mathematics

MDPI

Article

An Enhancing Differential Evolution Algorithm with a Rank-Up Selection: RUSDE

Kai Zhang [1,†] and Yicheng Yu [2,*,†]

[1] Department of Computer Science and Engineering, National Chung Hsing University, Taichung 402, Taiwan; kai.zhang@smail.nchu.edu.tw
[2] Cyberspace Security Research Center, Pengcheng Laboratory, Shenzhen 518055, China
* Correspondence: yuych@pcl.ac.cn
† These authors contributed equally to this work and co-first authors.

Abstract: Recently, the differential evolution (DE) algorithm has been widely used to solve many practical problems. However, DE may suffer from stagnation problems in the iteration process. Thus, we propose an enhancing differential evolution with a rank-up selection, named RUSDE. First, the rank-up individuals in the current population are selected and stored into a new archive; second, a debating mutation strategy is adopted in terms of the updating status of the current population to decide the parent's selection. Both of the two methods can improve the performance of DE. We conducted numerical experiments based on various functions from CEC 2014, where the results demonstrated excellent performance of this algorithm. Furthermore, this algorithm is applied to the real-world optimization problem of the four-bar linkages, where the results show that the performance of RUSDE is better than other algorithms.

Keywords: evolutionary algorithm; differential evolution; rank-up based; four-bar mechanism

Citation: Zhang, K.; Yu, Y. An Enhancing Differential Evolution Algorithm with a Rank-Up Selection: RUSDE. *Mathematics* **2021**, *9*, 569. https://doi.org/10.3390/math9050569

Academic Editor: Alfredo Milani

Received: 7 February 2021
Accepted: 3 March 2021
Published: 7 March 2021

1. Introduction

In real-world optimization problems, people have proposed various algorithms inspired by evolutions of lives, social behaviors, or physical phenomena, e.g., Genetic Algorithm (GA) [1], Ant Colony Optimization Algorithm (ACO) [2], Particle Swarm Optimization (PSO) [3], Cuckoo Search (CS) [4], Teach-Learning-Based Optimization (TLBO) [5–7], Krill Herd (KH) [8], Backtracking Search Optimization Algorithm (BSA) [9] and differential evolution (DE) [10]. Among them, the differential evolution (DE) algorithm and its variants showed excellent performance in competitions held under the IEEE Congress on Evolutionary Computation (CEC) conference series [11,12]. In recent years, DE has been widely applied in diverse fields, such as geography [13], chemistry [14], engineering design [15–18], image processing [19,20], and software development [21]. Additionally, there are many successful applications of the DE in these fields [22–24].

DE starts from a randomly generated initial population. A new individual is generated by summing the vector differences of any two individuals and the third individual. Then the new individual is compared with the corresponding individual in the current population. When the fitness of the new individual is better than that of the current individual, the old individual will be replaced by the new one in the next generation. Through continuous evolution, competition, retention, and deletion, the optimal solution is obtained.

However, in DE exists two problems. First, successful solutions may stop generating, and second, no fixed points can be converged; these two may lead to stagnation. In order to improve DE's performance and reduce the stagnation, some researchers proposed diverse versions. Qin adjusted the parameters of DE and determined adaptively a more suitable generation strategy along with its parameter settings in the search process (SaDE) [25]. Brest introduced an efficient technique for adapting control parameter settings associated with DE (jDE) [26]. Zhang created an optimal archive that contains new updating individuals,

that helps to generate the better one (JADE) [27]. Lampinen introduced an adaptive strategy with the control parameters on the base of JADE, and obtained better results (SHADE) [28]. Guo improved the optimal archive with all updating individuals and built a new Successful-Parent-Selecting framework (SPSDE) [29]. Later, she introduced the concept of eigenvector and proposed a new crossover method (EIGDE) [30]. The ranking idea was utilized to search for a better position, for example, the low-ranking individuals learn from the top-ranking ones [31,32]. Lixin combined adaptive population classification, adaptive control parameters, and mutation to create an Individual Dependent Mechanism (IDE) [33]. Some other concepts or methods were introduced, such as, neighborhood-based mutation operator (DEGL) [34], ensemble of parameters and mutation (EPSDE) [35], (MDEpBX) [36], a similarity-based mutant vector generation strategy (DE-SIM) [37], multipopulation-based mutation operators (CAMDE) [38], random neighbors based strategy (RNDE) [39], and adaptive lagrange interpolation search (ADELI) [40].

By far, plenty of algorithms adopt the two learning strategies: The individuals with low ranking learn from the ones with high ranking, such as rank-jDE [31] and TLBO [5]; the individuals without updating learn from the ones with updating, such as JADE [27], SHADE [28], and SPSDE [29]. These two learning strategies had been proved to be valid in improving the performance of DE. Therefore, this study proposes a new learning strategy to further reduce the stagnation of DE and accelerate the convergence. The individuals with no rank updating learn from the ones with rank updating. That is, enhancing differential evolution algorithm with a rank-up selection (RUSDE). The main idea of the proposed framework contains two parts: one is that an archive is created during the process which consists of the rank-up individuals, the other one is that a learning strategy has been adopted in terms of the continuous updating numbers of the top-rank individuals and the current individual. We discuss the influence of the archive size, test the RUSDE's performance by CEC 2014 benchmarks [41]. The results show that RUSDE performs better than the other 11 algorithms. We apply the RUSDE into the optimal synthesis problem of four-bar mechanisms, and obtain a considerable result.

The outline of the rest of this paper is organized as follows. We discuss the original DE in Section 2, and RUSDE is proposed in Section 3. Section 4 shows the experimental results of different functions that show the superiority of RUSDE. Section 5 takes an example for the synthesis problem of four-bar mechanism. Section 6 concludes the paper.

2. Related Works

2.1. Differential Evolution

Differential Evolution is a powerful evolutionary algorithm that approaches a single objective optimization problem [10]. Its process contains four steps.

Step 1, Initialization

Initialize the population X, whose size is N, for D dimensional problems, the i^{th} individual in d^{th} dimension is randomly generated within the bound.

Step 2, Mutation

The mutation is used to generate a new population by a change or perturbation from parents. The selection methods are diverse. Reference [10] lists some of the most frequently used strategies, such as "DE/rand/1", "DE/best/1", "DE/current-to-best/1", "DE/rand/2", and "DE/best/2". To explain them, we take "DE/rand/1" for example, shown in Equation (1).

$$v_i^{child} = x_a + F(x_b - x_c) \tag{1}$$

where x_a, x_b, and x_c are different individuals randomly chosen from population X, they are all different from the index i; F is the user-specified scaling factor in the range (0, 2).

Step 3, Crossover

For increasing the population diversity, a new child individual $u_{i,j}^{child}$ is generated by crossover operation The crossover scheme of DE can be written as Equation (2):

$$u_{i,d}^{child} = \begin{cases} v_{i,d}^{child} \text{ if } (\text{rand}(0,1) < \text{CR}) \text{ or } (d = d_{rand}) \\ x_{id} \text{ Otherwise} \end{cases} \quad d = 1, 2, \ldots, \text{D} \tag{2}$$

where d_{rand} is randomly chosen from [1,D]; CR is a user-defined crossover rate in the range of [0, 1].

Step 4, Selection

The greedy selection mechanism is implemented to yield the population survival into the next generation in terms of the fitness value after the crossover process. Only the better one can survive. The selection procedure can be mathematically expressed as Equation (3):

$$x_i = \begin{cases} u_i^{child} \text{ if } (f(u_i^{child}) < f(x_i)) \\ x_i \text{ Otherwise} \end{cases} \tag{3}$$

2.2. Differential Evolution Variants

In recent years, a number of DE variants occur to improve DE's performance. We divide them into four categories.

Improving mutation, crossover, or both strategies: For example, the opposition number is generated in population initialization (ODE) to realize the generation jumping and the local improvement [42,43]; the differential covariance matrix's i information is extracted for the determination of the search direction, similar to CMA-ES [44] combined with DE (DCMA) [45,46]. The covariance matrix learning is also used to create a proper coordinate system for the crossover operator, such as eigenvector (EIGDE) [30] and bimodal distribution parameter setting [47]. Besides, there are orthogonal crossover [48], adaptive Lagrange interpolation search(ADELI) [40], Gaussian mutation [49], and multiple strategies [50,51].

Adaptive control parameter strategies: The control parameters (scaling factor and crossover rate) have an effect on the performance. To balance the exploration and exploitation, adaptive or self-adaptive mechanisms are proposed, such as SaDE [25], jDE [26], SHADE [28], JADE [27], LTMA [52], IPOPCMAES [53] and jSO [54].

Alternating the parents: Some researchers select the parents according to the fitness values or rank to generate a better solution, such as neighborhood and directional Information [55], successful-parent-selecting framework (SPSDE) [29], and rank-based selection (rank-jDE) [31].

Building the population structure: The multiply populations strategies are introduced to adaptively enhance the performance of DE, such as IDE [33] and CAMDE [38]; An aging strategy is employed to jump out of the local optimum [56]. The pseudo-code for DE is given in Algorithm 1.

Algorithm 1 The pseudo code of DE/rand/1

```
Initialize a population X(N*D); Maximum iteration number MaxDT;
for G = 1 to MaxDT do
    for i = 1 to N do
        Randomly select x_a, x_b, x_c from the population X;
        Generate v_i by Equation (1);
        for d = 1 to D do
            Generate u_i by Equation (2);
        end for d
    end for i
    for i = 1 to N do
        Compute the fitness of u_i^child;
        if f(u_i^child) < f(x_i) then
            Replace x_i with u_i^child;
        end if
    end for i
end for G
```

3. The Proposed Methods

By far, the main idea of most ranking algorithms is that individuals with low ranking learn from individuals with high ranking. This is similar to human social behavior. Elites are in the top position and attract attention from people, hence everyone likes to learn from them. However, there is another group of people who have also attracted attention. Although they are not ranked high in their industries, their rankings have recently risen fast for some reason, hence people have begun to study the reasons for their rise and learn from it. Similar to the development of the national economy, some developed countries have advanced technology, but there will always be a period of slow economic growth, while some developing countries, which have relatively worse technology, have rapid economic growth. As a result, people will analyze the reason why these countries' economies grow rapidly and learn from them to develop their own economies. This prompts us to propose the ranking-up algorithm in this paper which contains two parts.

First, in order to increase the learning speed of DE, the individuals with no rank up learn from the ones with rank up. The detailed process is that we store the individuals whose rank is up recently in an archive Y. The size of the archive is M, Y is a matrix with $M * D$ and D is the dimension.

To judge the rank-up of the individual, we can use Equation (4):

$$rankupvalue(i, G) = rankvalue(i, G) - rankvalue(i, G - 1) \tag{4}$$

where *rankupvalue*(*i*,*G*) is the i^{th} individual's rank-up value in G^{th} generation; *rankvalue*(*i*,*G*) is the i^{th} individual's rank index in G^{th} generation. Hence, if *rankupvalue*(*i*,*G*) < 0, it means this individual's rank is up, then it will be stored into the archive Y. The method will obey the rule that the newest one replaces the oldest one. Second, a discussion is presented in terms of the current population. We use *flagrank*(*i*) to indicate the consecutive no-rising number of the current i^{th} individual's rank, and *flagbest* is indicated to the consecutive no-updating number of the global optimal individual, shown in Equations (5) and (6). We adapt the parent's selection method to perform the mutation process. The learning behavior of the current individual is discussed in six cases according to the current individual's ranking status: If the current individual's rank rises, then the individual will be used as the initial position of the learning behavior; if the current individual's rank does not rise twice consecutively, the individual will be replaced by a random individual from the archive Y; if the current individual's rank does not rise for one consecutive time, the individual will be replaced by a random individual from the current population X; for the global optimal individual of the current population, if it is not updated, all individuals will not learn from it. Otherwise, all individuals learn from it. As shown in Equation (7).

$$flagrank(i) = \begin{cases} 0, \text{ the } i^{th} \text{ individual's rank up} \\ flagrank(i) + 1 \text{ Otherwise} \end{cases} \tag{5}$$

$$flagbest = \begin{cases} 0, \text{ the best individual's fitness update} \\ flagbest + 1 \text{ Otherwise} \end{cases} \tag{6}$$

$$V_i = \begin{cases} Y1 + F_i(Y2 - Y3) + F_i(x_{best} - Y3), \\ \text{if } flagrank(i) >= 2,\ flagbest = 0 \\ Y1 + F_i(Y2 - Y3\ ), \\ \text{if } flagrank(i) >= 2,\ flagbest \neq 0 \\ X_i + F_i(Y2 - Y3) + F_i(x_{\text{best}} - Y3), \\ \text{if } flagrank(i) = 0,\ flagbest = 0 \\ X_i + F_i(Y2 - Y3), \\ \text{if } flagrank(i) = 0,\ flagbest \neq 0 \\ X1 + F_i(x_{better} - X2) + F_i(x_{best} - X2), \\ \text{if } flagrank(i) = 1,\ flagbest = 0 \\ X1 + F_i(x_{better} - X2), \\ \text{if } flagrank(i) = 1,\ flagbest \neq 0 \end{cases} \tag{7}$$

where $Y1$, $Y2$, and $Y3$ are individuals randomly selected from the archive Y; $X1$ and $X2$ are two random individuals in the current population; x_{best} is the global optimal individual in the current population; X_{better} is an individual whose rank is better than the current individual's; X_i is the current individual; F_i is the i^{th} individual's learning factor. In this study, F_i is set the same as Ref [26], shown in Equation (8).

$$F_i = \begin{cases} 0.1 + 0.9 * rand,\ rand < 0.1 \\ 0.5 \text{ Otherwise} \end{cases} \tag{8}$$

Crossover: in this study, we choose the same method as [26] to achieve cross behavior, shown in Equation (9).

$$u_{i,d}^{child} = \begin{cases} V_{i,j} \text{ if } rand \leq \text{CR or } j = rn(i) \\ X_{i,j} \text{ if } rand > \text{CR or } j \neq rn(i) \end{cases} \tag{9}$$

where $rn(i)$ represents a random integer from the set of integers 1, 2, ..., D, which is used to ensure that at least one dimension in $u_{i,d}^{child}$ comes from $V_{i,j}$. CR is a crossover operator rate whose value range is (0, 1). In this study, CR is set to 0.9, the same as in the literature [26].

Selection: when the new individual u_i^{child} is generated, its function fitness value will be calculated, compare the fitness value with the one of X_i, only the better one can survive.

In summary, the pseudo-code of the rank-up selection DE algorithm is shown in Algorithm 2.

In this paper, the crossover and selection part of RUSDE is the same with that of jDE [26]. Hence, RUSDE proposed in this paper is based on the frame of jDE. Besides, in order to alleviate the problem of stagnation of DE and improve the individual's updating rate, most algorithms create an archive with updated individuals, and then randomly extract individuals from the archive to generate the offspring, such as SPS frame [29]. RUSDE generates the new archive with the rank-up individuals. The difference is that the fitness value of the rank-up individual must be updated, while the rank of the individuals whose fitness value updated may not rise. In other words, this filter is more strict. Therefore, in the numerical experiments, RUSDE will be compared with jDE and SPS-jDE to show its performance.

