 30 runs but DELC has the lower standard deviation. Moreover, COMDE has the better "mean", "worst" and standard deviation results than those obtained by HEA-ACT and DEDS. In fact, the worst result provided by COMDE is better than the best result obtained by SBO. Finally, from the t-test results, the results obtained by COMDE are equal to DELC, DEDS and HEA-ACT and are superior to the results produced by SBO. It is to be noted that the (TNFE) of COMDE is less than half of the (TNFE) used in DEDS, HEA-ACT and SBO and it also used 70% of the (TNFE) used in DELC. Thus, COMDE is considered the most efficient with the smallest (TNFE).

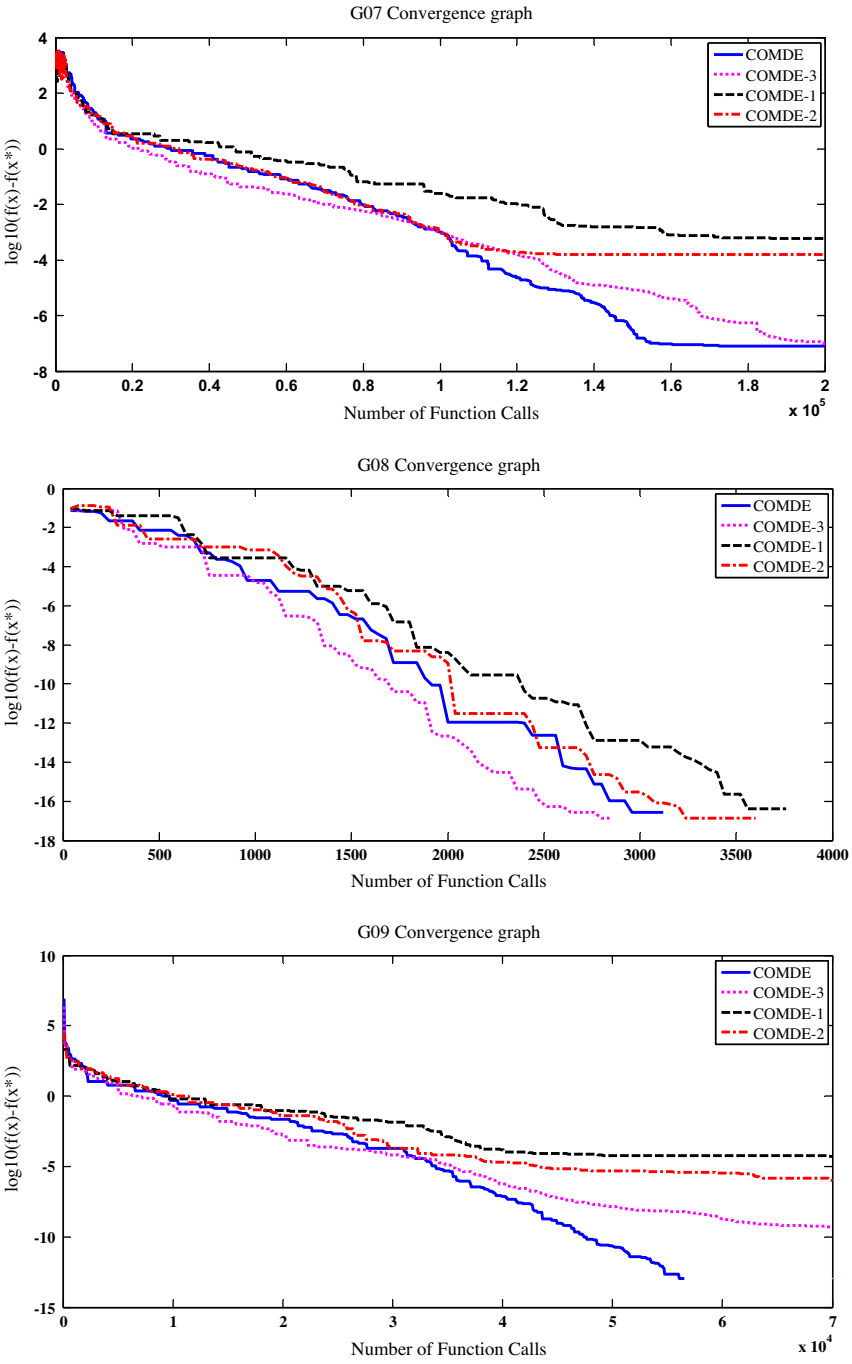

Fig. 6 (*continued*)

5.2.5. Pressure vessel design problem

This problem has been solved by [44,22,11–13]. The statistical results of all algorithms and COMDE are listed in Table 17. 30, 30, 30, 30 and 50 independent runs are performed by [44,22,11–13] algorithms, respectively. For the best results obtained by COMDE, the constraints are [0, −0.0358808290, −8.6147E−09, −63.3634041575], and the objective function value is 6059.714335048443. It can be seen from Table 17 that COMDE and DELC are able to find the optimal solution consistently in all the 30 runs. Moreover, from the *t*-test results, COMDE is superior to all other algorithms. Furthermore, the (TNFE) of COMDE is less than half of the (TNFE) used in CPSO, HPSO and CDE and it also used 60% of the (TNFE) used in FGA. Thus, COMDE is considered the most efficient with the smallest (TNFE). Based on the above analysis, results and comparisons,

