

Article

A Multi-Strategy Differential Evolution Algorithm with Adaptive Similarity Selection Rule

Liming Zheng and Yinan Wen *

College of Information Science and Technology, Jinan University, Guangzhou 510632, China

* Correspondence: nicky1567@stu2020.jnu.edu.cn

Abstract: The differential evolution (DE) algorithm is a simple and efficient population-based evolutionary algorithm. In DE, the mutation strategy and the control parameter play important roles in performance enhancement. However, single strategy and fixed parameter are not universally applicable to problems and evolution stages with diverse characteristics; besides, the weakness of the advanced DE optimization framework, called selective-candidate framework with similarity selection rule (SCSS), is found by focusing on its single strategy and fixed parameter greedy degree (GD) setting. To address these problems, we mainly combine the multiple candidates generation with multi-strategy (MCG-MS) and the adaptive similarity selection (ASS) rule. On the one hand, in MCG-MS, two symmetrical mutation strategies, “DE/current-to-pbest-w/1” and designed “DE/current-to-cbest-w/1”, are utilized to build the multi-strategy to produce two candidate individuals, which prevents the over-approximation of the candidate in SCSS. On the other hand, the ASS rule provides the individual selection mechanism for multi-strategy to determine the offspring from two candidates, where parameter *GD* is designed to increase linearly with evolution to maintain diversity at the early evolution stage and accelerate convergence at the later evolution stage. Based on the advanced algorithm jSO, replacing its offspring generation strategy with the combination of MCG-MS and ASS rule, this paper proposes multi-strategy differential evolution algorithm with adaptive similarity selection rule (MSDE-ASS). It combines the advantages of two symmetric strategies and has an efficient individual selection mechanism without parameter adjustment. MSDE-ASS is verified under the Congress on Evolutionary Computation (CEC) 2017 competition test suite on real-parameter single-objective numerical optimization, and the results indicate that, of the 174 cases in total, it wins in 81 cases and loses in 30 cases, and it has the smallest performance ranking value, of 3.05. Therefore, MSDE-ASS stands out compared to the other state-of-the-art DEs.

Keywords: evolutionary computation; differential evolution; multi-strategy; adaptive parameter; similarity selection



Citation: Zheng, L.; Wen, Y. A Multi-Strategy Differential Evolution Algorithm with Adaptive Similarity Selection Rule. *Symmetry* **2023**, *15*, 1697. <https://doi.org/10.3390/sym15091697>

Academic Editor: Theodore E. Simos

Received: 13 July 2023

Revised: 30 August 2023

Accepted: 31 August 2023

Published: 4 September 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Differential evolution (DE), proposed by Storn in 1996 [1], is a representative population-based evolutionary algorithm (EA). Similar to another representative EA, genetic algorithm (GA) [2,3], DE includes initialization, mutation, crossover, and selection. However, its mutant vector is generated by the parent difference vector in mutation and crossed with the parent individual vector to generate offspring in crossover, which is distinguished from GA. Due to the simple and efficient characteristics, it is common for DE to apply to optimization problems in various scientific and engineering fields, such as protein structure prediction [4], machine scheduling [5], electromagnetic devices optimization [6], combinatorial [7] and constrained multiobjective optimization [8]. In DE, the mutation strategy mainly searches individuals to determine the evolution direction, and the control parameter is mainly to adjust the evolution scale. These two aspects affect the convergence and diversity of the algorithm, which play important roles in performance enhancement. So, researchers usually focus on the improvement of the mutation strategy and the control parameter to propose the DE variants.

On the one hand, in addition to the mutation strategy of the original DE, “DE/rand/1” [1], there are some other mutation strategies, such as “DE/best/1” [1], “DE/current-to-best/1” [9], “DE/rand-to-best/1” [10], and “DE/current-to-pbest/1” [11]. On the other hand, some parameter setting suggestions were proposed in the past [1,12,13]. For example, Storn [1] suggested that the value of the scaling factor (F) is set to 0.5, and the population size (NP) should be between $5 \cdot D$ and $10 \cdot D$, where D is the dimension of the individual. According to Storn and Price [12] and Liu and Lampinen [13], a suitable range of F and CR are in the range $[0.5, 1]$ and $[0.8, 1]$, respectively. However, since different mutation strategies and parameter settings cause the algorithm to show different features, different problems and evolutionary stages with diverse characteristics cannot be well met by single strategy and fixed parameter setting. To handle this problem, multi-strategy and adaptive parameter setting have been introduced by researchers.

With the extensive study, researchers proposed many DE variants by considering the adaptation of the control parameter. Table 1 shows the nomenclature of these variants. Liu and Lampinen [14] proposed fuzzy adaptive differential evolution (FADE), which utilizes fuzzy logic controllers to adapt the control parameters F and CR . jDE and JADE, the former proposed by Brest et al. [15] and the latter proposed by Zhang and Sanderson [11], are both DEs with control parameter adaption mechanism. In [11,15], some formulas designed in them achieve the adaptive control parameters F and CR . SaDE, proposed by Qin et al. [16], also has the adaptation of the control parameter, which causes DE to choose the most appropriate control parameters, F and CR , in different stages of evolution. A two-level approach [17] in a novel DE proposed by Yu et al. estimates the optimization state to determine F and CR at the population level and further adjusts F and CR at the individual level. Tanabe and Fukunaga [18] designed a parameter adaptation technique to propose SHADE which allocates parameters based on historical successful evolution experience. Based on SHADE [18], by exploring the control parameter NP , Tanabe and Fukunaga [19] further designed an adaptive mechanism, called linear population size reduction (LPSR), which continually decreases NP through a linear function. Poláková et al. [20] achieved the adaptation of the parameter population size NP . This proposed mechanism is able to adaptively adjust the NP during the search with the linear reduction of the population diversity. A novel parameter with adaptive learning mechanism (PALM) for the enhancement of DE was proposed by Meng et al. [21]. In PALM, the adaption for CR is based on its success probability, and the adaptive learning for F is based on its success information as well as the associated fitness values. Adam et al. [22] modified the F and CR adaptation in SHADE to address the problem of premature convergence. Focusing on LSHADE [19] and distance-based Db-LSHADE algorithms in [22], the generalized Lehmer mean and linear bias reduction were proposed by Vladimir et al. [23] to control the parameter adaptation bias for performance enhancement. Arka et al. [24] proposed the nearest spatial neighborhood-based scheme for the improvement of parameter adaptation in the SHADE algorithm. These aforesaid DE variants are dedicated to realizing the adaption of the control parameters F , CR , and NP , but most of them still adopt the single strategy or design a novel mutation strategy. So, we mainly research the multi-strategy and find that some DE algorithms with multi-strategy were proposed by researchers in the past (these related works are described in Section 2.2).

Table 1. The list of nomenclature.

Nomenclature	Descriptions
JADE [11]	Adaptive differential evolution with optional external archive Fuzzy adaptive differential evolution New version of the DE algorithm with self-adapted control parameters
FADE [14]	
jDE [15]	

Table 1. Cont.

Nomenclature	Descriptions
SaDE [16]	Self-adaptive differential evolution
SHADE [18]	Success-history based adaptive differential evolution
LPSR [19]	Linear population size reduction
LSHADE [19]	SHADE using linear population size reduction
PALM [21]	Parameters with adaptive learning mechanism
Db-LSHADE [22]	LSHADE with distance-based parameter adaptation

From multi-strategy in the past, we notice that most of the schemes select an appropriate single strategy from multiple strategies to generate the offspring, such as SaDE [16], SaM-JADE [25], and EPSDE [26]. So, we focus on considering another multi-strategy approach in which the multi-strategy produces multiple candidate individuals respectively, and, then, the best individual is determined from multiple candidates as the offspring. For example, in CoDE [27], three strategies are adopted to generate three individuals respectively, and the individual with the best fitness value among three individuals is determined as the offspring. So, it is a crucial problem for this approach, how to design an effective individual selection mechanism to select the appropriate individual from the candidate pool. Aiming at the multi-strategy method and its important individual selection mechanism, we pay attention to an advanced DE optimization framework, namely, selective-candidate framework with similarity selection rule (SCSS), which was proposed by Zhang et al. [28] in 2020. It consists of multiple candidates generation (MCG) and similarity selection (SS) rule. On the one hand, the MCG section has its disadvantage because of single strategy, which may be because the small difference of candidates (these are discussed in detail in Section 3.1). For this drawback, introducing the multi-strategy may be an effective approach. On the other hand, the SS rule selects the best individual as the offspring from the candidate pool, which may be very suitable for multi-strategy as its individual selection mechanism. Moreover, the performance of SCSS may be limited by the fixed parameter GD in the SS rule.