Algorithm 2 The pseudo code of RUSDE

Initialize a population $X(N*D)$; Maximum iteration number $MaxDT$, size of archive $V(M)$ and let $count = 0$, $flagrank$ = zeros(N,1), $rankupvalue$ = zeros(N,$MaxDT$);Sort all individuals and calculate the current rank $rankvalue(i,1)$ of i^{th} individual. The first M individuals are stored into archive Y.
for G = 2 to $MaxDT$ **do**
  Sort all individuals in terms of fitness values;
  Record rank index of the i^{th} individual in $rankvalue(i, G)$
  Calculate the rank-up value $rankvalue(i, G)$ of i^{th} individual by Equation (4)
  for i = 1 to N **do**
    if $rankvalue(i, G) < 0$(% individual's rank up) **then**
      $count = count + 1$;
      $index$ = mod($count$,M);
      $Y(index,:) = x(i,:)$(%Store this individual into Y, and replace the earliest one);
      $flagrank(i) = 0$;
    else
      $flagrank(i) = flagrank(i) + 1$;
    end if
  end for i
  for i = 1 to N **do**
    Randomly select $X1$, $X2$, $X3$ from the population X and $Y1$, $Y2$, $Y3$ from the archive Y;
    Generate v_i by Equation (7);
    for d = 1 to D **do**
      Generate v_i by Equation (9);
    end for d
  end for i
  for i = 1 to N **do**
    Compute the fitness of u_i^{child};
    if $f(u_i^{child}) < f(x_i)$ **then**
      Replace x_i with u_i^{child};
      if $f(u_i^{child}) < f(gbest)$ **then**
        $gbest = u_i^{child}$;
        $flagbest = 0$;
      else
        $flagbest = flagbest + 1$;
      end if
    end if
  end for i
end for G

4. Experiments

4.1. Benchmarks and Experimental Settings

This study used a CEC 2014 benchmark problem [41] to evaluate RUSDE on the basis of comparison with other algorithms. The CEC 2014 benchmark functions can be divided into four categories: F1–F3 are unimodal functions, F4–F16 are simple multimodal functions, F17–F22 are hybrid functions, and F23–F30 are composition functions [57]. The shifted global optimum was determined by the random shift matrix $o = [o_1, o_2, \ldots, o_D]$, and the rotated matrix M was generated as described in a previous study [58]. The benchmarks are shown in Appendix A.

We compared the RUSDE with DE and its improved version, i.e., DErand [10], jDE [26], SaDE [25], Rank-jDE [31], SPS-jDE [29], jDE-EIG [30], LSHADE [59], and other well-known algorithms and improvements, i.e., KH [8], LBSA [60], DGSTLBO [61], SCA [62]. Each algorithm uses the same parameters as in the original literature. To be fair, the variables were set as follows: for 30-dimensional problems, swarm population N = 100 and function.

4.2. The Influence of Parameter M

In RUSDE, the size of the selection archive Y may have a crucial influence on the calculation of the optimal solution. We used CEC 2014 benchmark functions (see in Appendix A) to test the effect on the results with different parameter M values. For the 30-dimensional problems, we set different M values, N is set to 100, and FES = 200,000, the algorithms are performed for 20 times, respectively. In Table 1, we list the results of the average rank. As the parameter M increases, that means the size of the archive Y enlarges, and the information from Y may be out-of-date to lead the average rank get worse, while if the parameter M is too small, the diversity of Y becomes worse, which deteriorates the average rank. When M = 50, which is half of the population, the average ranking of RUSDE is the smallest.

Table 1. The influence of parameter M.

Func.	A1.	M = 20	M = 40	M = 50	M = 60	M = 80	M = 100
F1	Mean	2.94×10^{5}	5.10×10^{5}	3.96×10^{6}	4.01×10^{6}	5.00×10^{6}	3.43×10^{6}
F2	Mean	1.24×10^{-9}	2.19×10^{-8}	4.35×10^{-10}	1.76×10^{-8}	2.95×10^{-7}	1.84×10^{-6}
F3	Mean	3.48×10^{-10}	5.47×10^{-9}	1.04×10^{-13}	9.74×10^{-16}	1.92×10^{-15}	2.03×10^{-5}
F4	Mean	11.5	25.2	8.11	1.70	19	9.05
F5	Mean	20.6	20.6	20.4	20.4	20.4	20.3
F6	Mean	0.142	4.19×10^{-4}	0.135	12.3	14.4	14.3
F7	Mean	0.00	7.40×10^{-4}	0.00	0.00	0.00	1.11×10^{-17}
F8	Mean	0.00000011	0.0000358	6.16×10^{-10}	4.41×10^{-10}	1.52×10^{-9}	1.7×10^{-9}
F9	Mean	33.6	58.6	49.1	52.1	53.0	54.8
F10	Mean	40.8	59.8	4.91×10^{-3}	5.78×10^{-3}	6.90×10^{-3}	1.61×10^{-2}
F11	Mean	2.23×10^{3}	4.16×10^{3}	2.70×10^{3}	2.74×10^{3}	2.77×10^{3}	2.73×10^{3}
F12	Mean	0.569	0.952	0.517	0.536	0.489	0.549
F13	Mean	0.271	0.28	0.237	0.239	0.225	0.292
F14	Mean	0.29	0.292	0.278	0.255	0.262	0.251
F15	Mean	3.96	10.0	6.36	6.59	6.28	6.52
F16	Mean	10.9	11.2	10.0	9.95	10.1	10.2
F17	Mean	5.53×10^{-3}	4.04×10^{3}	1.29×10^{3}	6.64×10^{2}	6.48×10^{2}	6.34×10^{2}
F18	Mean	44.8	47.2	14.4	15.2	17.7	13.2
F19	Mean	3.32	4.13	4.57	4.82	5.02	6.15
F20	Mean	11.6	12.0	12.1	17.0	9.11	9.47
F21	Mean	9.48×10^{2}	8.27×10^{2}	3.26×10^{2}	3.15×10^{2}	3.61×10^{2}	3.47×10^{2}
F22	Mean	1.29×10^{2}	1.01×10^{2}	1.85×10^{2}	1.70×10^{2}	1.38×10^{2}	2.21×10^{2}
F23	Mean	3.15×10^{2}	3.15×10^{2}	3.15×10^{2}	3.15×10^{2}	3.15×10^{2}	3.15×10^{2}
F24	Mean	2.23×10^{2}	2.23×10^{2}	2.23×10^{2}	2.23×10^{2}	2.24×10^{2}	2.24×10^{2}
F25	Mean	2.03×10^{2}	2.03×10^{2}	2.05×10^{2}	2.05×10^{2}	2.05×10^{2}	2.05×10^{2}
F26	Mean	1.00×10^{2}	1.00×10^{2}	1.00×10^{2}	1.00×10^{2}	1.00×10^{2}	1.00×10^{2}
F27	Mean	3.14×10^{2}	3.09×10^{2}	3.00×10^{2}	3.00×10^{2}	3.00×10^{2}	3.04×10^{2}
F28	Mean	8.04×10^{2}	8.16×10^{2}	7.95×10^{2}	8.10×10^{2}	8.09×10^{2}	8.14×10^{2}
F29	Mean	1.06×10^{3}	9.73×10^{2}	9.58×10^{2}	7.95×10^{2}	8.00×10^{2}	1.30×10^{3}
F30	Mean	1.08×10^{3}	1.29×10^{3}	1.29×10^{3}	1.40×10^{3}	1.78×10^{3}	1.87×10^{3}
Average Rank		3.07	4.20	2.37	2.57	3.07	3.80

evaluations FEs = 300,000; for 50-dimensional problems, N = 100 and FEs = 500,000; The experiment is carried out on a PC with a 3.6G HZ CPU and 32G memory, and the software used is MATLAB R2016b in a Windows 64-bit environment.

In order to discuss the statistical significance of the influence of the parameter M, we use Friedman test to assess the impact of it. Assume h0: Different parameters M have no effect on the operating results of the RUSDE. Fromthe F test results, F-score is 5.117, which is larger than 2.545 (the significance level is taken to 0.05, Freedom is 10). This means that this result negates the original hypothesis of Friedman's test. Therefore, the parameter M has significant difference in the calculation of the optimal value of RUSDE. Therefore, in the later experiments, we all take M = 50.

4.3. Comparisons with Other Algorithms and Statistical Analysis of The Results

We used CEC 2014 to test 11 evolutionary algorithms (EAs) with 30 and 50-dimensional problems, and ranked the average value. The best, mean, and std values are shown in Tables 2 and 3. It can be seen that for the 30-dimensional problems, the average rank of LSHADE, RUSDE, and SaDE were in the first, second, and third places, respectively. RUSDE was ranked first for F2, F7, F9, F12, F16, F19, and F26, with a total of seven first ranks. LSHADE was ranked first for F1, F7, F9, F11, F15, F17, F18, F20, F21, F26, and F29, with a total of 12 first ranks, in the first place. The third place were SaDE; for the 50-dimensional problems, the average rank of LSHADE, RUSDE, and SPS-jDE were first, second, and third, respectively. LSHADE was ranked first for F1, F9, F11, F18, F21, F26, F29, and F30, with a total of eight first ranks in the first place. RUSDE was ranked first for F16 and F19, with a total of two first ranks in the third place. SPS-jDE was ranked first for F6, F7, F11, F12, F13, and F15, with a total of six first ranks in the second place. From the comprehensive perspective of the results, the average ranking of the RUSDE algorithm was the second smallest, which indicated that RUSDE has the best comprehensive performance except for LSHADE.

Suppose the variance of all algorithms obeys normal distribution. We used a *t*-test with double tail to compare RUSDE with the other 10 algorithms. The significant level was set to 0.05. Assume h0: there is no difference between the compared algorithms and RUSDE. To determine whether the difference is statistically significant, the *t*-test calculates a t-value (the *p*-value is obtained directly from this t-value). The t-value measures the size of the difference relative to the variation in in the RUSDE and other algorithms. The greater the magnitude of T, the greater the evidence against the null hypothesis. A smaller *p*-value means that there is stronger evidence in favor of the alternative hypothesis. P and T stand for *p*-value and t-value in Tables 4 and 5. "+", "-", and " = " indicated significantly better than, worse than, and no significant difference from the other 10 algorithms, respectively. The results were shown in Tables 4 and 5. It illustrated that for the 30-dimensional problems, the number of "+" for the other 10 algorithms was greater than the number of "-", and for the SaDE and SPS-jDE, the number of "+" was slightly more than the number of "-". That indicated the performance of RUSDE was slightly better than that of SaDE and SPS-jDE, slightly worse than LSHADE, and better than the remaining eight algorithms; For the 50-dimensional problems, the number of "+" for the other 10 algorithms is greater than the number of "-", and for the SPS-jDE algorithm, the number of "+" is equal to "-". This shows that the performance of RUSDE is not significantly different from that of SPS-jDE and LSHADE, and it is better than the remaining nine algorithms.

The Friedman test was used to evaluate the effect of the algorithm [63]. The null hypothesis (H0) means there is no difference between the algorithms, and the alternative hypothesis (H1) means there is at least one algorithm that obtained different results compared with the others. There are 12 algorithms in total, which means the freedom is 11. When we set the confidence level to 0.05, the critical value is 2.4663. According to the calculations, the F-scores for the mean value ranks of 30D and 50D are 19.869 and 23.008, respectively, which are much higher than the critical value, hence demonstrating that hypothesis H1 can be accepted. The F-tests show that significant differences existed between the algorithm; therefore, we use the post hoc Duncan's test to assess the differences between RUSDE and the other 11 algorithms. Table 6 illustrates the results based on the significance (*p*-value). It shows that for the 30D problems, RUSDE performs better than the other algorithms, except for SaDE, SPS-jDE, and LSHADE, and for the 50D problems, RUSDE obtains better performance than the other algorithms, except for SaDE, rank-jDE, SPS-jDE, and LSHADE.

Table 2. The comparative results of 30-dimensional problems for CEC 2014.

