Hindawi
Complexity
Volume 2021, Article ID 5515701, 24 pages
https://doi.org/10.1155/2021/5515701

WILEY | Hindawi

Research Article

A Customized Differential Evolutionary Algorithm for Bounded Constrained Optimization Problems

Wali Khan Mashwani,[1] **Zia Ur Rehman**,[1] **Maharani A. Bakar**,[2] **Ismail Koçak**,[3] **and Muhammad Fayaz**[4]

[1]*Instuite of Numerical Sciences, Kohat University of Science& Technology, Kohat, Pakistan*
[2]*Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu, Kuala Nerus Terengganu, Malaysia*
[3]*Faculty of Economics and Administrative Sciences, Kirikkale University, Ankara, Turkey*
[4]*Department of Computer Science, University of Central Asia, Naryn, Kyrgyzstan*

Correspondence should be addressed to Maharani A. Bakar; maharani@umt.edu.my

Received 16 January 2021; Revised 16 February 2021; Accepted 24 February 2021; Published 10 March 2021

Academic Editor: Atif Khan

Bound-constrained optimization has wide applications in science and engineering. In the last two decades, various evolutionary algorithms (EAs) were developed under the umbrella of evolutionary computation for solving various bound-constrained benchmark functions and various real-world problems. In general, the developed evolutionary algorithms (EAs) belong to nature-inspired algorithms (NIAs) and swarm intelligence (SI) paradigms. Differential evolutionary algorithm is one of the most popular and well-known EAs and has secured top ranks in most of the EA competitions in the special session of the IEEE Congress on Evolutionary Computation. In this paper, a customized differential evolutionary algorithm is suggested and applied on twenty-nine large-scale bound-constrained benchmark functions. The suggested C-DE algorithm has obtained promising numerical results in its 51 independent runs of simulations. Most of the 2013 IEEE-CEC benchmark functions are tackled efficiently in terms of proximity and diversity.

1. Introduction

Bound optimization is the mathematical process of optimizing an objective function in the presence of constraints imposed on the decision space. These variables in the decision space may be continuous, discrete, or mixed. The basic elements of optimization are decision variables, objective function, and constrained functions. The standard form for a single-objective, nonlinear, constrained optimization problem is described as follows [1]:

$$\begin{aligned} &\text{minimize } f(\mathbf{x}), \quad \mathbf{x} = (x_1, x_2, \ldots, x_n) \in \Omega \subseteq R^n \\ &\text{subject to} \\ &\quad c_i(\mathbf{x}) \le 0, \quad i = 1, \ldots, l, \\ &\quad c_j(\mathbf{x}) = 0, \quad j = l+1, \ldots, p, \\ &\quad x_k^l \le x_k \le x_k^u, \quad k = 1, \ldots, n, \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)^T \in \Omega$ is the candidate solution with n decision variables/real parameters, $f(x)$ is the objective function, $c_i(x) \le 0$ represents p inequality constrained functions, $c_j(x) = 0$ denotes q equality constrained functions, and $x_k^l \le x_k \le x_k^u$ are bound constrained over the search space $x \in \Omega$. An optimum solution $\mathbf{x}^*$ that is optimal in some certain neighbourhood is called a local optimum. Mathematically, by a local optimum, we mean a solution $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \le f(\mathbf{x})$ for all $\mathbf{x}$ in some neighbourhood of $\mathbf{x}^*$, Then, optimum solution $\mathbf{x}^*$ is called global optimum solution if $f(\mathbf{x}^*) \le f(\mathbf{x}), \forall \mathbf{x} \in \Omega \subseteq R^n$.

In the last two decades, various optimization methods were developed and are still developing for dealing with a diverse test suite of benchmark functions and real-world optimization problems [2–6]. These algorithms are inspired by Darwin's theory of evolution with Darwinian's principle "Survival of the fittest". The main intrinsic operators in the framework of EAs

to evolve population include mutation and crossover operators, by using mutation and crossover operator with elitism scheme [7]. EAs are nature-inspired-based techniques [8–10]. In general, EAs can be categorized into nine different groups including the biology-based [11, 12], physics-based [13, 14], social-based [15–18], music-based [19], chemical-based [20], sport-based [21], mathematics-based [22], swarm-based [23–29], and hybrid methods [30–37]. All these algorithms operate on a uniform and random set of solutions generated within the search domain of the given optimization and search problems. Among the evolutionary algorithms, the genetic algorithm (GA) [38], evolutionary strategy (ES) [39], evolutionary programming (EP) [5, 29], and genetic programming (GP) [40] are classical paradigms in evolutionary computing field [3]. Differential evolution (DE) [41], particle swarm optimization (PSO) [24], and ant colony optimization (ACO) [23, 42, 43], cuckoo search (CS) [44], and teaching learning-based optimization (TLBO) algorithm [45, 46] are comparatively recently developed EAs and successfully applied to various test suites of optimization problems and many real-world problems [18, 47–50].

Differential evolution (DE) is one of the well-established population-based evolutionary algorithms, which was first introduced by Storn and Price for solving the Chebyshev polynomial fitting problems [51, 52]. DE is capable of handling nonlinear, linear, and multimodel objective functions and solving directional preferences over existing relays in the power system. The DE has also worked to train the weight of the neural network to deal with real-world problems. The basic idea of this technique is to avoid repetition of the existing set of solutions. Classic DE has two main control parameters F and CR that need to be settled efficiently. The settlement of these parameters including the scaling factor, F, and the probability of crossover,CR, is as discussed in [31, 41, 53–61]. Crossover is the main source of exploration, the mutation is utilized for exploitation purposes, and selection operators brought down the pressure for the survival of fittest individuals to evolve the population. The adjustment of various parameters used in search operator implementation is mainly problem-dependent, and their proper settings are quite difficult and time-consuming while performing trial and error experiments [62].

Different search operators behave differently at different levels of the optimization search process while dealing with complicated optimization and search problems. It is quite difficult to determine that this particular crossover or mutation or selection operator is useful in this particular proposition optimization process. In this paper, a customized differential evolutionary algorithm is suggested for solving benchmark functions designed for the special session of the 2013 IEEE Congress of Evolutionary Computation (CEC'13) [63]. The proposed C-DE algorithm employs more than one mutation operator flowing by crossover to generate a new set of solutions for the next generation of the evolutionary search process. Experimental results demonstrated that the proposed C-DE algorithm is more promising as compared to the algorithm that employs a single mutation operator for population evolution.

The rest of the paper is organized as follows. Section 2 introduces the framework of the proposed customized differential evolutionary algorithm. Section 3 demonstrates the experimental results and characteristics of the used benchmark functions. Section 4 finally concludes this paper.

2. A Customized Differential Evolutionary Algorithm for Numerical Optimization Problems

Differential evolution (DE) is a stochastic algorithm that was first developed by Kenneth Price in 1994 [40]. DE is one of the most promising global search techniques that converge toward the known optimal solution of the problem at hand in a significant manner [60]. In general, DE has five different mutation strategies; among them, rand/1 and best/1rand/1 have performed better in our case. Among the five DE variants, DE rand/1 has faced premature convergence issues and slow convergence behaviour as compared to DE best/1rand/1. The proposed customized DE employs DE rand/1 and DE best/1rand/1 mutation strategies simultaneously to alleviate premature convergence of original differential evolution. The customized DE does the switching of aforementioned mutation strategies with linearly reducing weight parameter r from 0.25 to 0.01 during their whole course of optimization. (Figure 1).

Mutation strategies are the most distinctive components of differential evolution which are described as follows [64]:

(1) Rand/1: mathematically, it is formulated as follows:

$$\mathbf{y}_i^t = \mathbf{x}_{r_1}^t + F \times \left(\mathbf{x}_{r_2}^t - \mathbf{x}_{r_3}^t\right), \quad \text{where } r_1 \neq r_2 \neq r_3, \tag{2}$$

where $\mathbf{x}_{r_1}^t$, $\mathbf{x}_{r_2}^t$, and $\mathbf{x}_{r_3}^t$ are randomly taken distinct solutions from population other than $\mathbf{x}_i^t$ (target solution). $F \in [0.3, 0.7]$ is the scaling factor which is randomly generated. $\mathbf{y}_i^t$ is the mutant vector. In Algorithm 1, the mutation strategy as described in equation (2) is used and solved the CEC-2013 unconstrained suite of benchmark functions. The abovementioned mutation strategy is good in exploration while comparatively weak in exploitation because of which it does not converge efficiently keeping in view the average value of most of the used benchmark functions (Algorithm 2).

(2) Best/rand/1: mathematically, it is formulated as follows:

$$\mathbf{y}_i^t = \mathbf{x}_{\text{best}}^t + F \times \left(\mathbf{x}_{r_1}^t - \mathbf{x}_{r_2}^t\right), \quad \text{where } r_1 \neq r_2, \tag{3}$$

where $\mathbf{x}_{\text{best}}^t$ is the best solution in the current population, $\mathbf{x}_{r_1}^t$ and $\mathbf{x}_{r_2}^t$ are randomly chosen distinct solutions, and $\mathbf{x}_i^t$ is the i^{th} target solution. $F \in [0.3, 0.7]$ is the scaling factor which is randomly generated. $\mathbf{y}_i^t$ is the mutant vector. In Algorithm 3, the mutation strategy as described in equation (3) has shown better performance in exploitation and comparatively not good in exploration that induces a premature convergence.

(3) Current/rand/2: mathematically, it is formulated as follows:

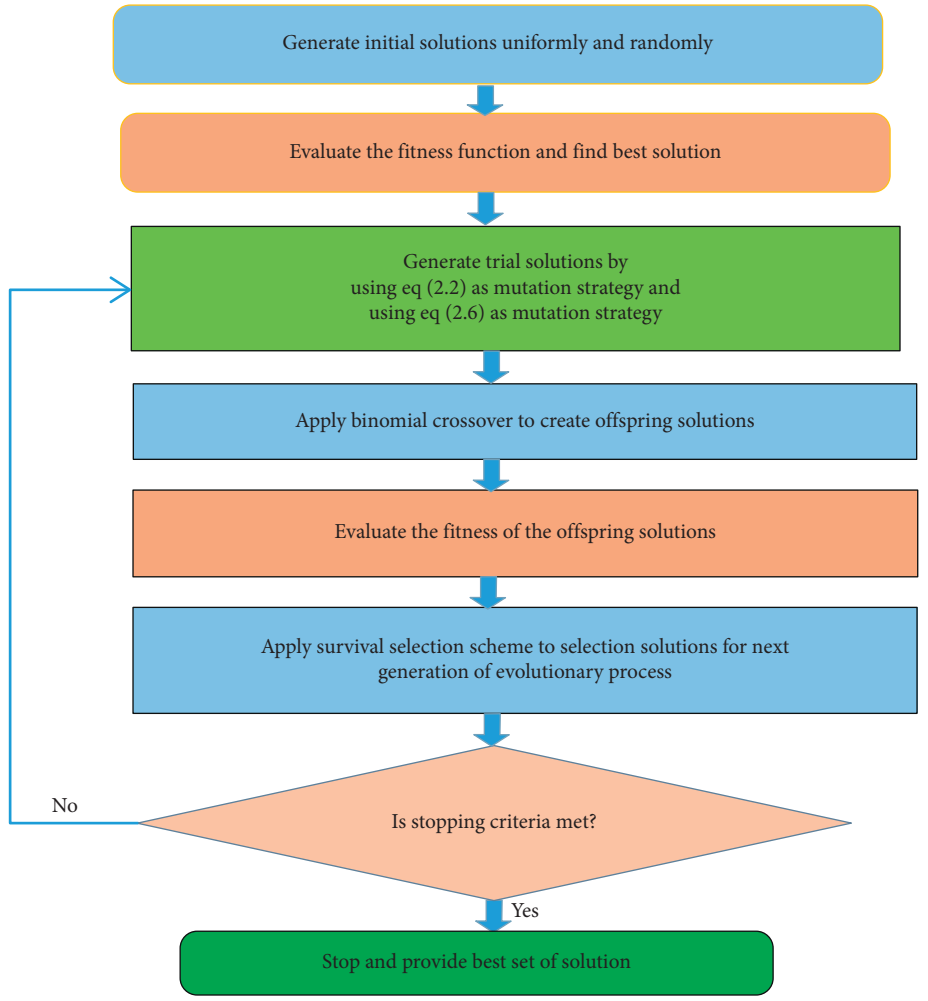

Figure 1: Flowchart of the proposed customized DE (CDE) algorithm.

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$, upper scaling bound = $beta_{\max}$, crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs **do**
(8) Generate $MutantPop$ from Pop using rand/1 mutation strategy;
(9) Use binomial crossover to generate $NewPop$;
(10) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(11) Set FEVs = FEVs + $nPop$;
(12) Update Pop by comparing Pop with $NewPop$ through their fitness values;
(13) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(14) **end while**

Algorithm 1: The pseudocode of DE rand/1 variant.