Therefore, in this paper, we mainly combine multiple candidates generation with multi-strategy (MCG-MS) and the adaptive similarity selection (ASS) rule, and further propose a novel multi-strategy DE algorithm, namely, multi-strategy differential evolution algorithm with the adaptive similarity selection rule (MSDE-ASS). The main contributions of MSDE-ASS are summarized as follows.

- (1) On the one hand, in MCG-MS, two symmetrical mutation strategies, “DE/current-to-pbest-w/1” from advanced DE algorithm jSO [29] and the designed “DE/current-to-cbest-w/1”, are utilized to build the multi-strategy to produce two candidate individuals with different trends, which prevents the over-approximation of the candidate in SCSS;
- (2) On the other hand, the ASS rule provides the individual selection mechanism for the MCG-MS to determine the offspring from two candidates through distance measure, where parameter GD is designed to increase linearly with evolution. This adaption divides the algorithm into two symmetrical stages, to maintain diversity at the early evolution stage and accelerate convergence at the later evolution stage;
- (3) Based on advanced jSO, replacing its offspring generation strategy with the combination of MCG-MS and ASS rule, a novel multi-strategy DE algorithm MSDE-ASS is proposed. It combines the advantages of two symmetric strategies and has an efficient individual selection mechanism without parameter adjustment. MSDE-ASS is verified under the Congress on Evolutionary Computation (CEC) 2017 competition test suite on real-parameter single-objective numerical optimization, and the results indicate that, of the 174 cases in total, it wins in 81 cases and loses in 30 cases, and it has smallest performance ranking value, of 3.05. Therefore, MSDE-ASS stands out compared to the other state-of-the-art DEs.

The rest of this paper is organized as follows. Section 2 introduces DE and the related work about strategy in DE. Section 3 describes the concrete content of the proposed algorithm. Section 4 presents the experimental study and relevant discussions, and Section 5 concludes this paper.

2. Related Work

This section introduces the detailed procedures of DE and summarizes the related work about strategy in DE. Table 2 provides the list of mutation strategies.

2.1. DE

DE is a population-based stochastic search algorithm for global optimization [1]. Its detailed procedures including four basic components are provided as follows.

Initialization: Every individual in the population is represented as $x_{i,g} = (x_{1,i,g}, x_{2,i,g}, x_{3,i,g}, \dots, x_{D,i,g})$, $i = 1, 2, \dots, NP$, where NP is the population size, D is the dimension, and g is the current generation index. When initialization begins, i.e., $g = 0$, the j -th dimension of the i -th individual is obtained by

$$x_{j,i,0} = x_{j,min} + rand_{i,j}[0, 1] \cdot (x_{j,max} - x_{j,min}) \quad (1)$$

where $x_{j,min}$ and $x_{j,max}$ denote the lower and upper bounds, respectively. $rand_{i,j}[0, 1]$ is a random number between 0 and 1, conforming to uniform distribution.

Mutation: Three individuals are sampled randomly from the current population as parent to generate the donor vector $v_{i,g}$, which is shown as

$$v_{i,g} = x_{r1,g} + F \cdot (x_{r2,g} - x_{r3,g}) \quad (2)$$

where $F \in [0, 1]$ is scaling factor and $r1, r2, r3 \in [1, NP]$, $i \neq r1 \neq r2 \neq r3$. When the value of the donor vector $v_{i,g}$ is out of lower bound $x_{j,min}$ or upper bound $x_{j,max}$, the Formula (3) on boundary constraints can be used.

$$v_{j,i,g} = \begin{cases} (x_{j,i,g} + x_{j,min})/2, & \text{if } v_{j,i,g} < x_{j,min} \\ (x_{j,i,g} + x_{j,max})/2, & \text{if } v_{j,i,g} > x_{j,max} \end{cases} \quad (3)$$

Crossover: The trial vector, which is represented as $u_{i,g} = (u_{1,i,g}, u_{2,i,g}, u_{3,i,g}, \dots, u_{D,i,g})$, $i = 1, 2, \dots, NP$, is formed by exchanging elements from the donor vector $v_{j,i,g}$ and the target vector $x_{j,i,g}$ on the dimension level. The binomial crossover is outlined as follows

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & \text{if } rand_{i,j}[0, 1] \leq CR \text{ or } j = j_{rand} \\ x_{j,i,g}, & \text{otherwise} \end{cases} \quad (4)$$

where $CR \in [0, 1]$ is the crossover rate and $j_{rand} \in [1, NP]$ represents a random integer.

Selection: The selection operation is applied to pick up the superior individual from the target and the trial vector for the next generation with the purpose of letting the population size be maintained at NP . For a minimization optimization problem, which is given by

$$x_{j,i,g+1} = \begin{cases} u_{j,i,g}, & \text{if } f(u_{j,i,g}) < f(x_{j,i,g}) \\ x_{j,i,g}, & \text{otherwise} \end{cases} \quad (5)$$

where $f(x_{j,i,g})$ and $f(u_{j,i,g})$ represent the objective function value of the individual $x_{j,i,g}$ and $u_{j,i,g}$, respectively. From Formula (5), it is obvious that the trial vector $u_{j,i,g}$ with better fitness value (i.e., smaller objective function value) replaces the individual $x_{j,i,g}$ as the target individual $x_{j,i,g+1}$ for the next generation $g + 1$.

2.2. Strategy in DE

In the past, many strategies were designed by researchers, which can be divided into single strategy and multi-strategy.

On the one hand, there are a large number of algorithms using single strategy such as FADE [14], jDE [15], ODE [30], and so on. JADE [11], SHADE [18], LSHADE [19], and iL-SHADE [31] all utilize the mutation strategy “DE/current-to-pbest/1” to generate the donor vector. In the DE algorithm proposed by Islam et al., a new mutation operator “DE/current-to-gr_best/1” [32] is modified from the conventional mutation scheme of “DE/current-to-best/1” to prevent the premature convergence of the algorithm. Yu et al. [17] presented a new adaptive DE. In this algorithm, they design a new mutation strategy named “DE/lbest/1” to enhance the diversity capability of the algorithm. In ADE-ALC [33], a novel mutation strategy, “DE/current-to-leader/1”, is designed by Fu et al. to generate the mutant individual with strong convergence. A modified JADE version with sorting crossover rate introduced by Zhou et al. is called JADE_sort [34]. This algorithm uses the same single mutation strategy as the original JADE [11]. Zheng et al. [35] suggested a new DE variant called collective information-powered differential evolution (CIPDE). In CIPDE, a new collective information-based mutation (CIM), namely, “DE/current-to-cbest/1”, is developed to generate the individual. Compared with “DE/current-to-pbest/1”, it has a stronger convergence capability. jSO [29], proposed by Brest et al., uses a new weighted version of the single mutation strategy “DE/current-to-pbest-w/1” to further balance the convergence and diversity. Meng et al. [21] introduced the parameters with adaptive learning mechanism (PALM) for the enhancement of DE aiming to avoid the mis-interaction of control parameters in the evolution. For the mutation operator, as with JADE [11], the powerful single mutation strategy “DE/current-to-pbest/1” is also adopted in this algorithm. In the NBOLDE, Wu et al. [36] develop a novel neighborhood mutation strategy based on “DE/current-to-best/1”, namely, “DE/neighbor-to-neighbor/1”, to maintain the balance between convergence and diversity. By designing the mutation strategy based on historical population and novel parameter adaptive mechanisms, a novel DE variant [37] was proposed by Meng and Yang, called Hip-DE.