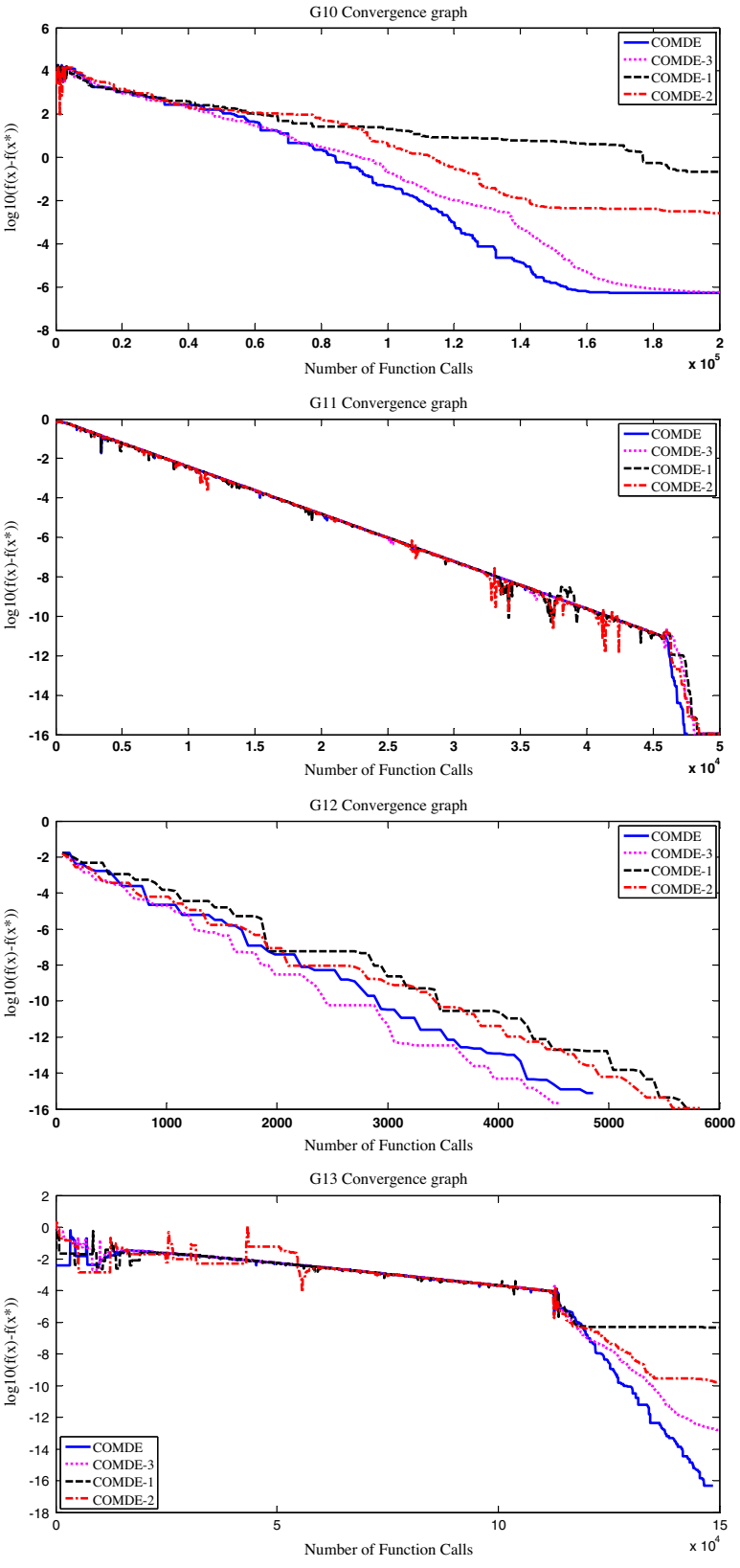

Fig. 6 (*continued*)

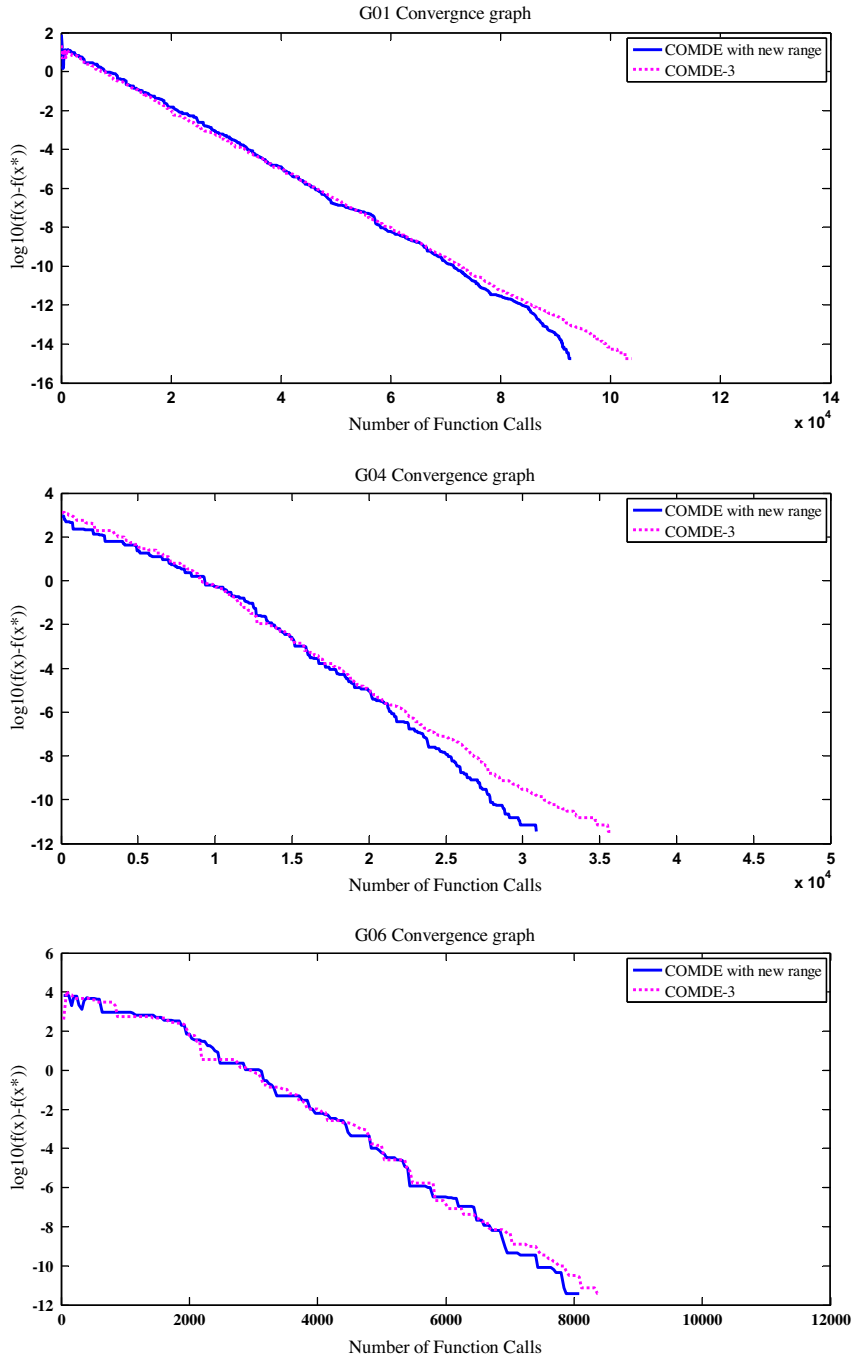

Fig. 7. The convergence graph of COMDE with new range and COMDE_3 for g01, g04, g06, g08, and g12.

the proposed COMDE algorithm is of better searching quality, efficiency and robustness for solving constrained engineering design problems and besides its performance is superior and competitive with all compared algorithms.