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE	RUSDE
F1	Mean	1.27×10^{6}	1.20×10^{5}	3.82×10^{5}	2.54×10^{8}	1.59×10^{5}	6.87×10^{5}	1.42×10^{4}	1.00×10^{5}	8.17×10^{4}	1.77×10^{5}	2.00×10^{2}	1.13×10^{5}
	Std.	4.16×10^{5}	1.01×10^{5}	2.11×10^{5}	8.21×10^{7}	1.72×10^{5}	3.89×10^{5}	1.40×10^{4}	6.46×10^{4}	8.81×10^{4}	1.14×10^{5}	1.60×10^{-5}	6.53×10^{4}
F2	Mean	1.28×10^{4}	4.65×10^{-23}	2.85×10^{-2}	1.65×10^{10}	25.1	0.00	1.82×10^{-23}	0.00	2.52×10^{-24}	0.00	1.19×10^{-17}	0.00
	Std.	6.42×10^{3}	1.03×10^{-22}	0.115	2.49×10^{9}	1.12×10^{2}	0.00	5.62×10^{-23}	0.00	1.13×10^{-23}	0.00	1.15×10^{-17}	0.00
F3	Mean	3.83×10^{4}	8.53×10^{-30}	3.96×10^{-4}	3.73×10^{4}	1.46	4.95×10^{-25}	4.77×10^{-28}	1.87×10^{-29}	1.25×10^{-27}	9.77×10^{-28}	2.13×10^{-22}	3.08×10^{-28}
	Std.	1.08×10^{4}	2.37×10^{-29}	1.61×10^{-3}	6.46×10^{3}	3.97	6.89×10^{-25}	1.22×10^{-27}	4.28×10^{-29}	3.11×10^{-27}	1.92×10^{-27}	3.21×10^{-22}	1.32×10^{-27}
F4	Mean	65.6	7.25	92.6	1.12×10^{3}	21.0	0.566	0.343	6.04×10^{-3}	0.106	4.10	0.557	0.403
	Std.	37.2	20.7	27.4	3.08×10^{2}	37.8	0.295	0.22	1.63×10^{-2}	0.155	14.9	0.451	0.212
F5	Mean	20.0	20.2	20.9	20.9	21.0	20.5	20.6	20.5	20.6	20.4	20.2	20.4
	Std.	6.45×10^{-4}	3.45×10^{-2}	3.84×10^{-2}	5.29×10^{-2}	4.80×10^{-2}	4.42×10^{-2}	5.36×10^{-2}	4.58×10^{-2}	4.73×10^{-2}	0.116	3.83×10^{-2}	0.111
F6	Mean	20.5	7.06	20.8	34.4	0.663	0.361	0.806	1.33	0.593	0.581	10.3	0.432
	Std.	3.28	1.87	2.70	2.59	1.00	0.765	1.49	1.08	0.881	0.72	0.815	0.82
F7	Mean	2.81×10^{-2}	1.08×10^{-2}	8.91×10^{-2}	1.43×10^{2}	1.48×10^{-2}	3.70×10^{-4}	1.84×10^{-3}	2.46×10^{-3}	4.93×10^{-4}	0.00	0.00	0.00
	Std.	1.24×10^{-2}	1.34×10^{-2}	0.109	21.8	3.04×10^{-3}	1.65×10^{-3}	6.02×10^{-3}	4.52×10^{-3}	2.20×10^{-3}	0.00	0.00	0.00
F8	Mean	1.04×10^{2}	8.88×10^{-17}	86.9	2.44×10^{2}	12.8	0.00	0.00	0.00	4.84	3.41	2.11×10^{-4}	0.497
	Std.	25.6	3.97×10^{-16}	20.3	18.5	4.65	0.00	0.00	0.00	4.03	1.99	2.10×10^{-4}	0.685
F9	Mean	1.10×10^{2}	38.5	1.01×10^{2}	2.74×10	62.3	95.8	38.4	63.9	91.1	34.6	22.6	34.2
	Std.	25.9	7.21	19.0	16.4	55.6	9.84	12.4	26.4	7.52	9.75	3.73	10.9
F10	Mean	3.70×10^{3}	4.47	2.48×10^{3}	5.97×10^{3}	1.96×10^{2}	48.4	1.12	12.6	3.31×10^{2}	22.1	6.66	4.38
	Std.	8.89×10^{2}	2.08	6.03×10^{2}	4.28×10^{2}	1.63×10^{2}	14.1	1.08	7.50	64.6	36.2	1.52	3.03
F11	Mean	3.80×10^{3}	1.94×10^{3}	3.53×10^{3}	7.14×10^{3}	6.43×10^{3}	4.23×10^{3}	3.59×10^{3}	4.08×10^{3}	4.47×10^{3}	2.07×10^{3}	1.42×10^{3}	2.11×10^{3}
	Std.	6.51×10^{2}	1.91×10^{2}	8.74×10^{2}	2.30×10^{2}	1.10×10^{3}	3.41×10^{2}	3.09×10^{2}	3.12×10^{2}	3.58×10^{2}	5.87×10^{2}	1.97×10^{2}	5.84×10^{2}
F12	Mean	0.258	0.288	2.00	2.53	2.21	0.882	0.813	0.901	1.06	0.202	0.228	0.177
	Std.	0.141	4.45×10^{-2}	0.523	0.246	0.361	0.129	0.110	8.95×10^{-2}	0.143	0.110	2.73×10^{-2}	9.74×10^{-2}
F13	Mean	0.498	0.261	0.478	3.03	0.274	0.30	0.263	0.258	0.280	0.130	0.188	0.182
	Std.	9.24×10^{-2}	5.02×10^{-2}	0.113	0.266	4.34×10^{-2}	4.22×10^{-2}	4.95×10^{-2}	3.96×10^{-2}	3.06×10^{-2}	3.66×10^{-2}	2.00×10^{-2}	5.04×10^{-2}
F14	Mean	0.299	0.225	0.258	47.8	0.276	0.277	0.270	0.276	0.256	0.286	0.244	0.261
	Std.	4.74×10^{-2}	4.18×10^{-2}	5.01×10^{-2}	10.3	3.12×10^{-2}	2.87×10^{-2}	4.05×10^{-2}	3.24×10^{-2}	3.74×10^{-2}	4.09×10^{-2}	1.68×10^{-2}	3.61×10^{-2}
F15	Mean	16.6	5.83	45.1	3.90×10^{3}	14.2	9.94	7.01	9.33	9.56	3.03	2.65	3.43
	Std.	4.72	1.70	23.4	4.11×10^{3}	1.70	0.719	1.01	0.862	0.952	1.08	0.341	1.15
F16	Mean	13.0	9.54	11.1	12.8	12.1	11.1	10.8	10.8	11.41.14E+01	9.42	9.33	9.09
	Std.	0.408	0.40	0.705	0.291	0.308	0.265	0.353	0.348	0.277	0.931	0.463	1.15
F17	Mean	1.44×10^{5}	1.29×10^{4}	4.40×10^{3}	6.45×10^{6}	6.24×10^{3}	1.53×10^{4}	3.96×10^{3}	9.04×10^{3}	1.40×10^{3}	7.49×10^{3}	3.66×10^{2}	9.16×10^{3}
	Std.	7.44×10^{4}	1.27×10^{4}	2.13×10^{3}	3.17×10^{6}	4.72×10^{3}	1.52×10^{4}	4.30×10^{3}	9.00×10^{3}	6.92×10^{2}	6.43×10^{3}	91.9	8.20×10^{3}
F18	Mean	9.04×10^{2}	81.9	2.84×10^{3}	1.71×10^{8}	15.6	64.3	33.8	1.28×10^{2}	33.1	1.02×10^{3}	7.19	26.5
	Std.	7.71×10^{2}	36.1	4.34×10^{3}	9.66×10^{7}	9.62	55.4	17.3	4.29×10^{2}	18.2	3.02×10^{3}	3.20	22.9
F19	Mean	20.5	6.06	14.0	83.3	2.87	4.92	4.64	4.33	4.15	2.99	3.36	2.74
	Std.	18.6	1.35	2.01	25.1	0.708	0.617	0.740	1.02	0.964	0.834	0.556	0.549
F20	Mean	2.37×10^{4}	35.0	3.74×10^{2}	1.63×10^{4}	22.2	19.0	19.3	15.9	18.4	13.7	5.14	19.0
	Std.	6.99×10^{3}	15.4	1.19×10^{2}	5.64×10^{3}	21.0	7.74	5.56	7.28	9.51	4.02	1.67	18.1

Table 2. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE	RUSDE
F21	Mean	1.14×10^5	5.20×10^3	4.19×10^3	1.80×10^6	7.60×10^2	1.34×10^3	5.21×10^2	1.22×10^3	3.65×10^2	1.67×10^3	3.12×10^2	1.79×10^3
	Std.	4.84×10^4	8.00×10^3	3.67×10^3	9.79×10^5	9.40×10^2	8.29×10^2	3.02×10^2	1.24×10^3	1.02×10^2	2.08×10^3	69.9	1.94×10^3
F22	Mean	5.84×10^2	1.65×10^2	4.19×10^2	7.85×10^2	1.08×10^2	61.7	95.0	1.44×10^2	81.6	2.74×10^2	57.2	1.83×10^2
	Std.	1.94×10^2	87.0	2.38×10^2	1.08×10^2	1.34×10^2	58.6	77.1	1.05×10^2	62.4	1.32×10^2	47.0	1.80×10^2
F23	Mean	3.15×10^2	3.15×10^2	2.00×10^2	3.66×10^2	3.15×10^2	3.15×10^2	3.15×10^2	3.15×10^2	3.15×10^2	3.15×10^2	3.15×10^2	3.15×10^2
	Std.	4.28×10^{-3}	1.17×10^{-13}	2.26×10^{-14}	10.2	9.22×10^{-14}	5.83×10^{-14}	7.72×10^{-14}	5.83×10^{-14}	5.22×10^{-14}	8.35×10^{-14}	5.83×10^{-14}	6.12×10^{-14}
F24	Mean	2.19×10^2	2.29×10^2	2.00×10^2	2.00×10^2	2.25×10^2	2.24×10^2	2.26×10^2	2.29×10^2	2.24×10^2	2.25×10^2	2.24×10^2	2.24×10^2
	Std.	9.88	5.33	3.37×10^{-6}	4.89×10^{-2}	1.24	1.22	3.34	5.48	1.28	3.22	0.596	1.74
F25	Mean	2.06×10^2	2.07×10^2	2.00×10^2	2.29×10^2	2.03×10^2	2.03×10^2	2.04×10^2	2.04×10^2	2.05×10^2	2.04×10^2	2.03×10^2	2.04×10^2
	Std.	3.34	2.89	5.83×10^{-14}	5.38	0.437	0.421	1.07	0.999	1.08	0.715	0.186	0.524
F26	Mean	1.80×10^2	1.00×10^2	1.35×10^2	1.03×10^2	1.05×10^2	1.00×10^2	1.00×10^2	1.00×10^2	1.00×10^2	1.00×10^2	1.00×10^2	1.00×10^2
	Std.	40.8	7.18×10^{-2}	48.7	0.403	22.3	5.86×10^{-2}	2.89×10^{-2}	3.41×10^{-2}	4.05×10^{-2}	3.62×10^{-2}	2.79×10^{-2}	3.37×10^{-2}
F27	Mean	8.26×10^2	4.62×10^2	2.00×10^2	8.01×10^2	3.59×10^2	3.64×10^2	3.56×10^2	3.41×10^2	3.72×10^2	3.22×10^2	3.00×10^2	3.16×10^2
	Std.	1.37×10^2	70.6	2.26×10^{-14}	3.43×10^2	45.1	48.4	51.4	40.2	45.5	38.1	1.19×10^{-4}	34.0
F28	Mean	1.47×10^3	9.45×10^2	5.73×10^2	2.07×10^3	8.33×10^2	8.04×10^2	8.21×10^2	8.29×10^2	8.41×10^2	8.32×10^2	7.97×10^2	8.09×10^2
	Std.	4.27×10^2	1.53×10^2	6.20×10^2	3.24×10^2	36.3	21.7	24.8	28.1	49.5	30.0	11.8	63.8
F29	Mean	4.45×10^5	8.88×10^5	1.12×10^6	1.15×10^7	8.50×10^2	1.01×10^3	7.83×10^2	8.43×10^2	7.67×10^2	9.26×10^2	7.18×10^2	8.62×10^2
	Std.	1.93×10^6	2.74×10^6	2.21×10^6	6.25×10^6	2.14×10^2	1.10×10^2	1.95×10^2	1.31×10^2	94.3	3.38×10^2	4.90	1.58×10^2
F30	Mean	7.96×10^3	2.34×10^3	6.08×10^3	2.18×10^5	9.65×10^2	1.42×10^3	1.12×10^3	1.76×10^3	1.07×10^3	1.63×10^3	1.01×10^3	1.68×10^3
	Std.	2.00×10^3	9.37×10^2	5.80×10^3	8.02×10^4	6.08×10^2	6.33×10^2	3.78×10^2	9.87×10^2	5.55×10^2	6.92×10^2	2.96×10^2	7.08×10^2
Avg.	Rank	9.60	6.30	7.93	11.33	6.87	5.70	4.70	5.23	5.43	4.77	2.70	3.70

Table 3. The comparative results of 50-dimensional problems for CEC 2014.

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE	RUSDE
F1	Mean	2.79×10^6	8.45×10^5	2.27×10^6	6.60×10^8	9.38×10^5	3.02×10^6	3.37×10^5	1.01×10^6	8.04×10^5	1.57×10^6	1.01×10^5	1.10×10^6
	Std.	4.74×10^5	4.60×10^5	9.25×10^5	1.58×10^8	4.37×10^5	1.15×10^6	1.85×10^5	3.61×10^5	2.76×10^5	6.08×10^5	3.62×10^4	1.98×10^5
F2	Mean	2.32×10^4	2.07×10^{-2}	1.25×10^6	5.41×10^{10}	5.06×10^3	85.5	36.3	0.657	3.34×10^3	4.18×10^3	2.98×10^2	2.20×10^2
	Std.	1.09×10^4	4.24×10^{-2}	5.39×10^6	5.87×10^9	4.01×10^3	1.04×10^2	1.61×10^2	1.16	2.87×10^3	5.58×10^3	1.93×10^2	3.27×10^2
F3	Mean	8.26×10^4	2.67	2.10×10^2	8.83×10^4	2.90×10^3	8.06	6.78	0.684	6.28×10^2	62.7	1.93×10^3	40.6
	Std.	1.25×10^4	6.53	5.31×10^2	1.39×10^4	1.54×10^3	11.4	16.7	1.12	7.74×10^2	73.8	2.62×10^3	96.9
F4	Mean	97.1	89.0	1.88×10^2	7.86×10^3	93.4	94.9	93.0	62.8	77.0	95.3	92.7	93.9
	Std.	37.4	38.8	52.3	1.73×10^3	14.6	3.22	33.2	33.1	32.0	2.43	4.07	3.14
F5	Mean	20.0	20.4	21.1	21.2	21.1	20.8	20.8	20.8	20.9	20.5	20.5	20.6
	Std.	8.48×10^{-4}	2.93×10^{-2}	3.37×10^{-2}	2.72×10^{-2}	4.65×10^{-2}	4.21×10^{-2}	4.47×10^{-2}	3.90×10^{-2}	4.34×10^{-2}	8.75×10^{-2}	5.02×10^{-2}	0.175
F6	Mean	43.4	18.9	43.1	66.3	4.51	2.66	5.52	4.61	4.64	1.74	28.6	3.24
	Std.	3.41	2.58	3.84	3.74	1.91	1.79	2.46	2.89	2.28	1.78	1.49	1.85
F7	Mean	0.114	1.27×10^{-2}	0.402	5.16×10^2	7.13×10^{-3}	3.70×10^{-4}	3.57×10^{-3}	1.60×10^{-3}	5.18×10^{-3}	2.61×10^{-16}	2.88×10^{-11}	3.00×10^{-16}
	Std.	3.00×10^{-2}	1.22×10^{-2}	0.715	60.7	1.15×10^{-2}	1.65×10^{-3}	5.33×10^{-3}	5.07×10^{-3}	4.99×10^{-3}	1.31×10^{-16}	2.86×10^{-11}	1.91×10^{-16}