Table 2. The list of mutation strategies.

Mutation Strategies	Formulas	Characteristics
DE/rand/1 [1]	$v_{i,g} = x_{r1,g} + F \cdot (x_{r2,g} - x_{r3,g})$	$x_{r1,g}$, $x_{r2,g}$, and $x_{r3,g}$ are sampled randomly; strong diversity.
DE/best/1 [1]	$v_{i,g} = x_{best,g} + F \cdot (x_{r2,g} - x_{r3,g})$	The best individual $x_{best,g}$ as the base vector; strong convergence.
DE/current-to-best/1 [9]	$v_{i,g} = x_{i,g} + F \cdot (x_{best,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	Use the current best individual $x_{best,g}$; strong convergence.
DE/current-to-pbest/1 [11]	$v_{i,g} = x_{i,g} + F \cdot (x_{pbest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	Use the top p% best individual $x_{pbest,g}$; more diversity than DE/current-to-best/1.
DE/lbest/1 [17]	$v_{i,g} = x_{lbest,g} + F \cdot (x_{r2,g} - x_{r3,g})$	$x_{lbest,g}$ denotes the best individual in the group with respect to target vector i ; more diversity than DE/best/1.
DE/current-to-pbest-w/1 [29]	$v_{i,g} = x_{i,g} + F_w \cdot (x_{pbest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	Use weighting-based scaling factor F_w ; further balance convergence and diversity.
DE/current-to-gr_best/1 [32]	$v_{i,g} = x_{i,g} + F \cdot (x_{grbest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	Use the best individual of a dynamic group $x_{grbest,g}$; more diversity than DE/current-to-best/1.
DE/current-to-leader/1 [33]	$v_{i,g} = x_{i,g} + F \cdot (x_{leader,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	$x_{leader,g}$ represents the best individual during the proposed leader's lifetime; strong convergence.
DE/current-to-cbest/1 [35]	$v_{i,g} = x_{i,g} + F \cdot (x_{cbest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	$x_{cbest,g}$ is the centroid of the multiple best vectors; more convergence than DE/current-to-pbest/1.
DE/neighbor-to-neighbor/1 [36]	$v_{i,g} = x_{r3,g} + F \cdot (x_{best,g} - x_{r3,g}) + F \cdot (x_{r1,g} - x_{r2,g})$	Replace $x_{i,g}$ with a random individual $x_{r3,g}$ in the neighborhood; balance convergence and diversity.

On the other hand, there are some DE algorithms with multi-strategy. The algorithm SaDE [13] adaptively selects an appropriate mutation strategy from the strategy candidate pool by learning previous successful and failed experiences in evolution. Focusing on the further improvement of JADE, Gong et al. [25] proposed SaM-JADE, by adding

strategy adaptation mechanism (SaM) in JADE. In SaM, a strategy parameter is set to adaptively select the best mutation strategy from the multi-strategy. EPSDE [26], developed by Mallipeddi et al., also owns a pool of mutation strategies, which determines the suitable mutation strategy according to the success information. In CoDE [27], Wang et al. utilized three strategies to generate three individuals, respectively, and, then, chose the individual with the best fitness value as the offspring from three individuals after estimating their fitness values. Ali et al. [38] suggested a new differential evolution algorithm, sTDE-dR. In sTDE-dR, the population is divided into some subgroups, and different subgroups are allocated to different mutation strategies. IDE, presented by Tang et al. [39], has an adaptive mutation strategy component called individual-dependent mutation (IDM) strategy. Concretely, according to the fitness ranking information, the IDM strategy put each individual into the superior set or inferior set to assign different mutation strategies. Similar to the IDM strategy in IDE, in MPADe, Cui et al. [40] split the population into three sub-populations and designed three novel mutation strategies for sub-populations. Fan and Yan [41] developed an adaptive mutation strategy with a roulette wheel in the ZEPDE algorithm. The sum of the difference between the fitness value of each individual and the max fitness value was adopted to calculate the cumulative probability (cp) of each mutation strategy. With parameter cp obtained, the roulette wheel can run to achieve multiple mutation strategies adaption. Liu et al. [42] presented a new historical and heuristic DE (HHDE), utilizing both the historical experience and heuristic information for the adaptation. They combined success and failure evolution information and fitness ranking information to adapt multiple mutation strategies and determine the parameters. In NDE proposed by Tian and Gao [43], a neighborhood-based mutation (NM) strategy employed neighborhood information and individual information to design two novel mutation operators and a probability parameter to accomplish the adaption of the mutation strategy. An underestimation-based multi-mutation strategy (UMS) [44] was designed by Xiao et al. for DE. In the UMS, a set of candidates were simultaneously produced for each parent through multi-strategy and, then, a cheap abstract convex underestimation model was built to obtain offspring from the candidate pool. Mohamed et al. [45] suggested a novel triangular mutation in the proposed DE (EFADE). This designed mutation and mutation “DE/rand/1” were assigned to the algorithm through a nonlinear probability rule. Sun et al. [46] presented a new DE algorithm named CSDE. CSDE adopted two mutation operators to generate individuals and provided an adaptive mechanism based on the historical success rate to co-ordinate these two. Attia et al. [47] proposed a novel algorithm MSaDE. In this DE variant, the multi-mutation strategies included “DE/rand/1”, “DE/best/1”, and designed mutation technique, and only one mutation strategy was selected to form the trial vector in each generation. In their algorithm GPDE, Sun et al. [48] designed a novel Gaussian mutation and improved common mutation schemes to adaptively produce new mutant individuals. Wang et al. [49] proposed a self-adaptive mutation DE algorithm based on PSO (DEPSO). A modified “DE/rand/1” was presented based on an elite archive to address the tendency of premature convergence and another PSO-based mutation scheme was introduced to accelerate the convergence speed. Deng et al. [50] developed a novel DE, namely, DSM-DE. In this algorithm, two designed dynamic speciation-based mutation strategies were utilized to produce two individuals in which the better individual was selected as the offspring after fitness estimation. Xia et al. [51] proposed a fitness-based adaptive DE algorithm, which achieved the adaption of the combination of mutation strategy with the F and CR parameters. It belongs to the multi-strategy because it has three breeding strategies. Yan and Tian [52] presented a novel differential evolution algorithm (TLADE). In this new algorithm, they mainly designed two prediction-based mutation operators through neighborhood information of each individual and, then, used the two-level information, the history search information of the neighbors of the individual and that of the whole population, to implement a two-level adaptive (TLA) mechanism, aiming to effectively integrate the advantages of two different mutation operators. In the algorithm APSDE, Wang et al. [53] designed the accompanying evolution, using failure information

to establish the accompanying population. This accompanying population was utilized to achieve the mutation strategy adaptation.

In summary, after researching many DE algorithms with single strategy and multi-strategy, these works in the past can be roughly summarized, which are shown in Table 3. For single strategy, a series of mutation strategies were designed and adopted by researchers in the past. Focusing on combining the advantages of the different mutation strategies, multi-strategy may be a great research method. For multi-strategy in the past, on the one hand, and most of them follow this thought, the best strategy is chosen from multi-strategy to produce offspring, where the strategy adaption is the common way to implement. On the other hand, like CoDE [27], UMS [44], and DSM-DE [50], the idea is that the offspring is determined from multiple individuals produced by multi-strategy, but there is a lack of further research—it may be a good direction to explore.

Table 3. Strategy in DE.