6. Discussion

In this section, some experiments are carried out so as to identify the key parameters that significantly lead to the perfect performance of our proposed approach by studying the effects of the modified basic differential evolution, new directed mutation scheme, the crossover rate equation as well as the dynamic tolerance equation for equality constraints on the

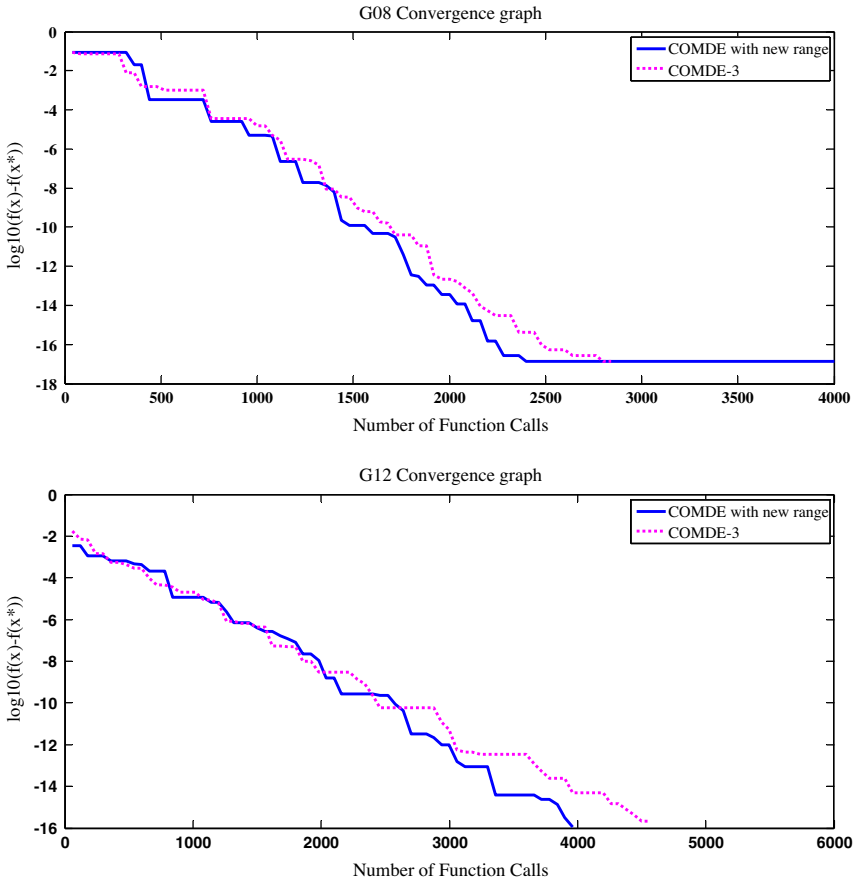

Fig. 7 (*continued*)

performance of COMDE algorithm. Practically, in order to make a fair comparison, the same parameters of CR equation, dynamic tolerance equation for equality constraints and (TNFE) are used in all experiments.

6.1. Effectiveness of new mutation scheme

In this subsection, some trials have been preformed to evaluate the benefits and effectiveness of the proposed modified basic mutation and new directed mutation on the performance of COMDE. Three different versions of COMDE have been tested and compared against the proposed one.

1. Version 1: COMDE with new directed mutation only and F_l is randomly generated within $(0, 1)$ (denoted as COMDE_1).
2. Version 2: COMDE with modified directed mutation only and F_l is randomly generated within $[(-1, 0) \cup (0, 1)]$ (denoted as COMDE_2).
3. Version 3: COMDE without new directed mutation, only using modified basic mutation scheme, as previously mentioned (denoted as COMDE_3).

The TNFE, the best, the median, the mean, the worst, and the standard deviation are listed in Table 18. The results of Table 18 explain that COMDE_1 was only able to find the global optimal solution in all 30 runs consistently for g08, g11, and g12. Moreover, the global optimal is reached in almost all problems. However, the standard deviation over 30 runs for all the remaining functions except g01, g02, and g03 considerably increases. It is clear that, as previously mentioned in Section 4.1, the directed mutation resembles the concept of gradient. Accordingly, the poor performance of COMDE_1 algorithm is due to the fact that it has great local exploitation tendency with weak global exploration ability that lead to rapid loss of the diversity that causes premature convergence as well as stagnation with almost benchmark problems. On the other hand, in order to increase exploration capability, the opposite direction is allowed, i.e. some vectors follow the direction from the best to the worst by different perturbations. Obviously, it can be seen that the performance of COMDE_2 is better than the performance of COMDE_1 as well as the standard deviation significantly deceases for all benchmark problems.

Table 19
Statistical results of our approach COMDE with two CR initial values.

