mathematics

MDPI

Article

Success History-Based Adaptive Differential Evolution Using Turning-Based Mutation

Xingping Sun, Linsheng Jiang, Yong Shen *, Hongwei Kang * and Qingyi Chen

School of Software, Yunnan University, Kunming 650000, China; sunxp@ynu.edu.cn (X.S.); jls@mail.ynu.edu.cn (L.J.); devas9@ynu.edu.cn (Q.C.)

* Correspondence: sheny@ynu.edu.cn (Y.S.); hwkang@ynu.edu.cn (H.K.)

Received: 24 August 2020; Accepted: 7 September 2020; Published: 11 September 2020

Abstract: Single objective optimization algorithms are the foundation of establishing more complex methods, like constrained optimization, niching and multi-objective algorithms. Therefore, improvements to single objective optimization algorithms are important because they can impact other domains as well. This paper proposes a method using turning-based mutation that is aimed to solve the problem of premature convergence of algorithms based on SHADE (Success-History based Adaptive Differential Evolution) in high dimensional search space. The proposed method is tested on the Single Objective Bound Constrained Numerical Optimization (CEC2020) benchmark sets in 5, 10, 15, and 20 dimensions for all SHADE, L-SHADE, and jSO algorithms. The effectiveness of the method is verified by population diversity measure and population clustering analysis. In addition, the new versions (Tb-SHADE, TbL-SHADE and Tb-jSO) using the proposed turning-based mutation get apparently better optimization results than the original algorithms (SHADE, L-SHADE, and jSO) as well as the advanced DISH and the jDE100 algorithms in 10, 15, and 20 dimensional functions, but only have advantages compared with the advanced j2020 algorithm in 5 dimensional functions.

Keywords: single objective optimization; differential evolution; success-history; premature convergence; turning-based mutation

1. Introduction

The single objective global optimization problem involves finding a solution vector $\boldsymbol{x} = (x_1, \ldots, x_D)$ that minimizes the objective function $f(\boldsymbol{x})$, where D is the dimension of the problem. The task of black box optimization is to solve the global optimization problem without clear objective function form or structure, that is, f is a "black box". This problem appears in many problems of engineering optimization, where complex simulations are used to calculate the objective function.

The differential evolution (DE) algorithm, proposed by Price and Storm in 1995, laid the foundation for a series of successful algorithms for continuous optimization. DE is a random black box search method, which was originally designed for numerical optimization problems [1], and it's also an evolutionary algorithm that ensures that every next generation has better solutions than the previous generation: a phenomenon known as elitism. The extensive study fields of DE are summarized lately in the references [2].

Studies on DE have yielded a number of improvements [3–17] to the classical DE algorithm, and the status of research on it can be easily obtained by noting the results of the Continuous Optimization Competition and the Evolutionary Computing Conference (CEC).

A popular variant of DE [18] is the algorithm proposed by Fukunaga and Tanabe called Success History-based Adaptive Differential Evolution (SHADE) [19]. In the optimization process, the scale factor F and the crossover rate CR of control parameters are adjusted to adapt to the given problem, and the "current to $p$best/1" mutation strategy and the external archive of poor quality solutions in

JADE [20] are combined in SHADE. The SHADE algorithm ranked third in CEC2013. In the second year, the author proposed an improved scheme, adding a linear reduction to the population size called L-SHADE to improve the convergence rate of SHADE [21]. L-SHADE won the CEC2014 competition. The winners in the subsequent years were SPS-L-SHADE-EIG [22] (CEC2015), LSHADE-EpSin [23] (joint winner of CEC2016), and jSO [24] (CEC2017). These algorithms are all based on L-SHADE, which makes it one of the most effective variants of SHADE [25]. With the exception of the jSO, the other winners benefited from general enhancements in the area [26]. Consequently, this study applies an improved method to the SHADE, L-SHADE and jSO algorithms. LSHADE-ESP [27] came in second in CEC2018 and the jDE100 [28] won CEC2019. And the j2020 [29] algorithm, which was proposed on CEC2020 recently, is also within the reference range. Enhanced versions of these DE algorithms add new mechanisms or parameters for optimization, similar to those in other optimization algorithms [30], as described in substantive surveys of these areas [25,31–37]. Moreover, theoretical analysis supporting DE has also been provided, such as in [38–41].

The DE consists of three main steps: mutation, crossover, and selection. Many proposals [6,10,11,14, 24] have been made to improve the mutation process to improve optimization performance. For instance, four strategies of combining mutation and crossover was used in SHADE4 [6], SHADE44 [10] and L-SHADE44 [11] to create a new trial individual and realize an adaptive mechanism. A novel multi-chaotic framework was used in MC-SHADE [14] to generate random numbers for the parent selection process in mutation process. A new weighted mutation strategy with parameterization enhancement was used in jSO [24] to enhance adaptability. This paper also focuses on improving this process in the DE algorithm, especially SHADE-based algorithms.

The CEC2020 [42] single-objective boundary-constrained numerical optimization benchmark sets are designed to determine the improvement in performance obtained by increasing the number of the calculation of the fitness function of an optimization algorithm. There are thus two motivations for this study. First, we need to solve the problem of premature convergence of algorithms based on SHADE in high dimensional search spaces on CEC2020 benchmark sets, so that they can maintain a high population diversity and a longer exploration phase. Second, the improvement to the algorithm should be simple, should not excessively increase complexity, and should not render the proposed algorithm incomprehensible and less applicable, as discussed in [43]. We proposed a method using turning-based mutation, and apply it to the SHADE, L-SHADE, and jSO algorithms to yield good performance while using relatively simple algorithm structure. Through experimental analysis involving 10, 15, and 20 dimensions, the improved algorithms achieved better performance than the original algorithms as well as the advanced DISH [44] and jDE100 algorithms on CEC2020 benchmark sets, but were slightly worse than the j2020 algorithm. We also use population diversity measure and population clustering analysis to verify the effectiveness of the proposed method.

Section 2 describes the process of evolution from the DE algorithm to the SHADE, L-SHADE, and jSO algorithms, and turning-based mutation is introduced in Section 3. The experimental settings and results are described in Sections 4 and 5, respectively. Section 6 discusses the results, and the conclusion of this paper is given in Section 7.

2. DE SHADE L-SHADE and jSO

2.1. Differential Evolution

The DE consists of three main steps: mutation, crossover, and selection. In mutation, the attribute vector of the selected individual $\boldsymbol{x}$ is combined in a simple vector operation to generate the mutated vector $\boldsymbol{v}$. The scale factor F of the control parameter is used in this operation. In the crossover step, according to the probability given by the crossover rate CR of the control parameter, the trial vector $\boldsymbol{u}$ is created by selecting the attribute from the original vector $\boldsymbol{x}$ or mutated vector $\boldsymbol{v}$. Finally, in the selection step, the trial vector $\boldsymbol{u}$ is evaluated by the objective function and the fitness $f(\boldsymbol{u})$ is compared

with the fitness of the selected vector $f(\boldsymbol{x})$. The vector with the better fitness value survives to the next generation.

This paper focuses on improving the mutation process, so the paragraphs below describe the mutation process of the DE algorithm. The complete steps of DE can be referred to the literature [1]. The mutation strategy of DE/rand/1/bin can be expressed as follows:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r1,G} + F_i \times (\boldsymbol{x}_{r2,G} - \boldsymbol{x}_{r3,G}) \tag{1}$$

where $\boldsymbol{v}_{i,G}$ is the mutated vector, and $\boldsymbol{x}_{r1,G}$, $\boldsymbol{x}_{r2,G}$, and $\boldsymbol{x}_{r3,G}$ are three different individuals randomly selected from the population. F_i is the scaling factor, and G is the index of the current generation.

If any dimension of the mutated vector $\boldsymbol{v}_{j,i,G}$ is outside the boundary of the search range $[x_{min}, x_{max}]$, we perform the following correction for boundary-handling to handle infeasible solutions [45]:

$$\boldsymbol{v}_{j,i,G} = \begin{cases} \frac{x_{min} + \boldsymbol{x}_{j,i,G}}{2} & if\ \boldsymbol{v}_{j,i,G} < x_{min} \\ \frac{x_{max} + \boldsymbol{x}_{j,i,G}}{2} & if\ \boldsymbol{v}_{j,i,G} > x_{max} \end{cases} \tag{2}$$

where j is the dimensional index and i is the individual index.

The pseudo-code of the DE/rand/1/bin algorithm is shown in Algorithm 1.

Algorithm 1 DE/rand/1/bin

```
initialize P, NP, F, CR and MaxFES;
while FES < MaxFES do
  for each individual x do
    use mutation Formula (1) to create mutated vector v;
    execute boundary-handling (2) to handle infeasible solutions;
    use binomial crossover to create trial vector u;
    use selection of classical DE to create individual of next generation;
  end for
end while
return the best found solution.
```

It can be seen from the description of DE algorithm that users need to set three control parameters: crossover rate CR, scaling factor F and population size NP. The setting of these parameters is very important to the performance of DE.

Fine-tuning the control parameter is a time-consuming task, because of which most advanced variants of DE use parameter adaptation. This is also why Tanabe and Fukunaga proposed the SHADE [19] algorithm in 2013. Because the algorithms used in this paper are based on SHADE, it is described in more detail below.

2.2. SHADE

In the control parameters of SHADE, crossover rate CR and scaling factor F are discussed. The algorithm is based on JADE [20], proposed by Sanderson and Zhang, and so they share many mechanisms [18]. The major difference between them is in historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ with their update mechanisms. The next subsections describe the historical memory update of SHADE and the difference between DE and SHADE algorithm in initialization, mutation, crossover and selection, respectively. The complete steps of SHADE can be referred to the literature [19].

2.2.1. Initialization

In SHADE, the population is initialized in the same manner as in DE, but there are two additional components—historical memory and external archive—that also need to be initialized.

Initialize the control parameters stored in the historical memory, crossover rate CR and scale factor F to 0.5:

$$\boldsymbol{M}_{CR,i} = \boldsymbol{M}_{F,i} = 0.5; \forall i = 1, \cdots, H, \tag{3}$$

where H is the size of the user-defined historical memory, and the index k to update the historical memory is initialized to one.

In addition, the initialization of the external archive of poor quality solutions is empty, i.e., $\boldsymbol{A} = \varnothing$.

2.2.2. Mutation

In contrast to DE/rand/1/bin, the "current to *p*best/1" mutation strategy is used in SHADE:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F_i \times \left(\boldsymbol{x}_{pbest,G} - \boldsymbol{x}_{i,G}\right) + F_i \times \left(\boldsymbol{x}_{r1,G} - \boldsymbol{x}_{r2,G}\right) \tag{4}$$

$$p_i = rand[p_{min}, 0.2] \tag{5}$$

$$p_{min} = \frac{2}{NP} \tag{6}$$

where, $\boldsymbol{x}_{i,G}$ is the given individual, and $\boldsymbol{x}_{pbest,G}$ is an individual selected from the best $NP \times p_i$ ($p_i \in [0, 1]$) individuals randomly in the current population. Vector $\boldsymbol{x}_{r1,G}$ is an individual selected from the current population randomly, and $\boldsymbol{x}_{r2,G}$ is an individual selected from a combination of the external archive $\boldsymbol{A}$ and the current population randomly. Index $r1 \neq r2 \neq i$. F_i is a scaling factor, *rand* [] is a uniform random distribution and NP is the size of population. The $\boldsymbol{v}_{i,G}$ is the mutated vector and G is the index of the current generation. The greed of the "current-to-*p*best/1" mutation strategy depends on the control parameter p_i, which is calculated as shown in Equations (5) and (6). It balances exploration and exploitation capabilities (a small value of p is more greedy). The scaling factor F_i is generated using the following formula:

$$F_i = randc_i(\boldsymbol{M}_{F,ri}, 0.1) \tag{7}$$

where $randc_i$ () is the Cauchy distribution, and $\boldsymbol{M}_{F,ri}$ is randomly selected from historical memory $\boldsymbol{M}_F$ (index ri is a uniformly distributed random value from [1, H]). If $F_i > 1$, let $F_i = 1$. If $F_i \leq 0$, Equation (7) is repeated to attempt to generate a valid value.