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$, upper scaling bound = $beta_{\max}$, crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs **do**
(8) Generate $MutantPop$ from Pop using rand/2 mutation strategy;
(9) Use binomial crossover to generate $NewPop$;
(10) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(11) Set FEVs = FEVs + $nPop$;
(12) Update Pop by comparing Pop with $NewPop$ through their fitness values;
(13) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(14) **end while**

Algorithm 2: The pseudocode of DE rand/2 variant.

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$, upper scaling bound = $beta_{\max}$, crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs **do**
(8) Generate $MutantPop$ from Pop using best/rand/2 strategy;
(9) Use binomial crossover to generate $NewPop$;
(10) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(11) Set FEVs = FEVs + $nPop$;
(12) Update Pop by comparing Pop with $NewPop$ through their fitness values;
(13) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(14) **end while**

Algorithm 3: The pseudocode of DE best/rand/2 variant.

$$\mathbf{y}_i^t = \mathbf{x}_i^t + F \times \left(\mathbf{x}_{r_2}^t - \mathbf{x}_{r_3}^t\right) + F \times \left(\mathbf{x}_{r_4}^t - \mathbf{x}_{r_5}^t\right), \quad \text{where } r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5, \tag{4}$$

where $\mathbf{x}_i^t$ is the target solution, $\mathbf{x}_{r_1}^t$, $\mathbf{x}_{r_2}^i t$, $\mathbf{x}_{r_3}^t$, and $\mathbf{x}_{r_4}^t$ are randomly taken distinct solutions from population other than target solution. $F \in [0.3, 0.7]$ is the scaling factor which is randomly generated. $\mathbf{y}_i^t$ is the mutant vector. The mutation strategy as described in equation (4) is employed in Algorithm 4 for creating a new solution. The above-described mutation strategy is good in exploitation while comparatively weak in exploration.

(4) Rand/2: mathematically, it is formulated as follows:

$$\mathbf{y}_i^t = \mathbf{y}_{r_1}^t + F \times \left(\mathbf{x}_{r_2}^t - \mathbf{x}_{r_3}^t\right) + F \times \left(\mathbf{x}_{r_4}^t - \mathbf{x}_{r_5}^t\right), \tag{5}$$

where $\mathbf{x}_{r_1}^t$, $\mathbf{x}_{r_2}^t$, $\mathbf{x}_{r_3}^t$, $\mathbf{x}_{r_4}^t$, and $\mathbf{x}_{r_5}^t$ are randomly taken distinct solutions from population other than target solution. $F \in [0.3, 0.7]$ is the scaling factor which is randomly generated. $\mathbf{y}_i^t$ is the mutant vector. The mutation strategy as described in equation (5) has been used in Algorithm 4 and performed good in exploration while less effective in exploitation.

(5) Best/rand/2: mathematically, it is formulated as follows:

$$\mathbf{y}_i^t = \mathbf{x}_{\text{best}}^t + F \times \left(\mathbf{x}_{r_1}^t - \mathbf{x}_{r_2}^t\right) + F \times \left(\mathbf{x}_{r_3}^t - \mathbf{x}_{r_4}^t\right), \tag{6}$$

where $\mathbf{x}_{\text{best}}^t$ is the best solution in the current population and $\mathbf{x}_{r_1}^t$, $\mathbf{x}_{r_2}^t$, $\mathbf{x}_{r_3}^t$, and $\mathbf{x}_{r_4}^t$ are taken randomly. $F \in [0.3, 0.7]$ is the scaling factor which is randomly generated within the specified bound. $\mathbf{y}_i^t$ is the mutant vector. The mutation strategy as described in equation (6) has been employed in Algorithm 4 to maintain exploitation versus exploration issue in order to avoid premature convergence during population evolution.

(6) Best/rand/1: mathematically, it is formulated as follows:

$$\mathbf{y}_i^t = \mathbf{x}_{r_1}^t + F \times \left(\mathbf{x}_{r_2}^t - \mathbf{x}_{r_3}^t\right) + F \times \left(\mathbf{x}_{\text{best}}^t - \mathbf{x}_{r_1}^t\right), \tag{7}$$

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$, upper scaling bound = $beta_{\max}$, crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs**do**
(8) Generate $MutantPop$ from Pop using current/rand/2 mutation strategy;
(9) Use binomial crossover to generate $NewPop$;
(10) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(11) Set FEVs = FEVs + $nPop$;
(12) Update Pop by comparing Pop with $NewPop$ through their fitness values;
(13) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(14) **end while**

Algorithm 4: The pseudocode of DE current/rand/2 variant.

where $\mathbf{x}_{r_1}^t$, $\mathbf{x}_{r_2}^t$, $\mathbf{x}_{r_3}^t$, $\mathbf{x}_{r_4}^t$, and $\mathbf{x}_{r_5}^t$ are chosen randomly. The scaling factor, $F \in [0.3, 0.7]$, is adjusted in all algorithms, and $\mathbf{y}_i^t$ is the mutant vector as used in Algorithm 5.

The proposed C-DE technique as outlined in Algorithm 6 employs the mutation strategies as described in equations (2) and (6) along with crossover as described in equation (8) to create its new set of solutions for the next generation of population evolution.

2.1. Crossover. In the current research, a binomial crossover operator is used. In the binomial crossover, every target vector is mixed with its corresponding mutant vector to get the trial vector as follows:

$$\mathbf{u}_i^t = \begin{cases} \mathbf{y}_{i,j}^t, & \text{if } rand \le \text{CR or } j = randi \ \in \{1{:}\ n\}; \\ \mathbf{x}_{i,j}^t, & \text{otherwise;} \end{cases} \tag{8}$$

where j is the coordinate counter, $\mathbf{y}_i^t$ is the mutant vector, CR is the probability of crossover, $randi$ is a randomly generated integer, n denotes the dimension of search space, $\mathbf{x}_i^t$ is the target vector, and $\mathbf{u}_i^t$ is the trial vector corresponding to $\mathbf{x}_i^t$.

2.2. Selection. The selection scheme for parent solution and their trial solution vector compete with each other based on their corresponding fitness values. The selection scheme is illustrated as follows:

$$\mathbf{x}_i^{t+1} = \begin{cases} \mathbf{u}_i^t, & \text{if } f\left(\mathbf{u}_i^t\right) < f\left(\mathbf{y}_i^t\right); \\ \mathbf{y}_i^t, & \text{otherwise;} \end{cases} \tag{9}$$

where $\mathbf{u}_i^t$ is the trial vector, $\mathbf{x}_i^t$ is the target vector, and $f(\mathbf{x}_i^t)$ and $f(\mathbf{u}_i^t)$ are the fitness values of the i^{th}-trial and i^{th}-target vector, respectively.

3. Simulation Results, Comparison, and Discussion

Due to the flurry of EAs recently developed, their performances are mainly analyzed by using different test suites of optimization and search problems. In the last few years, several unconstrained benchmark functions (i.e., bound-constrained) and constrained test problems are designed in special sessions of the IEEE Conference on Evolutionary Computation (IEEE-CEC) series for EA competition [63]. In a study of this paper, we have used twenty-eight different unconstrained continuous benchmark functions whose characteristics are given in Table 1. The first five test problems, $F_1 - F_5$, are unimodal functions. Most of the evolutionary algorithms are getting trap in the local basin of attraction while solving these test problems. The problems from F_6 to F_{20} are multimodal functions containing more than one local minima. These problems are much difficult to deal with as compared to unimodal functions. The functions F_{21} from F_{28} are composite functions.

3.1. PC Configuration and Simulation Environment. The simulations were conducted on a PC with 8 GB RAM and Intel(R)Core(TM)i3-7100U CPU @ 2.40 GHz processor. All simulations are done in MATLAB R2013a environment with $popsize = 100$. Each benchmark function has been solved in different dimensions including $n = 10, 30, 50$ by executing 51 times independently. Maximum function evaluation was fixed to $10000 \times n$. The algorithms for each problem are run 51 times independently to obtain the final best solution. $S_m \in [0, 1]$ denotes the switching of mutation strategies.

Tables 2 and 3 present the numerical results of C-DE as outlined in Algorithm 6 in comparison with DE-1 as described in Algorithm 1 and DE-5 hereby outlined in Algorithm 5. The numerical results for F_1 to F_{20} benchmark functions in terms of minimum, maximum, average, median, and standard deviation in function values are summarized in Table 2. Similarly, the numerical results for the benchmark functions denoted by F_{21} to F_{28} are given in Table 3.

Tables 4 and 5 demonstrate the numerical results in respect of the proposed C-DE as outlined in Algorithm 6 versus DE-1 as given in Algorithm 1 and DE-5 hereby outlined in Algorithm 5. The numerical results for F_1 to F_{20} benchmark functions in terms of minimum, maximum,

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$, upper scaling bound = $beta_{\max}$, crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs **do**
(8) Generate $MutantPop$ from Pop using best/1rand/1 strategy;
(9) Use binomial crossover to generate $NewPop$;
(10) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(11) Set FEVs = FEVs + $nPop$;
(12) Update Pop by comparing Pop with $NewPop$ through their fitness values;
(13) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(14) **end while**

Algorithm 5: The pseudocode of DE best/1rand/1 variant.

(1) Set Pop Size = $nPop$, Max Ftn Evaluations = MaxFEVs, lower scaling bound = $beta_{\min}$,upper scaling bound = $beta_{\max}$,crossover probability = pCR;
(2) Randomly generate initial Pop of size $nPop$;
(3) Set Function Evaluations Counter: FEVs = 0;
(4) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in Pop$;
(5) Set FEVs = FEVs + $nPop$;
(6) $BestSol = \mathbf{x}^*$ such that $\mathbf{x}^* = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(7) **while** FEVs < MaxFEVs **do**
(8) Take r randomly generated real number between 0 and 1;
(9) **if** $rand < S_m$ **then**
(10) Generate $MutantPop$ from Pop using rand/1 strategy;
(11) **else**
(12) Generate $MutantPop$ from Pop usingbest1/rand1 strategy;
(13) **end if**
(14) Use binomial crossover to generate $NewPop$;
(15) Evaluate $f(\mathbf{x}_i^t)$ for each $\mathbf{x}_i^t \in NewPop$;
(16) Set FEVs = FEVs + $nPop$;
(17) Update Pop by comparing Pop and $NewPop$ through their fitness values;
(18) $BestSol = \mathbf{x}^*$ such that $f(\mathbf{x}^*) = \min\{f(\mathbf{x}), \forall \mathbf{x} \in Pop\}$;
(19) **end while**

Algorithm 6: The pseudocode of the proposed customized DE algorithm.

average, median, and standard deviation in function values are summarized in Table 4. Similarly, the numerical results regarding the benchmark functions denoted by F_{21} to F_{28} are given in Table 5.

Tables 6 and 7 provide the experimental results approximated by the proposed C-DE herewith outlined in Algorithm 6 against DE-1 as given in Algorithm 1 and DE-5 hereby outlined in Algorithm 5. The numerical results for F_1 to F_{19} benchmark functions in terms of minimum, maximum, average, median, and standard deviation in function values are summarized in Table 6. Similarly, the numerical results regarding the benchmark functions denoted by F_{20} to F_{28} are given in Table 7.

Figure 2 displays the convergence graph of the F_1 to F_5 benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 2 depicts the F_1 to F_5 benchmark functions as solved in ten dimensions, the second panel is for thirty dimensions, and the third panel is for fifty dimensions.

Figure 3 displays the convergence graph of the F_6 to F_{10} benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 3 depicts the F_6 to F_{10} benchmark functions as solved in ten dimensions, the second panel is for thirty dimensions, and the third panel is for fifty dimensions.

Figure 4 displays the convergence graph of the F_{11} to F_{15} benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 4 depicts the F_{11} to F_{15} benchmark functions as solved in ten dimensions, the second panel is for thirty dimensions, and the third panel is for fifty dimensions.

Table 1: The characteristics of the CEC'13 test function [63].