Problem	Features	COMDE CR = 0.2	COMDE CR = 0.5	COMDE CR = 0.8
g01	Best	**−15.000000**	**−15.000000**	**−15.000000**
	Median	**−15.000000**	**−15.000000**	**−15.000000**
	Mean	**−15.000000**	**−15.000000**	**−15.000000**
	Worst	**−15.000000**	**−15.000000**	**−15.000000**
	SD	1.589E−13	1.97E−13	**1.226E−14**
	TNFE	130,000	130,000	130,000
g02	Best	−0.803618	**−0.803619**	−0.803618
	Median	**−0.803616**	**−0.803616**	−0.792605
	Mean	**−0.802881**	−0.801238	−0.788027
	Worst	**−0.792604**	−0.785265	−0.729086
	SD	**2.7E−03**	5.0E−03	1.69E−02
	TNFE	200,000	200,000	200,000
g03	Best	**−1.000000049**	**−1.000000049**	**−1.000000048**
	Median	**−1.000000005**	**−1.000000039**	**−1.000000023**
	Mean	−0.999999896	**−1.000000027**	−0.999999938
	Worst	−0.999999272	**−0.99999994**	−0.999999506
	SD	2.35E−07	**3.026E−08**	1.64E−07
	TNFE	150,000	150,000	150,000
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**
	Median	**−30665.539**	**−30665.539**	**−30665.539**
	Mean	**−30665.539**	**−30665.539**	**−30665.539**
	Worst	**−30665.539**	**−30665.539**	**−30665.539**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g05	Best	**5126.4981094**	**5126.4981094**	**5126.4981094**
	Median	**5126.4981094**	**5126.4981094**	**5126.4981094**
	Mean	5194.175620	**5126.4981094**	**5126.4981094**
	Worst	5388.9861357	**5126.4981094**	**5126.4981094**
	SD	1.015613E+02	**0.00E+00**	**0.00E+00**
	TNFE	200,000	200,000	200,000
g06	Best	**−6961.813875**	**−6961.813875**	**−6961.813875**
	Median	**−6961.813875**	**−6961.813875**	**−6961.813875**
	Mean	**−6961.813875**	**−6961.813875**	**−6961.813875**
	Worst	**−6961.813875**	**−6961.813875**	**−6961.813875**
	SD	2.13E−12	**0.00E+00**	**0.00E+00**
	TNFE	12,000	12,000	12,000
g07	Best	**24.306209**	**24.306209**	**24.306209**
	Median	**24.306209**	**24.306209**	**24.306209**
	Mean	**24.306209**	**24.306209**	**24.306209**
	Worst	**24.306210**	24.306211	**24.306210**
	SD	**2.91E−07**	4.7E−07	3.0E−07
	TNFE	200,000	200,000	200,000
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	**−0.095825**
	Mean	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	**−0.095825**	**−0.095825**
	SD	**2.07E−17**	**9.00E−18**	1.31E−18
	TNFE	4000	4000	4000
g09	Best	**680.630057**	**680.630057**	**680.630057**
	Median	**680.630057**	**680.630057**	**680.630057**
	Mean	**680.630057**	**680.630057**	**680.630057**
	Worst	**680.630057**	**680.630057**	**680.630057**
	SD	9.3E−14	4.071E−13	0.00E+00
	TNFE	70,000	70,000	70,000
g10	Best	**7049.248020**	**7049.248020**	**7049.248020**
	Median	**7049.248020**	**7049.248020**	**7049.248020**
	Mean	7049.248194	7049.248077	**7049.248022**
	Worst	7049.250693	7049.248615	**7049.248047**
	SD	5.29E−04	1.5E−04	**9.8E−06**
	TNFE	200,000	200,000	200,000
g11	Best	**0.749999999**	**0.749999999**	**0.749999999**
	Median	**0.749999999**	**0.749999999**	**0.749999999**
	Mean	**0.749999999**	**0.749999999**	**0.749999999**
	Worst	**0.749999999**	**0.749999999**	**0.749999999**

(*continued on next page*)

Table 19 (*continued*)

Problem	Features	COMDE CR = 0.2	COMDE CR = 0.5	COMDE CR = 0.8
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g12	Best	**−1.000000**	**−1.000000**	**−1.000000**
	Median	**−1.000000**	**−1.000000**	**−1.000000**
	Mean	**−1.000000**	**−1.000000**	**−1.000000**
	Worst	**−1.000000**	**−1.000000**	**−1.000000**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	6,000	6,000	6,000
g13	Best	**0.0539415**	**0.0539415**	**0.0539415**
	Median	**0.0539415**	**0.0539415**	**0.0539415**
	Mean	0.092427623	**0.0539415**	0.06677021
	Worst	0.43880260	**0.0539415**	0.43880260
	SD	1.174E−01	**1.4E−17**	7.026E−02
	TNFE	150,000	150,000	150,000

A result in boldface indicates the best result or the global optimum. NA means not available.

Nonetheless, its main deficiency is the lack of good exploration mutation that needs to escape from local solutions and avoid stagnation. With respect to COMDE_3 that is similar to COMDE without new directed mutation rule it almost provides similar "best", "median", "mean", "worst" results for all benchmark problems except g02, but COMDE is more robust than COMDE_3 with lower standard deviation for all test functions except g01. Typically, it can be noticed that COMDE and COMDE_3 show a very competitive performance in terms of high final quality solution and efficiency with a slight superiority to COMDE in robustness. However, convergence behavior is another important factor that must be considered in comparison among all proposed algorithms. Therefore, in order to analyze the convergence behavior of each algorithm compared, the convergence graph of the median run has been plotted for each test problem. Fig. 6 presents the convergence characteristics of all algorithms. From Fig. 6, it can be observed that COMDE_3 converged to better solutions faster than all other versions in g01, g04, g06, g08, and g12. However, COMDE has converged to the better solution faster than all other versions in g02, g07, g09, g10, and g13. The convergence behavior has been similar in g03, g05, g11 which are equality constrained problems. Concretely, there is not a clear pattern or obvious relation between the convergence behavior and the characteristics of the problems. Nevertheless, it is easily driven that the local scaling factor F_l is responsible for the slow convergence of COMDE in problems g01, g04, g06, g08, and g12, as previously mentioned in subsection 4.1 about the effect of the scaling factor on the search process. Therefore, the values of F_l must be decreased by replacing the range of F_l to be generated within small range such [0.2, 0.4] instead of [0.4, 0.6] to be suited with these problems i.e. to speed up convergence. Thus, in order to prove our conclusion, the convergence graph of median runs of COMDE with the new range [0.2, 0.4] and COMDE_3 was plotted for g01, g04, g06, g08, and g12. Fig. 7 presents the convergence characteristics of COMDE with a new range and COMDE_3 for problems g01, g04, g06, g08, and g12. Therefore, as can be seen from Fig. 7, the range of the local scaling factor F_l plays a vital role on the convergence behavior of the benchmark problems. Thus, after the above analysis and discussion, although the two proposed algorithms COMDE and COMDE_3 show competitive performance in terms of quality of solution and efficiency, COMDE is superior to all versions i.e. COMDE_1, COMDE_2, and COMDE_3 in terms of convergence rate and robustness. Accordingly, the main benefits of the new directed mutation are the fast convergence speed and the extreme robustness which are the weak points of all evolutionary algorithms. Ultimately, the performance of COMDE and COMDE_3 is in agreement with the no-free-lunch theorem [48]. The analysis above indicates that different test functions may need two different ranges for F_l or the average range may be suitable for both. Based on the above discussion and analysis, how to control the local scaling factor F_l to be suitable with all benchmark problems will be further point of research. Finally, it can be concluded that the optimal control setting for F_l is to be self-adapted.

6.2. Analysis of crossover rate equation

In this subsection, some trials have been preformed to evaluate the benefits and effectiveness of the proposed crossover rate equation on the performance on the COMDE algorithm. It is to be noted that COMDE has been used and tested and the same parameters of crossover equation, dynamic tolerance equation for equality constraints and (TNFE) are used in all experiments.

6.2.1. Effect of initial value

As previously mentioned in section 4, the constant crossover (CR) practically controls the diversity of the population. Thus, the performance of COMDE has been investigated under two different CR values (0.2 and 0.8) and the results have been compared to those obtained by the value of 0.5. The TNFE, the best, the median, the mean, the worst, and the standard deviation for each CR value are listed in Table 19. As shown in Table 19, COMDE can find the global optimal consistently for all benchmark problems except for problems g02, g05, and g13 when initial value of CR is 0.2 and 0.8. Additionally, for g02, the performance of COMDE when CR is initially 0.2 is better than its performance with CR initially 0.5 or 0.8, respectively. How-

Table 20
Statistical results of our approach COMDE with two power values for CR equation.