The boundary handling of SHADE is identical to that of DE, as shown in Equation (2).

2.2.3. Crossover

DE has 2 classic crossover strategies, i.e., binomial and exponential. The crossover strategies of SHADE is the same as that of DE/rand/1/bin, i.e., binomial crossover. However, the crossover rate of DE/rand/1/bin is set in advance while the CR_i of SHADE is generated by the following formula:

$$CR_i = randn_i(\boldsymbol{M}_{CR,ri}, 0.1) \tag{8}$$

where $randn_i$ () is Gaussian distribution, and $\boldsymbol{M}_{CR,ri}$ are randomly selected from historical memory $\boldsymbol{M}_{CR}$ (index r_i is a uniformly distributed random value from [1, H]). If $CR_i > 1$, let $CR_i = 1$; if $CR_i < 0$, let $CR_i = 0$.

2.2.4. Selection

The process of selection of SHADE is the same as that of DE. However, the external archive needs to be updated during selection. If a better trail individual is generated, the original individual $\boldsymbol{x}_{i,G}$ is stored in the external archive. If the external archive exceeds capacity, one of them is randomly deleted.

2.2.5. Historical Memory Update

Historical memory update is also an important operation in SHADE. The historical memories $\boldsymbol{M}_{CR}$ and $\boldsymbol{M}_F$ are initialized by Formula (3) but their contents change with the iteration of the algorithm.

These memories store the "successful" crossover rate CR and scaling factor F. "Successful" here means that the trail vector $\boldsymbol{u}$ is selected instead of the original vector $\boldsymbol{x}$ to survive to the next generation. In each generation, the values of these "successful" CR and F are first stored in arrays $\boldsymbol{S}_{CR}$ and $\boldsymbol{S}_F$, respectively. After each generation, a unit of each of the historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ is updated. The updated unit is specified by index k, which is initialized to one and increases by one after each generation. If k exceeds the memory capacity H, it is reset to one. The following formula is used to update the k-th unit of historical memory:

$$\boldsymbol{M}_{CR,k,G+1} = \begin{cases} mean_{WA}(\boldsymbol{S}_{CR}) \; if \; \boldsymbol{S}_{CR} \neq \varnothing \\ \boldsymbol{M}_{CR,k,G} \; otherwise \end{cases} \tag{9}$$

$$\boldsymbol{M}_{F,k,G+1} = \begin{cases} mean_{WL}(\boldsymbol{S}_F) \; if \; \boldsymbol{S}_F \neq \varnothing \\ \boldsymbol{M}_{F,k,G} \; otherwise \end{cases} \tag{10}$$

If all individuals in the G-th generation fail to generate a better trail vector, i.e., $\boldsymbol{S}_F = \boldsymbol{S}_{CR} = \emptyset$, the historical memory will not be updated. The weighted Lehmer mean W_L and weighted mean W_A are calculated using the following formulas, respectively:

$$mean_{WA}(\boldsymbol{S}_{CR}) = \sum_{k=1}^{|S_{CR}|} \boldsymbol{w}_k \times \boldsymbol{S}_{CR,k}, \tag{11}$$

$$mean_{WL}(\boldsymbol{S}_F) = \frac{\sum_{k=1}^{|S_F|} \boldsymbol{w}_k \times \boldsymbol{S}_{F,k}^2}{\sum_{k=1}^{|S_F|} \boldsymbol{w}_k \times \boldsymbol{S}_{F,k}} \tag{12}$$

To improve the adaptability of the parameters, the weight vector $\boldsymbol{w}$ is calculated based on the absolute value of the difference that is obtained by subtracting the objective function value of the given vector from that of the trail vector in current generation G, as follows:

$$\boldsymbol{w}_k = \frac{\Delta f_k}{\sum_{k=1}^{|S_{CR}|} \Delta f_k} \tag{13}$$

where $\Delta f_k = \left| f\left(\boldsymbol{u}_{k,G}\right) - f\left(x_{k,G}\right) \right|$ in (13).

The pseudo-code of the SHADE algorithm is shown in Algorithm 2.

Algorithm 2 SHADE

```
initialize P, NP, F, CR, A, H and MaxFES;
initialize M_F, M_CR by (3);
while FES < MaxFES do
    for each individual x do
        use mutation Formulas (7) and (8) to select F and CR;
        use Formula (4) to create mutated vector v;
        execute boundary-handling (2) to handle infeasible solutions;
        use binomial crossover to create trial vector u;
        use selection of classical DE to create individual of next generation;
        update external archive A;
    end for
    use Formulas (9) and (10) to update historical memory M_F and M_CR;
end while
return the best found solution.
```

2.3. Linear Decrease in Population Size: L-SHADE

In [21], a linear reduction of population size was introduced to SHADE to improve its performance. The basic thought is to gradually reduce the population size during evolution to improve exploitation

capabilities. In L-SHADE, the population size is calculated after each generation using Formula (14). If the new population size NP_{new} is smaller than the previous population size NP, the all individuals are sorted on the basis of the value of the objective function, and the worst $NP\text{-}NP_{new}$ individuals are cut. Also, the size of external archives/$\boldsymbol{A}$/decreases synchronously with population size:

$$NP_{new} = round\left(NP_{init} - \frac{FES}{MAXFES} \times \left(NP_{init} - NP_f\right)\right) \tag{14}$$

where NP_f and NP_{init} are the final and initial population size, respectively. *MaxFES* and *FES* are the maximum and current number of the calculation of the fitness function, respectively. And *round* () is a rounding function.

2.4. Weighted Mutation Strategy with Parameterization Enhancement: jSO

The jSO [24] algorithm won the CEC2017 single-objective real parameter optimization competition [46]. It is a type of iL-SHADE algorithm that uses a weighted mutation strategy [47]. The iL-SHADE algorithm extends L-SHADE by initializing all parameters in the historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ to 0.8, statically initializing the last unit of historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ to 0.9, updating $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ with the weighted Lehmer average value, limiting the crossover rate CR and scaling factor F in the early stage, and p is calculated for the "current-to-$p$best/1" mutation strategy as:

$$p = p_{min} + \frac{FES}{MAXFES}(p_{max} - p_{min}) \tag{15}$$

where p_{min} and p_{max} are the minimum and maximum value of p, respectively. *FES* and *MaxFES* are the current and maximum number of the calculation of the fitness function, respectively.

The jSO algorithm sets $p_{max} = 0.25$ and $p_{min} = p_{max}/2$, initial population size to $NP_{init} = 25\sqrt{D}\log D$, and the size of the historical memory to $H = 5$. All parameters in $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ are initialized to 0.3 and 0.8, respectively, and the weighted mutation strategy current-to-$p$best-w/1 is used:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F_w \times \left(\boldsymbol{x}_{pbest,G} - \boldsymbol{x}_{i,G}\right) + F_i \times (\boldsymbol{x}_{r1,G} - \boldsymbol{x}_{r2,G}), \tag{16}$$

where F_w is calculated as:

$$F_w = \begin{cases} 0.7F_i, & FES < 0.2MAXFES, \\ 0.8F_i, & FES < 0.4MAXFES, \\ 1.2F_i, & otherwise. \end{cases} \tag{17}$$

3. Turning-Based Mutation

The opposition-based DE (ODE) algorithm was proposed by Shahryar et al. [48]. The opposition-based learning (OBL) was used for generation jumping and population initialization, and the opposite numbers was used to improve the convergence rate of DE. Shahryar et al. let all vectors of the initial population take the opposite number in the initialization and allowed the trail vectors to take the opposite number in the selection operation. They then compared their fitness values and selected the vector with the better fitness to accelerate the convergence of the DE algorithm. We refer to the idea of "opposition" in the above algorithm, but the purpose of this paper is to change the direction of mutation under certain conditions to maintain population diversity and enable a longer exploration phase.

Suppose that the search space is two-dimensional (2D). There is a ring-shaped region, the center of which is the global suboptimal individual $x_{pbest,G}$. The outer radius of the ring is OR and the inner radius is IR. If the Euclidean distance $Distance$ between the given individual and the global suboptimal individual is smaller than the outer radius OR and larger than the inner radius IR, the differential vector de_i from the mutation Formulas (1) and (4) takes the opposite number, and some dimensions are

randomly selected to assign random values within the search range. Experiments have verified that better outer radius *OR* and inner radius *IR* can be calculated as:

$$OR_{init} = \sum_{j=1}^{D} \sqrt[2]{\left(\frac{x_{max} - x_{min}}{2}\right)^2} \tag{18}$$

$$IR = \sum_{j=1}^{D} \sqrt[2]{\left(\frac{x_{max} - x_{min}}{40}\right)^2} \tag{19}$$

$$OR = OR_{init} + \frac{IR - OR_{init}}{MaxFES} \times FES \tag{20}$$

where OR_{init} is the initial value of the outer radius and *IR* is the inner radius, which is also the minimum value of the outer radius. The outer radius *OR* decreases with an increase in the number of fitness evaluations. *MaxFES* and *FES* are the maximum and current number of the calculation of the fitness function, respectively, and x_{max} and x_{min} are the upper and lower bounds of the search range, respectively.

The Euclidean distance *Distance* between the given individual and the global suboptimal individual is calculated as:

$$Distance = \sqrt[2]{\sum_{j=1}^{D} \left(\boldsymbol{x}_{j,pbest,G} - \boldsymbol{x}_{j,i,G}\right)^2} \tag{21}$$

The differential vector de_i from the mutation Equations (1) and (4) is calculated as:

$$de_i = (F_i \; or \; F_w) \times \left(\boldsymbol{x}_{pbest,G} - \boldsymbol{x}_{i,G}\right) + F_i \times (\boldsymbol{x}_{r1,G} - \boldsymbol{x}_{r2,G}) \tag{22}$$

The pseudo-code of the operation on the differential vector de_i in turning-based mutation is shown as Operation 1:

Operation 1 operation on de_i

```
if Distance > IR and Distance < OR then
    de_i = −de_i;
    M = randi(D), R = randperm(D);
    for d = 1 to M do
        de_i(R(d)) = rand (x_max−x_min) + x_min;
    end for
end if
```

where $\boldsymbol{R}$ is the randomly disordered dimension index array, *M* is the number of randomly selected dimensions, and x_{max} and x_{min} are the upper and lower bounds of the search range, respectively.

Finally, the mutation operation is performed as shown in Equation (23):

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + de_i \tag{23}$$

If the Euclidean distance *Distance* between the given individual and the global suboptimal individual is smaller than the outer radius *OR* and larger than the inner radius *IR*, the improved method changes the direction of mutation of the given individual to maintain the population diversity and a longer exploration phase, thus enhancing the global search ability and the ability to escape the local optimum. Then, with an increase in number of fitness evaluations, the performance of the algorithm can be improved. If the Euclidean distance *Distance* between the given individual and the global suboptimal individual is smaller than or equal to the inner radius *IR*, the former is

allowed to mutate in the original direction. This enables the given individual to quickly converge to the global optimal or suboptimal position to avoid the problem of non-convergence caused by turning-based mutation.

Since Equation (21) and Operation 1 need to be executed in the mutation process of each individual, the overall time complexity [42] of the improved algorithms is slightly higher than that of the original algorithms, as shown in Tables 1–3.

Table 1. Time complexity specified by CEC2020 technical document-SHADE vs. Tb-SHADE.