Table 3. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE	RUSDE
F8	Mean	2.25×10^{2}	1.71×10^{-10}	1.87×10^{2}	5.13×10^{2}	34.3	36.7	0.995	10.7	75.7	14.5	15.1	3.68
	Std.	31.8	6.95×10^{-10}	28.5	28.3	6.90	6.29	1.12	8.74	8.39	4.14	1.71	2.04
F9	Mean	2.47×10^{2}	1.08×10^{2}	2.17×10^{2}	5.78×10^{2}	2.14×10^{2}	2.52×10^{2}	1.55×10^{2}	1.95×10^{2}	2.23×10^{2}	71.3	60.2	63.0
	Std.	45.2	21.8	27.4	34.4	1.38×10^{2}	17.2	29.5	65.4	8.79	20.6	8.50	14.3
F10	Mean	6.70×10^{3}	18.4	5.61×10^{3}	1.25×10^{4}	1.43×10^{3}	1.18×10^{3}	63.5	7.46×10^{2}	2.92×10^{3}	1.88×10^{2}	3.39×10^{2}	34.2
	Std.	1.03×10^{3}	6.49	1.13×10^{3}	7.71×10^{2}	1.50×10^{3}	2.48×10^{2}	31.5	1.54×10^{2}	3.13×10^{2}	1.54×10^{2}	70.3	43.5
F11	Mean	6.86×10^{3}	4.83×10^{3}	6.84×10^{3}	1.35×10^{4}	1.32×10^{4}	9.82×10^{3}	8.60×10^{3}	9.64×10^{3}	9.84×10^{3}	4.36×10^{3}	4.86×10^{3}	5.11×10^{3}
	Std.	8.68×10^{2}	4.63×10^{2}	1.33×10^{3}	3.92×10^{2}	2.70×10^{2}	2.66×10^{2}	5.35×10^{2}	4.57×10^{2}	4.71×10^{2}	1.02×10^{3}	4.55×10^{2}	9.90×10^{2}
F12	Mean	0.367	0.404	2.91	3.48	3.44	1.35	1.30	1.35	1.56	0.119	0.417	0.290
	Std.	0.281	4.10×10^{-2}	0.234	0.377	0.289	0.122	0.129	0.140	0.162	4.73×10^{-2}	4.64×10^{-2}	0.227
F13	Mean	0.667	0.476	0.603	4.69	0.438	0.432	0.391	0.39	0.411	0.233	0.273	0.335
	Std.	7.69×10^{-2}	8.10×10^{-2}	9.53×10^{-2}	0.294	5.40×10^{-2}	4.91×10^{-2}	4.48×10^{-2}	4.27×10^{-2}	4.74×10^{-2}	4.98×10^{-2}	1.52×10^{-2}	6.77×10^{-2}
F14	Mean	0.355	0.284	0.366	1.34×10^{2}	0.385	0.376	0.318	0.434	0.311	0.339	0.313	0.340
	Std.	4.14×10^{-2}	3.69×10^{-2}	0.136	18.5	0.128	0.163	4.10×10^{-2}	0.193	2.69×10^{-2}	9.14×10^{-2}	1.93×10^{-2}	9.99×10^{-2}
F15	Mean	43.2	21.2	1.03×10^{3}	1.57×10^{5}	30.1	24.1	21.7	22.6	23.7	6.81	8.42	7.91
	Std.	7.37	4.32	9.18×10^{2}	8.10×10^{4}	1.61	1.45	2.36	1.64	1.89	1.47	1.10	2.58
F16	Mean	22.1	18.7	20.7	22.7	22.1	20.8	20.6	20.6	21.0	18.6	18.7	18.1
	Std.	0.602	0.364	0.620	0.166	0.203	0.280	0.365	0.321	0.238	0.974	0.316	1.07
F17	Mean	2.84×10^{5}	6.92×10^{4}	2.16×10^{5}	4.99×10^{7}	5.37×10^{4}	1.34×10^{5}	4.38×10^{4}	6.30×10^{4}	3.60×10^{4}	8.69×10^{4}	1.76×10^{3}	5.40×10^{4}
	Std.	9.00×10^{4}	3.01×10^{4}	2.22×10^{5}	1.73×10^{7}	2.18×10^{4}	1.01×10^{5}	3.04×10^{4}	3.04×10^{4}	2.14×10^{4}	7.71×10^{4}	4.65×10^{2}	3.18×10^{4}
F18	Mean	2.73×10^{3}	7.18×10^{2}	2.29×10^{3}	1.52×10^{9}	7.06×10^{2}	3.88×10^{2}	5.02×10^{2}	7.03×10^{2}	3.52×10^{2}	4.96×10^{2}	92.0	2.13×10^{2}
	Std.	1.37×10^{3}	6.70×10^{2}	1.36×10^{3}	3.63×10^{8}	8.39×10^{2}	4.34×10^{2}	5.97×10^{2}	5.95×10^{2}	3.13×10^{2}	4.91×10^{2}	17.5	2.74×10^{2}
F19	Mean	50.1	30.6	57.9	2.88×10^{2}	12.8	17.9	26.7	20.4	14.7	9.74	11.6	8.24
	Std.	33.3	28.2	26.7	29.9	4.59	9.74	10.6	10.7	6.99	2.60	0.575	1.67
F20	Mean	2.53×10^{4}	3.62×10^{2}	9.11×10^{2}	3.61×10^{4}	7.68×10^{2}	2.00×10^{2}	1.62×10^{2}	1.48×10^{2}	1.89×10^{2}	5.06×10^{2}	5.72×10^{3}	3.08×10^{2}
	Std.	7.84×10^{3}	2.06×10^{2}	3.31×10^{2}	1.13×10^{4}	5.41×10^{2}	76.7	1.16×10^{2}	77.6	78.6	4.50×10^{2}	5.69×10^{3}	2.01×10^{2}
F21	Mean	2.07×10^{5}	7.84×10^{4}	1.75×10^{4}	1.09×10^{7}	3.31×10^{4}	6.24×10^{4}	1.75×10^{4}	4.73×10^{4}	1.27×10^{4}	5.96×10^{4}	1.02×10^{3}	4.12×10^{4}
	Std.	8.20×10^{4}	5.12×10^{4}	8.73×10^{3}	5.07×10^{6}	1.72×10^{4}	4.73×10^{4}	1.64×10^{4}	4.00×10^{4}	8.03×10^{3}	2.61×10^{4}	2.20×10^{2}	3.31×10^{4}
F22	Mean	1.67×10^{3}	6.16×10^{2}	1.20×10^{3}	2.43×10^{3}	8.02×10^{2}	6.17×10^{2}	4.28×10^{2}	3.76×10^{2}	6.24×10^{2}	7.73×10^{2}	4.41×10^{2}	6.79×10^{2}
	Std.	3.18×10^{2}	1.97×10^{2}	3.44×10^{2}	3.03×10^{2}	4.61×10^{2}	2.71×10^{2}	1.87×10^{2}	2.47×10^{2}	1.92×10^{2}	2.74×10^{2}	1.15×10^{2}	2.45×10^{2}
F23	Mean	3.44×10^{2}	3.44×10^{2}	2.00×10^{2}	7.00×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}	3.44×10^{2}
	Std.	8.12×10^{-2}	1.39×10^{-13}	9.22×10^{-15}	65.6	1.26×10^{-13}	1.78×10^{-13}	1.58×10^{-13}	2.61×10^{-14}	1.66×10^{-13}	1.14×10^{-13}	1.08×10^{-12}	4.52×10^{-14}
F24	Mean	2.74×10^{2}	2.81×10^{2}	2.00×10^{2}	2.41×10^{2}	2.76×10^{2}	2.69×10^{2}	2.73×10^{2}	2.72×10^{2}	2.72×10^{2}	2.70×10^{2}	2.75×10^{2}	2.71×10^{2}
	Std.	11.5	3.32	1.52×10^{-7}	60.7	2.47	2.90	2.21	2.60	2.26	2.68	0.917	1.95
F25	Mean	2.00×10^{2}	2.28×10^{2}	2.00×10^{2}	2.69×10^{2}	2.08×10^{2}	2.08×10^{2}	2.16×10^{2}	2.08×10^{2}	2.10×10^{2}	2.09×10^{2}	2.05×10^{2}	2.09×10^{2}
	Std.	1.63×10^{-3}	4.93	4.88×10^{-14}	27.1	1.22	1.47	7.60	2.41	8.58	1.40	0.301	1.97
F26	Mean	1.90×10^{2}	1.50×10^{2}	1.60×10^{2}	1.05×10^{2}	1.55×10^{2}	1.05×10^{2}	1.10×10^{2}	1.30×10^{2}	1.45×10^{2}	1.05×10^{2}	1.00×10^{2}	1.30×10^{2}
	Std.	30.6	51.1	50.0	0.495	50.9	22.3	30.7	46.9	50.9	22.3	2.62×10^{-2}	46.9
F27	Mean	1.48×10^{3}	9.03×10^{2}	2.00×10^{2}	2.08×10^{3}	4.92×10^{2}	3.91×10^{2}	4.56×10^{2}	4.91×10^{2}	4.79×10^{2}	3.81×10^{2}	3.61×10^{2}	4.28×10^{2}
	Std.	1.13×10^{2}	82.1	2.26×10^{-14}	1.89×10^{2}	58.4	37.9	50.6	72.7	60.1	57.9	1.32×10^{2}	58.2

Table 3. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE	RUSDE
F28	Mean	2.82×10^3	1.73×10^3	2.00×10^2	5.81×10^3	1.18×10^3	1.10×10^3	1.19×10^3	1.13×10^3	1.20×10^3	1.11×10^3	1.13×10^3	1.11×10^3
	Std.	7.93×10^2	4.99×10^2	2.53×10^{-14}	8.73×10^2	1.09×10^2	41.3	61.4	65.7	86.1	30.3	35.1	42.9
F29	Mean	5.53×10^6	1.11×10^7	1.78×10^7	2.01×10^8	1.38×10^3	1.70×10^3	1.18×10^3	1.48×10^3	1.28×10^3	1.48×10^3	9.34×10^2	1.76×10^6
	Std.	2.46×10^7	2.00×10^7	2.37×10^7	4.66×10^7	1.95×10^2	5.05×10^2	2.16×10^2	3.85×10^2	2.32×10^2	2.62×10^2	67.0	7.88×10^6
F30	Mean	3.11×10^4	1.06×10^4	2.18×10^4	1.90×10^6	9.38×10^3	8.70×10^3	9.16×10^3	9.18×10^3	9.87×10^3	9.03×10^3	8.68×10^3	8.93×10^3
	Std.	7.92×10^3	2.09×10^3	1.39×10^4	7.91×10^5	6.13×10^2	3.88×10^2	4.45×10^2	8.10×10^2	5.65×10^2	4.57×10^2	3.77×10^2	4.10×10^2
Avg.	Rank	9.10	5.87	7.93	11.33	7.50	5.87	4.80	5.13	6.17	4.23	3.67	4.13

Table 4. Comparison between RUSDE and other algorithms using a *t*-test with double tail for 30-dimensional problems.

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
F1	T	12.3	0.245	5.43	13.8	1.1	6.51	−6.63	−0.635	−1.28	2.16	−7.73
	P	7.54×10^{-15}	0.807	3.44×10^{-6}	2.00×10^{-16}	0.269	1.14×10^{-7}	7.92×10^{-8}	0.529	0.207	3.68×10^{-2}	2.56×10^{-9}
		+	=	+	+	=	+	-	=	=	+	-
F2	T	8.94	2.01	1.11	29.5	1.00	0.00	1.45	0.00	1.00	0.00	4.64
	P	6.96×10^{-11}	5.15×10^{-2}	0.275	8.33×10^{-28}	0.324	0.00	0.155	0.00	0.324	0.00	4.06×10^{-5}
		+	=	=	+	=	=	=	=	=	=	+
F3	T	15.8	−1.01	1.10	25.8	1.64	3.21	0.420	−0.980	1.25	1.28	2.97
	P	2.63×10^{-18}	0.317	0.279	1.04×10^{-25}	0.110	2.73×10^{-3}	0.677	0.333	0.220	0.207	5.14×10^{-3}
		+	=	=	+	=	+	=	=	=	=	+
F4	T	7.85	1.48	15.0	16.2	2.44	2.02	−0.866	−8.32	−5.05	1.11	1.39
	P	1.79×10^{-9}	0.148	1.39×10^{-17}	1.26×10^{-18}	1.97×10^{-2}	5.09×10^{-2}	0.392	4.32×10^{-10}	1.15×10^{-5}	0.273	0.174
		+	=	+	+	+	=	=	-	-	=	=
F5	T	−16.8	−6.60	19.8	19.1	20.0	4.34	4.96	4.97	7.71	−1.10	−8.07
	P	3.53×10^{-19}	8.72×10^{-8}	1.18×10^{-21}	4.65×10^{-21}	9.21×10^{-22}	1.00×10^{-4}	1.49×10^{-5}	1.47×10^{-5}	2.76×10^{-9}	0.278	9.28×10^{-10}
		-	-	+	+	+	+	+	+	+	=	-
F6	T	26.5	14.5	32.3	55.9	0.797	−0.285	0.980	2.96	0.596	0.608	38.3
	P	4.11×10^{-26}	4.63×10^{-17}	3.14×10^{-29}	4.22×10^{-38}	0.431	0.777	0.333	5.24×10^{-3}	0.555	0.547	5.50×10^{-32}
		+	+	+	+	=	=	=	+	=	=	+
F7	T	10.2	3.62	3.66	29.2	2.18	1.00	1.37	2.44	1.00	0.00	0.00
	P	2.23×10^{-12}	8.49×10^{-4}	7.70×10^{-4}	1.18×10^{-27}	3.56×10^{-2}	0.324	0.178	1.96×10^{-2}	0.324	0.00	0.00
		+	+	+	+	+	=	=	+	=	=	=
F8	T	18.1	−3.25	19.1	58.8	11.7	−3.25	−3.25	−3.25	4.75	6.18	−3.25
	P	3.05×10^{-20}	2.43×10^{-3}	4.87×10^{-21}	6.17×10^{-39}	3.64×10^{-14}	2.43×10^{-3}	2.43×10^{-3}	2.43×10^{-3}	2.91×10^{-5}	3.23×10^{-7}	2.44×10^{-3}
		+	-	+	+	+	-	-	-	+	+	-
F9	T	12.0	1.46	13.6	54.4	2.21	18.8	1.13	4.64	19.2	0.123	−4.55
	P	1.72×10^{-14}	0.154	3.84×10^{-16}	1.16×10^{-37}	3.29×10^{-2}	7.84×10^{-21}	0.265	4.04×10^{-5}	3.52×10^{-21}	0.903	5.40×10^{-5}
		+	=	+	+	+	+	=	+	+	=	-
F10	T	18.5	0.12	0.183	0.623	5.26	0.136	−4.53	4.57	0.226	2.18	3.01