Problem	Features	COMDE (power = 1)	COMDE (power = 4)	COMDE (power = 7)
g01	Best	**−15.000000**	**−15.000000**	**−15.000000**
	Median	**−15.000000**	**−15.000000**	**−15.000000**
	Mean	**−15.000000**	**−15.000000**	**−15.000000**
	Worst	**−15.000000**	**−15.000000**	**−15.000000**
	SD	8.187E−13	**1.97E−13**	1.8E−14
	TNFE	130,000	130,000	130,000
g02	Best	−0.803618	**−0.803619**	−0.803618
	Median	−0.803614	**−0.803616**	−0.79375009
	Mean	**−0.801779**	−0.801238	−0.795913962
	Worst	**−0.792604**	−0.785265	−0.771018233
	SD	**4.172E−03**	5.0E−03	8.309E−03
	TNFE	200,000	200,000	200,000
g03	Best	−0.996777547	**−1.000000049**	**−1.000000049**
	Median	−0.987860335	**−1.000000039**	−0.999999999
	Mean	−0.986874017	**−1.000000027**	−0.999999935
	Worst	−0.975235147	**−0.999999940**	−0.999999503
	SD	6.384E−03	**3.026E−08**	1.671E−07
	TNFE	150,000	150,000	150,000
g04	Best	**−30665.539**	**−30665.539**	**−30665.539**
	Median	**−30665.539**	**−30665.539**	**−30665.539**
	Mean	**−30665.539**	**−30665.539**	**−30665.539**
	Worst	**−30665.539**	**−30665.539**	**−30665.539**
	SD	1.906E−10	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g05	Best	5126.5827206	**5126.4981094**	**5126.4981094**
	Median	5171.4741126	**5126.4981094**	**5126.4981094**
	Mean	5225.2787115	**5126.4981094**	**5126.4981094**
	Worst	5483.6209106	**5126.4981094**	**5126.4981094**
	SD	1.16850E+02	**0.00E+00**	**0.00E+00**
	TNFE	200,000	200,000	200,000
g06	Best	−6961.813875	**−6961.813875**	**−6961.813875**
	Median	−6961.813875	**−6961.813875**	**−6961.813875**
	Mean	−6961.813875	**−6961.813875**	**−6961.813875**
	Worst	−6961.813875	**−6961.813875**	**−6961.813875**
	SD	1.09E−10	**0.00E+00**	**0.00E+00**
	TNFE	12,000	12,000	12,000
g07	Best	24.306221	**24.306209**	**24.306209**
	Median	24.306234	**24.306209**	**24.306209**
	Mean	24.306254	**24.306209**	**24.306209**
	Worst	24.306396	24.306211	**24.306209**
	SD	4.643E−05	4.7E−07	**1.969E−07**
	TNFE	200,000	200,000	200,000
g08	Best	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	**−0.095825**
	Mean	**−0.095825**	**−0.095825**	**−0.095825**
	Worst	**−0.095825**	**−0.095825**	**−0.095825**
	SD	**4.63E−18**	9.00E−18	1.034E−17
	TNFE	4000	4000	4000
g09	Best	**680.630057**	**680.630057**	**680.630057**
	Median	**680.630057**	**680.630057**	**680.630057**
	Mean	**680.630057**	**680.630057**	**680.630057**
	Worst	680.630058	**680.630057**	**680.630057**
	SD	2.703E−07	**4.071E−13**	2,692E−10
	TNFE	70,000	70,000	70,000
g10	Best	7049.2487219	**7049.248020**	**7049.248020**
	Median	7049.2503701	**7049.248020**	**7049.248020**
	Mean	7049.2565049	**7049.248077**	7049.248115
	Worst	7049.2841218	**7049.248615**	7049.249432
	SD	1.154E−02	**1.5E−04**	3.642E−04
	TNFE	200,000	200,000	200,000
g11	Best	**0.749999999**	**0.749999999**	**0.749999999**
	Median	**0.749999999**	**0.749999999**	**0.749999999**
	Mean	**0.749999999**	**0.749999999**	**0.749999999**
	Worst	**0.749999999**	**0.749999999**	**0.749999999**

(*continued on next page*)

Table 20 (*continued*)

Problem	Features	COMDE (power = 1)	COMDE (power = 4)	COMDE (power = 7)
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g12	Best	**−1.000000**	**−1.000000**	**−1.000000**
	Median	**−1.000000**	**−1.000000**	**−1.000000**
	Mean	**−1.000000**	**−1.000000**	**−1.000000**
	Worst	**−1.000000**	**−1.000000**	**−1.000000**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	6000	6000	6000
g13	Best	**0.0539415**	**0.0539415**	**0.0539415**
	Median	**0.0539415**	**0.0539415**	**0.0539415**
	Mean	0.2032798	**0.0539415**	0.0795989
	Worst	0.8921662	**0.0539415**	0.4388026
	SD	2.457E−01	**1.4E−17**	9.764E−02
	TNFE	150,000	150,000	150,000

A result in boldface indicates the best result or the global optimum. NA means not available.

ever, the performance of COMDE was deteriorated when CR initially decreased from 0.5 to 0.2 with g05 and g13. Moreover, it also prematurely converged with problem g13 when an increase to 0.8 was introduced, but it was so good with g05. In fact, this is the proper trade-off; since the three problems are completely different in their features, i.e. g02 is different from g05 and g13. On the other hand, there has been a slight change in the robustness of g10, but the performance of COMDE did not change and almost remained stable on all the remaining problems. Therefore, it is clear that when the size of the feasible region is large with respect to the search space combined with high dimensionality (problem g02), the initial value of CR must be close to zero, e.g. to depend mostly on the target vector values in the recombination step. However, when the size of feasible region or the feasibility metric is very small combined with moderate dimensionality and/or non-linear equality and inequality constraints (problems g10, g05 and g13), the initial value of CR must be equal to 0.5 or higher, e.g. to depend mostly on the mutant vector in the recombination step. In general, the performance of COMDE with initial value of $CR = 0.5$ is better than 0.2 and 0.8. Indeed, may be due to the fact the diversity and the convergence rate are balanced for almost benchmark problems when initial value of $CR = 0.5$. Since at this initial level, the information provided by target and mutant is equal that can initially balance the exploration and exploitation abilities at the early stage of the search. After that, through generations, in order to advance diversity that can lead to get more feasible solutions; the information provided by the target vector is considerably decreased and only utilized with probability of 0.05. Contrarily, the information provided by the mutant vector is exponentially increased to 0.95 which means that the trial vector is almost equal to the mutant vector. Finally, based on the above discussion, it is obvious that the initial value of $CR = 0.5$ is a good initial choice for COMDE performance.