	No.	Properties	$F_i^* = F_i(x^*)$
UF	1	Sphere function	−1400
	2	Rotated high conditioned elliptic function	−1300
	3	Rotated bent cigar function	−1200
	4	Rotated discus function	−1100
	5	Different power function	−1000
MF	6	Rotated Rosenbrock's function	−900
	7	Rotated Schaffers F7 function	−800
	8	Rotated Ackley's function	−700
	9	Rotated Weierstrass function	−600
	10	Rotated Griewank's function	−500
	11	Rastrigin's function	−400
	12	Rotated Rastrigin's function	−300
	13	Noncontinuous rotated Rastrigin's function	−200
	14	Schwefel's function	−100
	15	Rotated Schwefel's function	100
	16	Rotated Katsuura function	200
	17	Lunacek bi-Rastrigin function	300
	18	Rotated Lunacek bi-Rastrigin function	400
	19	Expanded Griewank's plus Rosenbrock's function	500
	20	Expanded Scaffer's F6 function	600
CF	21	Composition function 1 ($n = 5$, rotated)	700
	22	Composition function 2 ($n = 3$, unrotated)	800
	23	Composition function 3 ($n = 3$, rotated)	900
	24	Composition function 4 ($n = 3$, rotated)	1000
	25	Composition function 5 ($n = 3$, rotated)	1100
	26	Composition function 6 ($n = 5$, rotated)	1200
	27	Composition function 7 ($n = 5$, rotated)	1300
	28	Composition function 8 ($n = 5$, rotated)	1400

Table 2: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_1 to F_{20} in $n = 10$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{01}	−1400	DE-1	−1400	−1400	−1400	−1400	0.000000
		DE-5	−1400	−1400	−1400	−1400	0.000000
		C-DE	−1400	−1400	−1400	−1400	0.000000
F_{02}	−1300	DE-1	593583	4.56813e + 06	2.28216e + 06	2.11013e + 06	891922.510621
		DE-5	249403	2.24398e + 06	1.04699e + 06	987371	426831.918032
		C-DE	310525	2.42089e + 06	1.14756e + 06	996518	512003.868545
F_{03}	−1200	DE-1	75046.1	4.66235e + 06	1.55526e + 06	1.1441e + 06	1141062.835342
		DE-5	286063	781260	345872	305354	94580.836448
		C-DE	−559.296	781166	56405	18616.2	131613.067626
F_{04}	−1100	DE-1	2663.83	20828.9	11876	12016.1	3735.150538
		DE-5	2661.7	13432.2	7847.18	7823.6	2023.165850
		C-DE	2574.66	11249.8	6114.03	6033.66	1982.673559
F_{05}	−1000	DE-1	−1000	−1000	−1000	−1000	0.000000
		DE-5	−1000	−1000	−1000	−1000	0.000000
		C-DE	−1000	−1000	−1000	−1000	0.000000
F_{06}	−900	DE-1	−881.658	−880.015	−880.179	−880.158	0.213022
		DE-5	−879.096	−870.071	−870.678	−870.187	1.999259
		C-DE	−899.941	−890.165	−890.814	−890.186	2.263107
F_{07}	−800	DE-1	−784.364	−769.68	−778.597	−778.547	3.576671
		DE-5	−789.323	−781.563	−787.441	−787.721	1.694367
		C-DE	−799.435	−791.131	−797.985	−798.181	1.262155
F_{08}	−700	DE-1	−679.784	−679.479	−679.635	−679.636	0.068761
		DE-5	−679.813	−679.505	−679.631	−679.62	0.064325
		C-DE	−679.819	−679.523	−679.657	−679.635	0.075517

Table 2: Continued.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{09}	−600	DE-1	−588.254	−585.522	−586.761	−586.734	0.616831
		DE-5	−591.155	−587.466	−588.721	−588.543	0.787579
		C-DE	−596.588	−593.206	−594.598	−594.474	0.722405
F_{10}	−500	DE-1	−488.93	−487.95	−488.6	−488.633	0.199213
		DE-5	−491.612	−491.105	−491.349	−491.362	0.139360
		C-DE	−499.622	−499.119	−499.386	−499.392	0.113407
F_{11}	−400	DE-1	−400	−400	−400	−400	0.000000
		DE-5	−400	−400	−400	−400	0.000000
		C-DE	−400	−400	−400	−400	0.000000
F_{12}	−300	DE-1	−281.09	−263.199	−269.691	−269.351	3.606150
		DE-5	−284.128	−270.525	−277.174	−276.694	2.946380
		C-DE	−292.656	−280.911	−286.92	−286.936	3.047453
F_{13}	−200	DE-1	−183.951	−166.48	−174.742	−174.268	3.819038
		DE-5	−188.699	−174.674	−181.719	−182.099	3.220455
		C-DE	−193.282	−181.128	−186.944	−186.921	2.858693
F_{14}	−100	DE-1	−94	−93.9375	−93.9988	−94	0.008745
		DE-5	−95.8751	−89.0452	−95.0246	−95.6877	1.514591
		C-DE	−99.9375	−96.3352	−99.5179	−99.7502	0.905209
F_{15}	100	DE-1	966.064	1624.91	1345.42	1363.52	141.570897
		DE-5	850.871	1581.88	1269.04	1266.33	165.250631
		C-DE	778.43	1452.19	1184.57	1212.61	150.366735
F_{16}	200	DE-1	200.555	201.516	201.119	201.137	0.196826
		DE-5	200.51	201.527	201.122	201.127	0.223394
		C-DE	200.649	201.478	201.098	201.109	0.203207
F_{17}	300	DE-1	320.122	320.122	320.122	320.122	0.000001
		DE-5	311.542	318.288	318.01	318.122	0.924608
		C-DE	310.122	310.286	310.128	310.122	0.023545
F_{18}	400	DE-1	433.641	453.204	444.794	445.414	3.694887
		DE-5	429.56	449.267	437.735	437.257	3.894859
		C-DE	421.706	438.851	429.938	429.753	3.321748
F_{19}	500	DE-1	510.299	510.76	510.595	510.599	0.089374
		DE-5	504.153	504.684	504.538	504.548	0.082422
		C-DE	500.355	500.752	500.527	500.532	0.079466
F_{20}	600	DE-1	602.688	603.565	603.182	603.201	0.210304
		DE-5	602.184	603.466	602.781	602.815	0.299490
		C-DE	602.163	603.359	602.785	602.773	0.276907

Table 3: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_{21} to F_{28} in $n = 10$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{21}	700	DE-1	966.512	1200.19	1167.02	1200.19	56.998745
		DE-5	1000.01	1200.19	1196.27	1200.19	28.031778
		C-DE	900.624	1100.19	1092.95	1100.19	36.339134
F_{22}	800	DE-1	930.345	1058.03	987.216	982.962	29.935014
		DE-5	907.239	1106.44	976.237	997.545	51.389783
		C-DE	810.153	1029.34	899.141	904.74	53.322517
F_{23}	900	DE-1	1809.8	2484.84	2246.86	2289.89	162.930066
		DE-5	1684.24	2609.9	2141.83	2145.54	200.732186
		C-DE	1619.94	2523.09	2093.09	2101.43	198.583802
F_{24}	1000	DE-1	1181.5	1238.29	1213.19	1212.26	16.478379
		DE-5	1150.46	1238.26	1224.55	1231.54	19.882749
		C-DE	1141.11	1218.45	1208.85	1213.64	15.683668
F_{25}	1100	DE-1	1291.09	1334.34	1329.7	1331.29	7.180292
		DE-5	1266.16	1333.75	1320.13	1322.16	11.632105
		C-DE	1257.56	1314.85	1305.88	1308.8	10.946832

Table 3: Continued.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{26}	1200	DE-1	1342.6	1394.56	1366.7	1365.16	10.154832
		DE-5	1334.52	1425.07	1384.18	1387.6	36.985466
		C-DE	1311.73	1400.07	1349.87	1341.56	30.335554
F_{27}	1300	DE-1	1703.24	1840.04	1770.84	1767.39	34.869565
		DE-5	1620.08	1852.55	1650.55	1620.71	64.877728
		C-DE	1600.07	1824.26	1620.84	1600.49	55.037487
F_{28}	1400	DE-1	1716.3	1800	1797.84	1800	12.230642
		DE-5	1550	1750	1723.52	1750	67.131550
		C-DE	1500.61	1700	1688.44	1700	46.714278

Table 4: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_1 to F_{20} in $n = 30$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{01}	−1400	DE-1	−1400	−1400	−1400	−1400	0.000000
		DE-5	−1400	−1400	−1400	−1400	0.000000
		C-DE	−1400	−1400	−1400	−1400	0.000000
F_{02}	−1300	DE-1	1.36903e + 08	2.08058e + 08	1.77072e + 08	1.76426e + 08	14956353.317846
		DE-5	1.19104e + 08	1.54637e + 08	1.33283e + 08	1.33653e + 08	7763339.936747
		C-DE	1.47534e + 07	6.42165e + 07	3.17636e + 07	3.12834e + 07	8995583.667657
F_{03}	−1200	DE-1	1.07468e + 09	1.30442e + 09	1.13644e + 09	1.12995e + 09	42393532.103937
		DE-5	1.05797e + 08	1.76261e + 08	1.20783e + 08	1.1947e + 08	14758359.088589
		C-DE	103655	5.90805e + 07	1.37612e + 07	9.87793e + 06	13305275.710745
F_{04}	−1100	DE-1	35354	75681	57139.5	56623.9	9376.123241
		DE-5	18653.7	36440.7	25639.8	25351.4	4364.116408
		C-DE	15683.9	32184.6	24428.4	24524.1	4004.135290
F_{05}	−1000	DE-1	−1000	−1000	−1000	−1000	0.000000
		DE-5	−1000	−1000	−1000	−1000	0.000000
		C-DE	−1000	−1000	−1000	−1000	0.000000
F_{06}	−900	DE-1	−783.569	−756.81	−777.918	−781.984	8.219750
		DE-5	−863.853	−800.281	−851.899	−853.199	12.856533
		C-DE	−883.444	−816.652	−858.659	−856.692	12.862273
F_{07}	−800	DE-1	−752.53	−726.067	−740.613	−741.917	6.037184
		DE-5	−771.429	−737.11	−762.271	−763.958	7.427720
		C-DE	−791.822	−768.024	−783.293	−784.084	5.870443
F_{08}	−700	DE-1	−674.203	−673.974	−674.058	−674.05	0.050295
		DE-5	−669.238	−668.974	−669.045	−669.029	0.048735
		C-DE	−679.167	−678.961	−679.048	−679.04	0.043403
F_{09}	−600	DE-1	−561.328	−555.523	−558.644	−558.826	1.427762
		DE-5	−559.276	−551.544	−554.985	−554.761	1.682517
		C-DE	−570.355	−563.838	−566.394	−566.198	1.325391
F_{10}	−500	DE-1	−483.713	−447.89	−471.111	−472.988	7.436859
		DE-5	−485.98	−484.975	−485.704	−485.754	0.222811
		C-DE	−499.936	−498.639	−499.626	−499.719	0.269271
F_{11}	−400	DE-1	−390	−390	−390	−390	0.000000
		DE-5	−390	−386.02	−389.083	−389.005	1.011598
		C-DE	−400	−399.005	−399.746	−400	0.437924
F_{12}	−300	DE-1	−129.647	−87.7556	−105.127	−104.396	9.743013
		DE-5	−181.252	−111.655	−139.279	−138.231	12.860856
		C-DE	−189.566	−130.269	−150.997	−149.485	12.653504
F_{13}	−200	DE-1	543.649	606.651	587.113	588.029	10.751144
		DE-5	223.974	285.604	257.156	257.869	13.872433
		C-DE	−76.0579	−24.1726	−46.6554	−45.4933	10.970706
F_{14}	−100	DE-1	958.621	1163.13	1076.09	1073.97	37.599756
		DE-5	1068.22	2325.26	1756.67	1732.49	352.891847
		C-DE	503.907	1368.46	870.066	877.497	193.156721

Table 4: Continued.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{15}	100	DE-1	6740.64	7841.49	7423.28	7454.15	263.687213
		DE-5	6182.54	7797.06	7189.49	7202.74	312.036005
		C-DE	6651.8	7656.34	7232.33	7269.4	287.708609
F_{16}	200	DE-1	211.829	212.918	212.478	212.507	0.266427
		DE-5	213.686	214.943	214.427	214.456	0.287333
		C-DE	201.688	202.936	202.454	202.452	0.288254
F_{17}	300	DE-1	351.495	356.482	353.565	353.493	1.325248
		DE-5	360.308	373.106	367.184	367.747	2.924574
		C-DE	347.203	361.075	355.355	355.148	2.813337
F_{18}	400	DE-1	582.99	647.337	631.422	632.877	11.084542
		DE-5	576.88	621.743	605.017	606.259	10.266823
		C-DE	576.732	615.316	596.713	597.625	8.919994
F_{19}	500	DE-1	515.067	517.608	516.468	516.501	0.599903
		DE-5	524.311	526.192	525.428	525.509	0.411725
		C-DE	503.979	506.237	505.276	505.386	0.497524
F_{20}	600	DE-1	612.548	613.437	613.046	613.049	0.211167
		DE-5	611.933	613.14	612.602	612.616	0.268802
		C-DE	611.798	613.08	612.546	612.596	0.285701

Table 5: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_{21} to F_{28} in $n = 30$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{21}	700	DE-1	1010	1110	1083.33	1110	40.626595
		DE-5	950	1193.54	1055.47	1050	84.276686
		C-DE	900	1143.54	998.798	1000	71.457633
F_{22}	800	DE-1	2410.21	3350.83	2881.94	2855.53	227.697529
		DE-5	1997.4	3982.73	2865.52	2768.51	410.507035
		C-DE	1703.32	3285.89	2248.57	2238.3	326.629221
F_{23}	900	DE-1	7761.69	8933.49	8467.14	8491.29	276.464768
		DE-5	7610.14	8945.91	8282.95	8319.51	315.448232
		C-DE	7385.71	8893.86	8156.47	8238.79	345.393637
F_{24}	1000	DE-1	1285.61	1301.26	1295.22	1295.04	3.201388
		DE-5	1268.06	1328.19	1297.39	1298.54	14.328935
		C-DE	1212.5	1278.27	1248.74	1251.1	14.766463
F_{25}	1100	DE-1	1442.27	1457.12	1451.45	1451.62	3.322013
		DE-5	1419.13	1445.03	1432.18	1432.2	5.749719
		C-DE	1374.23	1395.2	1385.46	1386.97	5.218833
F_{26}	1200	DE-1	1414.91	1425.42	1419.49	1419.41	2.455244
		DE-5	1411.07	1413.84	1411.94	1411.83	0.548844
		C-DE	1401.05	1403.91	1402.06	1402.02	0.606003
F_{27}	1300	DE-1	2512.9	2640.53	2579.62	2578.5	25.085286
		DE-5	2226.43	2554.19	2443.58	2456.83	72.958778
		C-DE	2129.69	2467.36	2361.89	2385.88	73.692495
F_{28}	1400	DE-1	1800	1800	1800	1800	0.000000
		DE-5	1750	1750	1750	1750	0.000000
		C-DE	1700	1700	1700	1700	0.000000

Figure 5 displays the convergence graph of the F_{16} to F_{20} benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 5 depicts the F_{16} to F_{20} benchmark functions as solved in ten dimensions, the second panel is for thirty dimensions, and the third panel is for fifty dimensions.