D	T0	T1	SHADE		Tb-SHADE	
			T2	(T2 − T1)/T0	T2	(T2 − T1)/T0
5	6.03E+01	2.52E+02	4.81E+03	7.56E+01	5.58E+03	8.84E+01
10	6.03E+01	3.05E+02	5.55E+03	8.70E+01	6.35E+03	1.00E+02
15	6.03E+01	3.39E+02	5.68E+03	8.86E+01	6.76E+03	1.06E+02
20	6.03E+01	4.09E+02	6.03E+03	9.32E+01	7.14E+03	1.12E+02

Table 2. Time complexity specified by CEC2020 technical document—L-SHADE vs. TbL-SHADE.

D	T0	T1	L-SHADE		TbL-SHADE	
			T2	(T2 − T1)/T0	T2	(T2 − T1)/T0
5	6.03E+01	2.52E+02	4.44E+03	6.95E+01	4.97E+03	7.82E+01
10	6.03E+01	3.05E+02	4.77E+03	7.40E+01	6.71E+03	1.06E+02
15	6.03E+01	3.39E+02	4.98E+03	7.70E+01	6.97E+03	1.10E+02
20	6.03E+01	4.09E+02	5.24E+03	8.01E+01	7.30E+03	1.14E+02

Table 3. Time complexity specified by CEC2020 technical document-jSO vs. Tb-jSO.

D	T0	T1	jSO		Tb-jSO	
			T2	(T2 − T1)/T0	T2	(T2 − T1)/T0
5	6.03E+01	2.52E+02	4.30E+03	6.71E+01	5.24E+03	8.27E+01
10	6.03E+01	3.05E+02	5.28E+03	8.25E+01	6.51E+03	1.03E+02
15	6.03E+01	3.39E+02	6.50E+03	1.02E+02	7.12E+03	1.12E+02
20	6.03E+01	4.09E+02	7.27E+03	1.14E+02	7.81E+03	1.23E+02

The pseudo-code of the Tb-SHADE algorithm (SHADE algorithm using turning-based mutation) is shown as Algorithm 3, that of the TbL-SHADE algorithm (L-SHADE algorithm using turning-based mutation) is shown as Algorithm 4, and that of the Tb-jSO algorithm (jSO algorithm using turning-based mutation) is shown as Algorithm 5. The improved parts of these algorithm are underlined.

Algorithm 3 Tb-SHADE

initialize $\boldsymbol{P}$, NP, F, CR, $\boldsymbol{A}$, H and $MaxFES$;
initialize $\boldsymbol{M}_F$, $\boldsymbol{M}_{CR}$ by (3);
initialize OR_{init} by (18), initialize IR by (19);
while $FES < MaxFES$ **do**
  Calculate OR by (20);
  for each individual x **do**
    use mutation Formulas (7) and (8) to select F and CR;
    Set *Distance* by (21);
    Execute Operation 1;
    use Formula (23) to create mutated vector v;
    execute boundary-handling (2) to handle infeasible solutions;
    use binomial crossover to create trial vector $\boldsymbol{u}$;
    use selection of classical DE to create individual of next generation;
    update external archive $\boldsymbol{A}$;
  end for
  use Formulas (9) and (10) to update historical memory $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$;
end while
return the best found solution.

Algorithm 4 TbL-SHADE

initialize $\boldsymbol{P}$, NP_{init}, NP_f, F, CR, $\boldsymbol{A}$, H and $MaxFES$;
initialize $\boldsymbol{M}_F$, $\boldsymbol{M}_{CR}$ by (3);
initialize OR_{init} by (18), initialize IR by (19);
while $FES < MaxFES$ **do**
  Calculate OR by (20);
  for each individual x **do**
    use mutation Formulas (7) and (8) to select F and CR;
    Set *Distance* by (21);
    Execute Operation 1;
    use Formula (23) to create mutated vector v;
    execute boundary-handling (2) to handle infeasible solutions;
    use binomial crossover to create trial vector $\boldsymbol{u}$;
    use selection of classical DE to create individual of next generation;
    update external archive $\boldsymbol{A}$;
  end for
  use Formulas (9) and (10) to update historical memory $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$;
use (14) to calculate NP_{new};
$NP = NP_{\text{new}}$, $|\boldsymbol{A}| = NP_{\text{new}}$;
end while
return the best found solution.

Algorithm 5 Tb-jSO

initialize $\boldsymbol{P}$, NP_{init}, NP_f, F, CR, $\boldsymbol{A}$, H and $MaxFES$;
initialize all values in $\boldsymbol{M}_F$ to 0.3 and $\boldsymbol{M}_{CR}$ to 0.8, but $\boldsymbol{M}_{F,H} = 0.9$ and $\boldsymbol{M}_{CR,H} = 0.9$;
initialize OR_{init} by (18), initialize IR by (19);
while $FES < MaxFES$ **do**
  Calculate OR by (20);
  for each individual $\boldsymbol{x}$ **do**
    use mutation Formulas (7) and (8) to select F and CR;
    use (17) to calculate F_w;
    if $FES < 0.6MaxFES$ **and** $F_{i,G} > 0.7$ **then**
      $F_{i,G} = 0.7$;
    end if
    if $FES < 0.25MaxFES$ **then**
      $CR_{i,G} = \max(CR_{i,G}, 0.7)$;
    else if $FES < 0.5MaxFES$ **then**
      $CR_{i,G} = \max(CR_{i,G}, 0.6)$;
    end if
    Set $Distance$ by (21);
    Execute Operation 1;
    use Formula (23) to create mutated vector $\boldsymbol{v}$;
    execute boundary-handling (2) to handle infeasible solutions;
    use binomial crossover to create trial vector $\boldsymbol{u}$;
    use selection of classical DE to create individual of next generation;
    update external archive $\boldsymbol{A}$;
  end for
  use Formulas (9) and (10) to update historical memory $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$;
  use (14) to calculate NP_{new};
$NP = NP_{\text{new}}$, $|\boldsymbol{A}| = NP_{\text{new}}$;
end while
return the best found solution.

4. Experimental Settings

To verify the improved method by experiments, the original algorithm, the improved algorithm and the advanced DISH and the jDE100 algorithms were tested on the Single Objective Bound Constrained Numerical Optimization (CEC2020) benchmark sets in 5, 10, 15 and 20 dimensions. The termination criteria, i.e., the maximum number of the calculation of the fitness function (*MaxFES*) and the minimum error value (*Min error value*), were set as in Table 4. The search range is $[x_{min}, x_{max}] = [-100, 100]$, and 30 independent repeated experiments were conducted. The parameter setting of most algorithm [19,21,24] is shown in Tables 5 and 6. In addition, the parameter setting of j2020 algorithm can be found in [29].

Table 4. Termination criteria.

D	*MaxFES*	*Min Error Value*
5	50,000	10^{-8}
10	1,000,000	10^{-8}
15	3,000,000	10^{-8}
20	10,000,000	10^{-8}

Table 5. Parameter setting of some algorithms.

Algorithm	NP	H	$\|A\|$	NP_{init}	NP_f	$MaxG$	M_{CRinit}	M_{Finit}
SHADE	100	NP	NP	–	–	$MaxFES/NP$	0.5	0.5
Tb-SHADE	100	NP	NP	–	–	$MaxFES/NP$	0.5	0.5
L-SHADE	Calculated by (18)	100	NP	100	4	Not fixed	0.5	0.5
TbL-SHADE	Calculated by (18)	100	NP	100	4	Not fixed	0.5	0.5
jSO	Calculated by (18)	5	NP	25logD	4	Not fixed	0.8 $M_{CR,H} = 0.9$	0.3 $M_{F,H} = 0.9$
Tb-jSO	Calculated by (18)	5	NP	25logD	4	Not fixed	0.8 $M_{CR,H} = 0.9$	0.3 $M_{F,H} = 0.9$
DISH	Calculated by (18)	5	NP	25logD	4	Not fixed	0.8 $M_{CR,H} = 0.9$	0.3 $M_{F,H} = 0.9$

Table 6. Parameter setting of jDE100.

Parameter	Value	Description
F_l	$\frac{5.0}{\sqrt{bNP}}$	lower limit of scale factor for the big population
F_l	$\frac{1.0}{\sqrt{bNP}}$	lower limit of scale factor for the small population
F_u	1.1	upper limit of scale factor
CR_l	0.0	lower limit of crossover parameter
CR_u	1.1	upper limit of crossover parameter
F_{init}	0.5	initial value of scale factor
CR_{init}	0.5	initial value of crossover parameter
τ_1	0.1	probability to self-adapt scale factor
τ_2	0.1	probability to self-adapt crossover parameter
bNP	1000	size of big population
sNP	25	size of small population
$ageLmt$	1×10^9	number of FEs when population restart needs to occurs
eps	1×10^{-16}	small value used to check if two value are similar
$myEqs$	25	reinitialization if myEqs% of individuals in the corresponding population have the similar function values
$MaxG$	Not fixed	the maximum number of generations

The hypothesis that the turning-based mutation can maintain a longer exploration phase can be verified by analyzing the clustering and density of the population during the optimization process. These two analyses are described in more detail below.

4.1. Cluster Analysis

The clustering algorithm selected in this experiment is density based noisy application spatial clustering (DBSCAN) [49], which is based on the clustering density rather than its center, so it can find clusters of arbitrary shape. DBSCAN algorithm needs to set two control parameters and a distance measurement. The settings are as follows:

(1) distance between core points, that is, Eps = 1% of the decision space; for the CEC2020 benchmark sets, Eps = 2;
(2) minimum number of points forming a cluster, that is, $MinPts$ = 4 (minimum number of mutation individuals); and
(3) distance measure uses Chebyshev distance [50]—if the distance between the corresponding attributes of two individuals is greater than 1% of the decision space, they are not considered as direct density reachable.

4.2. Population Diversity

The population diversity (PD) measure is taken from [51], which is based on the square root of the deviation sum, Equation (25), of individual components and their corresponding mean values, Equation (24):

$$\overline{x_j} = \frac{1}{NP}\sum_{i=1}^{NP} \boldsymbol{x}_{j,i} \tag{24}$$

$$PD = \sqrt{\frac{1}{NP}\sum_{i=1}^{NP}\sum_{j=1}^{D}\left(\boldsymbol{x}_{j,i} - \overline{x_j}\right)^2}. \tag{25}$$

where i is the iterator of members of the population and j is that of the component (dimension).

5. Results

Tables 7–18 compare the error values (when the error value was smaller than 10^{-8}, the corresponding value was considered optimal) obtained by the original algorithms (SHADE, L-SHADE, and jSO) and their improved versions using turning-based mutation (Tb-SHADE, TbL-SHADE and Tb-jSO, respectively). The results of a comparison are showed in the last column of each table. If the performance of the original version was significantly better, uses the "−" sign; if the performance of the improved version was significantly better, uses the sign "+"; if their performances were similar, "=" is used. The better performance values are displayed in bold, and the last row of these tables shows the results of an overall comparison. Tables 19–22 provide the error values obtained by the advanced algorithms DISH, jDE100 and j2020 on CEC2020. All tables provide the best, mean and std (standard deviation) values of 30 independent repetitions of the experiments.