6.2.2. Effect of power of dynamic setting

The performance of COMDE has been investigated under two different values for the power k of the dynamic setting $(1 - G/GEN)^k$ in the CR equations ((1) and (7)) and the results have been compared with those obtained by the value of 4. The TNFE, the best, the median, the mean, the worst, and the standard deviation for each power value are listed in Table 20. As depicted in Table 20, COMDE can find the global optimal consistently for all benchmark problems except for problems g02, g03, g05, g07, g10 and g13 when power value is 1 which means that the performance of COMDE was deteriorated when the value of CR linearly increased. Simply, it is clear that the search process slows down as CR linearly increased (g03) or it may be stagnated (g05, g07, g10, g13). However, as expected for g02, the performance of COMDE when the power is 1 is better than its performance with power 4 or 7, respectively. On the other hand, COMDE can find the global optimal consistently for all benchmark problems except for problems g02, and g13 only when power value is 7, which indicates that the performance of COMSE was improved as the CR values rapidly increased non-linearly. However, it may lead to premature convergence (g13). Nonetheless, the best performance for COMDE was found with power value 4, where COMDE can find the global optimal consistently for all benchmark problems with high quality solution and more robustness except for problems g02. In fact, it is clear that the loss of diversity that may cause stagnation tends to occur with linear increase of CR value, while the premature convergence tends to occur with high and rapid non-linear increase of CR value. Therefore, it can be concluded that the reasonable choice of the power value that balances the crossover rate over all benchmark problems is 4.

6.3. Analysis on equality constraints equation

In this subsection, some trials have been preformed to evaluate the benefits and effectiveness of the proposed dynamic tolerance equation for equality constraints on the performance of the COMDE algorithm. It is obvious that COMDE has been used and tested and the same parameters of crossover equation, dynamic tolerance equation for equality constraints and (TNFE) are used in all experiments.

Table 21
Statistical results of our approach COMDE with three values of initial tolerance.

Problem	Features	Initial = 0.5	Initial = 1	Initial = 2
g03	Best	**−1.000000049**	**−1.000000049**	**−1.000000049**
	Median	**−1.000000048**	−1.000000039	−1.000000038
	Mean	**−1.000000034**	−1.000000027	−1.000000020
	Worst	**−0.999999971**	−0.999999940	−0.999999898
	SD	**2.433E−08**	3.026E−08	4.194E−08
	TNFE	150,000	150,000	150,000
g05	Best	**5126.4981094**	**5126.4981094**	**5126.4981094**
	Median	**5126.4981094**	**5126.4981094**	**5126.4981094**
	Mean	5128.851906529	**5126.4981094**	**5126.4981094**
	Worst	5154.166988157	**5126.4981094**	**5126.4981094**
	SD	6.614E+00	0.00E+00	0.00E+00
	TNFE	200,000	200,000	200,000
g11	Best	**0.749999999**	**0.749999999**	**0.749999999**
	Median	**0.749999999**	**0.749999999**	**0.749999999**
	Mean	**0.749999999**	**0.749999999**	**0.749999999**
	Worst	**0.749999999**	**0.749999999**	**0.749999999**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g13	Best	**0.0539415**	**0.0539415**	**0.0539415**
	Median	0.4388026	0.4388026	**0.0539415**
	Mean	0.2976868	0.2592007	**0.0539415**
	Worst	0.4388026	0.4388026	**0.0539415**
	SD	1.886E−01	1.952E−01	**1.4E−17**
	TNFE	150,000	150,000	150,000

A result in boldface indicates the best result or the global optimum. NA means not available.

6.3.1. Effect of initial tolerance value

As previously mentioned in Section 4, the initial tolerance has a significant impact in creating the suitable initial feasible region that can attract too many infeasible points that can be improved through generations to be feasible. Consequently, the performance of COMDE has been investigated under three different values of initial tolerance $\varepsilon(t)$ (0.5, 1 and 2) for all equality constrained benchmark problems (g03, g05, g11, g13) and the results have been compared with those obtained by the value of 1 for (g03, g05, g11) and with those obtained by value of 2 for (g13). The TNFE, the best, the median, the mean, the worst, and the standard deviation for each CR value are listed in Table 21. Based on the results in Table 21, COMDE has been able to find the optimal solutions in 30 runs consistently to problems g03 and g11 with all tolerance values.

Table 22
Statistical results of our approach COMDE with two power values for dynamic tolerance equation.

Problem	Features	Power = 0.5	Power = 1	Power = 2
g03	Best	−0.999997539	**−1.000000049**	**−1.000000049**
	Median	−0.999992886	**−1.000000039**	−0.998551150
	Mean	−0.999991649	**−1.000000027**	−0.996280651
	Worst	−0.999980526	**−0.999999940**	−0.970110462
	SD	4.852E−06	**3.026E−08**	7.525E−03
	TNFE	150,000	150,000	150,000
g05	Best	**5126.4981094**	**5126.4981094**	**5126.4981094**
	Median	**5126.4981094**	**5126.4981094**	5168.07567903
	Mean	**5126.4981094**	**5126.4981094**	5199.0215893616
	Worst	**5126.4981094**	**5126.4981094**	5516.6729385989
	SD	7.83E−09	**0.00E+00**	1.043343E+02
	TNFE	200,000	200,000	200,000
g11	Best	**0.75000000**	**0.749999999**	**0.749999999**
	Median	**0.75000000**	**0.749999999**	**0.749999999**
	Mean	**0.75000000**	**0.749999999**	**0.749999999**
	Worst	**0.75000000**	**0.749999999**	**0.749999999**
	SD	1.259E−09	**0.00E+00**	**0.00E+00**
	TNFE	50,000	50,000	50,000
g13	Best	**0.0539415**	**0.0539415**	0.0539415
	Median	**0.0539415**	**0.0539415**	0.43880261
	Mean	**0.0539415**	**0.0539415**	0.31051558
	Worst	**0.0539415**	**0.0539415**	0.43880261
	SD	8.969E−13	**1.4E−17**	1.877E−01
	TNFE	150,000	150,000	150,000

A result in boldface indicates the best result or the global optimum. NA means not available.