Figure 6 displays the convergence graph of the F_{21} to F_{25} benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 6 depicts the F_{21} to F_{25} benchmark functions as solved in ten dimensions, the second panel is for thirty dimensions, and the third panel is for fifty dimensions.

Table 6: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_1 to F_{19} in $n = 50$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{01}	−1400	DE-1	−1400	−1400	−1400	−1400	0.000000
		DE-5	−1400	−1400	−1400	−1400	0.000000
		C-DE	−1400	−1400	−1400	−1400	0.000000
F_{02}	−1300	DE-1	1.82669e + 08	4.0298e + 08	3.08217e + 08	3.15245e + 08	40952346.718760
		DE-5	6.5501e + 07	1.46685e + 08	1.03416e + 08	1.01884e + 08	18482876.197602
		C-DE	5.44136e + 07	1.46107e + 08	1.04963e + 08	1.05681e + 08	20121866.384027
F_{03}	−1200	DE-1	1.10012e + 09	6.44772e + 09	3.55854e + 09	3.31787e + 09	1263221747.238202
		DE-5	37540.4	5.7581e + 08	9.78856e + 07	3.8351e + 07	132237941.475719
		C-DE	1.95311e + 06	3.23647e + 08	6.21553e + 07	3.62992e + 07	70023673.941721
F_{04}	−1100	DE-1	69182.1	119153	89629.8	90090.8	10106.552893
		DE-5	20820.9	41255.3	30832.5	30841.7	4612.877411
		C-DE	20496.2	38856.9	29036.1	28704.6	3791.979814
F_{05}	−1000	DE-1	−1000	−1000	−1000	−1000	0.000000
		DE-5	−1000	−1000	−1000	−1000	0.000000
		C-DE	−1000	−1000	−1000	−1000	0.000000
F_{06}	−900	DE-1	−836.081	−834.721	−835.527	−835.546	0.278565
		DE-5	−835.493	−780.995	−832.512	−833.476	7.423864
		C-DE	−855.053	−850.805	−852.701	−852.776	0.895707
F_{07}	−800	DE-1	−659.767	−624.433	−642.8	−642.167	8.870652
		DE-5	−738.024	−694.803	−716.596	−716.018	9.264348
		C-DE	−788.119	−756.809	−770.008	−769.713	7.322152
F_{08}	−700	DE-1	−676.032	−675.804	−675.865	−675.858	0.037308
		DE-5	−676.982	−676.812	−676.87	−676.86	0.041021
		C-DE	−678.984	−678.811	−678.874	−678.867	0.042407
F_{09}	−600	DE-1	−487.463	−477.117	−480.324	−480.172	1.930815
		DE-5	−500.784	−490.311	−493.893	−493.512	2.256670
		C-DE	−539.562	−530.812	−533.846	−532.959	2.223050
F_{10}	−500	DE-1	−482.098	−417.281	−460.08	−463.091	16.289936
		DE-5	−487.973	−486.858	−487.687	−487.729	0.227074
		C-DE	−499.786	−497.251	−498.559	−498.632	0.544408
F_{11}	−400	DE-1	−296.762	−271.293	−279.13	−278.952	4.588255
		DE-5	−350	−298.691	−342.965	−345.025	8.544754
		C-DE	−400	−331.719	−377.221	−371.827	20.613249
F_{12}	−300	DE-1	75.2158	125.882	103.524	104.096	13.276617
		DE-5	0.268876	81.9058	49.9213	53.4445	18.892895
		C-DE	−13.793	77.6588	34.6832	36.7967	17.050881
F_{13}	−200	DE-1	158.018	226.329	205.73	207.248	12.793081
		DE-5	123.648	181.726	158.425	159.411	13.288146
		C-DE	92.3484	167.1	142.689	147.413	17.611983
F_{14}	−100	DE-1	2757.28	2979.67	2825.25	2821.93	45.565094
		DE-5	2303.21	4082.56	3069.74	3039.67	409.883370
		C-DE	1375.05	3542.59	2571.58	2608.89	437.462372
F_{15}	100	DE-1	13765.6	15319	14673.9	14734.7	384.976680
		DE-5	13237	15094.1	14454.5	14549.9	399.510805
		C-DE	12992.9	14685.4	14102.7	14161.2	308.720813
F_{16}	200	DE-1	209.112	210.714	210.335	210.403	0.284461
		DE-5	212.755	213.932	213.389	213.37	0.285913
		C-DE	202.497	203.836	203.329	203.335	0.299347
F_{17}	300	DE-1	441.674	455.768	449.039	448.778	2.675904
		DE-5	498.309	530.053	516.227	516.817	6.054115
		C-DE	418.995	443.525	431.271	431.221	6.529947
F_{18}	400	DE-1	822.319	876.283	857.057	857.384	11.321625
		DE-5	788.248	868.463	840.602	843.171	13.809299
		C-DE	755.358	809.619	790.15	791.587	12.806493
F_{19}	500	DE-1	524.859	529.441	527.534	527.603	0.879431
		DE-5	522.265	525.845	524.077	524.195	0.823669
		C-DE	510.223	515.645	513.479	513.743	1.248977

Table 7: The experimental results obtained by DE-1 and DE-5 versus proposed C-DE by solving CEC'13 benchmark functions F_{20} to F_{28} in $n = 50$ dimensions.

P. no	Opt	Algorithm	Best	Worst	Mean	Median	Std
F_{20}	600	DE-1	622.329	623.044	622.787	622.79	0.169152
		DE-5	621.728	622.563	622.22	622.242	0.219888
		C-DE	621.065	622.68	622.148	622.21	0.296463
F_{21}	700	DE-1	1900	2822.19	2198	1900.11	404.866135
		DE-5	1900	2822.19	2518.61	2822.19	398.516988
		C-DE	900	1822.19	1446.28	1536.44	420.382683
F_{22}	800	DE-1	3505.93	5715.86	4396.73	4350.43	493.522833
		DE-5	3843.14	9364.91	6308.58	6003.59	1222.434466
		C-DE	2918.4	6161	4014.26	3946.52	731.765773
F_{23}	900	DE-1	15239.1	16998.2	16361.2	16433	377.677770
		DE-5	14668.2	17063.9	15928.8	16003.5	537.803418
		C-DE	13352.2	15814.8	15025.5	15024	452.814685
F_{24}	1000	DE-1	1456.81	1475.32	1467.97	1469.21	5.249819
		DE-5	1303.2	1449.87	1415.05	1423.02	26.753472
		C-DE	1219.05	1350.21	1307.84	1312.43	27.412943
F_{25}	1100	DE-1	1470.69	1495.4	1487.77	1488.68	5.477735
		DE-5	1463.79	1498.27	1482.96	1485.38	8.930093
		C-DE	1460.9	1484.33	1473.4	1473.52	5.722868
F_{26}	1200	DE-1	1434.88	1487.44	1455.53	1454.62	12.242195
		DE-5	1410.06	1664.94	1457.74	1419.11	87.860087
		C-DE	1408.06	1658.64	1431.65	1416.61	57.431295
F_{27}	1300	DE-1	3228.46	3399.89	3324.78	3324.33	38.861379
		DE-5	2950.16	3314.36	3157.77	3167.29	79.562492
		C-DE	2819.3	3270.46	3124.07	3138.61	88.484672
F_{28}	1400	DE-1	2800	2831.74	2800.63	2800	4.443258
		DE-5	1910	5016.76	2150.74	1910	833.557633
		C-DE	1800	4911.42	2101.47	1800	923.523003

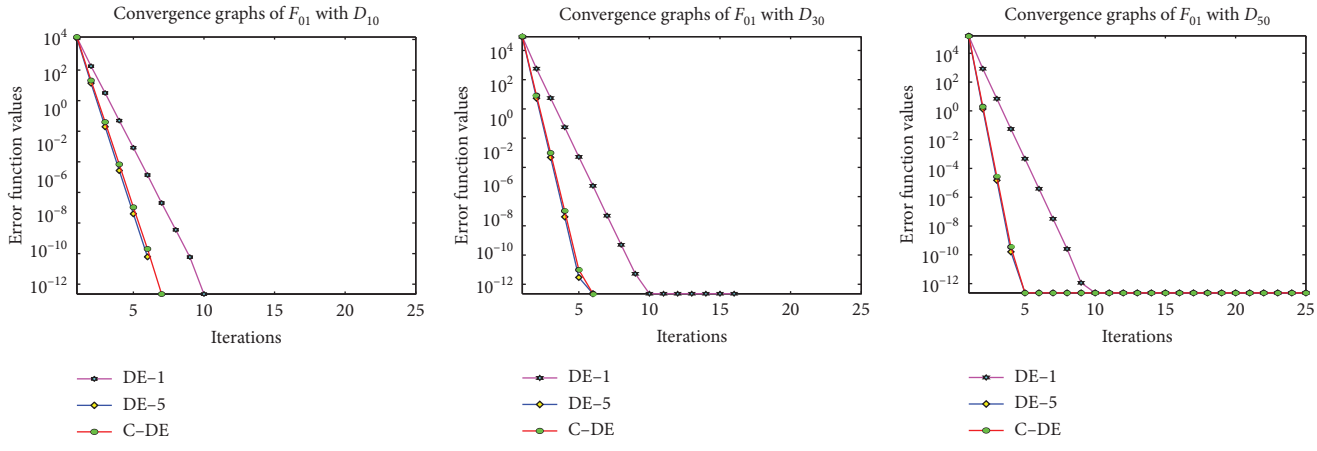

Figure 2: Continued.

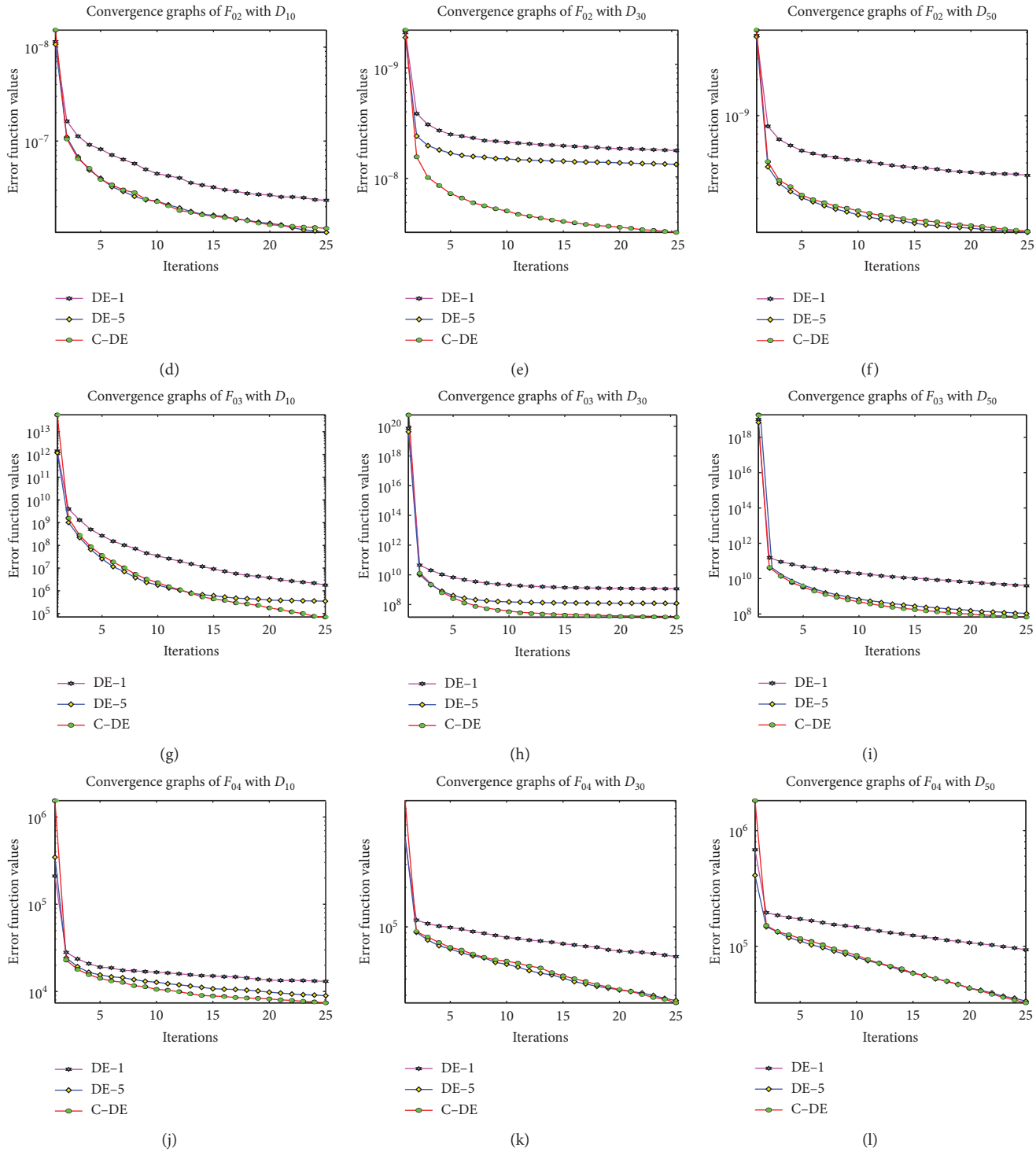

Figure 2: Continued.