Algorithm	Strategy
FADE [14], jDE [15], and ODE [30].	Single strategy, use DE/rand/1 [1].
JADE [11], SHADE [18], LSHADE [19], PALM [21], iL-SHADE [31], and JADE_sort [34].	Single strategy, DE/current-to-pbest/1 is designed in [11] and utilized in [18–21,31,34].
A new adaptive DE presented by Yu et al. [17], jSO [29], DE algorithm proposed by Islam et al. [32], ADE-ALC [33], CIPDE [35], NBOLDE [36], and Hip-DE [37].	Single strategy, design a new mutation strategy DE/lbest/1 [17], DE/current-to-pbest-w/1 [29], DE/current-to-gr_best/1 [32], DE/current-to-leader/1 [33], DE/current-to-cbest/1 [35], DE/neighbor-to-neighbor/1 [36], and the mutation strategy based on historical population [37], respectively.
SaDE [13], SaM-JADE [25], EPSDE [26], ZEPDE [41], HHDE [42], NM [43], EFADE [45], CSDE [46], MSaDE [47], GPDE [48], DEPSO [49], a fitness-based adaptive DE proposed by Xia et al. [51], TLADE [52], and APSDE [53].	Multi-strategy, mutation adaption, i.e., adaptively select an appropriate mutation strategy from the multiple strategies.
sTDE-dR [38], IDE [39], and MPADDE [40].	Multi-strategy, another approach to determine the best strategy from the multiple strategies, i.e., the population is divided into some subgroups, and different subgroups are allocated to different mutation strategies.
CoDE [27], UMS [44], and DSM-DE [50].	Another multi-strategy, the offspring is determined from multiple individuals produced by multi-strategy. In CoDE [27] and DMS-DE [50], the individual with the best fitness value is selected as the offspring from multiple individuals by estimating all candidates. In UMS [44], a cheap abstract convex underestimation model is built to obtain offspring from the candidate pool.

3. Proposed Method

Section 3 proposes the motivation, introduces the details of the MSDE-ASS through three parts, and describes the complete procedure of the proposed algorithm, respectively.

3.1. Motivation

SCSS [28] is a DE optimization framework. We focus on its two components, multiple candidates generation (MCG) and the similarity selection (SS) rule. In the MCG section, M candidates are independently generated using single strategy, which means that each parent in the population has its own candidate pool [28]. And, then, in the SS rule section, the offspring is determined from each candidate pool to participate in the DE selection. This process is controlled by the parameter greed degree (GD) and the Euclidean distance between the parent and each candidate.

Although it is advanced, SCSS still has shortcomings. On the one hand, in the MCG section, each current solution owns a pool of candidates but the high similarity of these candidates may be observed because of single strategy. Compared with the single strategy, mutation strategies with different features in multi-strategy are more likely to generate

individuals with low similarity. Therefore, introducing the multi-strategy may be an effective approach.

In order to clearly reveal the deficiency of single strategy and the advantage of multi-strategy in MCG, we plot Figure 1 to describe two methods of different multiple candidates generation (here, we set the candidate number $M = 2$, according to [28]), where the situation in the left subplot maybe occurs in single strategy, and the situation in the right subplot is more likely to occur in multi-strategy; besides, we use the green dots, the blue dots, and the orange dots to represent the current solution x , the candidate $u1$, and the candidate $u2$, respectively. Also, $d1$ and $d2$ measure the distance between x and $u1$, and the distance between x and $u2$, respectively.

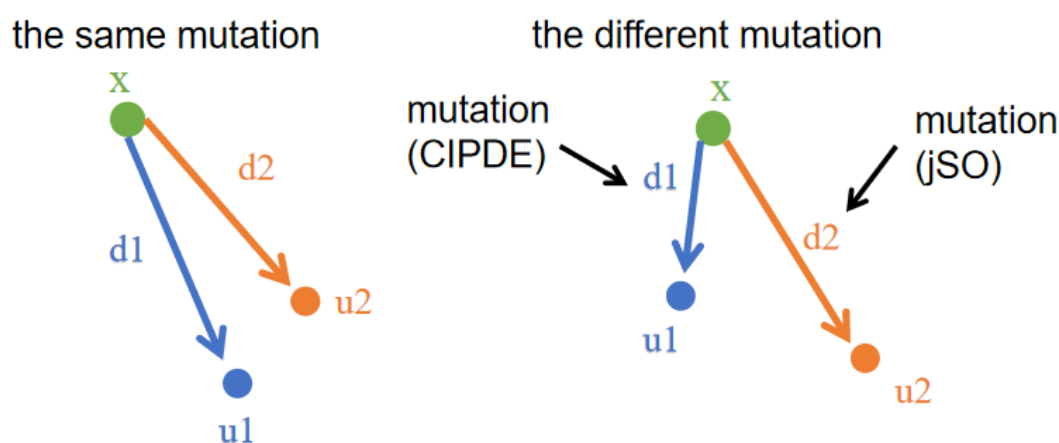


Figure 1. Two different multiple candidates generation (the green dots represent the current solution x , the blue dots are the candidate $u1$, and the orange dots are the candidate $u2$; also, $d1$ is the distance between x and $u1$, and $d2$ is the distance between x and $u2$).

As observed in Figure 1 left subplot, the candidates $u1$ and $u2$ are produced by the current solution x . Because they are the result of using the same mutation strategy, they do not have a clear trend of convergence or diversity, i.e., the calculated distances $d1$ and $d2$ are too approximate. That affects the effectiveness of the SS rule and, finally, limits the performance enhancement of the algorithm. On the contrary, as shown in Figure 1 right subplot, we can see that x produces two candidates, $u1$ and $u2$, utilizing two symmetrical mutation strategies, respectively, i.e., the convergence strategy (mutation of CIPDE) and the diversity strategy (mutation of jSO). Compared with the multiple candidates generation in the left subplot, the multiple candidates generation in the right subplot prevents the over-approximation of the distance between $d1$ and $d2$. Thus, it is obvious that, if we use the multi-strategy as in Figure 1 right subplot, i.e., adopt the two symmetrical mutation strategies to generate the candidates, the MCG will be able to reduce the possibility of occurrence of the situation shown in Figure 1 left subplot, which will further enhance the effectiveness of SS rule.

On the other hand, parameter adaption is a commonly used idea to improve the performance of the algorithm. In Section 1, we mention that many DE variants were developed based on the parameter adaption since different problems and different stages of evolution have different appropriate parameter settings. However, in the SS rule, parameter GD is set to a fixed value to control the convergence and diversity of SCSS. This means that we can consider implementing adaptive GD to enhance the performance of SCSS.

In summary, we analyze the deficiency of adopting single strategy and the advantage of introducing multi-strategy in MCG. It is indicated that the over-approximation problem of the distance between two candidates is likely to arise in single strategy, which limits the SS rule and, thus, limits the performance, while the multi-strategy adopting two symmetrical mutation strategies is more likely to generate candidates with larger distance differences, which can reduce the likelihood of this bad phenomenon occurring and,

thus, further enhance the effectiveness of SS rule. For the SS rule, we discuss the necessity for adapting GD . The fixed parameter GD controls the convergence and diversity of SCSS and the fixed parameter is not universally applicable to problems and evolution stages with diverse characteristics, which means that the performance may be further enhanced by the commonly used parameter adaption. Therefore, according to these motivations, we propose the multiple candidates generation with multi-strategy (MCG-MS), where two symmetric mutation strategies are introduced in Section 3.2.1, and the adaptive similarity selection (ASS) rule as the individual selection mechanism required for multi-strategy, where the adaptive GD is introduced in Section 3.2.2. To design a state-of-the-art DE, based on the advanced algorithm jSO, replacing its offspring generation strategy with the combination of MCG-MS and the ASS rule, we further propose a multi-strategy differential evolution algorithm with the adaptive similarity selection rule (MSDE-ASS), where the complete procedure is introduced in Section 3.2.3.

3.2. MSDE-ASS

This section is divided into three parts to introduce the details of the MSDE-ASS.

3.2.1. Mutation Strategy in MCG-MS

In the MCG-MS section, we first consider the mutation strategies of the CIPDE [35] and jSO [29], which are shown in Formulas (6) and (9).