Table 7. SHADE vs. Tb-SHADE on CEC2020 in 5*D*.

f	SHADE			Tb-SHADE			Result
	Best	**Mean**	**Std**	**Best**	**Mean**	**Std**	
1	1.70E−09	6.44E−09	2.40E−09	8.41E−01	5.37E+00	3.83E+00	−
2	3.78E−01	1.73E+00	1.80E+00	3.43E+00	1.30E+01	6.19E+00	−
3	1.32E+00	5.07E+00	1.07E+00	1.42E+00	6.73E+00	2.16E+00	=
4	1.23E−02	9.32E−02	4.43E−02	4.30E−02	1.71E−01	8.96E−02	=
5	1.65E−09	6.89E−09	2.39E−09	3.71E−02	1.05E+00	6.62E−01	=
6	5.41E−10	6.48E−09	2.54E−09	2.49E−02	5.89E−02	1.81E−02	=
7	2.83E−10	6.62E−09	3.01E−09	1.32E−05	2.12E−04	6.34E−04	=
8	4.29E−09	3.34E+00	1.83E+01	4.09E−01	2.23E+00	1.61E+00	=
9	1.00E+02	1.07E+02	3.65E+01	2.07E+01	7.80E+01	2.78E+01	+
10	3.00E+02	3.46E+02	8.65E+00	5.62E+01	2.76E+02	7.49E+01	+
				2 + 2 −			

Table 8. SHADE vs. Tb-SHADE on CEC2020 in 10*D*.

f	SHADE			Tb-SHADE			Result
	Best	**Mean**	**Std**	**Best**	**Mean**	**Std**	
1	2.12E−09	7.52E−09	2.16E−09	1.50E−09	7.38E−09	2.13E−09	=
2	1.91E−01	5.14E+00	4.17E+00	3.67E+00	2.26E+01	1.22E+01	−
3	1.05E+01	1.18E+01	7.29E−01	3.71E+00	1.03E+01	3.49E+00	+
4	9.90E−02	1.63E−01	2.75E−02	3.67E−05	2.54E−02	1.47E−02	=
5	8.82E−09	3.54E−01	1.91E−01	9.23E−07	5.48E+00	5.13E+00	−
6	7.17E−02	1.95E−01	1.01E−01	1.68E−01	5.28E−01	2.05E−01	=
7	4.09E−06	1.39E−01	1.76E−01	3.97E−02	1.83E−01	1.19E−01	=
8	1.00E+02	1.00E+02	6.24E−07	2.15E−04	1.86E+01	1.76E+01	+
9	1.00E+02	2.92E+02	8.72E+01	1.00E+02	1.00E+02	4.45E−02	+
10	3.98E+02	4.16E+02	2.29E+01	1.00E+02	1.21E+02	6.61E+01	+
				4 + 2 −			

Table 9. SHADE vs. Tb-SHADE on CEC2020 in 15*D*.

f	SHADE			Tb-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	6.48E−09	8.35E−09	9.82E−10	2.06E−09	7.43E−09	2.01E−09	=
2	1.68E−01	7.53E+00	2.16E+01	2.46E+00	3.08E+01	2.79E+01	−
3	1.56E+01	1.57E+01	2.05E−01	3.64E+00	8.46E+00	2.91E+00	+
4	1.78E−01	2.74E−01	3.69E−02	4.06E−02	7.41E−02	3.22E−02	=
5	1.31E+00	5.07E+01	5.74E+01	2.41E+01	4.89E+01	1.76E+01	−
6	7.43E−02	3.78E−01	2.22E−01	3.26E−01	2.61E+00	2.98E+00	=
7	4.18E−01	2.06E+01	4.50E+01	3.02E−01	3.07E+00	2.01E+00	+
8	1.00E+02	1.00E+02	0.00E+00	8.74E−09	5.20E+01	4.49E+01	+
9	3.38E+02	3.87E+02	9.71E+00	1.00E+02	1.46E+02	1.01E+02	+
10	4.00E+02	4.00E+02	0.00E+00	1.00E+02	2.07E+02	7.85E+01	+
				5 + 2 −			

Table 10. SHADE vs. Tb-SHADE on CEC2020 in 20*D*.

f	SHADE			Tb-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	5.87E−09	8.38E−09	9.25E−10	2.21E−09	7.94E−09	1.76E−09	=
2	8.52E−09	2.49E−01	5.14E−01	9.38E−02	1.97E+00	1.38E+00	=
3	2.04E+01	2.06E+01	3.27E−01	6.66E−09	1.46E+01	8.57E+00	+
4	2.27E−01	3.78E−01	4.93E−02	3.50E−01	4.09E−01	3.48E−02	=
5	2.06E+01	2.04E+02	8.51E+01	7.38E+00	1.56E+02	1.11E+02	+
6	9.26E−02	2.04E−01	6.62E−02	2.62E−01	3.84E−01	6.17E−02	=
7	3.55E−01	4.53E+01	5.54E+01	3.06E+00	2.49E+01	2.23E+01	+
8	1.00E+02	1.00E+02	2.23E−13	1.00E+02	1.00E+02	1.89E−13	=
9	4.01E+02	4.05E+02	2.00E+00	1.00E+02	3.99E+02	6.89E+01	+
10	4.14E+02	4.14E+02	9.83E−03	4.10E+02	4.13E+02	9.93E−01	=
				4 + 0 −			

Table 11. L-SHADE vs. TbL-SHADE on CEC2020 in 5*D*.

f	L-SHADE			TbL-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	3.32E−09	7.16E−09	1.95E−09	1.86E−09	5.06E−05	2.77E−04	=
2	6.54E−10	9.34E−02	1.14E−01	1.30E−05	4.24E−01	1.22E+00	=
3	5.15E+00	5.17E+00	6.51E−02	6.14E−01	2.96E+00	1.73E+00	+
4	9.92E−03	6.52E−02	3.08E−02	7.35E−07	1.58E−02	1.81E−02	=
5	2.24E−09	6.88E−09	2.02E−09	2.19E−09	3.60E−05	1.97E−04	=
6	8.44E−10	5.15E−09	2.84E−09	6.25E−11	6.00E−09	5.02E−09	=
7	1.51E−09	6.13E−09	2.69E−09	7.16E−10	4.76E−09	2.96E−09	=
8	5.19E−09	7.95E−09	1.44E−09	6.25E−09	2.77E−04	1.04E−03	=
9	7.81E−09	9.67E+01	1.83E+01	4.50E−09	6.61E+01	4.36E+01	+
10	3.00E+02	3.44E+02	1.20E+01	8.19E−09	2.71E+02	8.34E+01	+
				3 + 0 −			

Table 12. L-SHADE vs. TbL-SHADE on CEC2020 in 10*D*.

f	L-SHADE			TbL-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	5.88E−09	8.29E−09	1.22E−09	1.70E−09	6.65E−09	2.33E−09	=
2	3.44E−04	1.13E+00	1.53E+00	6.39E−02	7.09E+00	8.75E+00	−
3	1.04E+01	1.07E+01	2.92E−01	4.00E+00	9.40E+00	3.00E+00	+
4	9.87E−02	1.52E−01	2.23E−02	9.53E−09	1.90E−02	1.10E−02	=
5	3.83E−09	4.32E+00	2.20E+01	5.39E−02	1.60E+00	1.25E+00	=
6	2.33E−02	9.59E−02	5.58E−02	7.39E−02	3.75E−01	1.38E−01	=
7	1.06E−07	1.39E−01	2.02E−01	1.26E−06	2.14E−03	2.71E−03	=
8	7.59E−09	9.67E+01	1.83E+01	9.62E−09	3.20E+01	2.15E+01	+
9	1.00E+02	2.91E+02	8.70E+01	5.77E−09	1.02E+02	5.05E+01	+
10	3.98E+02	4.16E+02	2.29E+01	1.00E+02	1.40E+02	1.03E+02	+
				4 + 1 −			

Table 13. L-SHADE vs. TbL-SHADE on CEC2020 in 15*D*.

f	L-SHADE			TbL-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	3.53E−09	7.72E−09	1.70E−09	1.35E−09	7.01E−09	2.54E−09	=
2	3.64E−12	3.73E−01	7.99E−01	8.33E−02	1.08E+00	1.74E+00	=
3	1.56E+01	1.56E+01	1.41E−01	4.45E−09	2.55E+00	1.55E+00	+
4	2.07E−01	2.64E−01	3.91E−02	9.87E−03	4.13E−02	1.23E−02	=
5	2.46E+00	5.21E+01	6.26E+01	7.51E+00	3.40E+01	1.58E+01	+
6	2.43E−03	4.23E−01	1.52E+00	1.13E−01	1.82E+00	3.20E+00	=
7	6.63E−02	1.65E+01	4.13E+01	4.35E−01	9.54E−01	3.53E−01	+
8	1.00E+02	1.00E+02	0.00E+00	7.71E−09	6.02E+01	4.20E+01	+
9	3.00E+02	3.80E+02	2.36E+01	1.00E+02	1.80E+02	1.27E+02	+
10	4.00E+02	4.00E+02	0.00E+00	1.00E+02	2.20E+02	1.27E+02	+
				6 + 0 −			

Table 14. L-SHADE vs. TbL-SHADE on CEC2020 in 20*D*.

f	L-SHADE			TbL-SHADE			Result
	Best	Mean	Std	Best	Mean	Std	
1	4.35E−09	8.54E−09	1.33E−09	1.50E−09	7.72E−09	2.26E−09	=
2	9.46E−11	3.44E−02	3.88E−02	3.12E−02	5.21E−01	7.45E−01	=
3	2.04E+01	2.06E+01	3.93E−01	1.74E−09	2.28E+00	1.58E+00	+
4	2.57E−01	3.66E−01	3.95E−02	2.97E−02	7.35E−02	2.85E−02	=
5	3.02E+01	2.46E+02	1.02E+02	1.13E+01	2.42E+02	1.39E+02	+
6	8.46E−02	1.85E−01	7.59E−02	1.97E−01	3.13E−01	5.12E−02	=
7	2.23E−01	4.34E+01	5.64E+01	1.58E+00	4.15E+01	3.84E+01	=
8	1.00E+02	1.00E+02	1.89E−13	5.14E+01	9.40E+01	1.45E+01	+
9	4.00E+02	4.04E+02	2.37E+00	1.00E+02	4.12E+02	9.59E+01	+
10	4.14E+02	4.14E+02	1.08E−02	3.99E+02	4.03E+02	4.09E+00	+
				5 + 0 −			

Table 15. jSO vs. Tb-jSO on CEC2020 in 5*D*.

f	jSO			Tb-jSO			Result
	Best	Mean	Std	Best	Mean	Std	
1	1.34E−09	6.70E−09	2.16E−09	2.24E−09	7.69E−09	2.26E−09	=
2	9.71E−09	4.11E−01	1.24E+00	8.54E−09	2.01E+00	3.43E+00	=
3	6.13E−01	4.92E+00	1.22E+00	2.91E−08	2.34E+00	1.75E+00	=
4	8.28E−09	6.28E−02	3.35E−02	1.92E−09	3.03E−02	3.47E−02	=
5	3.14E−09	2.08E−02	1.14E−01	3.85E−09	7.48E−02	2.35E−01	=
6	1.00E−09	5.84E−09	2.36E−09	4.08E−10	6.74E−09	2.84E−09	=
7	1.57E−10	5.51E−09	2.94E−09	3.82E−11	4.59E−09	3.30E−09	=
8	4.22E−09	7.95E−09	1.63E−09	5.87E−09	8.58E−09	1.08E−09	=
9	5.55E−09	9.67E+01	1.83E+01	6.29E−09	9.33E+01	2.54E+01	=
10	3.00E+02	3.46E+02	8.65E+00	3.00E+02	3.08E+02	1.80E+01	+
				1 + 0 −			

Table 16. jSO vs. Tb-jSO on CEC2020 in 10*D*.

f	jSO			Tb-jSO			Result
	Best	Mean	Std	Best	Mean	Std	
1	4.05E−09	7.91E−09	1.39E−09	4.56E−09	8.54E−09	1.28E−09	=
2	3.12E−01	6.99E+00	4.53E+00	3.54E+00	2.03E+01	2.14E+01	−
3	1.04E+01	1.19E+01	6.21E−01	2.62E+00	8.28E+00	2.85E+00	+
4	9.86E−02	1.58E−01	3.20E−02	1.97E−02	5.51E−02	3.94E−02	=
5	6.31E−09	2.61E−01	2.94E−01	9.79E−09	1.50E+00	9.92E−01	=
6	1.95E−02	1.05E−01	8.44E−02	2.93E−02	3.63E−01	1.75E−01	=
7	6.94E−07	7.13E−02	1.60E−01	1.23E−06	4.87E−02	1.01E−01	=
8	1.00E+02	1.00E+02	0.00E+00	6.98E−09	1.45E+00	4.49E+00	+
9	1.00E+02	3.02E+02	6.89E+01	1.00E+02	1.19E+02	6.22E+01	+
10	3.98E+02	4.04E+02	1.57E+01	1.00E+02	3.88E+02	5.44E+01	+
				4 + 1 −			