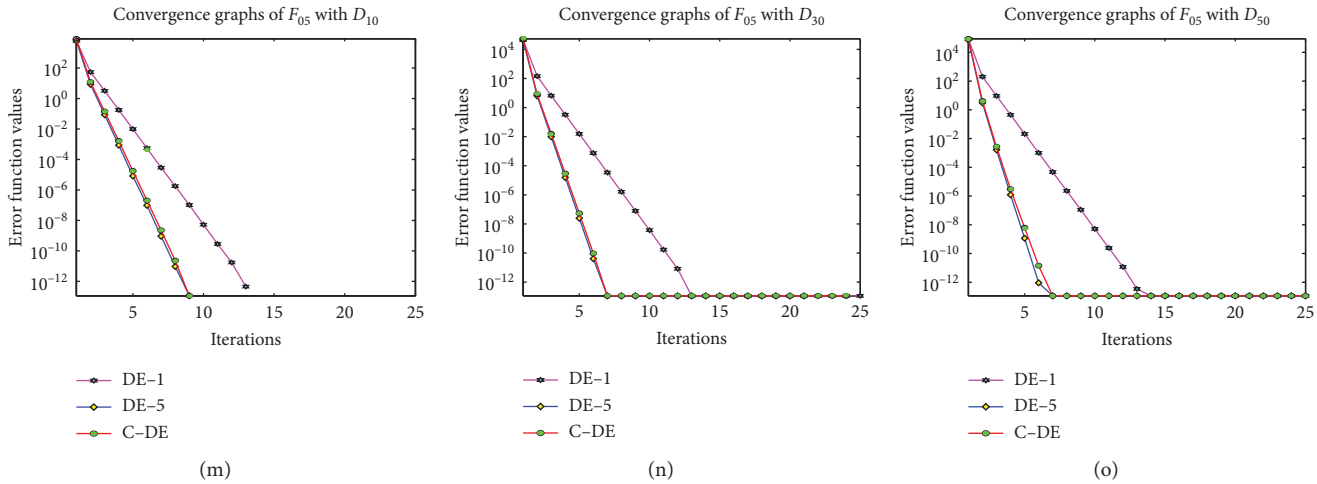

Figure 2: Evolution in the best function values of F_1 to F_5 solved with DE-1, DE-5, and C-DE in n = 10, 30, and 50 dimensions.

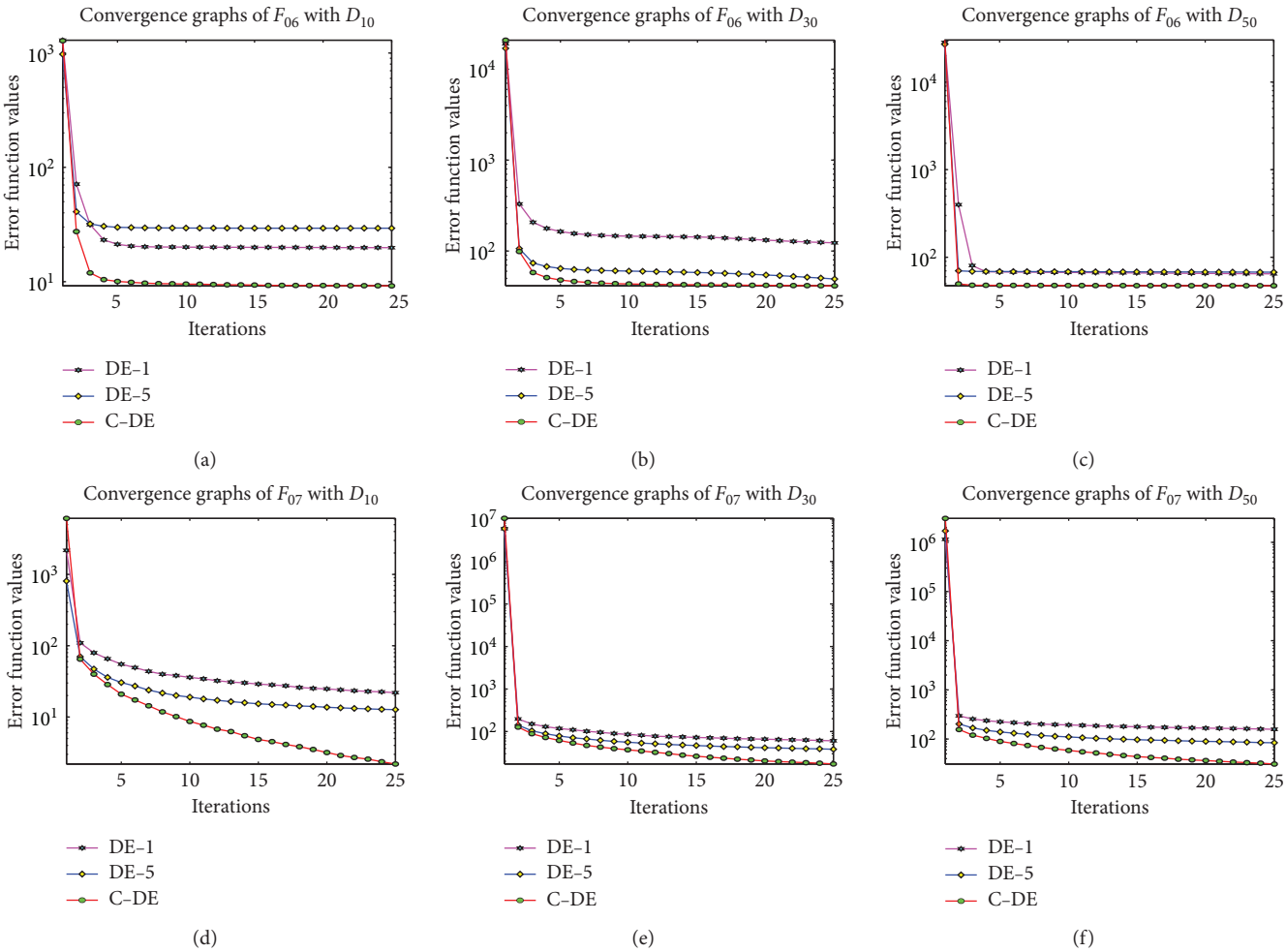

Figure 3: Continued.

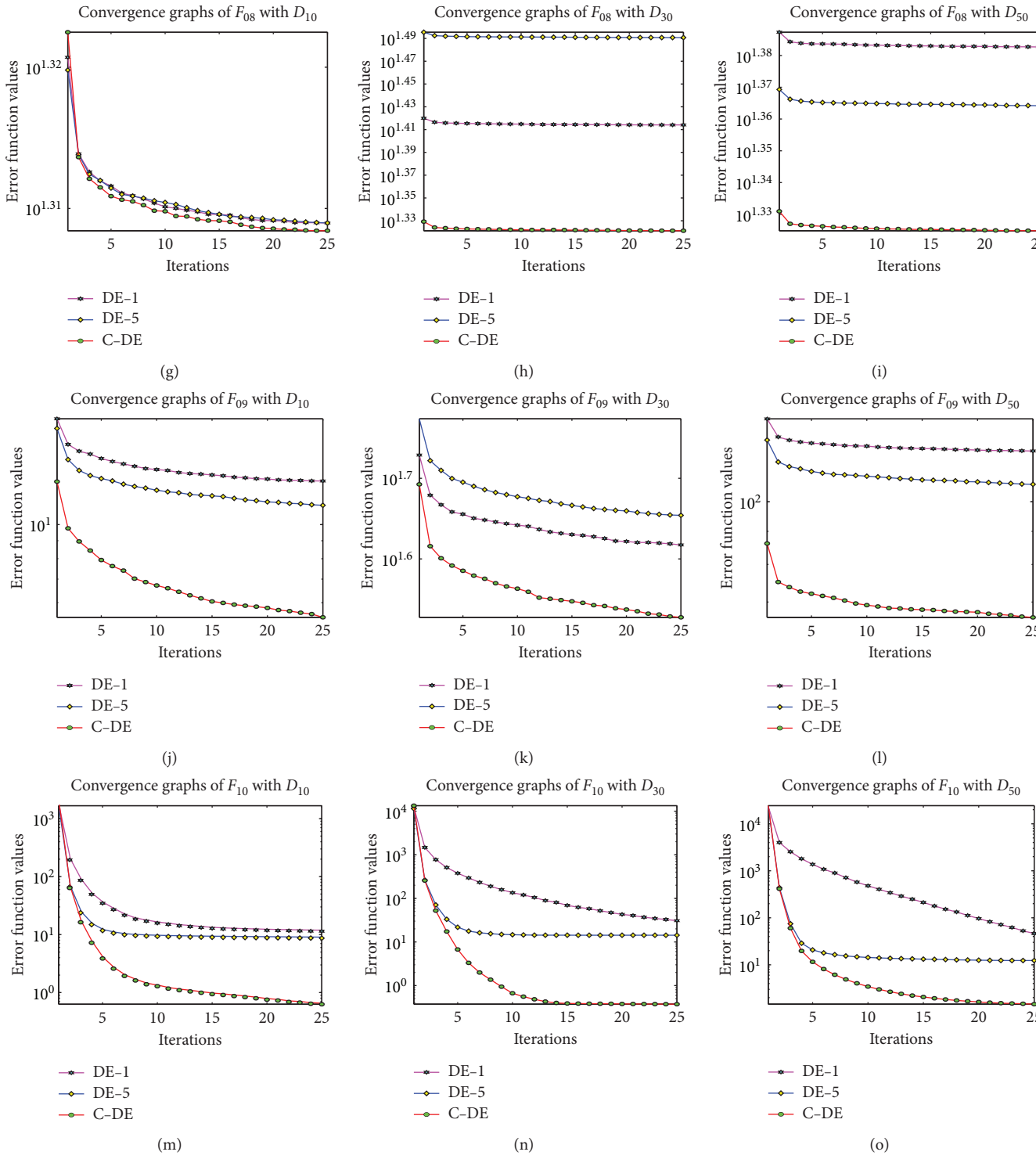

Figure 3: Evolution in the best function values of F_6 to F_{10} solved with DE-1, DE-5, and C-DE in $n = 10$, 30, and 50 dimensions.