However, with small tolerance value of 0.5, COMDE prematurely converged with problems g05 and g13. In addition, it is only able to reach the optimal solution in "best" and "median" cases as well as "best" with g05 and g13, respectively. Moreover, with tolerance value 1, it could not reach the optimal solution in all 30 runs consistently to problem g13, but it could with problem g05. Nevertheless, by increasing the initial tolerance to value 2, COMDE was able to find the optimal solutions in 30 runs consistently to all the constrained problems. As a result, it can be deduced that the initial tolerance value is mainly responsible for the poor or good performance of COMDE with equality constrained problems. Actually, problems g05 and g13 are very difficult to solve and they have the following common features: moderate dimensionality (four or five decision variables), non-linear objective functions, more than one equality constraint (three in both), and feasibility metric of zero. On the other hand, g03 and g11 have only one common feature which is one equality constraint, but g03 is more difficult than g11. Based on the analysis above, it is very difficult to generate feasible solutions during the initial search process with these types of problems. However, by using large tolerance value, i.e. by widening the true feasible area, more feasible solutions can be generated to satisfy the equality constraints in the initial generation. Then, through generation, while the initial feasible region is contracted, the algorithm improves the previous feasible solutions to satisfy the re-defined feasible space. Therefore, Table 21, it is clear that the initial tolerance value of $\varepsilon(t) = 1$ is a reasonable choice for g03, g05 and g11, but $\varepsilon(t) = 2$ is a good choice for g13.

6.3.2. Effect of power of dynamic setting

The performance of COMDE has also been investigated under two different values for the power k of the dynamic setting $(1 - G/GEN)^k$ in the dynamic tolerance equation (0.5 and 2) and the results have been compared with those obtained by the value of 1. The TNFE, the best, the median, the mean, the worst, and the standard deviation for each power value are listed in Table 22. Based on the results in Table 22, COMDE was able to find the optimal solutions in 30 runs consistently for problem g11 with all power values. Moreover, with small power value of 0.5, it could also reach the optimal solution in all 30 runs consistently for all problems but with lower standard deviation than those obtained with power 1. However, with power value of 2, it has only been able to reach the optimal solution in "best" case with problems g03, g05 and g13, and it prematurely converged in all other three cases (median, mean, and worst). As a result, as the initial tolerance value, the results in Table 22 support our previous discussion concerning the considerable effect of the power of the dynamic setting in the dynamic tolerance equation on the performance of COMDE with equality constraints problems. Therefore, it can be concluded that as the power k in Eq. (14) increases, the widening feasible region shrinks rapidly to the true feasible that may lead to converge to local optimal or it may also significantly deteriorate the search process as can be seen with all problems except g11. On the other hand, if the value of the power k is within (0, 1), the widening feasible region contracted very slowly that lead to obtain less robust results. In conclusion, $k = 1$ is a reasonable choice for all problems with equality constraints g03, g05, g11 and g13.

7. Conclusion

This study represents a significant step and a considerable trend towards the progress of existing intelligent approaches to global constrained optimization. In this paper, a new Constrained Optimization based on Modified Differential Evolution algorithm (COMDE) is proposed for solving constrained numerical and engineering optimization problems. In order to enhance the local search ability and advance the convergence rate, a new directed mutation rule is presented and it is combined with the basic mutation strategy. It is noteworthy to mention that the modified basic mutation strategy or the new directed mutation strategy is applied to an individual in the population with a probability of 0.5. Furthermore, two new global and local scaling factors are introduced as two new uniform random variables instead of keeping them constant through generations so as to globally cover the whole search space as well as biasing the search direction to follow the best vector direction. Additionally, a dynamic non-linear increased crossover probability function is formulated to balance the global exploration and the local exploitation. Furthermore, a modified constraint handling technique based on feasibility and sum of constraints violation is utilized. Moreover, a new dynamic tolerance technique for handling equality constraints is proposed. The proposed COMDE algorithm has been compared with 28 recent Evolutionary Algorithms based on Genetic Algorithm, Evolutionary Strategy, Particle Swarm Optimization, Culture Evolution, and Differential Evolution as well as traditional direct methods that are designed for solving constrained optimization problems on a set of difficult state-of-the-art constrained numerical and engineering optimization benchmark problems. The experimental results and comparisons have shown that the COMDE algorithm performs better in constrained optimization problems with different types, complexity and dimensionality; it performs better with regard to the search process efficiency, the final solution quality, the convergence rate, and robustness, when compared with other algorithms. Finally, the performance of the COMDE algorithm is superior to and competitive with other recent and well-known constrained optimization evolutionary algorithms. The effectiveness and benefits of the new directed mutation strategy and modified basic strategy used in COMDE have been experimentally investigated and compared. It is found that the COMDE with modified basic mutation strategy combined with new mutation scheme is superior to and competitive with COMDE algorithm without new directed mutation scheme in terms of convergence rate as well as robustness. Meanwhile, the effect of the parameters of the crossover probability function and the parameters of the dynamic tolerance equation on the performance of COMDE have been investigated and evaluated by different experiments. Current research effort focuses on how to control the local scaling factor by self-adaptive

mechanism. Additionally, future research will investigate the performance of the COMDE algorithm in solving unconstrained and constrained multi-objective optimization problems as well as real world applications.