Firstly, the mutation strategy of CIPDE is the following formulas.

$$v_{i,g} = x_{i,g} + F \cdot (x_{cBest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g}) \quad (6)$$

where $x_{cBest,g}$ is collective vector, computed by:

$$x_{cBest,g} = \sum_{k=1}^m w_k \cdot x_{i,k} \quad (7)$$

From Formula (7), we can know $x_{cBest,g}$ is obtained by finding the centroid of the m best vectors ($m \in [1, i]$, is a randomly generated integer). Specifically, the weighting factors w_k are utilized to describe the different contributions of different vectors. Considering a vector with a higher rank should contribute more [35], we let

$$w_k = \frac{m - k + 1}{1 + 2 + \dots + m} \quad (8)$$

where $k = 1, 2, \dots, m$.

Secondly, the mutation strategy of jSO is following the formulas.

$$v_{i,g} = x_{i,g} + F_w \cdot (x_{pBest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g}) \quad (9)$$

where $x_{pBest,g}$ is randomly chosen as one of the top $100 \cdot p\%$ individuals in the current population with $p \in (0, 1]$ and F_w is calculated:

$$F_w = \begin{cases} 0.7 \cdot F, & FEs < 0.2 \cdot Max_FEs \\ 0.8 \cdot F, & FEs < 0.4 \cdot Max_FEs \\ 1.2 \cdot F, & otherwise \end{cases} \quad (10)$$

From the Formula (9) we note, the mutation strategy of jSO is the modified version of the JADE mutation strategy “DE/current-to-pbest/1” [11], called “DE/current-to-pbest-w/1” in [29]. To better combine the advantage of the two symmetrical strategies, we further design the improved mutation strategy of CIPDE similar to jSO, called “DE/current-to-cbest-w/1”, which is shown in Formula (11).

$$v_{i,g} = x_{i,g} + F_w \cdot (x_{cBest,g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g}) \quad (11)$$

where $x_{cBest,g}$ defines as Formula (7) and F_w defines as Formula (10).

So, two symmetrical mutation strategies, “DE/current-to-cbest-w/1” and “DE/current-to-pbest-w/1”, are obtained to build the multi-strategy in MCG-MS. Specifically, as the convergence strategy, the mutation strategy “DE/current-to-cbest-w/1” is biased towards fast convergence because it seeks the local population centroid as the base vector in the mutation strategy. On the contrary, compared with the mutation strategy “DE/current-to-cbest-w/1”, the mutation strategy “DE/current-to-pbest-w/1” prefers to maintain population diversity as the diversity strategy because its base vector is from the top $100 \cdot p\%$ individuals.

3.2.2. Adaptive Parameter GD in the ASS Rule

In the ASS rule section, as shown in the following formulas, the adaptive parameter GD is designed.

$$GD = \frac{FEs}{Max_FEs} \quad (12)$$

where FEs is the consumed function evaluations and $MaxFEs$ is the allocated maximum function evaluations.

The value of parameter GD ranges from 0 to 1. The median value of 0.5 divides the algorithm into two symmetrical stages, convergence and diversity stages. When GD is larger than 0.5, the algorithm enhances convergence. Oppositely, when GD is smaller than 0.5, the algorithm tends to maintain diversity. So, from the Formula (12), $MaxFEs$ is fixed and GD increases from minimum to maximum with the FEs increasing. In the pre-middle stage of evolution ($GD < 0.5$), the algorithm is in the diversity stage with strong population diversity. With the GD value increasing, the algorithm enters the mid-late stage of evolution ($GD \geq 0.5$), i.e., the convergence stage, and the convergence capability of the algorithm is increasingly stronger. Through such a GD adaption, a better balance between the convergence and diversity of the algorithm can be implemented.

3.2.3. Complete Procedure of MSDE-ASS

The pseudo-code of the MSDE-ASS is presented in Algorithm 1 to describe its complete procedure.

As is shown in Algorithm 1, this new DE algorithm, similar to SCSS [28], also includes two components, multiple candidates generation with multi-strategy (MCG-MS) and the adaptive similarity selection (ASS) rule. After initializing population X via Formula (1) in line 1 and evaluating the fitness of X in line 2, the algorithm enters the loop, and, then, the fitness ranking obtained in line 4 is prepared for individual selection in the ASS rule. In the MCG-MS section, two candidates are generated with the proposed multi-strategy and two sets of parameters from the adaptive parameter mechanism of jSO [29] (lines 6–12). So, each current solution x_i owns a pool of candidates y_i^m . Especially, in line 8, symmetrical mutation strategies from Formulas (9) and (11) are introduced to produce two donor vectors, respectively. Then, their boundary problem is repaired via Formula (3) in line 9 and two candidates are gained through the binomial crossover in the following line. In line 11, the Euclidean distance $dist_i^m$ between each candidate y_i^m and the parent x_i are obtained, serving as the selection standard for candidate solutions in the ASS rule. For the ASS rule section, in line 14, the value of parameter GD at the current generation is determined by the Formula (12). Then, in lines 16–24, GD divides individuals sorted by fitness value into two groups, which controls the convergence and diversity of the MSDE-ASS. According to the opposite selection rules (minimum or maximum Euclidean distance) for these two different groups (lines 14 and 18), a solution is selected from the pool (lines 18 and 22) and corresponding parameters in use are also recorded (lines 19 and 23). After gaining the offspring Y , similar to line 2, we have evaluating the fitness of Y in line 26. Finally, in line 27, solutions are selected as new X from $X \cup Y$ using Formula (5) to enter the next iteration. The algorithm repeats the whole evolution process (line 4 to line 27) until the stopping criteria are met.

Algorithm 1 Pseudo-code for the MSDE-ASS

```

1: Initialize population  $X = \{x_1, x_2, x_3, \dots, x_{NP}\}$  via Formula (1)
2: Evaluate the fitness of  $X$ 
3: While the stopping criteria are not met Do
4: Determine the fitness ranking  $rank(i)$  of each individual  $i \in \{1, 2, 3, \dots, NP\}$ 
Component 1: Multiple Candidates Generation with Multi-strategy
5: For  $i = 1 : NP$ 
6:   For  $m = 1 : 2$ 
7:     Produce the control parameters of every individual  $CP^m = \{CP_1^m, CP_2^m, CP_3^m, \dots, CP_{NP}^m\}$  BY
THE ADAPTIVE PARAMETER MECHANISM OF JSO
8:     Produce DONOR VECTOR  $v_i^m$  FOR  $x_i$  via THE MUTATION FROM FORMULA (9)
AND (11)
9:     REPAIR THE BOUNDARY PROBLEM OF DONOR VECTOR  $v_i^m$  via FORMULA (3)
10:    Operate binomial crossover through FORMULA (4) to produce new solution  $y_i^m$ 
11:    Calculate  $dist_i^m = \text{Euclidian distance}(y_i^m, x_i)$ 
12:  End For
13: End For
Component 2: Similarity Selection Rule with Adaptive Greedy Degree
14: PARAMETER GD IS ADAPTIVELY DETERMINED BY FORMULA (12)
15: For  $i = 1 : NP$ 
16:   If  $rank(i) \leq \text{ceil}(NP \times GD)$ 
17:      $index = \text{argmin}(dist_i^m), m \in \{1, 2\}$ 
18:      $y_i = y_i^{index}$ 
19:      $CP_i = CP_i^{index}$ 
20:   Else
21:      $index = \text{argmax}(dist_i^m), m \in \{1, 2\}$ 
22:      $y_i = y_i^{index}$ 
23:      $CP_i = CP_i^{index}$ 
24:   End If
25: End For
26: Evaluate the fitness of  $Y$ 
27: Select solutions as new  $X$  from  $X \cup Y$  using Formula (5) to enter the next iteration
28: End While

```

3.3. Time Complexity

This subsection discusses the time complexity of the MSDE-ASS. According to [28], considering SCSS-DEs firstly, the complexity of SCSS-DEs is $O(M \cdot NP \cdot D^2 \cdot g_{max})$. As investigated in [28], $M = 2 \ll NP$ is sufficient for advanced DEs. Thus, the complexity of advanced SCSS-DEs remains as $O(NP \cdot D^2 \cdot g_{max})$. Because of the similar framework composition as the advanced SCSS-DEs and the two proposed improvements in MSDE-ASS which have no effect on the time complexity, the time complexity of MSDE-ASS is also $O(NP \cdot D^2 \cdot g_{max})$.