Figure 7 displays the convergence graph of the F_{26} to F_{28} benchmark functions solved by the proposed C-DE algorithm versus DE-1 and DE-5 after 51 independent runs of simulation with different random seeds. The first panel of Figure 7 depicts the F_{26} to F_{28} benchmark functions as solved in ten dimensions, the second panel is for thirty

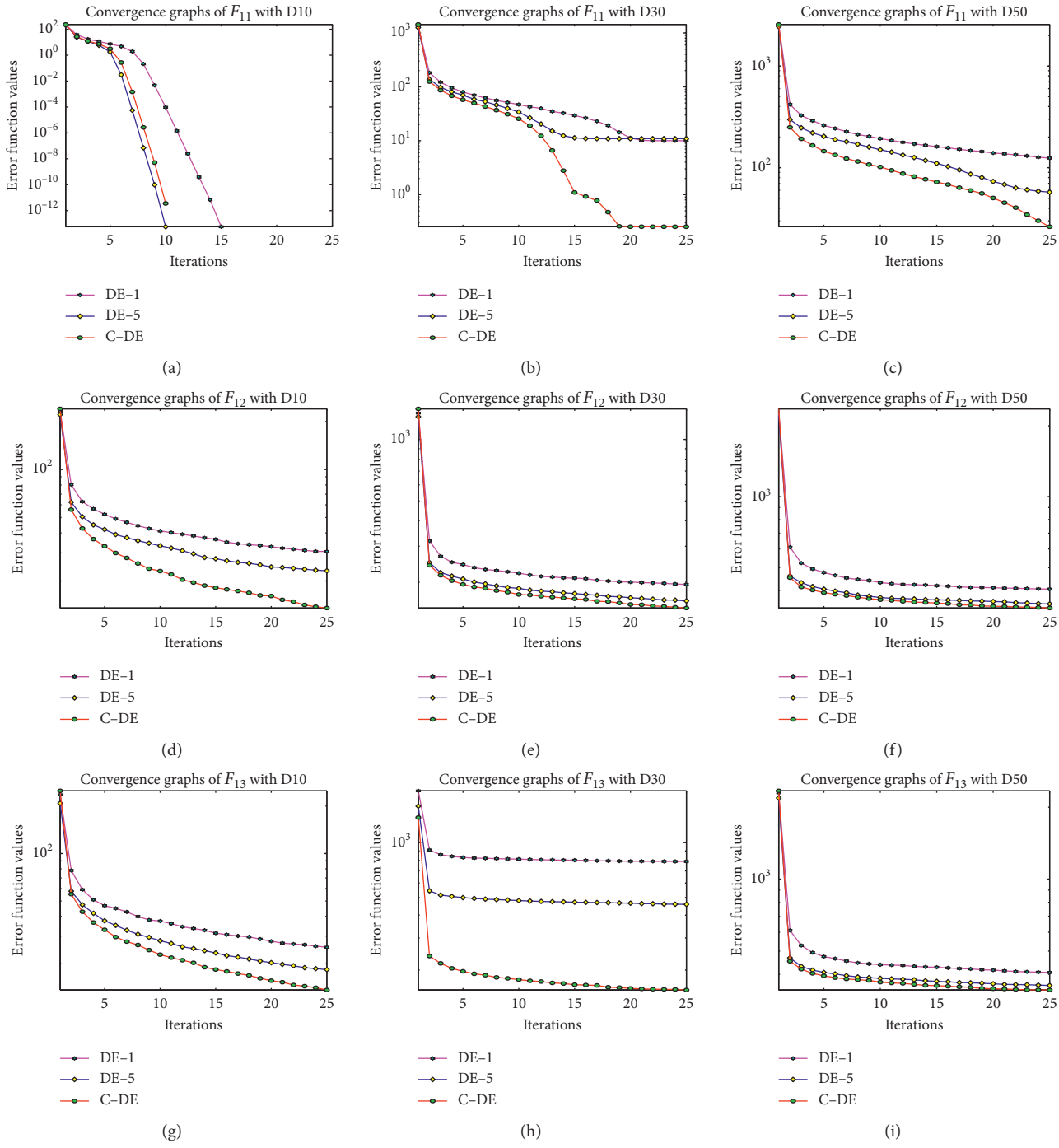

Figure 4: Continued.

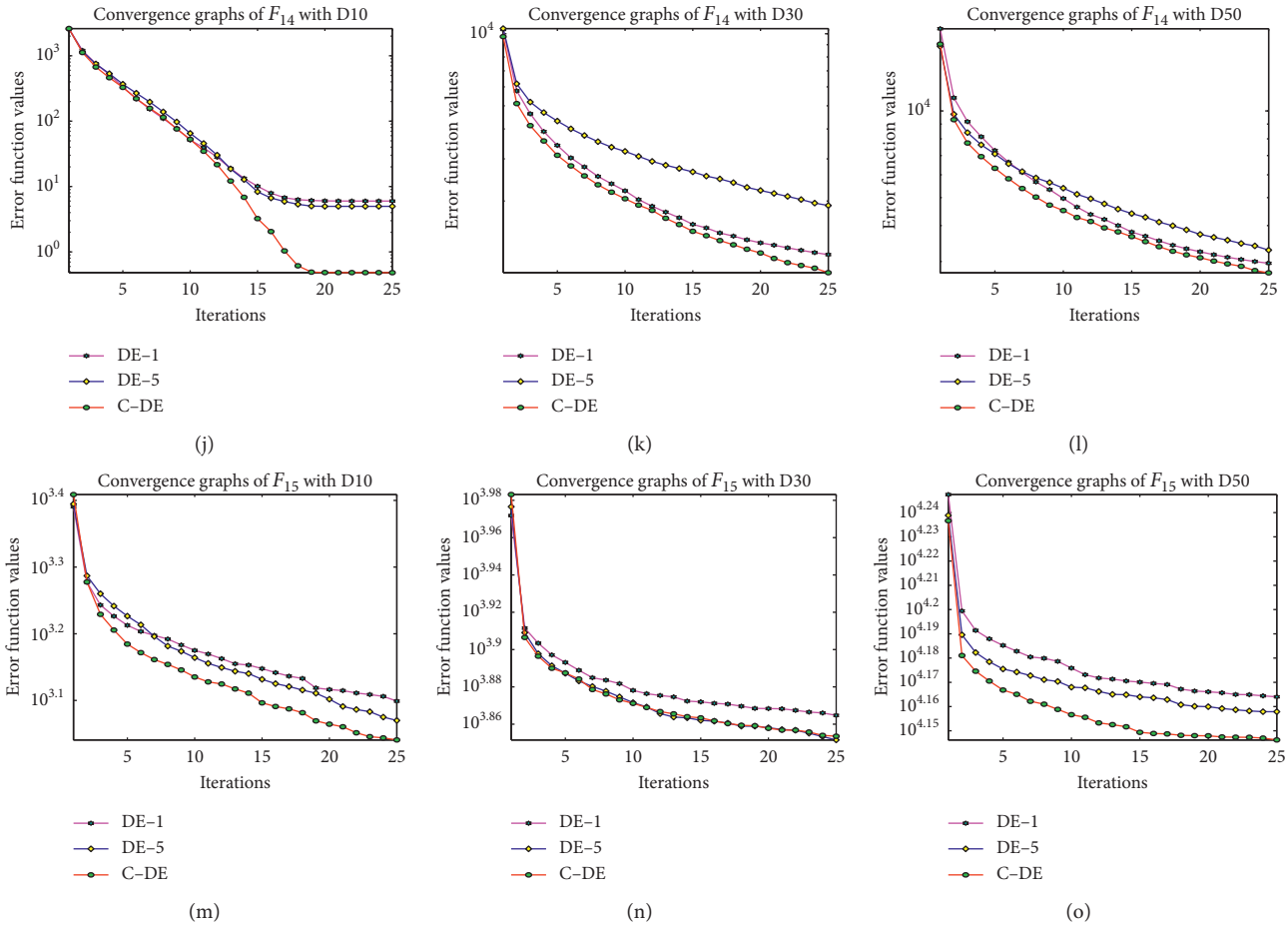

FIGURE 4: Evolution in the best function values of F_{11} to F_{15} solved with DE-1, DE-5, and C-DE in $n = 10$, 30, and 50 dimensions.

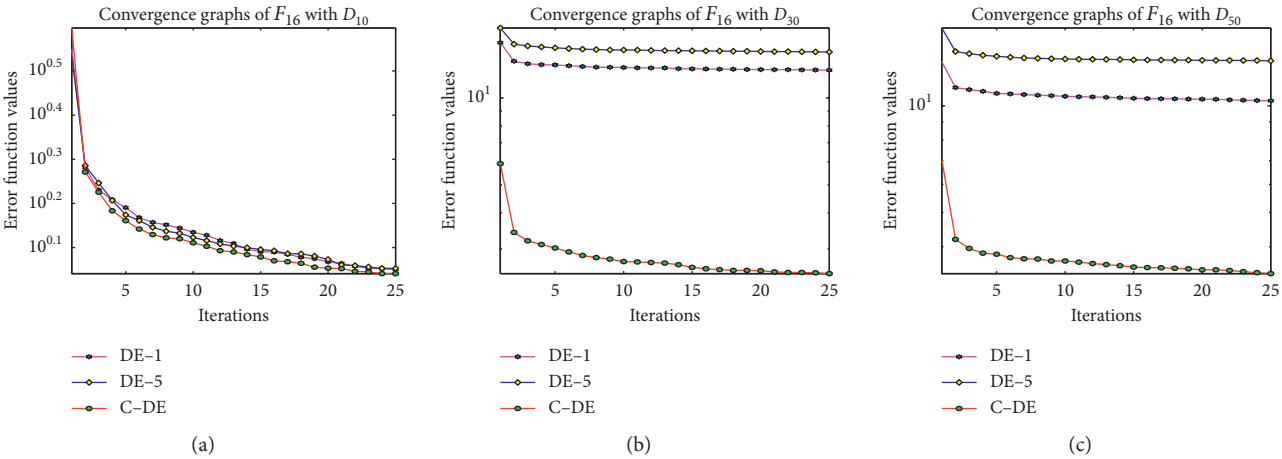

FIGURE 5: Continued.

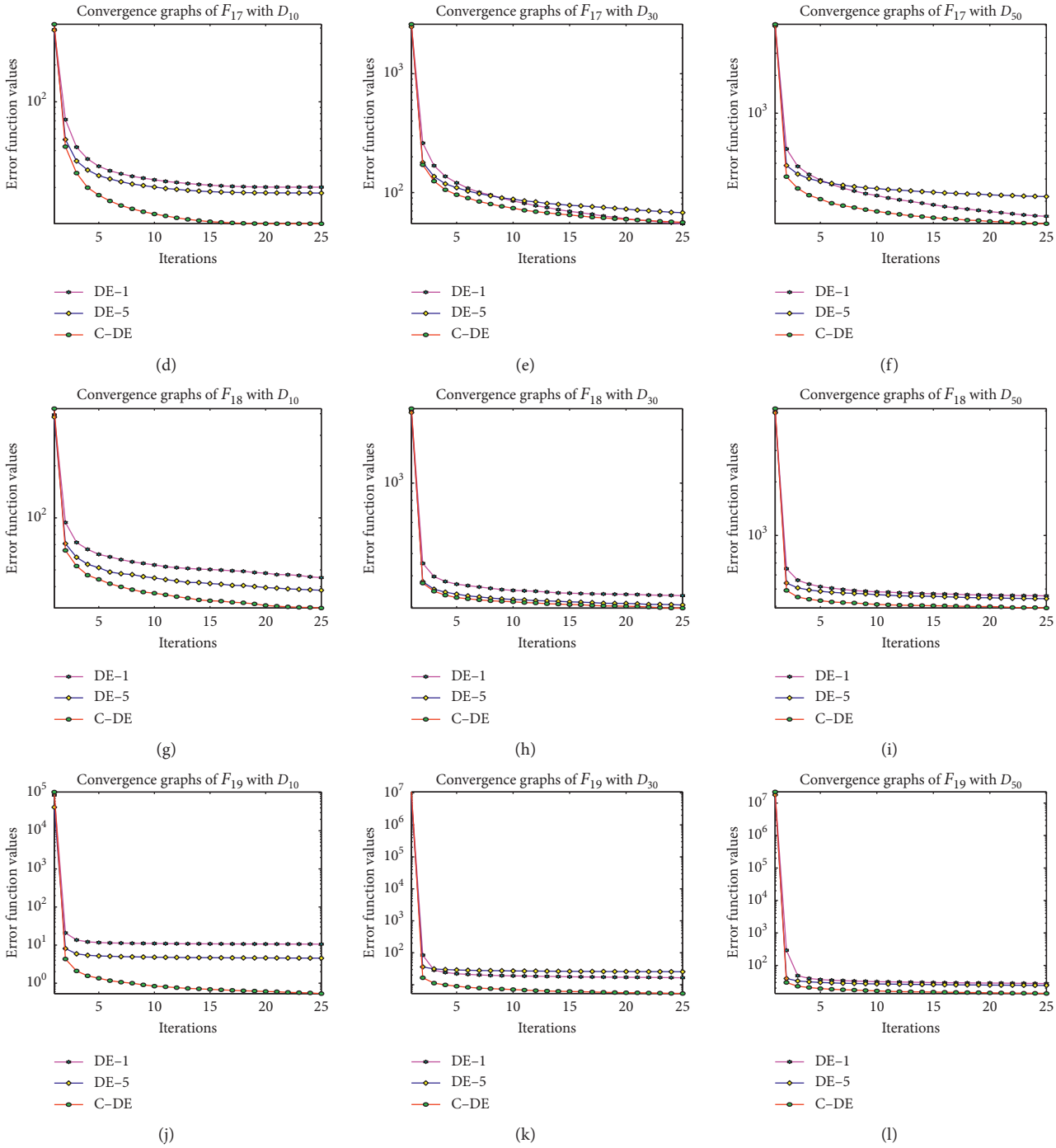

Figure 5: Continued.