Table 4. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
	P	1.25×10^{-20}	0.905	2.04×10^{-20}	7.19×10^{-40}	5.83×10^{-6}	3.30×10^{-16}	5.75×10^{-5}	5.07×10^{-5}	1.26×10^{-23}	3.53×10^{-2}	4.60×10^{-3}
		+	=	+	+	+	+	-	+	+	+	+
F11	T	8.58	−1.22	6.03	0.359	0.155	0.140	9.94	0.132	0.153	−0.247	−4.97
	P	2.03×10^{-10}	0.229	5.17×10^{-7}	6.47×10^{-31}	$5.38 \times 10^{-18}8$	1.30×10^{-16}	3.99×10^{-12}	8.60×10^{-16}	7.36×10^{-18}	0.806	1.45×10^{-5}
		+	=	+	+	+	+	+	+	+	=	-
F12	T	2.09	4.64	15.3	39.8	24.3	19.5	19.3	24.4	22.9	0.740	2.23
	P	4.30×10^{-2}	4.11×10^{-5}	7.81×10^{-18}	1.37×10^{-32}	9.77×10^{-25}	2.25×10^{-21}	3.01×10^{-21}	7.60×10^{-25}	7.80×10^{-24}	0.464	3.17×10^{-2}
		+	+	+	+	+	+	+	+	+	=	+
F13	T	13.4	4.96	10.7	47.1	6.12	8.00	5.11	5.25	7.42	−3.79	0.450
	P	5.72×10^{-16}	1.53×10^{-5}	5.21×10^{-13}	2.63×10^{-35}	3.85×10^{-7}	1.15×10^{-9}	9.31×10^{-6}	6.12×10^{-6}	6.61×10^{-9}	5.22×10^{-4}	0.655
		+	+	+	+	+	+	+	+	+	-	=
F14	T	2.83	−2.94	−0.239	20.7	1.35	1.55	0.729	1.37	−0.497	2.00	−1.97
	P	7.33×10^{-3}	5.55×10^{-3}	0.812	2.81×10^{-22}	0.186	0.130	0.470	0.179	0.622	5.31×10^{-2}	5.56×10^{-2}
		+	-	=	+	=	=	=	=	=	=	=
F15	T	12.1	5.23	7.95	4.23	23.4	21.5	10.4	18.4	18.4	−1.15	−2.91
	P	1.21×10^{-14}	6.46×10^{-6}	1.32×10^{-9}	1.41×10^{-4}	3.45×10^{-24}	7.21×10^{-23}	9.96×10^{-13}	1.64×10^{-20}	1.60×10^{-20}	0.256	5.94×10^{-3}
		+	+	+	+	+	+	+	+	+	=	-
F16	T	14.2	1.66	6.69	14.1	11.3	7.64	6.35	6.19	8.79	0.991	0.866
	P	7.96×10^{-17}	0.105	6.50×10^{-8}	1.16×10^{-16}	9.38×10^{-14}	3.42×10^{-9}	1.91×10^{-7}	3.09×10^{-7}	1.09×10^{-10}	0.328	0.392
		+	=	+	+	+	+	+	+	+	=	=
F17	T	8.09	1.11	−2.51	9.09	−1.38	1.60	−2.51	-3.87×10^{-2}	−4.22	−0.715	−4.79
	P	8.62×10^{-10}	0.273	1.66×10^{-2}	4.47×10^{-11}	0.177	0.117	1.63×10^{-2}	0.969	1.47×10^{-4}	0.479	2.53×10^{-5}
		+	=	-	+	=	=	-	=	-	=	-
F18	T	5.09	5.80	2.90	7.91	−1.96	2.82	1.14	1.06	1.01	1.47	−3.74
	P	1.0×10^{-5}	1.07×10^{-6}	6.17×10^{-3}	1.52×10^{-9}	5.68×10^{-2}	7.55×10^{-3}	0.262	0.298	0.318	0.150	6.08×10^{-4}
		+	+	+	+	=	+	=	=	=	=	-
F19	T	4.26	10.2	24.3	14.4	0.636	11.8	9.18	6.14	5.65	1.08	3.55
	P	1.30×10^{-4}	1.94×10^{-12}	9.98×10^{-25}	6.06×10^{-17}	0.529	2.93×10^{-14}	3.45×10^{-11}	3.70×10^{-7}	1.71×10^{-6}	0.287	1.05×10^{-3}
		+	+	+	+	=	+	+	+	+	=	+
F20	T	15.2	2.99	13.2	12.9	0.503	-1.17×10^{-2}	6.85×10^{-2}	−0.716	−0.133	−1.29	−3.41
	P	1.02×10^{-17}	4.85×10^{-3}	8.78×10^{-16}	1.69×10^{-15}	0.618	0.991	0.946	0.478	0.895	0.204	1.54×10^{-3}
		+	+	+	+	=	=	=	=	=	=	-
F21	T	10.4	1.85	2.59	8.20	−2.10	−0.928	−2.85	−1.09	−3.24	−0.182	−3.37
	P	1.20×10^{-12}	7.15×10^{-2}	1.34×10^{-2}	6.21×10^{-10}	4.21×10^{-2}	0.359	6.97×10^{-3}	0.284	2.47×10^{-3}	0.856	1.76×10^{-3}
		+	=	+	+	-	=	-	=	-	=	-
F22	T	6.78	−0.398	3.54	12.8	−1.49	−2.86	−2.00	−0.832	−2.37	1.82	−3.01
	P	4.92×10^{-8}	0.693	1.08×10^{-3}	2.16×10^{-15}	0.144	6.91×10^{-3}	5.25×10^{-2}	0.411	2.29×10^{-2}	7.68×10^{-2}	4.57×10^{-3}
		+	=	+	+	=	-	=	=	-	=	-
F23	T	0.00	1.93	-7.90×10^{15}	22.4	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	P	0.00	6.11 × 10−2	0.00	1.74×10^{-23}	0.00	1.00	1.00	1.00	1.00	1.00	1.00
		=	=	-	+	=	=	=	=	=	=	=

Table 4. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
F24	T	−2.36	3.37	−62.7	−62.4	1.55	−0.119	1.54	3.37	−0.545	0.834	−1.04
	P	2.36×10^{-2}	1.76×10^{-3}	5.79×10^{-40}	6.82×10^{-40}	0.128	0.906	0.132	1.75×10^{-3}	0.589	0.410	0.304
		-	+	-	-	=	=	=	+	=	=	=
F25	T	3.46	5.11	−30.5	21.3	−1.89	−2.37	1.42	0.459	3.52	0.449	−7.44
	P	1.36×10^{-3}	9.49×10^{-6}	2.64×10^{-28}	9.90×10^{-23}	6.66×10^{-3}	2.29×10^{-2}	0.165	0.649	1.15×10^{-3}	0.656	6.37×10^{-9}
		+	+	-	+	=	-	=	=	+	=	-
F26	T	8.78	0.00	3.22	28.0	1.02	0.00	0.00	0.00	0.00	−0.603	1.56
	P	1.12×10^{-10}	0.00	2.59×10^{-3}	5.59×10^{-27}	0.313	0.00	0.00	0.00	0.00	0.550	0.128
		+	=	+	+	=	=	=	=	=	=	=
F27	T	16.1	8.35	−15.2	6.30	3.41	3.64	2.88	2.10	4.44	0.537	−2.08
	P	1.36×10^{-18}	3.96×10^{-10}	9.33×10^{-18}	2.23×10^{-7}	1.54×10^{-3}	8.14×10^{-4}	6.44×10^{-3}	4.24×10^{-2}	7.58×10^{-5}	0.594	4.42×10^{-2}
		+	+	-	+	+	+	+	+	+	=	-
F28	T	6.70	3.66	−1.70	16.9	1.45	−0.328	76.5	1.28	1.77	1.43	−0.831
	P	6.20×10^{-8}	7.55×10^{-4}	9.79×10^{-2}	2.87×10^{-19}	0.154	0.745	0.449	0.209	8.40×10^{-2}	0.160	0.411
		+	+	=	+	=	=	=	=	=	=	=
F29	T	1.03	1.45	2.27	8.23	−0.186	3.47	−1.40	−0.410	−2.29	0.769	−4.05
	P	0.311	0.155	2.92×10^{-2}	5.64×10^{-10}	0.853	1.31×10^{-3}	0.170	0.684	2.76×10^{-2}	0.447	2.46×10^{-4}
		=	=	+	+	=	+	=	=	-	=	-
F30	T	13.3	2.55	3.38	12.1	−3.37	−1.16	−3.05	0.312	−3.00	−18.2	−3.86
	P	7.91×10^{-16}	1.50×10^{-2}	1.70×10^{-3}	1.44×10^{-14}	1.72×10^{-3}	0.255	4.12×10^{-3}	0.757	4.71×10^{-3}	0.857	4.21×10^{-4}
		+	+	+	+	-	=	-	=	-	=	-
+		26	13	21	29	12	14	8	13	12	3	6
=		2	14	4	0	16	13	16	15	12	26	9
-		2	3	5	1	2	3	6	2	6	1	15

Table 5. Comparison between RUSDE and other algorithms using a *t*-test with double tail for 50-dimensional problems.

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
F1	T	14.7	−2.26	5.53	18.6	−1.49	7.35	−12.6	−0.979	−3.87	3.31	−0.221
	P	2.77×10^{-17}	2.98×10^{-2}	2.52×10^{-6}	1.07×10^{-20}	0.145	8.34×10^{-9}	4.15×10^{-15}	0.334	4.15×10^{-4}	2.03×10^{-3}	2.54×10^{-23}
		+	-	+	+	=	+	-	=	−	+	-
F2	T	9.42	−3.00	1.04	41.2	5.37	−1.75	−2.25	−2.99	4.86	3.16	0.928
	P	1.75×10^{-11}	4.72×10^{-3}	0.305	3.83×10^{-33}	4.16×10^{-6}	8.86×10^{-2}	3.05×10^{-2}	4.83×10^{-3}	2.07×10^{-5}	3.05×10^{-3}	0.359
		+	-	=	+	+	=	-	-	+	+	=
F3	T	29.6	−1.75	1.41	28.3	8.32	−1.49	−1.54	−1.84	3.37	0.810	3.23
	P	7.13×10^{-28}	8.86×10^{-2}	0.167	3.71×10^{-27}	4.31×10^{-10}	0.144	0.132	7.31×10^{-2}	1.74×10^{-3}	0.423	2.54×10^{-3}
		+	=	=	+	+	=	=	=	+	=	+
F4	T	0.389	−0.561	8.05	20.1	−0.145	0.974	−0.124	−4.18	−2.35	1.56	−1.07
	P	0.70	0.578	9.78×10^{-10}	7.55×10^{-22}	0.885	0.336	0.902	1.64×10^{-4}	2.42×10^{-2}	0.127	0.292
		=	=	+	+	=	=	=	-	-	=	=

Table 5. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
F5	T	−14.5	−3.77	14.4	14.8	14.3	5.77	5.57	5.23	7.23	−1.14	−2.07
	P	4.93×10^{-17}	5.63×10^{-4}	6.15×10^{-17}	2.54×10^{-17}	7.63×10^{-17}	1.17×10^{-6}	2.25×10^{-6}	6.42×10^{-6}	1.19×10^{-8}	0.261	4.53×10^{-2}
		-	-	+	+	+	+	+	+	+	=	-
F6	T	46.2	22.0	41.8	67.6	2.13	−1.00	3.31	1.79	2.14	−2.61	47.8
	P	5.15×10^{-35}	3.11×10^{-23}	2.25×10^{-33}	3.28×10^{-41}	3.96×10^{-2}	0.322	2.04×10^{-3}	8.19×10^{-2}	3.90×10^{-2}	1.29×10^{-2}	1.48×10^{-35}
		+	+	+	+	+	=	+	=	+	-	+
F7	T	17.0	4.66	2.52	38.0	2.78	1.00	3.00	1.41	4.64	−0.750	4.51
	P	2.20×10^{-19}	3.87×10^{-5}	1.62×10^{-2}	7.65×10^{-32}	8.46×10^{-3}	0.324	4.77×10^{-3}	0.166	4.04×10^{-5}	0.458	6.08×10^{-5}
		+	+	+	+	+	=	+	=	+	=	+
F8	T	31.1	−8.05	28.7	80.2	19.0	22.3	−5.16	3.48	37.3	10.5	19.1
	P	1.28×10^{-28}	9.68×10^{-10}	2.35×10^{-27}	5.23×10^{-44}	5.34×10^{-21}	1.93×10^{-23}	8.14×10^{-6}	1.28×10^{-3}	1.54×10^{-31}	9.27×10^{-13}	4.47×10^{-21}
		+	-	+	+	+	+	-	+	+	+	+
F9	T	17.3	7.70	22.3	61.8	4.88	37.8	12.6	8.81	42.6	1.49	−0.729
	P	1.23×10^{-19}	2.83×10^{-9}	1.98×10^{-23}	9.61×10^{-40}	1.95×10^{-5}	9.16×10^{-32}	3.70×10^{-15}	1.03×10^{-10}	1.05×10^{-33}	0.144	0.471
		+	+	+	+	+	+	+	+	+	=	=
F10	T	29.0	−1.60	22.1	72.0	4.17	20.1	2.44	19.9	40.7	4.30	16.5
	P	1.68×10^{-27}	0.118	2.61×10^{-23}	3.07×10^{-42}	1.72×10^{-4}	7.02×10^{-22}	1.94×10^{-2}	1.10×10^{-21}	5.79×10^{-33}	1.13×10^{-4}	6.19×10^{-19}
		+	=	+	+	+	+	+	+	+	+	+
F11	T	5.89	−1.15	4.65	35.3	35.0	20.5	13.9	18.6	19.3	−2.39	−1.07
	P	8.13×10^{-7}	0.258	3.92×10^{-5}	1.17×10^{-30}	1.54×10^{-30}	3.70×10^{-22}	1.88×10^{-16}	1.16×10^{-20}	3.30×10^{-21}	2.20×10^{-2}	0.290
		+	=	+	+	+	+	+	+	+	-	=
F12	T	0.954	2.21	36.0	32.4	38.3	18.3	17.3	17.8	20.4	−3.29	2.45
	P	0.346	3.35×10^{-2}	5.89×10^{-31}	2.64×10^{-29}	5.61×10^{-32}	1.78×10^{-20}	1.30×10^{-19}	5.19×10^{-20}	4.44×10^{-22}	2.15×10^{-3}	1.90×10^{-2}
		=	+	+	+	+	+	+	+	+	-	+
F13	T	14.5	5.97	10.3	64.6	5.30	5.18	3.10	3.03	4.13	−5.45	−4.01
	P	4.53×10^{-17}	6.30×10^{-7}	1.65×10^{-12}	1.83×10^{-40}	5.16×10^{-6}	7.63×10^{-6}	3.60×10^{-3}	4.33×10^{-3}	1.94×10^{-4}	3.23×10^{-6}	2.77×10^{-4}
		+	+	+	+	+	+	+	+	+	-	-
F14	T	0.610	−2.35	0.671	32.4	1.22	0.832	−0.929	1.93	−1.29	-3.07×10^{-2}	−1.21
	P	0.546	2.42×10^{-2}	0.506	2.85×10^{-29}	0.229	0.411	0.359	6.16×10^{-2}	0.206	0.976	0.234
		=	-	=	+	=	=	=	=	=	=	=
F15	T	20.2	11.8	4.99	8.67	32.6	24.5	17.7	21.5	22.1	−1.65	0.810
	P	6.32×10^{-22}	2.66×10^{-14}	1.36×10^{-5}	1.53×10^{-10}	2.25×10^{-29}	7.44×10^{-25}	6.49×10^{-20}	6.73×10^{-23}	2.88×10^{-23}	0.106	0.423
		+	+	+	+	+	+	+	+	+	=	=
F16	T	14.4	2.55	9.35	18.9	16.5	11.0	9.95	10.1	11.9	1.52	2.37
	P	5.24×10^{-17}	1.51×10^{-2}	2.13×10^{-11}	6.97×10^{-21}	6.51×10^{-19}	2.19×10^{-13}	3.89×10^{-12}	2.41×10^{-12}	2.41×10^{-14}	0.136	2.29×10^{-2}
		+	+	+	+	+	+	+	+	+	=	+
F17	T	10.8	1.56	3.24	12.9	-3.11×10^{-2}	3.40	−1.03	0.917	−2.10	1.77	−7.34
	P	3.88×10^{-13}	0.128	2.49×10^{-3}	2.03×10^{-15}	0.975	1.60×10^{-3}	0.308	0.365	4.29×10^{-2}	8.49×10^{-2}	8.65×10^{-9}
		+	=	+	+	=	+	=	=	-	=	-
F18	T	8.12	3.12	6.72	18.8	2.50	1.52	1.97	3.35	1.50	2.25	−1.97
	P	7.91×10^{-10}	3.46×10^{-3}	5.91×10^{-8}	8.29×10^{-21}	1.70×10^{-2}	0.137	5.65×10^{-2}	1.85×10^{-3}	0.143	3.06×10^{-2}	5.57×10^{-2}
		+	+	+	+	+	=	=	+	=	+	=