4. Simulation

In this section, the effectiveness of the proposed MSDE-ASS is investigated through comprehensive experiments conducted using the CEC2017 (the file and codes of the CEC2017 test suite can be downloaded from <https://github.com/P-N-Suganthan/CEC2017-BoundConstrained>) [54] benchmark function sets. This test suite consists of 29 functions with diverse mathematics properties, which are characterized by unimodal functions F1 and F3, simple multi-modal functions F4–F10, hybrid functions F11–F20, and composition functions F21–F30. For each test function, the final solution error values are obtained after achieving the stop criteria which is set to $10^4 \times D$ function evaluations (D is the dimension) [54]. And 51 independent runs are performed, while the mean and standard

deviations of the finally obtained solution error values are reported. The error of the solution is used to measure the performance of the algorithm, which is defined as

$$SE = f(x) - f(x^*) \quad (13)$$

where $f(x)$ is the fitness value of the optimal x achieved with $10^4 \times D$ function evaluations (D is the dimension) and $f(x^*)$ is the fitness value of the global optimal x^* .

In order to draw statistically sounded conclusions, the Wilcoxon rank-sum test with 5% significance level is applied to compare the performances. The symbols “−”, “=”, and “+” represent that the proposed algorithm performs significantly worse than, similar to, or better than the compared algorithms, respectively. All the algorithms were implemented in MATLAB and run on an Intel core i7 3.40 GHz PC with 8 GB of RAM in Windows 10 system.

4.1. Comparison with Single Strategy

In order to test the effectiveness of the multi-strategy in MSDE-ASS, because the mutation strategies in MSDE-ASS are taken from “DE/current-to-cbest-w/1” and “DE/current-to-pbest-w/1” [29], we let MSDE-ASS compare to two advanced DE algorithms with single strategy, jSOc (“DE/current-to-cbest-w/1” as the mutation strategy) and jSO (“DE/current-to-pbest-w/1” as the mutation strategy). It is noticed that jSOc is different from jSO (the MATLAB codes of jSO can also be downloaded from <https://github.com/P-N-Suganthan/CEC2017-BoundConstrained>). It replaces the mutation strategy of jSO with “DE/current-to-cbest-w/1” and other contents are the same as the jSO. The parameter settings of all algorithms are the same as those in [29] for fair comparisons.

Comparison results on 50- D CEC2017 functions are summarized in Table 4. Statistically, of the 58 cases in total, MSDE-ASS wins in 32 cases (=20 + 12) and only loses in 7 (=2 + 5) cases. MSDE-ASS is more excellent than jSOc on 20 functions and only loses on 2 functions. Compared with jSO, MSDE-ASS wins on 12 functions and underperforms on 5 functions. It is demonstrated that the multi-strategy in MSDE-ASS can effectively enhance the performance of the algorithm.

Table 4. Performance results of jSOc and jSO with MSDE-ASS on 29 50- D CEC2017 benchmark problems.

Algorithm	jSOc	jSO
+/=/:	20/7/2	12/12/5

After comparing the performances among MSDE-ASS, jSOc, and jSO, we further discuss the characteristics of the mutation strategies of the three algorithms. To describe the population convergence and diversity capability of these mutation strategies, we introduce the metrics, population diversity (div), which is calculated as:

$$\text{div} = \frac{1}{NP} \sum_{i=1}^{NP} \|x_i - c\| \quad (14)$$

where $\|x_i - c\|$ is the Euclidean distance between x_i and c , and $c = \frac{1}{NP} \sum_{i=1}^{NP} x_i$ is the center of the population. Figure 2 depicts the convergence of the div value on F5 and F19 from 50- D CEC2017 test functions. In addition to the convergence graphics of the metrics div, we also record the best error value (BE) during the entire course of the evolution, aiming to perform a comprehensive analysis. The metrics BE is defined as follows, similar to Formula (13).

$$\text{BE} = f(x_{\text{best}}) - f(x^*) \quad (15)$$

where $f(x_{best})$ is the fitness value of the current best individual x_{best} and $f(x^*)$ is the fitness value of the global optimal x^* . Figure 3 describes the convergence curves about metrics BE on F5 and F19 from 50-D CEC2017 test functions.

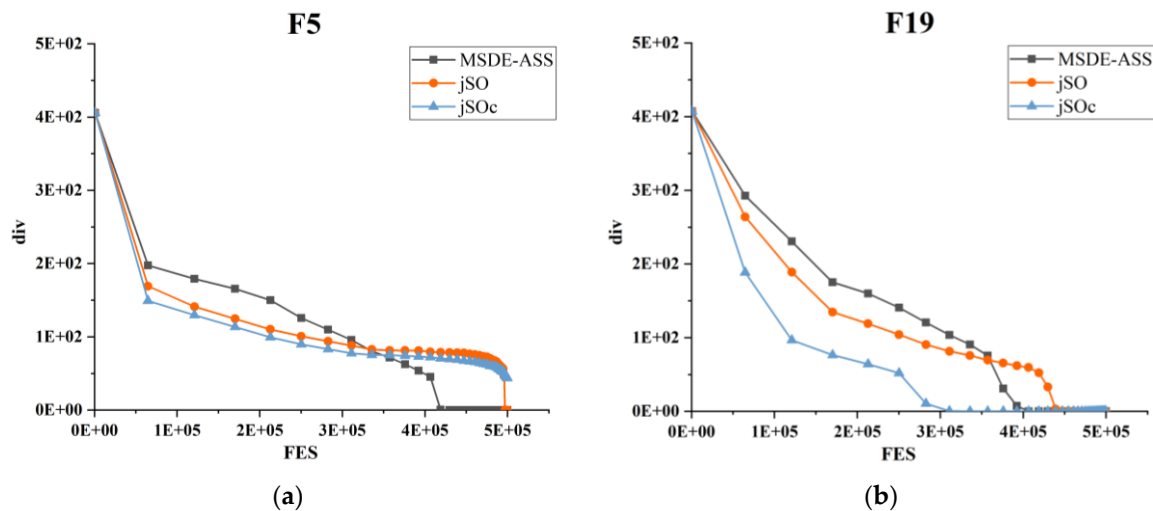


Figure 2. Convergence graphics of the population diversity (div) achieved by the MSDE-ASS, jSO, and jSOc on the 50-D CEC2017 functions: (a) F5; (b) F19.

From Figure 2a, on F5, the div value of jSOc reduces quickly and the div value of jSO decreases slower than jSOc. From Figure 2b, we can observe that, on F19, the convergence rate of jSOc and jSO have similar features as the situation on F5. So, it is indicated that the mutation strategy of jSOc has stronger convergence capability and the mutation strategy of jSO tends to maintain the population diversity. Notably, the proposed MSDE-ASS maintains more population diversity than jSO in the early evolutionary stage and accelerates convergence in the middle and later evolutionary stages. At the same time, from Figure 3, we can see the convergence case of BE achieved by the MSDE-ASS, jSO, and jSOc. jSOc is more excellent than jSO on F5 but jSO performs better than jSOc on F19. The MSDE-ASS gains the best performance among all algorithms on F5 and F19.