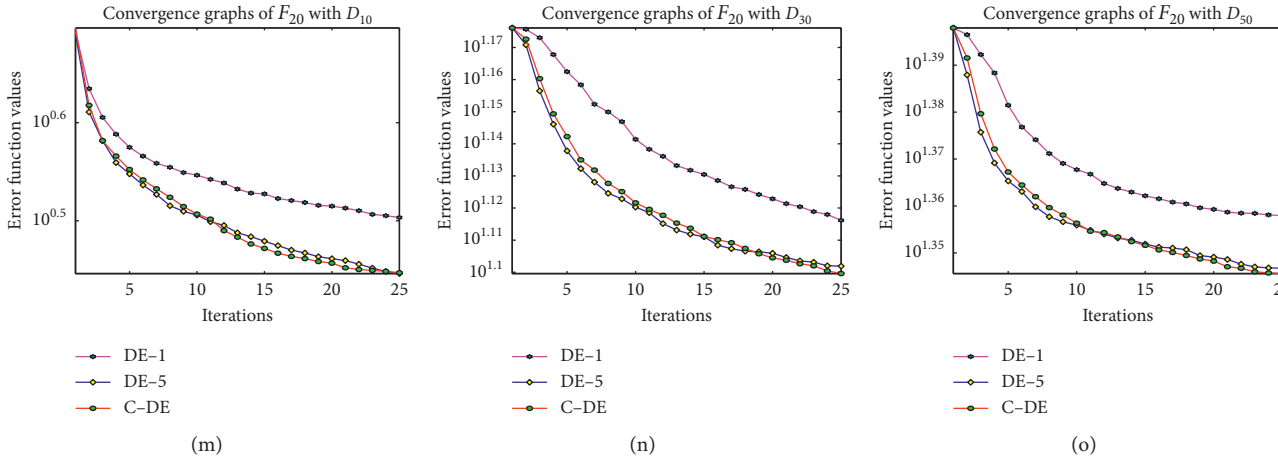

Figure 5: Evolution in the best function values of F_{16} to F_{20} solved with DE-1, DE-5, and C-DE in $n = 10$, 30, and 50 dimensions.

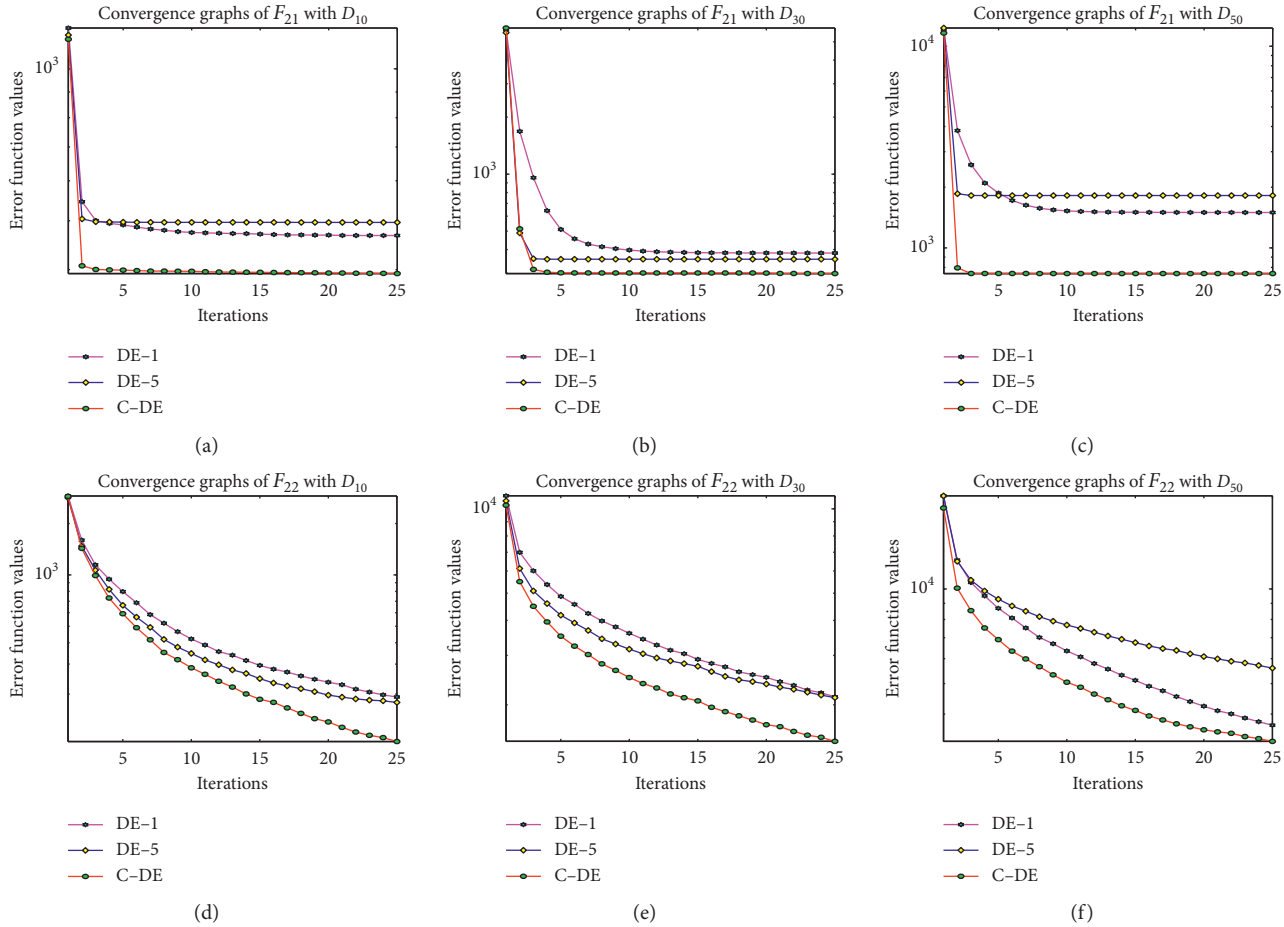

Figure 6: Continued.

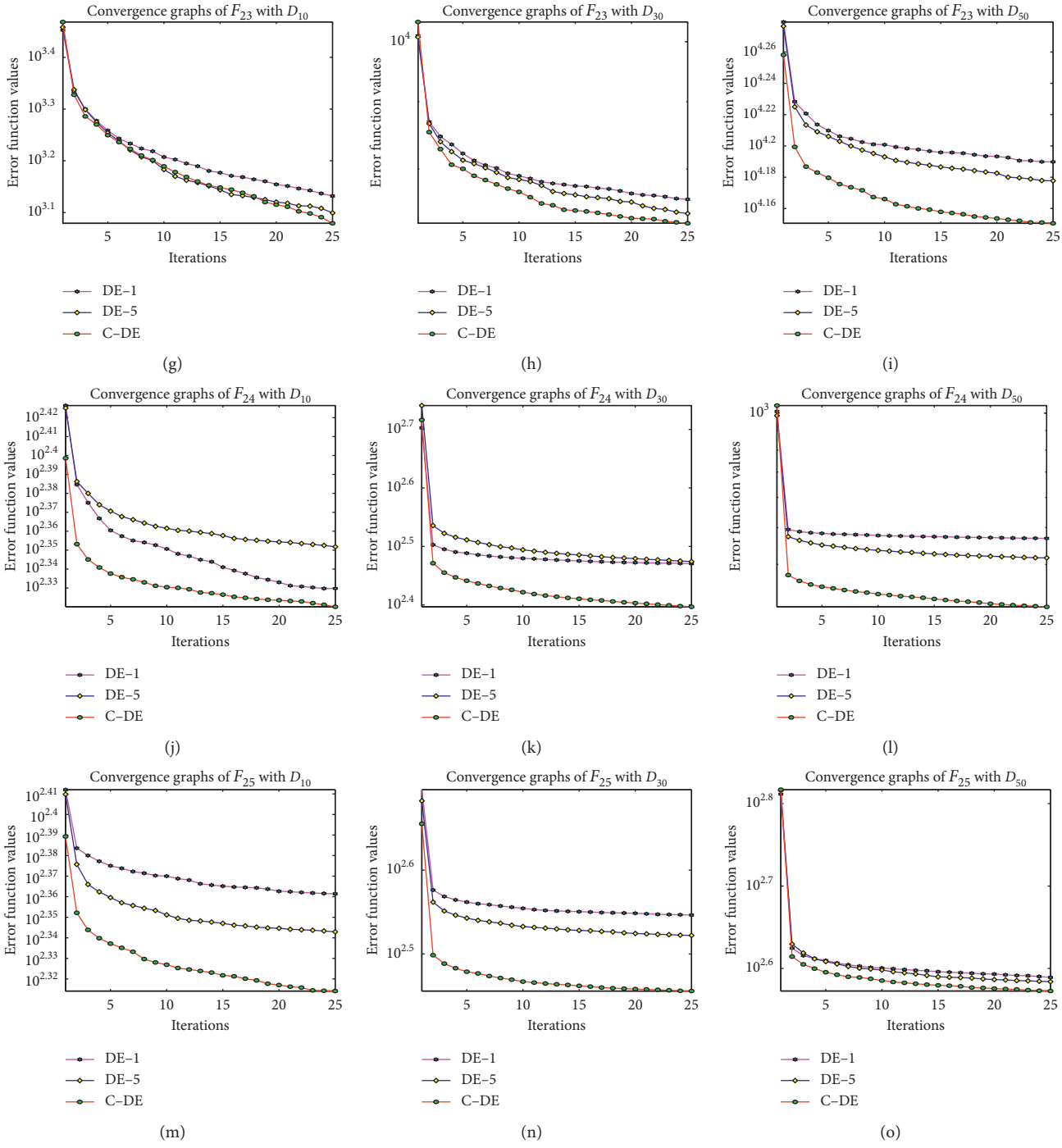

Figure 6: Evolution in the best function values of F_{21} to F_{25} solved with DE-1, DE-5, and C-DE in n = 10, 30, and 50 dimensions.

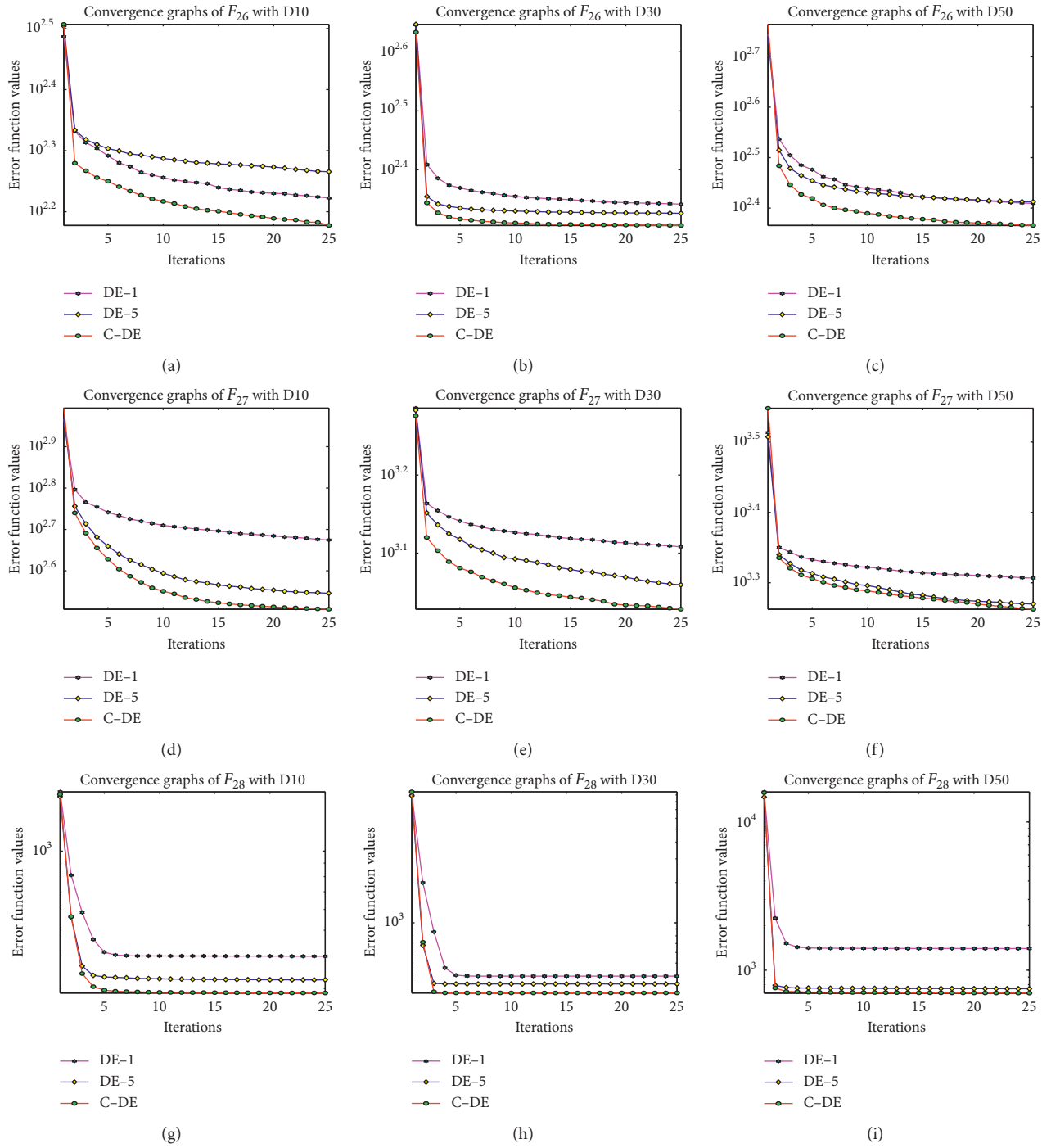

FIGURE 7: Evolution in the best function values of F_{26} to F_{28} solved with DE-1, DE-5, and C-DE in $n = 10$, 30, and 50 dimensions.

dimensions, and the third panel is for fifty dimensions.