Table 17. jSO vs. Tb-jSO on CEC2020 in 15*D*.

f	jSO			Tb-jSO			Result
	Best	Mean	Std	Best	Mean	Std	
1	4.88E−09	8.28E−09	1.34E−09	4.93E−09	8.62E−09	1.36E−09	=
2	4.16E−02	2.61E+01	4.47E+01	1.67E−01	2.04E+01	2.49E+01	+
3	1.56E+01	1.66E+01	5.14E−01	1.99E+00	4.75E+00	1.59E+00	+
4	1.78E−01	2.62E−01	3.76E−02	2.96E−02	9.31E−02	4.54E−02	=
5	1.15E+00	3.07E+00	2.07E+00	1.56E−01	1.13E+01	1.04E+01	−
6	3.33E−02	3.20E−01	3.15E−01	8.05E−02	2.70E−01	1.26E−01	=
7	1.24E−01	7.15E−01	2.13E−01	1.06E−01	4.85E−01	2.48E−01	=
8	1.00E+02	1.00E+02	0.00E+00	7.83E−09	5.54E+01	4.08E+01	+
9	3.86E+02	3.89E+02	8.36E−01	1.00E+02	2.55E+02	1.48E+02	+
10	4.00E+02	4.00E+02	0.00E+00	4.00E+02	4.00E+02	0.00E+00	=
				4 + 1 −			

Table 18. jSO vs. Tb-jSO on CEC2020 in 20*D*.

f	jSO			Tb-jSO			Result
	Best	Mean	Std	Best	Mean	Std	
1	5.80E−09	8.53E−09	1.32E−09	6.70E−09	9.21E−09	8.21E−10	=
2	6.25E−02	1.99E+00	1.60E+00	1.74E+00	6.29E+00	3.45E+00	=
3	2.04E+01	2.13E+01	5.24E−01	2.56E+00	5.17E+00	1.63E+00	+
4	1.97E−01	3.53E−01	4.48E−02	5.92E−02	1.21E−01	5.05E−02	=
5	1.20E+00	6.93E+00	5.07E+00	4.16E−01	3.27E+01	4.81E+01	−
6	5.63E−02	9.75E−01	4.25E−01	5.93E−02	3.11E−01	1.49E−01	=
7	5.31E−03	1.12E−01	1.10E−01	1.79E−01	3.07E+00	4.29E+00	=
8	1.00E+02	1.00E+02	8.44E−14	1.00E+02	1.00E+02	2.53E−13	=
9	3.98E+02	4.01E+02	1.36E+00	4.15E+02	4.24E+02	5.11E+00	−
10	4.14E+02	4.14E+02	4.73E−04	3.99E+02	3.99E+02	6.47E−01	+
				2 + 2 −			

Table 19. DISH and jDE100 on CEC2020 in 5*D*.

f	DISH			jDE100			j2020		
	Best	Mean	Std	Best	Mean	Std	Best	Mean	Std
1	2.60E−09	7.84E−09	1.87E−09	1.19E+05	9.94E+05	1.05E+06	0.00E+00	0.00E+00	0.00E+00
2	4.32E−09	3.99E−01	1.22E+00	1.46E+02	3.26E+02	8.04E+01	1.91E−04	3.23E+00	3.74E+00
3	6.13E−01	5.11E+00	8.63E−01	1.07E+01	2.00E+01	4.14E+00	0.00E+00	3.42E+00	2.33E+00
4	2.90E−09	6.82E−02	4.43E−02	2.69E−01	1.32E+00	4.48E−01	0.00E+00	7.68E−02	6.40E−02
5	1.83E−09	6.24E−02	1.90E−01	1.97E+01	5.80E+01	2.42E+01	0.00E+00	1.37E−01	2.86E−01
6	2.01E−09	6.94E−09	2.32E−09	4.19E−01	1.47E+00	5.28E−01	–	–	–
7	1.49E−09	5.96E−09	2.57E−09	1.95E+00	2.69E+01	2.96E+01	–	–	–
8	5.76E−09	3.35E+00	1.83E+01	4.06E+00	2.18E+01	9.07E+00	0.00E+00	6.28E−01	2.39E+00
9	1.00E+02	1.07E+02	2.54E+01	1.04E+02	1.23E+02	1.07E+01	0.00E+00	2.05E+01	3.75E+01
10	3.00E+02	3.44E+02	1.20E+01	1.89E+02	3.38E+02	3.24E+01	0.00E+00	1.26E+02	9.03E+01

Table 20. DISH and jDE100 on CEC2020 in 10*D*.

f	DISH			jDE100			j2020		
	Best	Mean	Std	Best	Mean	Std	Best	Mean	Std
1	4.26E−09	8.83E−09	1.17E−09	5.88E+07	2.80E+08	1.48E+08	0.00E+00	0.00E+00	0.00E+00
2	6.25E−02	5.26E+00	3.92E+00	7.66E+02	1.06E+03	1.13E+02	0.00E+00	6.79E−01	1.16E+00
3	1.07E+01	1.19E+01	5.71E−01	6.45E+01	8.68E+01	1.23E+01	0.00E+00	8.06E+00	3.88E+00
4	1.28E−01	1.64E−01	2.53E−02	6.08E+00	1.01E+01	2.35E+00	0.00E+00	1.09E−01	9.04E−02
5	4.96E−09	2.43E−01	1.65E−01	4.28E+03	1.38E+04	6.69E+03	0.00E+00	3.02E−01	3.13E−01
6	1.97E−02	1.61E−01	1.28E−01	3.72E+01	1.15E+02	3.93E+01	2.91E−02	4.78E−01	2.49E−01
7	1.14E−07	3.31E−02	1.09E−01	5.59E+02	2.35E+03	1.37E+03	3.10E−07	6.73E−02	1.25E−01
8	1.00E+02	1.00E+02	0.00E+00	7.55E+01	1.35E+02	2.19E+01	0.00E+00	1.54E+00	4.00E+00
9	1.00E+02	2.67E+02	1.02E+02	1.63E+02	2.24E+02	2.24E+01	0.00E+00	8.00E+01	4.07E+01
10	3.98E+02	4.07E+02	1.84E+01	4.49E+02	4.73E+02	1.26E+01	1.00E+02	1.40E+02	8.12E+01

Table 21. DISH and jDE100 on CEC2020 in 15*D*.

f	DISH			jDE100			j2020		
	Best	**Mean**	**Std**	**Best**	**Mean**	**Std**	**Best**	**Mean**	**Std**
1	4.42E−09	8.41E−09	1.42E−09	6.48E+08	1.38E+09	5.36E+08	0.00E+00	0.00E+00	0.00E+00
2	1.67E−01	2.19E+01	4.02E+01	1.49E+03	2.02E+03	2.22E+02	0.00E+00	5.72E−02	4.32E−02
3	1.56E+01	1.67E+01	5.06E−01	1.41E+02	1.84E+02	2.41E+01	0.00E+00	6.78E+00	7.82E+00
4	1.78E−01	2.60E−01	3.93E−02	1.75E+01	7.69E+01	6.76E+01	0.00E+00	1.99E−01	7.47E−02
5	1.56E−01	2.54E+00	1.17E+00	3.17E+04	2.06E+05	9.67E+04	0.00E+00	7.58E+00	7.69E+00
6	2.07E−02	2.47E−01	2.06E−01	1.70E+02	3.36E+02	7.58E+01	1.65E−03	8.45E−01	2.09E+00
7	4.38E−01	7.59E−01	1.89E−01	1.51E+04	6.11E+04	3.24E+04	6.81E−02	9.83E−01	2.03E+00
8	1.00E+02	1.00E+02	0.00E+00	1.81E+02	2.82E+02	6.30E+01	0.00E+00	9.49E+00	2.74E+01
9	3.00E+02	3.84E+02	1.88E+01	2.98E+02	4.27E+02	4.89E+01	1.00E+02	1.23E+02	5.68E+01
10	4.00E+02	4.00E+02	0.00E+00	7.07E+02	8.67E+02	8.83E+01	1.00E+02	3.90E+02	5.48E+01

Table 22. DISH and jDE100 on CEC2020 in 20*D*.

f	DISH			jDE100			j2020		
	Best	**Mean**	**Std**	**Best**	**Mean**	**Std**	**Best**	**Mean**	**Std**
1	6.06E−09	8.44E−09	9.07E−10	1.76E+09	4.18E+09	1.38E+09	0.00E+00	0.00E+00	0.00E+00
2	6.25E−02	1.50E+00	1.69E+00	2.59E+03	3.11E+03	2.27E+02	0.00E+00	2.60E−02	2.47E−02
3	2.06E+01	2.16E+01	4.69E−01	2.29E+02	3.05E+02	3.33E+01	0.00E+00	1.44E+01	9.29E+00
4	2.56E−01	3.60E−01	4.24E−02	6.79E+01	4.38E+02	3.73E+02	2.98E−02	1.80E−01	7.84E−02
5	2.08E−01	6.77E+00	3.21E+00	3.58E+05	6.91E+05	1.88E+05	3.12E−01	7.78E+01	5.75E+01
6	3.67E−02	6.97E−01	4.64E−01	3.96E+02	6.86E+02	1.33E+02	6.84E−02	1.91E−01	1.01E−01
7	1.39E−02	1.17E−01	1.04E−01	2.25E+04	1.48E+05	5.43E+04	1.95E−02	1.98E+00	4.02E+00
8	1.00E+02	1.00E+02	2.23E−13	4.53E+02	7.18E+02	1.67E+02	0.00E+00	9.27E+01	2.21E+01
9	3.96E+02	4.01E+02	1.86E+00	5.04E+02	5.89E+02	3.25E+01	1.00E+02	3.39E+02	1.28E+02
10	4.14E+02	4.14E+02	5.12E−04	5.56E+02	8.34E+02	1.46E+02	1.00E+02	3.39E+02	1.28E+02

Convergence diagrams are shown in Figures 1–12. Figures 1–4 shows the convergence curves of SHADE and Tb-SHADE, respectively, for some test functions in 5*D*, 10*D*, 15*D*, and 20*D*, Figures 5–8 shows those of L-SHADE and TbL-SHADE for some test functions in 5*D*, 10*D*, 15*D*, and 20*D*. and Figures 9–12 shows those of the jSO and Tb-jSO, respectively, for some test functions in 5*D*, 10*D*, 15*D* and 20*D*. It is apparent that the red line of the turning-based mutation version of the algorithm was often slower to converge but attained better objective function values.

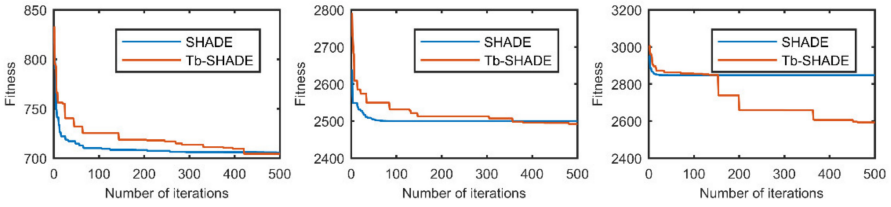

Figure 1. The selected average convergence of SHADE and Tb-SHADE on CEC2020 in 5*D* is compared. From left to right f3, f9 and f10.