Table 5. *Cont.*

Func.	A1.	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
F19	T	5.61	3.53	8.29	41.9	4.20	4.39	7.65	5.03	4.03	2.18	8.53
	P	1.92×10^{-6}	1.09×10^{-3}	4.70×10^{-10}	2.02×10^{-33}	1.56×10^{-4}	8.61×10^{-5}	3.31×10^{-9}	1.22×10^{-5}	2.58×10^{-4}	3.58×10^{-2}	2.34×10^{-10}
		+	+	+	+	+	+	+	+	+	+	+
F20	T	14.2	0.849	6.96	14.2	3.56	−2.24	−2.80	−3.31	−2.45	1.80	4.26
	P	7.98×10^{-17}	0.401	2.77×10^{-8}	9.79×10^{-17}	1.01×10^{-3}	3.12×10^{-2}	8.05×10^{-3}	2.04×10^{-3}	1.90×10^{-2}	7.94×10^{-2}	1.30×10^{-4}
		+	=	+	+	+	-	-	-	-	=	+
F21	T	8.40	2.73	−3.11	9.61	−0.975	1.64	−2.88	0.521	−3.75	1.95	−5.44
	P	3.46×10^{-10}	9.63×10^{-3}	3.56×10^{-3}	1.01×10^{-11}	0.336	0.109	6.53×10^{-3}	0.606	5.93×10^{-4}	5.84×10^{-2}	3.32×10^{-6}
		+	+	-	+	=	=	-	=	-	=	-
F22	T	10.9	−0.900	5.49	20.2	1.05	−0.763	−3.65	−3.90	−0.802	1.14	−3.94
	P	2.72×10^{-13}	0.374	2.81×10^{-6}	6.94×10^{-22}	0.301	0.450	7.89×10^{-4}	3.77×10^{-4}	0.428	0.260	3.37×10^{-4}
		+	=	+	+	=	=	-	-	=	=	-
F23	T	0.00	0.00	-1.40×10^{16}	24.31	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	P	0.00	0.00	0.00	9.69×10^{-25}	0.00	0.00	0.00	1.00	0.00	0.00	0.00
		=	=	-	+	=	=	=	=	=	=	=
F24	T	0.803	10.8	-1.64×10^{2}	−2.27	5.94	−3.21	2.33	0.486	0.262	−1.37	6.90
	P	0.427	4.04×10^{-13}	8.54×10^{-56}	2.92×10^{-2}	6.90×10^{-7}	2.71×10^{-3}	2.53×10^{-2}	0.629	0.795	0.178	3.34×10^{-8}
		=	+	-	-	+	-	+	=	=	=	+
F25	T	−20.1	16.5	−20.1	9.84	−2.61	−0.684	4.31	−1.41	0.730	0.515	−7.60
	P	8.02×10^{-22}	6.24×10^{-19}	7.76×10^{-22}	5.35×10^{-12}	1.29×10^{-2}	0.498	1.10×10^{-4}	0.166	0.470	0.610	3.85×10^{-9}
		-	+	-	+	-	=	+	=	=	=	-
F26	T	4.78	1.29	1.95	−2.38	1.61	−2.14	−1.59	4.39×10^{-3}	0.967	−2.15	−2.86
	P	2.60×10^{-5}	0.206	5.87×10^{-2}	2.23×10^{-2}	0.115	3.90×10^{-2}	0.120	0.997	0.340	3.77×10^{-2}	6.79×10^{-3}
		+	=	=	-	=	-	=	=	=	-	-
F27	T	36.6	21.1	−17.5	37.1	3.43	−2.41	1.60	2.98	2.72	−2.61	−2.10
	P	3.03×10^{-31}	1.42×10^{-22}	8.18×10^{-20}	1.87×10^{-31}	1.47×10^{-3}	2.10×10^{-2}	0.118	4.97×10^{-3}	9.89×10^{-3}	1.30×10^{-2}	4.27×10^{-2}
		+	+	-	+	+	-	=	+	+	-	-
F28	T	9.59	5.45	−95.3	24.1	2.06	−1.12	3.97	1.13	3.81	−0.433	0.959
	P	1.09×10^{-11}	3.24×10^{-6}	7.57×10^{-47}	1.34×10^{-24}	4.60×10^{-2}	0.270	3.05×10^{-4}	0.266	4.95×10^{-4}	0.667	0.344
		+	+	-	+	+	=	+	=	+	=	=
F29	T	0.652	1.94	2.87	18.9	−1.00	−1.00	−1.00	−1.00	−1.00	−1.00	−1.00
	P	0.518	6.04×10^{-2}	6.68×10^{-3}	6.74×10^{-21}	0.324	0.324	0.324	0.324	0.324	0.324	0.324
		=	=	+	+	=	=	=	=	=	=	=
F30	T	12.5	3.53	4.13	10.7	2.65	−1.92	1.60	1.16	5.98	0.742	−2.05
	P	5.14×10^{-15}	1.11×10^{-3}	1.93×10^{-4}	4.91×10^{-13}	1.16×10^{-2}	6.28×10^{-2}	0.118	0.252	6.06×10^{-7}	0.463	4.71×10^{-2}
		+	+	+	+	+	=	=	=	+	=	-
+		22	15	20	28	20	12	14	12	17	6	10
=		6	10	4	0	9	14	10	14	8	18	10
-		2	5	6	2	1	4	6	4	5	6	10

Table 6. The post hoc Duncan's test results for CEC 2014.

	KH	LBSA	DGSTLBO	SCA	DErand	jDE	SaDE	rank-jDE	jDE-EIG	SPS-jDE	LSHADE
30D	0.000	0.047	0.000	0.000	0.002	0.234	0.987	0.594	0.474	0.996	0.991
50D	0.000	0.762	0.001	0.000	0.000	0.369	1.000	0.988	0.116	1.000	1.000

4.4. Convergence Performance of RUSDE

Figure 1 shows the convergence curve of the RUSDE and 11 other algorithms in CEC 2014. It illustrated that for the unimodal function F2, RUSDE always had a faster convergence speed; for the multi-modal function F7, RUSDE had a faster convergence speed as well as jumping out of the local optimum; for the multi-modal functions F12, RUSDE converged at a normal speed in the early stage, but in the late stage, it could jump out of the local optimum and obtain a better solution; for composite functions and mixed functions F16, F19, and F26, RUSDE performed the superiority to other EAs.

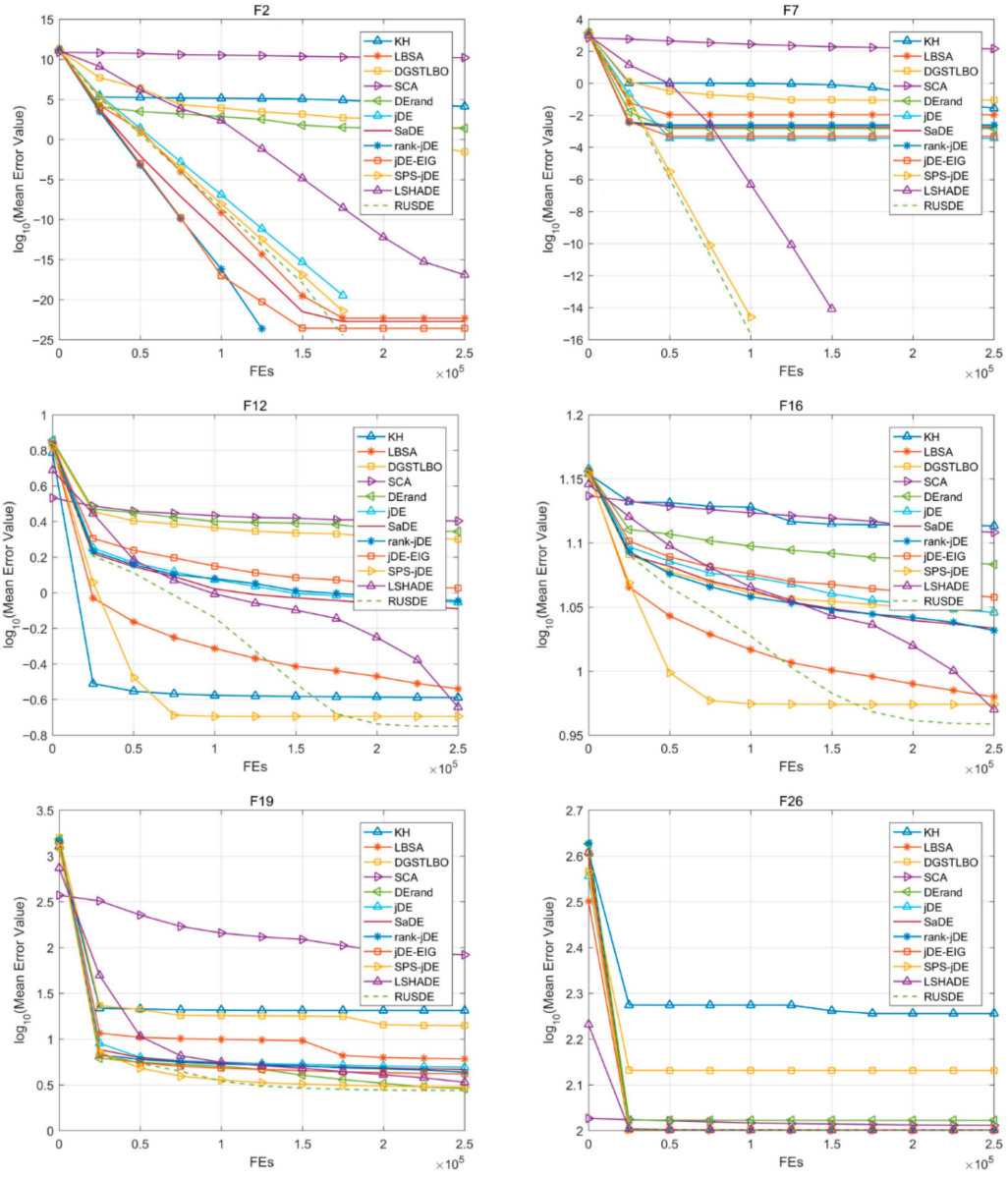

Figure 1. Convergence curves in 30D problems of the CEC 2014 benchmarks.

5. Application Example

The four-bar mechanism is a common mechanism widely used in many machines and devices. The dimensional synthesis of four-bar mechanisms aims to synthesize a mechanism with the minimum errors between the coupler points and desired points to meet the design requirements. A typical case presented in [64] is tested in this paper.

5.1. The Classic Case of Four-Bar Mechanism

The schematic of the four-bar mechanism together with the variables is shown in Figure 2. In the world coordinate system XOY, the position of coupler point C can be written as Equation (10).

$$\begin{cases} C_x^i(x) = x_0 + r_2 cos(\theta_2^i + \theta_0) + r_{cx} cos(\theta_3^i + \theta_0) - r_{cy} sin(\theta_3^i + \theta_0) \\ C_y^i(x) = y_0 + r_2 sin(\theta_2^i + \theta_0) + r_{cx} sin(\theta_3^i + \theta_0) + r_{cy} cos(\theta_3^i + \theta_0) \end{cases} \tag{10}$$

where r_1, r_2, r_3, and r_4 are the length of the links; r_{cx}, r_{cy} are the coordinates of coupler point C in the relative coordinate system X_cOY_c; x_0, y_0 are the coordinates of joint O_2 in the world coordinate system XOY; θ_0 is the angle of the stationary link with respect to the X-axis; θ_2^i is the i^{th} input angle corresponding to the i^{th} desired point in the relative coordinate system X_tOY_t.

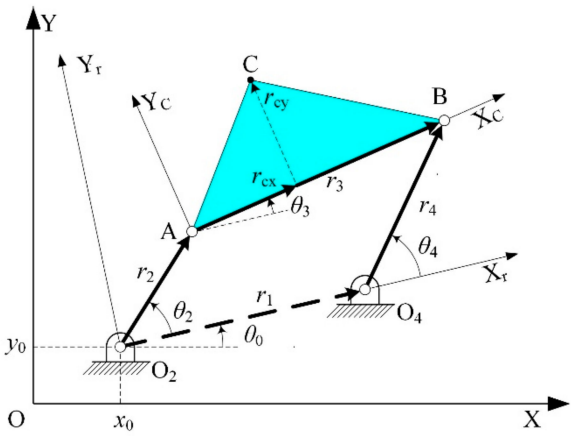

Figure 2. Variables of the four-bar mechanism.

5.2. The Constraints and Goal Function

The links' length must satisfy the Grashof condition, which can be expressed by Equation (11).

$$h1: r_1 + r_2 \leq r_3 + r_4 \tag{11}$$

In order to avoid the order defect, the sequence of input angles should fulfill the clockwise or counterclockwise rotation, shown in Equation (12).

$$h2: \begin{matrix} \theta_2^k < \theta_2^{k-1} < \cdots < \theta_2^2 < \theta_2^1 < \theta_2^n < \cdots < \theta_2^{k+1} \\ \theta_2^k < \theta_2^{k+1} < \cdots < \theta_2^n < \theta_2^1 < \theta_2^2 < \cdots < \theta_2^{k-1} \end{matrix} \tag{12}$$

The objective function can be divided into two parts. The first part is the sum of the square of Euclidean distances between the coupler points and the corresponding desired points; the second part is the penalty functions. The Grashof condition and the Sequence condition are introduced as the penalty functions. At last, the objective function of the optimal problem can be expressed as Equation (13)

$$f_{obj}(x) = \begin{cases} \sum_{i=1}^{n} [(C_x^i(x) - C_{dx}^i(x))^2 + (C_y^i(x) - C_{dy}^i(x))^2] \\ + h_1(x)M_1 + h_2(x)M_2 \end{cases} \tag{13}$$

where $h_1(x)$, $h_2(x)$ are the Grashof and Sequence condition, respectively. If the condition is satisfied, then the value is equal to 0, otherwise, it is set to 1; M_1, M_2 are high values to penalize the objective function, and they are set to 10^4.