References

[1] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Transactions on Evolutionary Computation 10 (6) (2006) 646–657.
[2] P. Chootinan, A. Chen, Constraint handling in genetic algorithms using a gradient-based repair method, Computers & Operations Research 33 (2009) 2263–2281.
[3] C.A.C. Coello, Theoretical and numerical constraint handling techniques used with evolutionary algorithms: a survey of the state of the art, Computer Methods in Applied Mechanics and Engineering 191 (2002) 1245–1287.
[4] C.A.C. Coello, R. Landa Becerra, Efficient evolutionary optimization through the use of a cultural algorithm, Engineering Optimization 36 (2) (2004) 219–236.
[5] S. Das, A. Abraham, U. Chakraborty, A. Konar, Differential evolution neighborhood-based mutation operator, IEEE Transactions on Evolutionary Computation 13 (3) (2009) 526–553.
[6] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2000) 311–338.
[7] E.Z. Elfeky, R.A. Sarker, D.L. Essam, A simple ranking and selection for constrained evolutionary optimization, in: Lecture Notes in Computer Science: Simulated Evolution and Learning, 2006, pp. 537–544.
[8] H.Y. Fan, J. Lampinen, A trigonometric mutation approach to differential evolution, Journal of Global optimization 27 (1) (2003) 105–129.
[9] V. Feoktistov, Differential Evolution in Search of Solutions, Springer, New York, 2006.
[10] A. Ghosh, S. Das, A. Chowdhury, R. Giri, An improved differential evolution algorithm with fitness-based adaptation of the control parameter, Information Sciences 181 (18) (2011) 3749–3765.
[11] Q. He, L. Wang, An effective co-evolutionary particle swarm optimization for constrained engineering design problems, Engineering Applications of Artificial Intelligence 20 (1) (2007) 89–99.
[12] Q. He, L. Wang, A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization, Applied Mathematics & Computation 186 (2) (2007) 1407–1422.
[13] F.Z. Huang, L. Wang, Q. He, An effective co-evolutionary differential evolution for constrained optimization, Applied Mathematics & Computation 186 (1) (2007) 340–356.
[14] D. Jia, G. Zhang, M.K. Khan, An effective memetic differential evolution algorithm based on chaotic local search, Information Sciences 181 (15) (2011) 3175–3187.
[15] S. Koziel, Z. Michalewicz, Evolutionary algorithms, homomorphous mapping, and constrained parameter optimization, Evolutionary Computation 7 (1) (1999) 19–44.
[16] J. Lampinen, I. Zelinka, On stagnation of the differential evolution algorithm, in: Proceedings of MENDEL 2000, 6th International Mendel Conference on Soft Computing, 2000, pp. 76–83.
[17] J. Lampinen, A constraint handling approach for the differential evolution algorithm, in: Proceedings of 2002 IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, May 2002, pp. 1468–1473.
[18] R. Landa Becerra, C.A.C. Coello, Cultured differential evolution for constrained optimization, Computer Methods in applied Mechanics and Engineering 195 (2006) 4303–4322.
[19] J.J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A.C. Coello, K. Deb, Problem definitions and evaluation criteria for the CEC2006 special session on constrained real-parameter optimization, 2006. <http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-06/CEC06.htm>.
[20] A. Menchaca-Mendez, C.A.C. Coello, A new proposal to hybridize the Nelder–Mead method to a differential evolution algorithm for constrained optimization, in: Proceeding 2009 IEEE Congress on Evolutionary Computation, 2009, pp. 223–230.
[21] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing 10 (2010) 629–640.
[22] E. Mezura-Montes, C.A.C. Coello, Constraint-handling in genetic algorithms through the use of dominance-based tournament selection, Advancement Engineering Informatics 16 (3) (2002) 193–203.
[23] E. Mezura-Montes, C.A.C. Coello, An improved diversity mechanism for solving constrained optimization problems using a multimembered evolution strategy, in: GECCO, 2004, pp. 700–712.
[24] E. Mezura-Montes, C.A.C. Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Transactions on Evolutionary Computation 9 (1) (2005) 1–17.
[25] E. Mezura-Montes, C.A.C. Coello, E.I. Tun-Morales, Simple feasibility rules and differential evolution for constrained optimization, in: IMICAI'2004, LNAI 2972, 2004, pp. 707–716.
[26] E. Mezura-Montes, M.E. Miranda-Varela, R. del Carmen Gómez-Ramón, Differential evolution in constrained numerical optimization: an empirical study, Information Sciences 180 (22) (2010) 4223–4262.
[27] E. Mezura-Montes, J. Velázquez-Reyes, C.A.C. Coello, Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization, in: GECCO, 2005, pp. 225–232.
[28] E. Mezura-Montes, C.A.C. Coello, J.V. Reyes, Increasing successful offspring and diversity in differential evolution for engineering design, in: Proceedings of the Seventh International Conference on Adaptive Computing in Design and Manufacture (ACDM 2006), April 2006, pp. 131–139.
[29] E. Mezura-Montes, A.G. Palomeque-Ortiz, Self adaptive and deterministic parameter control in differential evolution for constrained optimization, in: E. Mezura-Montes (Ed.), SCI 2009, Springer, Heidelberg, 2009, pp. 95–120 (198).
[30] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, third ed., Springer, Berlin, 1996.
[31] A.W. Mohamed, H.Z. Sabry, M. Khorshid, An alternative differential evolution algorithm for global Optimization, Journal of Advanced Research (2011), doi:10.1016/j.jare.2011.06.004.
[32] A. Muñoz, A. Hernández, E. Villa, Continuous constrained optimization with dynamic tolerance using COPSO algorithm, in: E. Mezura-Montes (Ed.), SCI 2009, Springer, Heidelberg, 2009, pp. 1–23 (198).
[33] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, Berlin, 2005.
[34] T. Ray, K.M. Liew, Society and civilization: an optimization algorithm based on the simulation of social behavior, IEEE Transactions on Evolutionary Computation 7 (4) (2003) 386–396.
[35] T.P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary of Computation 4 (3) (2000) 284–294.
[36] T.P. Runarsson, X. Yao, Search biases in constrained evolutionary optimization, IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews 35 (2) (2005) 233–243.
[37] A.E. Smith, D.W. Coit, Constraint handling techniques – penalty functions, in: T. Bäck, D.B. Fogel, Z. Michalewicz (Eds.), Handbook of Evolutionary Computation, Oxford University Press and Institute of Physics Publishing, Oxford, 1997 (Chapter C 5.2).
[38] R. Storn, K. Price, Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, Technical Report TR-95-012, ICSI, 1995.
[39] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.

[40] T. Takahama, S. Sakai, Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations, IEEE Transactions on Evolutionary Computation 9 (5) (2005) 437–451.
[41] B. Tessema, G. Yen, A self adaptive penalty function based algorithm for constrained optimization, in: Proceedings 2006 IEEE Congress on Evolutionary Computation, 2006, pp. 246–253.
[42] A.S.S.M. Barkat Ullah, R. Sarker, D. Cornforth, C. Lokan, AMA: a new approach for solving constrained real-valued optimization problems, Soft Computing 13 (2009) 741–762.
[43] S. Venkatraman, G.G. Yen, A generic framework for constrained optimization using genetic algorithms, IEEE Transactions on Evolutionary Computation 9 (4) (2005) 424–435.
[44] L. Wang, L. Li, An effective differential evolution with level comparison for constrained engineering design, Structural Multidisciplinary Optimization 41 (2010) 947–963.
[45] Y. Wang, Z. Cai, A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems, Frontiers of Computer Science in China 3 (1) (2009) 38–52.
[46] Y. Wang, Z. Cai, G. Guo, Y. Zhou, Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems, IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics 37 (3) (2007) 560–575.
[47] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Structural Multidisciplinary Optimization 37 (2009) 395–413.
[48] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[49] M. Zhang, W. Luo, XF. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, Information Sciences 178 (15) (2008) 3043–3074.