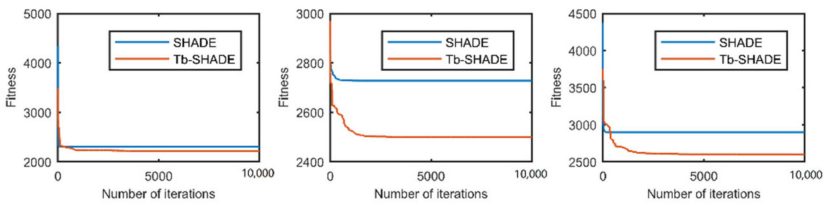

Figure 2. The selected average convergence of SHADE and Tb-SHADE is compared on CEC2020 in 10*D*. From left to right f8, f9 and f10.

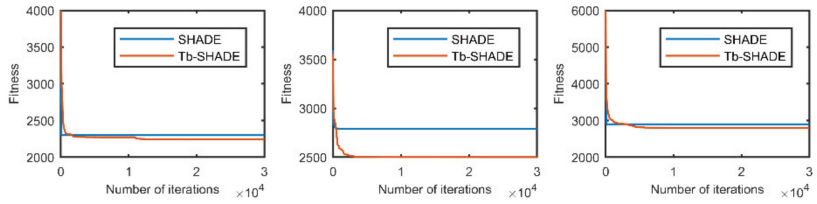

Figure 3. The selected average convergence of SHADE and Tb-SHADE is compared on CEC2020 in 15*D*. From left to right f8, f9 and f10.

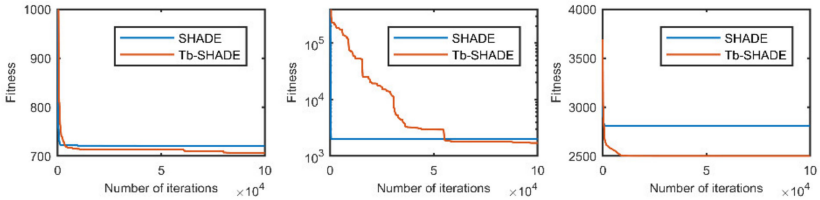

Figure 4. The selected average convergence of SHADE and Tb-SHADE is compared on CEC2020 in 20*D*. From left to right f3, f5 and f9.

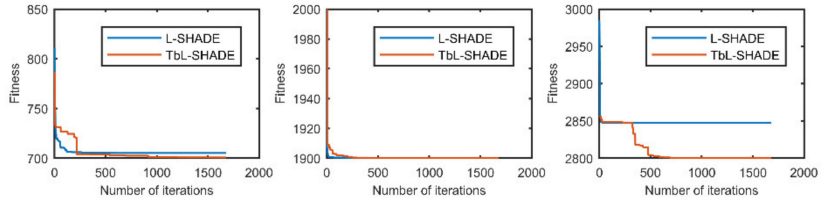

Figure 5. The selected average convergence of L-SHADE and TbL-SHADE is compared on CEC2020 in 5*D*. From left to right f3, f4 and f10.

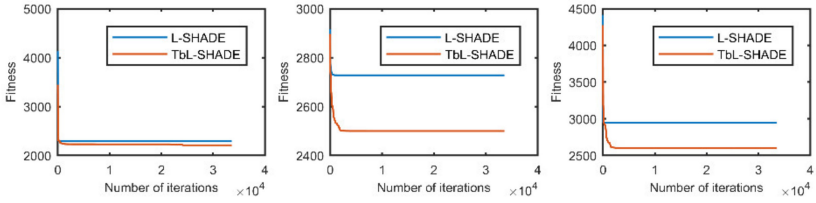

Figure 6. The selected average convergence of L-SHADE and TbL-SHADE is compared on CEC2020 in 10*D*. From left to right f8, f9 and f10.

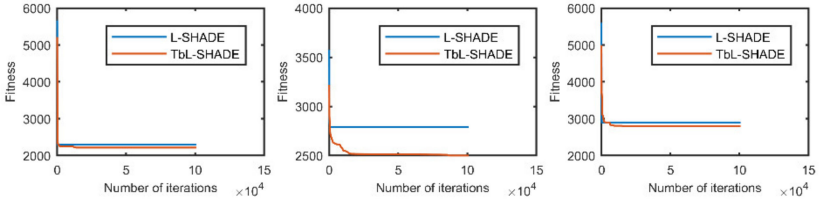

Figure 7. The selected average convergence of L-SHADE and TbL-SHADE is compared on CEC2020 in 15*D*. From left to right f8, f9 and f10.

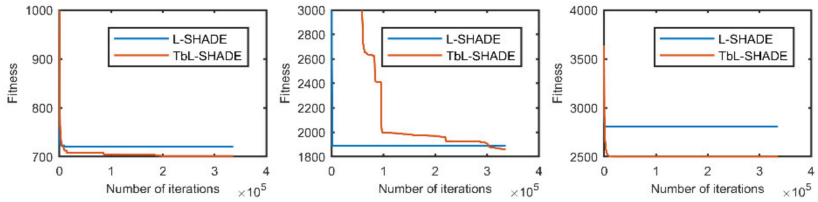

Figure 8. The selected average convergence of L-SHADE and TbL-SHADE is compared on CEC2020 in 20*D*. From left to right f3, f5 and f9.

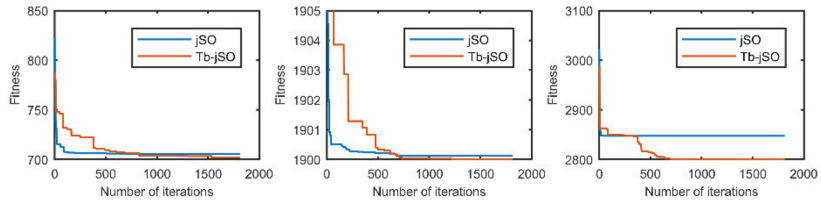

Figure 9. The selected average convergence of jSO and Tb-jSO is compared on CEC2020 in 5*D*. From left to right f3, f4 and f10.

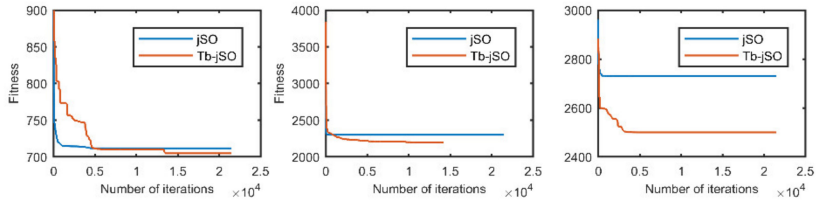

Figure 10. The selected average convergence of jSO and Tb-jSO is compared on CEC2020 in 10*D*. From left to right f3, f8 and f9.

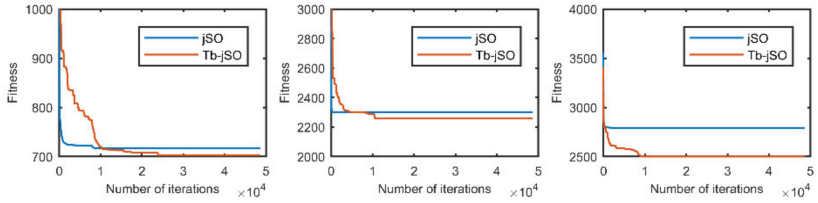

Figure 11. The selected average convergence of jSO and Tb-jSO is compared on CEC2020 in 15*D*. From left to right f3, f8 and f9.

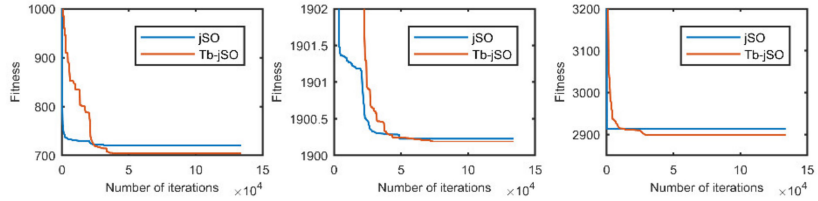

Figure 12. The selected average convergence of jSO and Tb-jSO is compared on CEC2020 in 20*D*. From left to right f3, f4 and f10.

Tables 23–34 shows the number of runs (#runs) of population aggregation, the average generation (Mean CO) of the first cluster during these runs, and the average population diversity (Mean PD) of these generations.

Table 23. Clustering and population diversity of SHADE and Tb-SHADE on the CEC2020 in 5*D*.

f	SHADE			Tb-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	3.04E+01	3.74E+01	30	2.29E+02	4.84E+01
2	12	3.97E+02	7.29E+01	0	–	–
3	12	4.01E+02	1.20E+01	0	–	–
4	13	2.83E+02	1.16E+01	0	–	–
5	30	9.24E+01	3.69E+01	30	4.40E+02	3.66E+01
6	30	1.13E+02	3.29E+01	30	4.61E+02	2.28E+01
7	30	7.66E+01	4.57E+01	30	3.63E+02	4.41E+01
8	30	6.46E+01	2.84E+01	29	4.42E+02	2.52E+01
9	30	7.91E+01	6.31E+01	30	4.13E+02	3.61E+01
10	30	3.57E+01	1.30E+01	30	3.87E+02	3.03E+01

Table 24. Clustering and population diversity of SHADE and Tb-SHADE on the CEC2020 in 10*D*.

f	SHADE			Tb-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	5.84E+01	1.88E+01	30	6.57E+02	8.30E+01
2	0	–	–	0	–	–
3	3	7.94E+03	1.35E+01	0	–	–
4	0	–	–	0	–	–
5	30	7.14E+02	3.92E+01	16	8.18E+03	8.16E+01
6	0	–	–	3	8.76E+03	8.21E+01
7	30	8.78E+02	2.36E+01	14	9.54E+03	3.66E+01
8	30	5.06E+01	1.48E+01	30	8.97E+02	1.39E+02
9	2	5.12E+03	3.09E+01	23	2.41E+03	1.12E+02
10	30	1.03E+02	2.37E+01	29	2.72E+03	7.69E+01

Table 25. Clustering and population diversity of SHADE and Tb-SHADE on the CEC2020 in 15*D*.

f	SHADE			Tb-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	7.08E+01	1.37E+01	30	8.93E+02	1.05E+02
2	0	–	–	0	–	–
3	30	1.65E+04	9.53E+00	0	–	–
4	0	–	–	0	–	–
5	29	8.34E+02	3.33E+01	0	–	–
6	3	1.44E+04	6.27E+01	0	–	–
7	30	6.96E+02	1.56E+01	6	2.72E+04	6.18E+01
8	30	6.59E+01	1.12E+01	30	1.10E+03	2.16E+02
9	25	1.52E+04	3.12E+01	2	8.86E+03	9.17E+01
10	30	1.12E+02	6.72E+00	30	3.22E+03	1.13E+02

Table 26. Clustering and population diversity of SHADE and Tb-SHADE on the CEC2020 in 20*D*.

f	SHADE			Tb-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	7.64E+01	1.19E+01	30	1.26E+03	8.92E+01
2	4	9.06E+04	7.74E+01	0	–	–
3	30	3.34E+04	9.12E+00	0	–	–
4	0	–	–	0	–	–
5	30	5.46E+02	1.86E+01	12	8.62E+04	1.27E+02
6	0	–	–	0	-	-
7	30	9.20E+02	2.59E+01	4	8.96E+04	4.63E+01
8	30	8.56E+01	1.06E+01	30	1.72E+03	2.82E+02
9	0	–	–	0	-	-
10	30	1.02E+02	6.21E+00	30	2.63E+03	1.26E+02

Table 27. Clustering and population diversity of L-SHADE and TbL-SHADE on the CEC2020 in 5*D*.