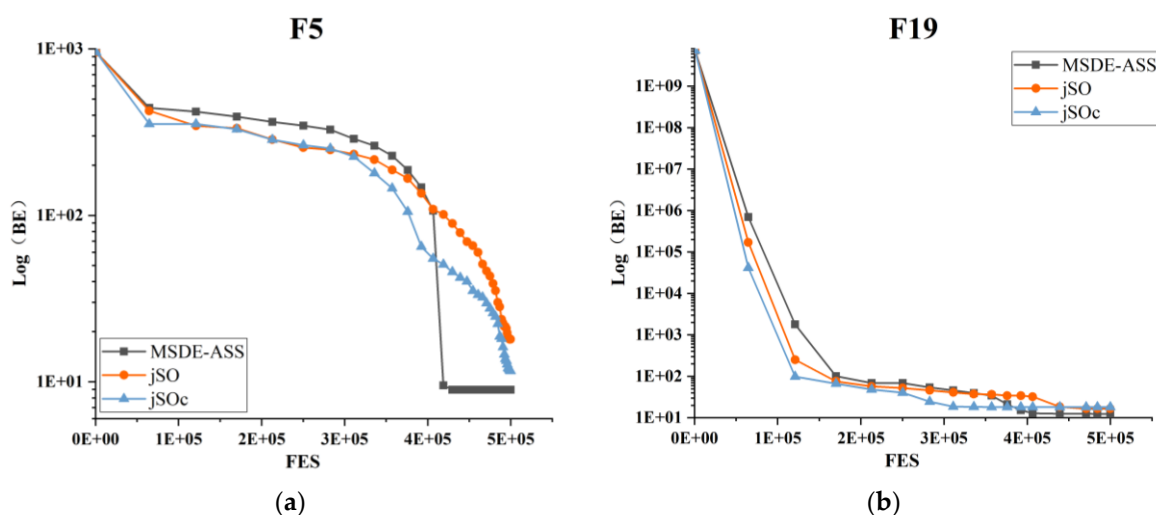


Figure 3. Convergence graphics of the best error value (BE) achieved by the MSDE-ASS, jSO, and jSOc on the 50-D CEC2017 functions: (a) F5; (b) F19.

Therefore, single mutation strategies with different characteristics are applicable to different test functions with different features. On F5, the mutation strategy of jSOc beats the mutation strategy of jSO because of its convergence ability. On F19, due to the diversity

of the mutation strategy of jSO, the obtained result is the opposite to that on F5. Importantly, the combination of multi-strategy with the ASS rule is able to effectively give full play to the advantage of multi-strategy. Facing the different test functions, the MSDE-ASS can still efficiently optimize.

4.2. Working Mechanism of MSDE-ASS

Aiming to research the working mechanism of MSDE-ASS, we design a series of experiments.

4.2.1. Source of Individual

Firstly, to explore the working mechanism of the multi-strategy of MCG-MS, we focus on the source of individuals of near and far Euclidean distance selected from the candidate pool. We design an individual ratio (IR) to describe it, which is calculated as:

$$IR = IN / TN \quad (16)$$

where IN defines as the number of individuals derived from a certain mutation strategy in all near or far individuals determined as offspring, and TN is the total number of near or far individuals as offspring. When IR equals to 0.5, the multi-strategy is invalid, because this situation means the source of individuals near and far is random, equivalent to the original SCSS with single strategy.

Concretely, for the multi-strategy of MCG-MS in MSDE-ASS, we record the number of the near and far individuals determined as offspring, TN_{near} and TN_{far} , and the number of individuals provided by “DE/current-to-cbest-w/1” in TN_{near} and generated by “DE/current-to-pbest-w/1” in TN_{far} , called IN_{cbest} and IN_{pbest} . According to Formula (16), the IR_{cbest} and IR_{pbest} are obtained and the results of these two metrics on the 29 50-D CEC2017 test suite are shown in Figure 4 (IR_{cbest} and IR_{pbest} are black and red lines, respectively, in the figure); besides, we consider two single strategies of MCG in SCSS, “DE/current-to-pbest-w/1” and “DE/current-to-cbest-w/1”, as the contrast group, calculating two IR_{pbest} (blue and green lines) and two IR_{cbest} (purple and brown lines), respectively, which are also presented in Figure 4.

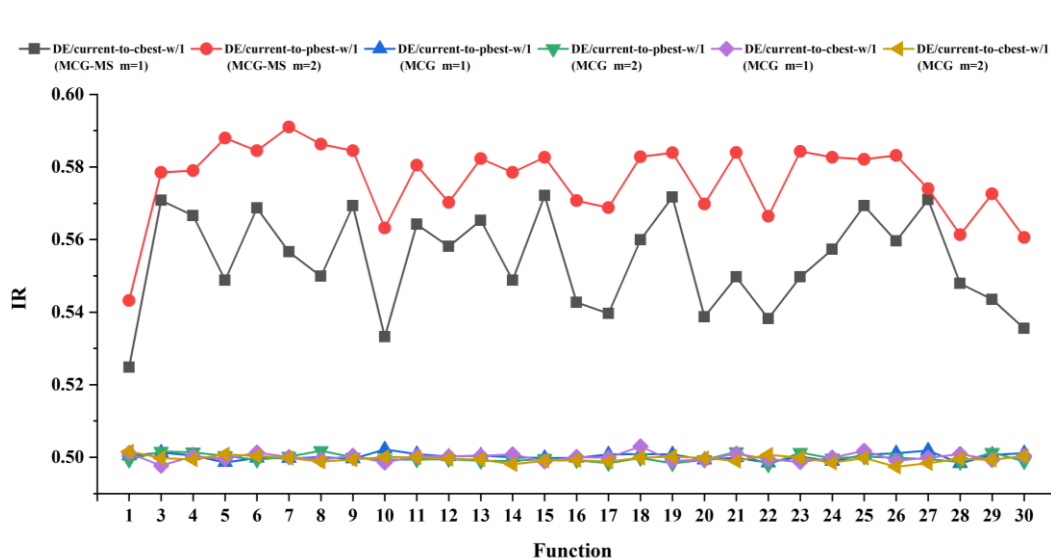


Figure 4. Individual ratio (IR) of multi-strategy of MCG-MS in MSDE-ASS and two single strategies of MCG in SCSS on 29 50-D CEC2017 benchmark problems ($m \in \{1, 2\}$, is the candidate index).

From Figure 4, we can find that, on the one hand, the values of IR_{cbest} and IR_{pbest} from multi-strategy on all test functions are more than 0.5. On the other hand, the values

of two IR_{pbest} and two IR_{cbest} from single strategy on all test functions are all close to 0.5. So, the mutation strategy “DE/current-to-pbest-w/1” and “DE/current-to-cbest-w/1” are not necessarily able to produce near and far individuals, respectively, but it is also distinguished from ineffective multi-strategy. On each test function, the mutation strategy “DE/current-to-cbest-w/1” and “DE/current-to-pbest-w/1” both have the tendency to produce the close and distant individual, respectively, which proves the effectiveness of the multi-strategy of MCG-MS in MSDE-ASS.

4.2.2. Mechanism of the ASS Rule

Secondly, to explore the working mechanism of the ASS rule, we designed a compared algorithm with the reverse operation version of the ASS rule named OMSDE-ASS and designed another compared algorithm with the random operation version of the ASS rule called RMSDE-ASS. Finally, on the 29 50-D CEC2017, we compared the performances among MSDE-ASS, OMSDE-ASS, and RMSDE-ASS, which are shown in Table 5.

Table 5. Performance results of OMSDE-ASS and RMSDE-ASS with MSDE-ASS on 29 50-D CEC2017 benchmark problems.

Algorithm	OMSDE-ASS	RMSDE-ASS
+/-/-:	21/6/2	14/13/2

From Table 5, compared with OMSDE-ASS, MSDE-ASS wins on most of the test functions and only loses in 2 cases. Because of the opposite operation in the ASS rule, the performance of the OMSDE-ASS is, naturally, worse than that of the MSDE-ASS. Similarly, compared with RMSDE-ASS, MSDE-ASS performs better in 14 cases and only underperforms in 2 cases. This result distinguishes between the ASS rule and random operations. In summary, the effectiveness of the ASS rule in MSDE-ASS is confirmed, which means that the ASS rule can effectively select the appropriate individual as the offspring to participate in the DE selection operation.