4. Conclusion

In this paper, three variants of differential evolutionary algorithms, namely, DE-1, DE-5, and C-DE, are suggested and their numerical results in terms of best, worst, mean, median, and Std are obtained for the unconstrained benchmark functions designed for the special session on evolutionary algorithm competition (the 2013 IEEE Conference on Evolutionary Computation). Among the three DE variants, C-DE as outlined in Algorithm 6 has obtained better experimental results as compared to DE-1 as reported in Algorithm 1 and DE-5 described in Algorithm 5. Based on comparative analysis, we conclude the following:

The best fitness values of C-DE are much better for mostly test problems with $n = 10, 30, 50$ subjects to the "No Free Lunch Theorem" concept in the parlance of evolutionary computing communities.

Based on the worst objective function values, C-DE has again prominent position among the other two compared algorithms in order to solve most benchmark functions with dimensions keeping in view the criterion of "No Free Lunch Theorem" which guarantees the transparency of the manuscript.

The mean objective function values indicate in respect of C-DE much better than the average function values of other DE variant algorithms.

The median objective function values found by C-DE are much promising keeping in view the median values of the competing algorithms which reflect the fairness of the proposed algorithm.

Competitors play a comparable role keeping in view the standard deviation values obtained for the used benchmark function.

The efficiency and robustness of the proposed algorithm are satisfactory which can be observed from the error function values as plotted against the number of iterations.

In future, we will be planning for the following tasks:

To check the efficiency of C-DE on other existing unconstrained suites to more elaborate the importance of the proposed algorithm

To do parameter sensitivity to ensure the correct use of parameters on specific type problems is included in the future work

For checking the robustness and efficiency of the proposed algorithm, we are intended to compare it with the advanced version of DE such as JADE, SHADE, and IDE

For customization, the proposed algorithm with other advanced variants of DE can be come in count in future

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. Furthermore, the MATLAB code of the benchmark functions is available on https://www3.ntu.edu.sg/home/epnsugan/

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Acknowledgments

The authors acknowledge the support of the Research Intensified Grant Scheme (RIGS) Fasa 1/2019, Vot Number 55192/7, Universiti Malaysia Terengganu.

References

[1] G. Wu, R. Mallipeddi, and P. Suganthan, *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*, National University of Defense Technology, Changsha, Hunan, 2017.

[2] I. N. daSilva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni, and S. F. dos ReisAlves, *Artificial Neural Networks: A Practical Course*, Springer Publishing Company, Berlin, Germany, 1st edition, 2016.

[3] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Springer Publishing Company, Berlin, Germany, 2nd edition, 2015.

[4] S.-C. Fang and S. Puthenpura, *Linear Optimization and Extensions: Theory and Algorithms*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[5] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Springer, Berlin, Germany, 1992.

[6] R. E. Miller, *Optimization: Foundations and Applications*, John Wiley & Sons, Hoboken, NJ, USA, 1999.

[7] X. Yu and M. Gen, "Introduction to evolutionary algorithms," in *Decision Engineering*, R. Roy, Ed., Springer, New York, NY, USA, 2011.

[8] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, Inc., New York, NY, USA, 1996.

[9] E. E. Agoston, *Introduction to Evolutionary Computing*, Springer Publishing Company, Berlin, Germany, 2nd edition, 2015.

[10] R. Mallipeddi and P. N. Suganthan, "Differential evolution algorithm with ensemble of populations for global numerical optimization," *Opsearch*, vol. 46, no. 2, pp. 184–213, 2009.

[11] S. Akyol and B. Alatas, "Plant intelligence based metaheuristic optimization algorithms," *Artificial Intelligence Review*, vol. 47, no. 4, pp. 417–462, 2017.

[12] B. I. S. MuhammadSulaiman, S. Abdellah, and O. B. Kirikchi, "A plant propagation algorithm for constrained engineering optimisation problems," *Mathematical Problems in Engineering*, vol. 2014, p. 10, 2014.

[13] S. Hong, D. Han, K. Cho, J. S. Shin, and J. Noh, "Physics-based full-body soccer motion control for dribbling and shooting," *ACM Transactions on Graphics*, vol. 38, no. 4, pp. 81–74, 2019.

[14] N. Siddique and H. Adeli, "Physics-based search and optimization: inspirations from nature," *Expert Systems*, vol. 33, no. 6, pp. 607–623, 2016.

[15] F. Johari, A. Zain, N. Mustaffa, and A. Udin, "Firefly algorithm for optimization problem," *Applied Mechanics and Materials*, vol. 421, 2013.

[16] L. Li and F. Liu, *Group Search Optimization for Applications in Structural Design*, Springer, Berlin, Germany, 2011.

[17] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pp. 65–74, Springer, Granada, Spain, May 2010.

[18] X. Yu and M. Gen, *Introduction to Evolutionary Algorithms*, Springer Science & Business Media, Berlin, Germany, 2010.

[19] J. H. Jeong and C. W. Ahn, Edited by Handa, Ed., "Automatic evolutionary music composition based on multi-objective genetic algorithm," in *18th Asia Pacific Symposium on Intelligent and Evolutionary Systems*, Ishibuchi, Y.-S. Ong, and K.-C. Tan, Eds., Springer International Publishing, Cham, Switzerland, 2015.

[20] F. T. Silva, M. X. Silva, and J. C. Belchior, "A new genetic algorithm approach applied to atomic and molecular cluster studies," *Frontiers in Chemistry*, vol. 7, p. 707, 2019.

[21] L. While and G. Kendall, "Scheduling the English football league with a multi-objective evolutionary algorithm," in

*Parallel Problem Solving from Nature–PPSN XIII*Springer International Publishing, Cham, Switzerland, 2014.

[22] H. Mühlenbein and T. Mahnig, *Mathematical Analysis of Evolutionary Algorithms*, Springer US, Boston, MA, USA, 2002.

[23] C. Blum, "Ant colony optimization: introduction and recent trends," *Physics of Life Reviews*, vol. 2, no. 4, pp. 353–373, 2005.

[24] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, October 1995.

[25] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2-3, pp. 235–306, 2002.

[26] D. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi, *The Bees Algorithm, Technical Note, Manufacturing Engineering Centre*, Cardiff University, Cardiff, UK, 2005.

[27] J. R. Rohan, C. N. Prasad, and P. Sadiq, "An introduction to the collective behaviour of swarm intelligence," in *Proceedings of the National Conference on Contemporary Research and Innovations in Computer Science*, Hyderabad, India, December 2017.

[28] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings*, pp. 69–73, IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, May 1998.

[29] Y. M. Xie and G. P. Steven, "A simple evolutionary procedure for structural optimization," *Computers & Structures*, vol. 49, no. 5, pp. 885–896, 1993.

[30] C. Grosan and A. Abraham, "Hybrid evolutionary algorithms: methodologies, architectures, and reviews," 2007.

[31] W. Khan, *Hybrid Multiobjective Evolutionary Algorithm Based on Decomposition*, PhD, Department of Mathematical Sciences, University of Essex, Colchester, UK, 2012.

[32] W. K. Mashwani, "Hybrid multiobjective evolutionary algorithms: a survey of the state-of-the-art," *International Journal of Computer Science Issues*, vol. 8, no. 6, pp. 374–392, 2011.

[33] W. K. Mashwani, "MOEA/D with DE and PSO: MOEA/D-DE+PSO," in *The Thirty-First SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pp. 217–221, Cambridge, UK, December, 2011.

[34] W. K. Mashwani, "Comprehensive survey of the hybrid evolutionary algorithms," *International Journal of Applied Evolutionary Computation (IJAEC)*, vol. 4, no. 2, pp. 1–19, 2013.

[35] W. K. Mashwani and A. Salhi, "A decomposition-based hybrid multiobjective evolutionary algorithm with dynamic resource allocation," *Applied Soft Computing*, vol. 12, no. 9, pp. 2765–2780, 2012.

[36] W. K. Mashwani, "Multiobjective memetic algorithm based on decomposition," *Applied Soft Computing*, vol. 21, pp. 221–243, 2014.

[37] X. Qian, X. Wang, Y. Su, and L. He, "An effective hybrid evolutionary algorithm for solving the numerical optimization problems," *Journal of Physics: Conference Series*, vol. 1004, 2018.

[38] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," *SIAM Journal on Computing*, vol. 2, no. 2, pp. 88–105, 1973.

[39] F. H. M. T. Back and H. Schwefel, "A survey of evolution strategies," in *Proceeding of the Fourth International Conference on "Genetic Algorithms*, Morgan Kauffman, Mateo, CA, USA San Diego, CA, USA, July 1991.

[40] J. R. Koza, *Genetic Programming II*, MIT press, Cambridge, UK, 1994.

[41] R. Storn and K. V. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," Technical Report TR-95-012, ICSI, New Delhi, India, 1995.

[42] J. Kim, T. Sharma, B. Kumar, G. Tomar, K. Berry, and W. Hyung, "Research article intercluster Ant colony optimization algorithm for wireless sensor network in dense environment," *International Journal of Distributed Sensor Networks*, vol. 151, 2014.

[43] L. Perez Caceres, M. Lopez-Ibanez, and T. Stutzle, "Ant colony optimization on a limited budget of evaluations," *Swarm Intellingence*, vol. 9, 2015.

[44] X.-S. Yang and S. Deb, "Cuckoo search via levy flights," in *Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing*, pp. 210–214, IEEE, Kitakyushu, Japan, December 2009.

[45] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 3, no. 4, pp. 535–560, 2012.

[46] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[47] A. Biswas, S. Dasgupta, B. K. Panigrahi et al., "Economic load dispatch using a chemotactic differential evolution algorithm," in *Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems, ser. HAIS '09*, pp. 252–260, León, Spain, September 2009.

[48] S. Chakraverty, D. M. Sahoo, and N. R. Mahato, *Concepts of Soft Computing-Fuzzy and ANN with Programming*, Springer, Berlin, Germany, 2019.

[49] P. N. Sugunthan, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," *Natural Computing*, vol. 341, 2005.

[50] O. Kramer, D. E. Ciaurri, and S. Koziel, *Derivative-Free Optimization*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[51] R. Storn and K. Price, "Minimizing the real functions of the ICEC'96 contest by differential evolution," in *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 842–844, IEEE, Nagoya, Japan, May 1996.

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

[53] U. K. Chakraborty, *Advances in Differential Evolution*, Springer Publishing Company, Berlin, Germany, 1st edition, 2008.

[54] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part.B*, vol. 42, no. 2, pp. 482–500, 2012.

[55] R. A. Khanum, M. A. Jan, W. K. Mashwani, N. M. Tairan, H. U. Khan, and H. Shah, "On the hybridization of global and local search methods," *Journal of Intelligent & Fuzzy Systems*, vol. 35, no. 3, pp. 3451–3464, 2018.

[56] W. K. Mashwani, "Enhanced versions of differential evolution: state-of-the-art survey," *International Journal Computing Sciences and Mathematics*, vol. 5, no. 2, pp. 107–126, 2014.
[57] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, pp. 61–106, 2010.
[58] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Transactions On Evolutionary Computation*, vol. 12, no. 1, pp. 107–125, 2008.
[59] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*, Springer-Verlag, Berlin, Germany, 2005.
[60] M. S. Rainer, *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*, Springer-Verlag, Berlin, Germany, 2005.
[61] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.
[62] F. G. Lobo, C. F. Lima, and Z. Michalewicz, *Parameter Setting in Evolutionary Algorithms*, Springer Publishing Company, Berlin, Germany, 1st edition, 2007.
[63] J. Liang, B. Qu, P. Suganthan, and A. Hernández-Díaz, *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China, 2013.
[64] W. K. Mashwani, S. N. A. Shah, S. B. Belhaouari, and A. Hamdi, "Ameliorated ensemble strategy-based evolutionary algorithm with dynamic resources allocations," *International Journal of Computational Intelligence Systems*, vol. 14, pp. 412–437, 2020.