f	L-SHADE			TbL-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	3.00E+01	3.75E+01	30	2.53E+02	4.85E+01
2	18	8.71E+02	3.20E+01	6	1.38E+03	1.66E+01
3	30	6.29E+02	7.01E+00	1	1.21E+03	6.85E+00
4	10	4.65E+02	1.06E+01	0	–	–
5	30	8.75E+01	3.42E+01	30	1.10E+03	2.17E+01
6	30	1.14E+02	3.01E+01	30	7.34E+02	2.30E+01
7	30	7.66E+01	4.29E+01	30	4.85E+02	4.13E+01
8	30	6.47E+01	2.08E+01	30	7.59E+02	1.97E+01
9	30	7.11E+01	6.15E+01	30	3.90E+02	2.77E+01
10	30	3.52E+01	1.07E+01	30	3.71E+02	2.93E+01

Table 28. Clustering and population diversity of L-SHADE and TbL-SHADE on the CEC2020 in 10*D*.

f	L-SHADE			TbL-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	5.96E+01	2.01E+01	30	6.61E+02	8.21E+01
2	10	2.88E+04	2.98E+01	9	3.12E+04	2.12E+01
3	20	2.65E+04	4.36E+00	2	3.19E+04	3.52E+00
4	1	2.86E+04	8.35E+00	2	3.21E+04	1.97E+00
5	30	7.31E+02	4.15E+01	15	2.89E+04	1.58E+01
6	6	2.31E+04	1.10E+01	0	–	–
7	30	9.46E+02	2.34E+01	30	2.30E+04	1.77E+01
8	30	5.75E+01	1.51E+01	30	8.86E+02	1.26E+02
9	10	1.63E+04	5.77E+01	26	6.57E+03	1.18E+02
10	30	1.08E+02	2.70E+01	28	4.24E+03	7.02E+01

Table 29. Clustering and population diversity of L-SHADE and TbL-SHADE on the CEC2020 in 15*D*.

f	L-SHADE			TbL-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	7.06E+01	1.45E+01	30	8.80E+02	1.05E+02
2	30	6.19E+04	4.25E+01	30	7.69E+04	3.52E+01
3	30	1.84E+04	7.97E+00	24	8.51E+04	2.14E+01
4	4	9.42E+04	6.03E+00	3	9.80E+04	1.64E+00
5	28	8.35E+02	3.80E+01	2	9.80E+04	1.21E+01
6	18	7.04E+04	1.09E+01	17	9.20E+04	2.24E+01
7	30	6.98E+02	1.54E+01	0	–	–
8	30	6.60E+01	1.15E+01	30	1.10E+03	2.16E+02
9	30	2.00E+04	2.82E+01	14	7.38E+04	1.05E+02
10	30	1.14E+02	6.87E+00	30	3.13E+03	1.08E+02

Table 30. Clustering and population diversity of L-SHADE and TbL-SHADE on the CEC2020 in 20*D*.

f	L-SHADE			TbL-SHADE		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	7.64E+01	1.22E+01	30	1.26E+03	8.95E+01
2	26	9.06E+04	6.00E+01	30	1.79E+05	6.37E+01
3	30	3.61E+04	8.72E+00	30	2.21E+05	4.83E+01
4	17	3.00E+05	1.14E+01	9	3.03E+05	6.49E+00
5	30	5.46E+02	1.63E+01	19	2.66E+05	7.85E+01
6	17	2.59E+05	1.75E+01	11	2.94E+05	1.89E+01
7	30	8.89E+02	1.92E+01	7	3.24E+05	1.18E+01
8	30	8.64E+01	8.66E+00	30	1.72E+03	2.80E+02
9	19	2.37E+05	3.59E+01	26	2.19E+05	1.11E+02
10	30	1.04E+02	6.78E+00	30	2.52E+03	1.26E+02

Table 31. Clustering and population diversity of jSO and Tb-jSO on the CEC2020 in 5*D*.

f	jSO			Tb- jSO		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	3.17E+01	4.57E+01	30	3.17E+02	4.18E+01
2	29	9.61E+02	4.16E+01	26	1.22E+03	4.92E+01
3	30	9.12E+02	9.36E+00	19	1.43E+03	2.33E+01
4	29	1.08E+03	8.68E+00	22	1.44E+03	1.39E+01
5	30	1.17E+02	3.36E+01	30	9.94E+02	2.81E+01
6	30	1.57E+02	3.68E+01	30	9.66E+02	1.54E+01
7	30	1.06E+02	4.38E+01	29	6.79E+02	3.89E+01
8	30	7.21E+01	1.74E+01	30	7.20E+02	2.13E+01
9	30	5.49E+01	6.52E+01	30	3.94E+02	2.70E+01
10	30	3.25E+01	9.66E+00	30	3.90E+02	2.97E+01

Table 32. Clustering and population diversity of jSO and Tb-jSO on the CEC2020 in 10*D*.

f	jSO			Tb-jSO		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	5.88E+01	3.93E+01	30	2.29E+03	7.97E+01
2	30	6.11E+03	1.12E+02	30	6.86E+03	1.65E+02
3	30	6.17E+03	1.48E+01	30	1.04E+04	4.81E+01
4	30	8.55E+03	1.28E+01	30	1.34E+04	2.44E+01
5	30	4.41E+03	3.63E+01	30	8.97E+03	5.36E+01
6	30	5.54E+03	2.38E+01	30	1.07E+04	5.42E+01
7	30	1.44E+03	3.57E+01	30	9.69E+03	3.94E+01
8	30	5.52E+01	9.59E+00	30	3.40E+03	5.53E+01
9	30	4.98E+03	4.19E+01	30	3.73E+03	4.66E+01
10	30	8.33E+01	2.45E+01	30	3.07E+03	6.99E+01

Table 33. Clustering and population diversity of jSO and Tb-jSO on the CEC2020 in 15*D*.

f	jSO			Tb- jSO		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	8.09E+01	4.39E+01	30	5.34E+03	1.02E+02
2	30	1.39E+04	1.21E+02	30	1.54E+04	1.66E+02
3	30	1.20E+04	1.37E+01	30	2.02E+04	5.26E+01
4	30	1.69E+04	1.57E+01	30	2.99E+04	2.84E+01
5	30	1.17E+04	4.95E+01	30	1.31E+04	1.24E+02
6	30	1.79E+04	2.23E+01	30	2.41E+04	5.03E+01
7	30	2.53E+03	3.56E+01	30	2.26E+04	3.76E+01
8	30	7.50E+01	7.39E+00	30	3.53E+03	4.65E+01
9	30	1.07E+04	3.03E+01	30	1.83E+04	4.07E+01
10	30	9.39E+01	6.56E+00	30	8.49E+03	8.97E+01

Table 34. Clustering and population diversity of jSO and Tb-jSO on the CEC2020 in 20*D*.

f	jSO			Tb-jSO		
	#runs	Mean CO	Mean PD	#runs	Mean CO	Mean PD
1	30	9.32E+01	3.60E+01	30	9.38E+03	9.36E+01
2	30	3.15E+04	1.17E+02	30	3.48E+04	1.69E+02
3	30	2.91E+04	1.37E+01	30	4.65E+04	6.03E+01
4	30	3.86E+04	1.80E+01	30	7.28E+04	3.40E+01
5	30	2.94E+04	6.35E+01	30	3.25E+04	1.28E+02
6	30	3.81E+04	2.77E+01	30	4.80E+04	9.32E+01
7	30	3.06E+04	1.92E+01	30	4.83E+04	6.16E+01
8	30	9.09E+01	7.40E+00	30	7.11E+03	4.78E+01
9	30	2.84E+04	4.81E+01	30	3.68E+04	1.08E+02
10	30	9.94E+01	6.62E+00	30	1.08E+04	1.27E+02

The rankings of the Friedman test [52] were obtained by using the average value (Mean) of each algorithm on all 10 test functions in Tables 7–22, and are shown in Tables 35–38. The related statistical values of the Friedman test are shown in Table 39. If the chi-square statistic was greater than the critical value, the null hypothesis was rejected. p represents the probability of the null hypothesis obtaining. The null hypothesis here was that there is no significant difference in performance among the nine algorithms considered here on CEC2020.

Table 35. The Friedman ranks of comparative algorithms on CEC2020 in 5D.

Rank	Name	F-Rank
0	TbL-SHADE	3.2
1	Tb-jSO	3.55
2	L-SHADE	3.8
3	jSO	4.05
4	j2020	4.95
5	DISH	5.05
6	SHADE	5.2
7	Tb-SHADE	6.6
8	jDE100	8.6

Table 36. The Friedman ranks of comparative algorithms on CEC2020 in 10D.

Rank	Name	F-Rank
0	j2020	3.2
1	Tb-jSO	3.6
2	TbL-SHADE	3.9
3	DISH	4.9
4	jSO	4.95
5	Tb-SHADE	5.05
6	L-SHADE	5.05
7	SHADE	5.75
8	jDE100	8.6

Table 37. The Friedman ranks of comparative algorithms on CEC2020 in 15D.

Rank	Name	F-Rank
0	j2020	3.15
1	TbL-SHADE	3.45
2	Tb-jSO	3.45
3	Tb-SHADE	4.35
4	DISH	4.7
5	jSO	5.2
6	L-SHADE	5.6
7	SHADE	6.1
8	jDE100	9

Table 38. The Friedman ranks of comparative algorithms on CEC2020 in 20D.

Rank	Name	F-Rank
0	j2020	2.35
1	TbL-SHADE	4.05
2	Tb-jSO	4.3
3	DISH	4.8
4	jSO	4.9
5	Tb-SHADE	5
6	L-SHADE	5.1
7	SHADE	5.5
8	jDE100	9

Table 39. Related statistical values obtained of Friedman test for $\alpha = 0.05$.

D	Chi-sq′	Prob > Chi-sq′ (p)	Critical Value·
5	34.25414365	3.65E−05	15.51
10	28.83468835	3.39E−04	15.51
15	38.8213628	5.31E−06	15.51
20	37.00654818	1.15E−05	15.51

6. Results and Discussion

The results on the CEC2020 benchmark sets are first discussed. As shown in Tables 7–18, the scores were two improvements against two instances of worsening (5D), four improvements and two instances of worsening (10D), five improvements and two instances of worsening (15D), and four improvements no instances of worsening (20D) in the case of SHADE; three improvements against zero instances of worsening (5D), four improvements and one worsening (10D), six improvements no worsening (15D), and five improvements and no worsening (20D) in the case of L-SHADE; and one improvement against no worsening (5D), four improvements and one worsening (10D), four improvements and one worsening (15D), and two improvements two instances of worsening (20D) in the case of jSO. In some test functions, the improved algorithm even escaped the local optimum and found the optimal value (if the error was smaller than 10^{-8}, the relevant value was considered optimal). Examples are f3 in Tables 10 and 13, and Table 14, f8 in Tables 9, 13 and 16, and Table 17, f9 in Table 12, and f10 in Table 11. In most cases, the improved version was clearly better than the original algorithm except for Tb-SHADE (5D) and Tb-jSO (20D).

According to the convergence curves in Figures 1–12, in most cases, the improved algorithm showed similar convergence to the original in the early stage of the optimization process, but it clearly maintained a longer exploration phase and achieved better values of the objective function in the middle and late stages; in a few cases (such as $f4$ in Figure 5), the improved algorithm had slower convergence but did not achieve a better objective function value than the original.

As the numerical analyses in Tables 23–34 show, in most cases, the improved algorithms exhibited fewer clusters (#runs), later clustering (mean CO), and higher population density (mean PD) than the original algorithm. But Tb-SHADE (5D) had a lower population density on $f6$–$f9$, as did TbL-SHADE (all dimensions) on $f2$–$f7$, where this might have been related to the linear decrease in the population size. Tb-jSO showed similar numbers of clusters in all dimensions and a lower population density on some test functions in 5D. Therefore, in most cases, the improved versions maintained the diversity of population and a longer exploration phase in the optimization process.