5.3. The Experimental Settings and Results

We compare five algorithms, KH, DGSTLBO, SCA, DE, and jDE, on the synthesis problem of four-bar mechanisms. The parameter settings of algorithms are $N = 100$; $D = 15$; $FEs = 100,000$.

Then, the design variables of this example are:

$$X = [r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, x_0, y_0, \theta_0, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6]$$

This problem requires tracing a trajectory along eighteen given points arranged in an irregular closed path and with prescribed timing. The desired points are:

$$\{C_d^i\} = [(0.5, 1.1); (0.4, 1.1); (0.3, 1.1); (0.2, 1); (0.1, 0.9); (0.005, 0.75); (0.02, 0.6); (0, 0.5); (0, 0.4); (0.03, 0.3); (0.1, 0.25); (0.15, 0.2); (0.2, 0.3); (0.3, 0.4); (0.4, 0.5); (0.5, 0.7); (0.6, 0.9); (0.6, 1)]$$

$$\{\theta_2^i\} = \{\theta_2^1 + (i-1)\tfrac{\pi}{9}\},\ i = 1, \cdots, 18$$

The values of the best, mean, and standard deviations obtained by the six algorithms for 30 runs are presented in Table 7. It demonstrates that the best solution obtained by RUSDE is the most accurate among the six algorithms. The convergence graph of the best values for the synthesis problem of the four-bar mechanism is shown in Figure 3. The results show that RUSDE has the fastest convergence speed and the highest convergence accuracy among the six competitive algorithms. Therefore, RUSDE is superior to the other five algorithms in convergence speed and accuracy.

Table 7. The results of the four-bar mechanism with the six algorithms.

Algorithm	KH	DGSTLBO	SCA	DErand	jDE	SUDE
r_1	55.42712	46.54123	23.70608	50.861491	41.744822	50.921529
r_2	32.90918	6.004795	6.032425	10.760724	9.7486632	10.555366
r_3	58.25947	21.47507	27.71803	27.302929	22.983601	25.963469
r_4	55.22884	52.0851	29.26344	45.535705	38.477082	44.544581
r_{cx}	−1.86351	39.11445	5.270246	27.279789	25.521236	24.896262
r_{cy}	24.41642	20.138	−29.3624	23.886111	18.161774	19.182391
x_0	11.77316	57.68502	2.196234	−4.925943	−0.655142	1.6991638
y_0	35.90325	11.74831	7.080032	59.604688	55.047776	57.587445
θ_0	6.282858	0.516842	1.051096	3.6943843	3.7322061	3.658706
${\theta_2}^1$	5.918298	5.492722	0.676841	1.7660023	1.4301979	1.6464615
${\theta_2}^2$	6.017541	0.000418	2.031381	2.470655	2.5290628	2.4183316
${\theta_2}^3$	6.10178	0.410755	2.52128	2.9248265	2.9902006	2.8825657
${\theta_2}^4$	6.175316	0.734077	5.97276	3.3459071	3.4651034	3.3720796
${\theta_2}^5$	6.262379	1.08325	6.030824	3.7737419	3.9645854	3.8854152
${\theta_2}^6$	6.281808	1.575339	0	4.2562343	4.6160322	4.5596146
f_{obj}	14.8	2.30	3.34×10^2	1.35×10^{-2}	0.421	8.71×10^{-2}
Mean	43.0	37.7	5.53×10^2	2.39	9.01	0.996
std	27.7	57.3	2.28×10^2	3.86	7.47	1.47

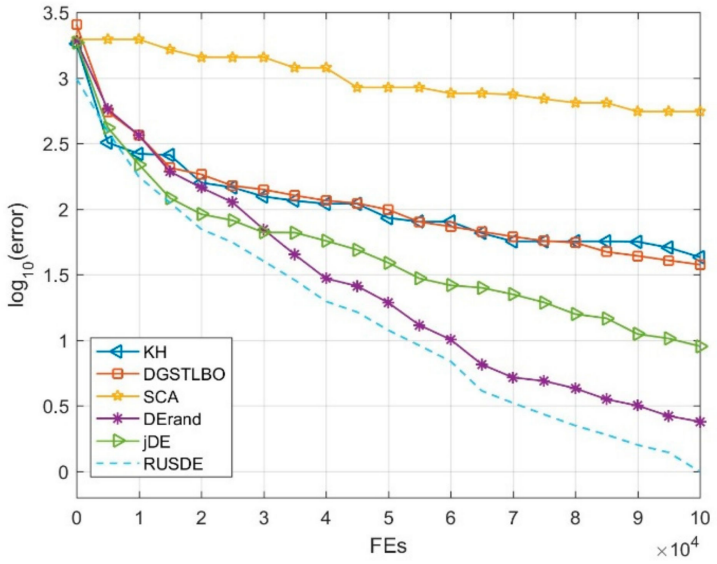

Figure 3. Convergence curves of the four-bar mechanism with the six algorithms.

6. Conclusions

In this study, we proposed an enhanced learning method to improve the performance of differential evolution (DE), named RUSDE. In this method, we selected the rank-up individuals and stored them into an archive Y. Compared with the fitness updated individuals, this selection was more strict; we adopted different mutation strategies inspired by simulating human social behavior. In this learning strategy, parents were selected in terms of their updating status. We used the CEC 2014 benchmarks comprising unimodal, basic multimodal, expanded multimodal, and hybrid problems to test the performance of RUSDE, and discussed the influence of the archive size M. The results showed that when the size was half of the population, the reasonable average rank could be obtained. We compared the performance of RUSDE with DErand, jDE, SaDE, Rank-jDE, SPS-jDE, jDE-EIG, KH, LBSA, DGSTLBO, LSHADE, and SCA. The numerical experiment results showed that RUSDE performed superior to DErand, jDE, SaDE, Rank-jDE, jDE-EIG, KH, LBSA, DGSTLBO, and SCA, except for SPS-jDE in the 30D and 50D problems. The statistical analysis results show that for the 30D problems, RUSDE performs better than the other algorithms, except for SaDE, SPS-jDE, and LSHADE, and for the 50D problems, RUSDE obtains better performance than the other algorithms, except for SaDE, rank-jDE, SPS-jDE, and LSHADE. An example of an application of the proposed method to optimize a four-bar mechanism showed that RUSDE performed better than the other algorithms. The limits of RUSDE are that the size of the selection archive Y has a crucial influence on the calculation of the optimal solution. For different problems, RUSDE with different archive sizes performs differently. In future research, an adaptive mechanism to decide the archive size M will be developed for RUSDE. Moreover, we will introduce this method to other similar algorithms and apply it to practical problems in the field of engineering control system.

Author Contributions: Conceptualization, K.Z.; Methodology, K.Z. and Y.Y.; Supervision, Y.Y.; Visualization, Y.Y.; Writing—original draft, K.Z.; Writing—review & editing, Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by a grant (#DMR-186-168) from National Chung Hsing University, Taichung, Taiwan.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Benchmarks in CEC 2014.

No.	Function Name	Expression	F_i*
F1	Rotated High Conditioned Elliptic Function	$F_1(x) = f_1(M(x - o_1)) + F_1*$	100
F2	Rotated Bent Cigar Function	$F_2(x) = f_2(M(x - o_2)) + F_2*$	200
F3	Rotated Discus Function	$F_3(x) = f_3(M(x - o_2)) + F_3*$	300
F4	Shifted and Rotated Rosenbrock's Function	$F_4(x) = f_4\left(M\left(\frac{2.048(x - o_4)}{100}\right)\right) + F_4*$	400
F5	Shifted and Rotated Ackley's Function	$F_5(x) = f_5(M(x - o_5)) + F_5*$	500
F6	Shifted and Rotated Weierstrass Function	$F_6(x) = f_6\left(M\left(\frac{0.5(x - o_6)}{100}\right)\right) + F_6*$	600
F7	Shifted and Rotated Griewank's Function	$F_7(x) = f_7\left(M\left(\frac{600(x - o_7)}{100}\right)\right) + F_7*$	700
F8	Shifted Rastrigin's Function	$F_8(x) = f_8\left(M\left(\frac{5.12(x - o_8)}{100}\right)\right) + F_8*$	800
F9	Shifted and Rotated Rastrigin's Function	$F_9(x) = f_8\left(M\left(\frac{5.12(x - o_9)}{100}\right)\right) + F_9*$	900
F10	Shifted Schwefel's Function	$F_{10}(x) = f_9\left(M\left(\frac{1000(x - o_{10})}{100}\right)\right) + F_{10}*$	1000
F11	Shifted and Rotated Schwefel's Function	$F_{11}(x) = f_9\left(M\left(\frac{1000(x - o_{11})}{100}\right)\right) + F_{11}*$	1100
F12	Shifted and Rotated Katsuura Function	$F_{12}(x) = f_{10}\left(M\left(\frac{5(x - o_{12})}{100}\right)\right) + F_{12}*$	1200
F13	Shifted and Rotated HappyCat Function	$F_{13}(x) = f_{11}\left(M\left(\frac{5(x - o_{13})}{100}\right)\right) + F_{13}*$	1300
F14	Shifted and Rotated HGBat Function	$F_{14}(x) = f_{12}\left(M\left(\frac{5(x - o_{14})}{100}\right)\right) + F_{14}*$	1400
F15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$F_{15}(x) = f_{13}\left(M\left(\frac{5(x - o_{15})}{100}\right) + 1\right) + F_{15}*$	1500
F16	Shifted and Rotated Expanded Scaffer's F6 Function	$F_{16}(x) = f_{14}(M(x - o_{16}) + 1) + F_{16}*$	1600
	$F(x) = g_1(M_1 z_1) + g_2(M_2 z_2) + \cdots + g_N(M_N z_N) + F*(x)$ where $z_1 = [y_{S_1}, y_{S_2}, \cdots, y_{S_{n1}}]$, $z_2 = [y_{S_{n1+1}}, y_{S_{n+2}}, \cdots, y_{S_{n1+n2}}]$, $\cdots$, $z_N = [y_{S_{\sum_{i=1}^{N-1} n_i + 1}}, y_{S_{\sum_{i=1}^{N-1} n_i + 2}}, \cdots, y_{S_D}]$ $n_1 = p_1 D$, $n_2 = p_2 D$, $\cdots$, $n_N = D - \sum_{i=1}^{N-1} n_i$		
F17	Hybrid Function 1	$N = 3$, $p = [0.3, 0.3, 0.4]$, $g = [f_9, f_8, f_1]$	1700
F18	Hybrid Function 2	$N = 3$, $p = [0.3, 0.3, 0.4]$, $g = [f_2, f_{12}, f_8]$	1800
F19	Hybrid Function 3	$N = 4$, $p = [0.2, 0.2, 0.3, 0.3]$, $g = [f_7, f_6, f_4, f_{14}]$	1900
F20	Hybrid Function 4	$N = 4$, $p = [0.2, 0.2, 0.3, 0.3]$, $g = [f_{12}, f_3, f_{13}, f_8]$	2000
F21	Hybrid Function 5	$N = 5$, $p = [0.1, 0.2, 0.2, 0.2\ 0.3]$, $g = [f_{14}, f_{12}, f_4, f_9, f_1]$	2100
F22	Hybrid Function 6	$N = 5$, $p = [0.1, 0.2, 0.2, 0.2\ 0.3]$, $g = [f_{10}, f_{11}, f_{13}, f_9, f_5]$	2200
F23	Composition Function 1	$N = 5$, $\sigma = [10, 20, 30, 40, 50]$, $\lambda = [1, 1 \times 10^{-6}, 1 \times 10^{-26}, 1 \times 10^{-6}, 1 \times 10^{-6}]$, $bias = [0, 100, 200, 300, 400]$, $g = [F_{4'}, F_{1'}, F_{2'}, F_{3'}, F_{1'}]$	2300
F24	Composition Function 2	$N = 3$, $\sigma = [20, 20, 20]$, $\lambda = [1, 1, 1]$, $bias = [0, 100, 200]$, $g = [F_{10'}, F_{9'}, F_{11'}]$	2400
F25	Composition Function 3	$N = 3$, $\sigma = [10, 30, 50]$, $\lambda = [0.25, 1, 1 \times 10^{-7}]$, $bias = [0, 100, 200]$, $g = [F_{11'}, F_{9'}, F_{1'}]$	2500
F26	Composition Function 4	$N = 5$, $\sigma = [10, 10, 10, 10, 10]$, $\lambda = [0.25, 1, 1 \times 10^{-7}, 2.5, 10]$, $bias = [0, 100, 200, 300, 400]$, $g = [F_{11'}, F_{13'}, F_{1'}, F_{6'}, F_{7'}]$	2600
F27	Composition Function 5	$N = 5$, $\sigma = [10, 10, 10, 10, 20]$, $\lambda = [10, 10, 2.5, 25, 1 \times 10^{-6}]$, $bias = [0, 100, 200, 300, 400]$, $g = [F_{14'}, F_{9'}, F_{11'}, F_{6'}, F_{1'}]$	2700
F28	Composition Function 6	$N = 5$, $\sigma = [10, 20, 30, 40, 50]$, $\lambda = [2.5, 10, 2.5, 5 \times 10^{-4}, 1 \times 10^{-6}]$, $bias = [0, 100, 200, 300, 400]$, $g = [F_{15'}, F_{13'}, F_{11'}, F_{16'}, F_{1'}]$	2800
F29	Composition Function 7	$N = 3$, $\sigma = [10, 30, 50]$, $\lambda = [1, 1, 1]$, $bias = [0, 100, 200]$, $g = [F_{17''}, F_{18''}, F_{19'}]$	2900
F30	Composition Function 8	$N = 3$, $\sigma = [10, 30, 50]$, $\lambda = [1, 1, 1]$, $bias = [0, 100, 200]$, $g = [F_{20'}, F_{21'}, F_{22'}]$	3000

Table A2. Definitions of the basic functions in CEC 2014.