The significant improvements in Tables 7–18 and the clustering analysis in Tables 23 and 24 can be linked. The results in the former set of tables with the "+" symbol were always connected with the occurrence of later clustering, none at all, or fewer instances of clusters of 30 (for the last option, see, for example, column #runs in Tables 24–26, $f3$). Consequently, the improvement in the performance effected by the updated version was related to the maintenance of population diversity and a longer exploration phase.

According to the Friedman ranking in Tables 35–38, Tb-SHADE, TbL-SHADE, and Tb-jSO were clearly better than the original algorithms and the advanced DISH and jDE100 in 10D, 15D, and 20D. But Tb-SHADE did not perform as well as SHADE in 5D and did not perform as well as DISH in 5D, 10D and 20D. In addition, the j2020 algorithm delivered the best performance and ranked first in 10D, 15D and 20D and one of the improved versions, TbL-SHADE, only delivered the best performance and ranked first in 5D. And jDE100 (winner of CEC2019), which ranks last in Tables 35–38, did not seem suitable for CEC2020. Table 39 shows that the null hypothesis was rejected in all dimensions, and thus the Friedman ranking was correct. All in all, the three improved algorithms obtained good optimization results in contrast to the original algorithm as well as the advanced DISH and jDE100 algorithms but were slightly worse than the advanced j2020 algorithm.

7. Conclusions

In this paper, a relatively simple and direct method using turning-based mutation was proposed and tested on Single Objective Bound Constrained Numerical Optimization (CEC2020) benchmark sets in 5, 10, 15, and 20 dimensions against the SHADE, L-SHADE, and jSO algorithms. The basic thought of the proposed method is to change the direction of mutation under certain conditions to maintain the population diversity and a longer exploration phase. It can thus avoid premature convergence and escape the local optimum to get better optimization results. The results of experiments showed that this method is effective on CEC2020 benchmark sets in 10, 15, and 20 dimensions. The strong point of the proposed method is that it can be applied to variants of SHADE easily. A disadvantage is that it increases the time complexity and its effectiveness lacks theoretical proof. Our future research in the area will focus on further experiments, and on applying the proposed method to more algorithms. For example, the improved method may be useful for some practical problems featuring constraints.

Author Contributions: Conceptualization, H.K.; methodology, H.K.; project administration, L.J. and Y.S.; software, X.S.; validation, X.S. and Q.C.; visualization, L.J. and Q.C.; formal analysis, H.K.; investigation, Q.C.; resources, Y.S.; data curation, L.J.; writing—original draft preparation, L.J.; writing—review and editing, H.K. and X.S.; supervision: X.S.; funding acquisition: Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China, grant number 61663046, 61876166. This research was funded by Open Foundation of Key Laboratory of Software Engineering of Yunnan Province, grant number 2015SE204.

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

References

1. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
2. Eltaeib, T.; Mahmood, A. Differential evolution: A survey and analysis. *Appl. Sci.* **2018**, *8*, 1945. [CrossRef]
3. Arafa, M.; Sallam, E.A.; Fahmy, M. An enhanced differential evolution optimization algorithm. In Proceedings of the 2014 Fourth International Conference on Digital Information and Communication Technology and its Applications (DICTAP), Bangkok, Thailand, 6–8 May 2014; pp. 216–225.
4. Awad, N.H.; Ali, M.Z.; Suganthan, P.N. Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastian, Spain, 5–8 June 2017; pp. 372–379.
5. Bujok, P.; Tvrdík, J. Adaptive differential evolution: SHADE with competing crossover strategies. In Proceedings of the International Conference on Artificial Intelligence and Soft Computing, Zakopane, Poland, 14–18 June 2015; pp. 329–339.
6. Bujok, P.; Tvrdík, J.; Poláková, R. Evaluating the performance of shade with competing strategies on CEC 2014 single-parameter test suite. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 5002–5009.
7. Liu, X.-F.; Zhan, Z.-H.; Zhang, J. Dichotomy guided based parameter adaptation for differential evolution. In Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, Madrid, Spain, 11–15 July 2015; pp. 289–296.
8. Liu, Z.-G.; Ji, X.-H.; Yang, Y. Hierarchical differential evolution algorithm combined with multi-cross operation. *Expert Syst. Appl.* **2019**, *130*, 276–292. [CrossRef]
9. Mohamed, A.W.; Hadi, A.A.; Fattouh, A.M.; Jambi, K.M. LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastian, Spain, 5–8 June 2017; pp. 145–152.
10. Poláková, R.; Tvrdík, J.; Bujok, P. L-SHADE with competing strategies applied to CEC2015 learning-based test suite. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 4790–4796.

11. Poláková, R.; Tvrdík, J.; Bujok, P. Evaluating the performance of L-SHADE with competing strategies on CEC2014 single parameter-operator test suite. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 1181–1187.
12. Sallam, K.M.; Sarker, R.A.; Essam, D.L.; Elsayed, S.M. Neurodynamic differential evolution algorithm and solving CEC2015 competition problems. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, 25–28 May 2015; pp. 1033–1040.
13. Viktorin, A.; Pluhacek, M.; Senkerik, R. Network based linear population size reduction in SHADE. In Proceedings of the 2016 International Conference on Intelligent Networking and Collaborative Systems (INCoS), Ostrawva, Czech Republic, 7–9 September 2016; pp. 86–93.
14. Viktorin, A.; Pluhacek, M.; Senkerik, R. Success-history based adaptive differential evolution algorithm with multi-chaotic framework for parent selection performance on CEC2014 benchmark set. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 4797–4803.
15. Viktorin, A.; Senkerik, R.; Pluhacek, M.; Kadavy, T. Distance vs. Improvement Based Parameter Adaptation in SHADE. In *Artificial Intelligence and Algorithms in Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 455–464.
16. Viktorin, A.; Senkerik, R.; Pluhacek, M.; Kadavy, T.; Zamuda, A. Distance based parameter adaptation for differential evolution. In Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, USA, 27 November–1 December 2017; pp. 1–7.
17. Zhao, F.; He, X.; Yang, G.; Ma, W.; Zhang, C.; Song, H. A hybrid iterated local search algorithm with adaptive perturbation mechanism by success-history based parameter adaptation for differential evolution (SHADE). *J. Eng. Optim.* **2020**, *52*, 367–383. [CrossRef]
18. Al-Dabbagh, R.D.; Neri, F.; Idris, N.; Baba, M.S. Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy. *Swarm Evol. Comput.* **2018**, *43*, 284–311. [CrossRef]
19. Tanabe, R.; 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; pp. 71–78.
20. Zhang, J.; Sanderson, A.C. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
21. 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; pp. 1658–1665.
22. Guo, S.-M.; Tsai, J.S.-H.; Yang, C.-C.; Hsu, P.-H. A self-optimization approach for L-SHADE incorporated with eigenvector-based crossover and successful-parent-selecting framework on CEC 2015 benchmark set. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, 25–28 May 2015; pp. 1003–1010.
23. Awad, N.H.; Ali, M.Z.; Suganthan, P.N.; Reynolds, R.G. An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 2958–2965.
24. 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 Sebastian, Spain, 5–8 June 2017; pp. 1311–1318.
25. Piotrowski, A.P.; Napiorkowski, J.J. Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure? *Swarm Evol. Comput.* **2018**, *43*, 88–108. [CrossRef]
26. Piotrowski, A.P. L-SHADE optimization algorithms with population-wide inertia. *Inf. Sci.* **2018**, *468*, 117–141. [CrossRef]
27. Stanovov, V.; Akhmedova, S.; Semenkin, E. LSHADE algorithm with rank-based selective pressure strategy for solving CEC 2017 benchmark problems. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
28. Brest, J.; Maučec, M.S.; Bošković, B. The 100-Digit Challenge: Algorithm jDE100. In Proceedings of the 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 10–13 June 2019; pp. 19–26.
29. Brest, J.; Maučec, M.S.; Bošković, B. Differential Evolution Algorithm for Single Objective Bound-Constrained Optimization: Algorithm j2020. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020; pp. 1–8.

30. Suganthan, P.N. Particle swarm optimiser with neighbourhood operator. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999; pp. 1958–1962.
31. Das, S.; Maity, S.; Qu, B.-Y.; Suganthan, P.N. Real-parameter evolutionary multimodal optimization—A survey of the state-of-the-art. *Swarm Evol. Comput.* **2011**, *1*, 71–88. [CrossRef]
32. Mezura-Montes, E.; Coello, C.A.C. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm Evol. Comput.* **2011**, *1*, 173–194. [CrossRef]
33. Neri, F.; Cotta, C. Memetic algorithms and memetic computing optimization: A literature review. *Swarm Evol. Comput.* **2012**, *2*, 1–14. [CrossRef]
34. Neri, F.; Tirronen, V. Recent advances in differential evolution: A survey and experimental analysis. *Artif. Intell. Rev.* **2010**, *33*, 61–106. [CrossRef]
35. Zamuda, A.; Brest, J. Self-adaptive control parameters' randomization frequency and propagations in differential evolution. *Swarm Evol. Comput.* **2015**, *25*, 72–99. [CrossRef]
36. Zhou, A.; Qu, B.-Y.; Li, H.; Zhao, S.-Z.; Suganthan, P.N.; Zhang, Q. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm Evol. Comput.* **2011**, *1*, 32–49. [CrossRef]
37. Piotrowski, A.P. Review of differential evolution population size. *Swarm Evol. Comput.* **2017**, *32*, 1–24. [CrossRef]
38. Opara, K.R.; Arabas, J. Differential Evolution: A survey of theoretical analyses. *Swarm Evol. Comput.* **2019**, *44*, 546–558. [CrossRef]
39. Poikolainen, I.; Neri, F.; Caraffini, F. Cluster-based population initialization for differential evolution frameworks. *Inf. Sci.* **2015**, *297*, 216–235. [CrossRef]
40. Weber, M.; Neri, F.; Tirronen, V. A study on scale factor/crossover interaction in distributed differential evolution. *Artif. Intell. Rev.* **2013**, *39*, 195–224. [CrossRef]
41. Zaharie, D. Influence of crossover on the behavior of differential evolution algorithms. *Appl. Soft Comput.* **2009**, *9*, 1126–1138. [CrossRef]
42. Yue, C.; Price, K.; Suganthan, P.; Liang, J.; Ali, M.; Qu, B.; Awad, N.; Biswas, P. Problem definitions and evaluation criteria for the CEC 2020 special session and competition on single objective bound constrained numerical optimization. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020.
43. Piotrowski, A.P.; Napiorkowski, J.J. Some metaheuristics should be simplified. *Inf. Sci.* **2018**, *427*, 32–62. [CrossRef]
44. Viktorin, A.; Senkerik, R.; Pluhacek, M.; Kadavy, T.; Zamuda, A. Distance based parameter adaptation for success-history based differential evolution. *Swarm Evol. Comput.* **2019**, *50*, 100462. [CrossRef]
45. Caraffini, F.; Kononova, A.V.; Corne, D. Infeasibility and structural bias in differential evolution. *Inf. Sci.* **2019**, *496*, 161–179. [CrossRef]
46. Awad, N.; Ali, M.; Liang, J.; Qu, B.; Suganthan, P.; Definitions, P. Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastian, Spain, 5–8 June 2017.
47. Brest, J.; Maučec, M.S.; Bošković, B. iL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 1188–1195.
48. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M. Opposition-based differential evolution. *IEEE Trans. Evol. Comput.* **2008**, *12*, 64–79. [CrossRef]
49. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD-96 Proceedings*; AAAI: Menlo Park, CA, USA, 1996; pp. 226–231.
50. Deza, M.M.; Deza, E. Encyclopedia of distances. In *Encyclopedia of Distances*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 1–583.
51. Poláková, R.; Tvrdík, J.; Bujok, P.; Matoušek, R. Population-size adaptation through diversity-control mechanism for differential evolution. In Proceedings of the MENDEL, 22th International Conference on Soft Computing, Brno, Czech Republic, 8–10 June 2016; pp. 49–56.
52. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.