No.	Function Name	Expression
F1	High Conditioned Elliptic Function	$f_1(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$
F2	Bent Cigar Function	$f_2(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$
F3	Discus Function	$f_3(x) = 10^6 x_1^2 + \sum_{i=2}^{D} x_i^2$
F4	Rosenbrock's Function	$f_4(x) = \sum_{i=1}^{D-1} \left(100(x_i^2 - x_{i+1}^2) + (x_i - 1)^2\right)$
F5	Ackley's Function	$f_5(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$
F6	Weierstrass Function	$f_6(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}\left[a^k \cos\left(2\pi b^k (x_i + 0.5)\right)\right]\right) - D\sum_{k=0}^{k\max}\left[a^k \cos\left(2\pi b^k \cdot 0.5\right)\right]$ where $a = 0.5$, $b = 3$, $k\max = 20$
F7	Griewank's Function	$f_7(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
F8	Rastrigin's Function	$f_8(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$
F9	Modified Schwefel's Function	$f_9(x) = 418.9829 \times \sum_{i=1}^{D} g(z_i)$ where $z_i = x_i + 4.209687462275036e + 002$ $g(z_i) = \begin{cases} z_i \sin(\lvert z_i \rvert^{1/2}) & \text{if } \lvert z_i \rvert \le 500 \\ (500 - \text{mod}(z_i, 500))\sin(\lvert 500 - \text{mod}(z_i, 500) \rvert) - \frac{(z_i - 500)^2}{10000D} & \text{if } z_i > 500 \\ (\text{mod}(\lvert z_i \rvert, 500) - 500)\sin(\text{mod}(\lvert z_i \rvert, 500) - 500) - \frac{(z_i + 500)^2}{10000D} & \text{if } z_i < -500 \end{cases}$
F10	Katsuura Function	$f_{10}(x) = \frac{10}{D^2}\prod_{i=1}^{D}\left(1 + i\sum_{j}^{32}\frac{\lvert 2^j x_i - round(2^j x_i)\rvert}{2^j}\right)^{\frac{10}{D^2}} - \frac{10}{D^2}$
F11	HappyCat Function	$f_{11}(x) = \left\lvert \sum_{i=1}^{D} x_i^2 - D \right\rvert^{1/4} + \left(0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right)/D + 0.5$
F12	HGBat Function	$f_{12}(x) = \left\lvert \left(\sum_{i=1}^{D} x_i^2\right)^2 - \left(\sum_{i=1}^{D} x_i\right)^2 \right\rvert^{1/2} + \left(0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right)/D + 0.5$
F13	Expanded Griewank's plus Rosenbrock's Function	$f_{13}(x) = f_7(f_4(x_1, x_2)) + f_7(f_4(x_2, x_3)) + \cdots + f_7(f_4(x_{D-1}, x_D)) + f_7(f_4(x_D, x_1))$
F14	Expanded Scaffer's F6 Function	$g(x, y) = 0.5 + \frac{(\sin^2\sqrt{x^2+y^2}) - 0.5}{1 + 0.001(x^2+y^2)}$ $f_{14}(x) = g(x_1, x_2) + g(x_2, x_3) + \cdots + g(x_{\text{Di}-1}, x_D) + g(x_D, x_1)$

References

1. Holland, J. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Application to Biology. Control and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1975.
2. Dorigo, M. Optimization, Learning and Natural Algorithms. Ph.D. Thesis, Politecnico di Milano, Milan, Italy, 1992.
3. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks—Conference Proceedings, Perth, WA, Australia, 27 November–1 December 1995.
4. Xin-She, Y.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 9–11 December 2009.
5. Rao, R.; Savsani, V.; Vakharia, D. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. Des.* **2011**, *43*, 303–315. [CrossRef]
6. Črepinšek, M.; Liu, S.-H.; Mernik, L. A note on teaching–learning-based optimization algorithm. *Inf. Sci.* **2012**, *212*, 79–93. [CrossRef]
7. Črepinšek, M.; Liu, S.-H.; Mernik, L.; Mernik, M. Is a comparison of results meaningful from the inexact replications of computational experiments? *Soft Comput.* **2016**, *20*, 223–235. [CrossRef]
8. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4831–4845. [CrossRef]
9. Civicioglu, P. Backtracking Search Optimization Algorithm for numerical optimization problems. *Appl. Math. Comput.* **2013**, *219*, 8121–8144. [CrossRef]
10. Rainer, S.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over con-tinuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359.
11. García, S.; Molina, D.; Lozano, M.; Herrera, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *J. Heuristics* **2009**, *15*, 617–644. [CrossRef]

12. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evo-lutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
13. Pan, Z.; Liang, H.; Gao, Z.; Gao, J. Differential evolution with subpopulations for high-dimensional seismic inversion. *Geophys. Prospect.* **2018**, *66*, 1060–1069. [CrossRef]
14. Elferik, S.; Hassan, M.; Alnaser, M. Adaptive Valve Stiction Compensation Using Differential Evolution. *J. Chem. Eng. Jpn.* **2018**, *51*, 407–417. [CrossRef]
15. Qiu, X.; Xu, J.-X.; Xu, Y.H.; Tan, K.C. A New Differential Evolution Algorithm for Minimax Optimization in Robust Design. *IEEE Trans. Cybern.* **2018**, *48*, 1355–1368. [CrossRef] [PubMed]
16. Aguitoni, M.C.; Pavão, L.V.; Siqueira, P.H.; Jiménez, L.; Ravagnani, M.A.D.S.S. Heat exchanger network synthesis using genetic algorithm and differential evolution. *Comput. Chem. Eng.* **2018**, *117*, 82–96. [CrossRef]
17. Ak, C.; Yildiz, A.; Yıldız, A. A Novel Closed-Form Expression Obtained by Using Differential Evolution Algorithm to Calculate Pull-In Voltage of MEMS Cantilever. *J. Microelectromech. Syst.* **2018**, *27*, 392–397. [CrossRef]
18. Zhao, W.-J.; Liu, E.-X.; Wang, B.; Gao, S.-P.; Png, C.E. Differential Evolutionary Optimization of an Equivalent Dipole Model for Electromagnetic Emission Analysis. *IEEE Trans. Electromagn. Compat.* **2018**, *60*, 1635–1639. [CrossRef]
19. Manjit, K.; Kumar, V. Colour image encryption technique using differential evolution in non-subsampled con-tourlet transform domain. *IET Image Process.* **2018**, *12*, 1273–1283.
20. Zhou, Z.; Gao, X.; Zhang, J.; Zhu, Z.; Hu, X. A novel hybrid model using the rotation forest-based differential evolution online sequential extreme learning machine for illumination correction of dyed fabrics. *Text. Res. J.* **2019**, *89*, 1180–1197. [CrossRef]
21. Wang, Y.; Zhou, M.; Song, X.; Gu, M.; Sun, J. Constructing Cost-Aware Functional Test-Suites Using Nested Differential Evolution Algorithm. *IEEE Trans. Evol. Comput.* **2017**, *22*, 334–346. [CrossRef]
22. Wang, L.; Zeng, Y.; Chen, T. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. *Expert Syst. Appl.* **2015**, *42*, 855–863. [CrossRef]
23. Marco, B.; Milani, A.; Santucci, V. Learning bayesian networks with algebraic differential evolution. In Proceedings of the International Conference on Parallel Problem Solving from Nature, Coimbra, Portugal, 8–12 September 2018.
24. Aleš, Z.; Sosa, J.D.H. Success history applied to expert system for underwater glider path planning using differential evolution. *Expert Syst. Appl.* **2019**, *119*, 155–170.
25. Qin, A.K.; Huang, V.L.; Suganthan, P.N. Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization. *IEEE Trans. Evol. Comput.* **2008**, *13*, 398–417. [CrossRef]
26. Brest, J.; Greiner, S.; Boskovic, B.; Mernik, M.; Zumer, V. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Trans. Evol. Comput.* **2006**, *10*, 646–657. [CrossRef]
27. Zhang, J.; Sanderson, A.C. JADE: Adaptive Differential Evolution with Optional External Archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
28. Ryoji, T.; Fukunaga, A. Success-history based parameter adaptation for differential evolution. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013.
29. Guo, S.-M.; Yang, C.-C.; Hsu, P.-H.; Tsai, J.S.H. Improving Differential Evolution with a Successful-Parent-Selecting Framework. *IEEE Trans. Evol. Comput.* **2015**, *19*, 717–730. [CrossRef]
30. Guo, S.-M.; Yang, C.-C. Enhancing Differential Evolution Utilizing Eigenvector-Based Crossover Operator. *IEEE Trans. Evol. Comput.* **2015**, *19*, 31–49. [CrossRef]
31. Gong, W.; Cai, Z. Differential Evolution with Ranking-Based Mutation Operators. *IEEE Trans. Cybern.* **2013**, *43*, 2066–2081. [CrossRef] [PubMed]
32. Guo, L.; Li, X.; Gong, W. Ranking-Based Differential Evolution for Large-Scale Continuous Optimization. *Comput. Inform.* **2018**, *37*, 49–75. [CrossRef]
33. Tang, L.; Dong, Y.; Liu, J. Differential Evolution with an Individual-Dependent Mechanism. *IEEE Trans. Evol. Comput.* **2014**, *19*, 560–574. [CrossRef]
34. Das, S.; Abraham, A.; Chakraborty, U.K.; Konar, A. Differential Evolution Using a Neighborhood-Based Mutation Operator. *IEEE Trans. Evol. Comput.* **2009**, *13*, 526–553. [CrossRef]
35. Mallipeddi, R.; Suganthan, P.; Pan, Q.; Tasgetiren, M. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Appl. Soft Comput.* **2011**, *11*, 1679–1696. [CrossRef]
36. Islam, S.M.; Das, S.; Ghosh, S.; Roy, S.; Suganthan, P.N. An Adaptive Differential Evolution Algorithm with Novel Mutation and Crossover Strategies for Global Numerical Optimization. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2012**, *42*, 482–500. [CrossRef]
37. Segredo, E.; Lalla-Ruiz, E.; Hart, E. A novel similarity-based mutant vector generation strategy for differential evolution. In Proceedings of the Genetic and Evolutionary Computation Conference, Kyoto, Japan, 15–19 July 2018.
38. Xu, B.; Tao, L.; Chen, X.; Cheng, W. Adaptive differential evolution with multi-population-based mutation operators for constrained optimization. *Soft Comput.* **2019**, *23*, 3423–3447. [CrossRef]
39. Peng, H.; Guo, Z.; Deng, C.; Wu, Z. Enhancing differential evolution with random neighbors based strategy. *J. Comput. Sci.* **2018**, *26*, 501–511. [CrossRef]
40. Huang, Q.; Zhang, K.; Song, J.; Zhang, Y.; Shi, J. Adaptive differential evolution with a Lagrange interpolation argument algorithm. *Inf. Sci.* **2019**, *472*, 180–202. [CrossRef]

41. Liang, J.; Qu, B.Y.; Suganthan, P.N. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. In *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report*; Nanyang Technological University: Singapore, 2013; p. 635.
42. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M.A. Opposition-Based Differential Evolution. *IEEE Trans. Evol. Comput.* **2008**, *12*, 64–79. [CrossRef]
43. Wang, H.; Wu, Z.; Rahnamayan, S. Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems. *Soft Comput.* **2010**, *15*, 2127–2140. [CrossRef]
44. Hansen, N.; Ostermeier, A. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evol. Comput.* **2001**, *9*, 159–195. [CrossRef]
45. Hansen, N.; Niederberger, A.S.P.; Guzzella, L.; Koumoutsakos, P. A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion. *IEEE Trans. Evol. Comput.* **2008**, *13*, 180–197. [CrossRef]
46. Ghosh, S.; Das, S.; Roy, S.; Islam, S.M.; Suganthan, P. A Differential Covariance Matrix Adaptation Evolutionary Algorithm for real parameter optimization. *Inf. Sci.* **2012**, *182*, 199–219. [CrossRef]
47. Wang, Y.; Li, H.-X.; Huang, T.; Li, L. Differential evolution based on covariance matrix learning and bimodal distribution parameter setting. *Appl. Soft Comput.* **2014**, *18*, 232–247. [CrossRef]
48. Wang, Y.; Cai, Z.; Zhang, Q. Enhancing the search ability of differential evolution through orthogonal crossover. *Inf. Sci.* **2012**, *185*, 153–177. [CrossRef]
49. Wang, H.; Rahnamayan, S.; Sun, H.; Omran, M.G.H. Gaussian Bare-Bones Differential Evolution. *IEEE Trans. Cybern.* **2013**, *43*, 634–647. [CrossRef] [PubMed]
50. Wang, Y.; Cai, Z.; Zhang, Q. Differential Evolution with Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Trans. Evol. Comput.* **2011**, *15*, 55–66. [CrossRef]
51. Dorronsoro, B.; Bouvry, P. Improving Classical and Decentralized Differential Evolution with New Mutation Operator and Population Topologies. *IEEE Trans. Evol. Comput.* **2011**, *15*, 67–98. [CrossRef]
52. Črepinšek, M.; Liu, S.-H.; Mernik, M.; Ravber, M. Long Term Memory Assistance for Evolutionary Algorithms. *Mathematics* **2019**, *7*, 1129. [CrossRef]
53. Auger, A.; Hansen, N. A restart CMA evolution strategy with increasing population size. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Scotland, UK, 2–5 September 2005; Volume 2, pp. 1769–1776.
54. Brest, J.; Maučec, M.S.; Bošković, B. Single objective real-parameter optimization: Algorithm jSO. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastián, Spain, 5–8 June 2017; pp. 1311–1318.
55. Cai, Y.; Wang, J. Differential evolution with neighborhood and direction information for numerical optimiza-tion. *IEEE Trans. Cybern.* **2013**, *43*, 2202–2215. [CrossRef]
56. Brest, J.; Korosec, P.; Šilc, J.; Zamuda, A.; Bošković, B.; Maučec, M.S. Differential evolution and differential ant-stigmergy on dynamic optimisation problems. *Int. J. Syst. Sci.* **2013**, *44*, 663–679. [CrossRef]
57. Zhang, K.; Huang, Q.; Zhang, Y. Enhancing comprehensive learning particle swarm optimization with local optima topology. *Inf. Sci.* **2019**, *471*, 1–18. [CrossRef]
58. Salomon, R. Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. *BioSystems* **1996**, *39*, 263–278. [CrossRef]
59. Tanabe, R.; Fukunaga, A.S. Improving the search performance of SHADE using linear population size reduction. In Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 6–11 July 2014.
60. Chen, D.; Zou, F.; Lu, R.; Wang, P. Learning backtracking search optimisation algorithm and its application. *Inf. Sci.* **2017**, *376*, 71–94. [CrossRef]
61. Zou, F.; Wang, L.; Hei, X.; Chen, D.; Yang, D. Teaching–learning-based optimization with dynamic group strategy for global optimization. *Inf. Sci.* **2014**, *273*, 112–131. [CrossRef]
62. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl. Based Syst.* **2016**, *96*, 120–133. [CrossRef]
63. Veček, N.; Črepinšek, M.; Mernik, M. On the influence of the number of algorithms, problems, and independent runs in the comparison of evolutionary algorithms. *Appl. Soft Comput.* **2017**, *54*, 23–45. [CrossRef]
64. Singh, R.; Chaudhary, H.; Singh, A.K. Defect-free optimal synthesis of crank-rocker linkage using nature-inspired optimization algorithms. *Mech. Mach. Theory* **2017**, *116*, 105–122. [CrossRef]