4.2.3. Adaptive Setting of the Parameter GD

Moreover, to verify the effectiveness of the adaptive GD in the ASS rule, the MSDE-ASS compares to the MSDE with a different fixed GD. For different fixed GD settings, because the GD value ranges from 0 to 1, by setting the step size to 0.2, five GD values 0.1, 0.3, 0.5, 0.7, and 0.9 are determined for validation. In addition to GD, other parameter settings of all algorithms are the same for fair comparisons. Performance comparison results on 29 50-D CEC2017 benchmark functions are summarized in Table 6.

Table 6. Performance results of fixed GD with adaptive GD on 29 50-D CEC2017 benchmark problems.

GD	0.1	0.3	0.5	0.7	0.9
+/-/-:	16/11/2	5/20/4	6/20/3	10/17/2	18/9/2

From the results of the performance comparison in Table 6, compared with MSDE with fixed GD, MSDE-ASS still wins in 14, 1, 3, 8, and 16 cases, respectively, after offsetting the number of winnings and losings. Whether the fixed value of the GD setting is small or large, the performance of the algorithm with adaptive GD is more excellent than them. It is proved that the adaptive GD setting is effective, and is able to gain better performance of the algorithm.

In summary, two components of the MSDE-ASS, MCG-MS, and the ASS rule, can be effectively co-ordinated in operation. That is why the MSDE-ASS is capable of effective performance enhancement.

4.3. Comparison with the State-of-the-Art DEs

To test the performance of the MSDE-ASS, it is compared with six state-of-the-art DEs [29,55–59]:

PaDE [55]: An enhanced L-SHADE algorithm with a novel adaptive mechanism of control parameters;

jSO [29]: An L-SHADE variant with fine-tuning parameter settings and weighting-based scaling factor at different evolution stages. It is the top-ranked DE algorithm in the CEC2017 competition;

EB-LSHADE [56]: An improved L-SHADE algorithm with a novel mutation strategy;

LSHADE-cnEpSin [57]: A modified L-SHADE algorithm integrating sinusoidal parameter adaptation and covariance matrix adaptation-based crossover;

EaDE [58]: An explicit adaptive DE algorithm with SCSS-L-SHADE and SCSS-L-CIPDE;

LSHADE-RSP [59]: An improved jSO algorithm with rank-based selective pressure strategy. It is the champion in the CEC2018 competition.

Table 7 presents the performance comparison results on 50-D CEC2017 functions (the detailed performance data are shown in Table A1).

From the results summarized in Table 7, of the 174 cases in total, MSDE-ASS wins in 81 cases ($=17 + 12 + 19 + 14 + 11 + 8$) and loses in 30 cases ($=5 + 5 + 4 + 8 + 4 + 4$). With respect to PaDE, jSO, EB-LSHADE, LSHADE-cnEpSin, EaDE, and LSHADE-RSP, the “+/-” metric achieved by MSDE-ASS is “17/5”, “12/5”, “19/4”, “14/8”, “11/4”, and “8/4”. Referring to these results, it is obvious that MSDE-ASS stands out among the state-of-the-art DEs.

Table 7. Performance results of state-of-the-art DEs with MSDE-ASS on 29 50-D CEC2017 benchmark problems.

	PaDE	jSO	EB-LSHADE	LSHADE-cnEpSin	EaDE	LSHADE-RSP
F1	=	=	=	=	=	=
F3	=	=	=	=	=	=
F4	=	=	+	—	=	=
F5	+	+	+	+	=	+
F6	—	—	—	—	—	—
F7	+	+	+	+	=	+
F8	+	+	+	+	—	+
F9	=	=	+	=	=	=
F10	=	=	=	=	=	+
F11	+	+	+	=	+	=
F12	+	=	+	—	+	=
F13	+	=	+	+	+	=
F14	+	=	+	+	+	—
F15	+	+	+	+	+	+
F16	+	+	+	=	=	=
F17	+	+	=	=	+	=
F18	+	+	+	+	+	+
F19	+	+	+	+	+	=
F20	+	—	+	—	=	=
F21	+	+	+	+	=	+
F22	—	=	=	—	=	=
F23	—	=	=	+	=	=
F24	—	—	—	+	—	=
F25	+	+	+	—	+	=
F26	=	=	—	+	=	=
F27	+	+	+	+	+	+
F28	+	—	+	—	=	—
F29	—	—	—	—	—	—
F30	=	=	+	+	+	=
+/-/-	17/7/5	12/12/5	19/6/4	14/7/8	11/14/4	8/17/4

In addition to the performance results presented in the table, Figure 5 shows more detailed experimental results on 8 test functions from the 50-D CEC2017 test suite, which are the unimodal function F1, the simple multimodal functions F5, F7, and F8, the hybrid functions F15 and F18, and the composition functions F21 and F27, respectively. These 8 subgraphs in Figure 5 describe the convergence curves about metrics BE of all compared algorithms. The horizontal coordinate axis *FES* is the fitness estimation number. The vertical co-ordinate axis $\log(\text{BE})$ represents the logarithm form of BE, which records the best error value defined in Formula (15) and displays the result more clearly compared to BE. Through Figure 5, the gap between the fitness value of the current best individual and the fitness value of the global optimal can be revealed throughout the evolution. The curves of the MSDE-ASS (black line) are mainly concerned.

So, the performance of algorithms on each test function can be revealed in Table 7 and Figure 5. Considering the function properties, the following are observed.

On the unimodal functions F1 and F3, as with the results on F1 provided by Figure 5a, all the algorithms converge to the best error value less than $1\text{E-}08$, which means that they all search for the optimal value. So, all the state-of-the-art algorithms perform very well, including MSDE-ASS.

On the simple multimodal functions F4–F10, MSDE-ASS outperforms all compared algorithms except EaDE on F5, F7, and F8, and performs worse than other competitors in all cases on F6. As seen in Figure 5b–d, on these functions, MSDE-ASS maintains the most population diversity of all algorithms at the early and middle stages, and converges very quickly at the later stage. This feature is the reason that MSDE-ASS stands out on F5, F7, and F8.

On the hybrid functions F11–F20, MSDE-ASS has the most outstanding performance. Compared with PaDE and EB-LSHADE, respectively, it exhibits better performance on all hybrid functions, and on other all hybrid functions except F17 (the comparable performance on F17). Compared with jSO, MSDE-ASS is superior on 6 functions (F11, F15–19) and only inferior on F20. Compared with LSHADE-cnEpSin, MSDE-ASS shows advantages for solving 5 functions i.e., F13–15, F18–19, and only shows disadvantages on 2 functions F12 and F20. Compared with EaDE, MSDE-ASS performs well on 8 functions (F11–15 and F17–19) and underperforms in none of the cases. Compared with LSHADE-RSP, MSDE-ASS wins on F15 and F18, and loses on F14. For illustration, on F15 and F18 from Figure 5e,f, MSDE-ASS exhibits performance superiority compared with the competitors because it reaches the smallest BE value among all algorithms. It is worth noting that the algorithm EaDE, which performs well on the simple multimodal functions like F5, F7, and F8, underperforms on the hybrid functions like F15 and F18. Unlike the EaDE, the algorithm MSDE-ASS has excellent performance on both.

On the composition functions F21–F30, MSDE-ASS is still very competitive. Compared with PaDE, it performs well on 4 functions (F21, F25, and F27–28) but also underperforms on the same number of functions (F22–24 and F29). Similar to the contrasting results of the PaDE, while winning in 3 cases F21, F25, and F27, MSDE-ASS performs worse than jSO in 3 cases, which are F24 and F28–29. Compared with EB-LSHADE, LSHADE-cnEpSin, and EaDE, respectively, MSDE-ASS is superior on 5, 6, and 3 functions and inferior on 3, 4, and 2 functions. After offsetting the number of winning and losing, MSDE-ASS are both more excellent than EB-LSHADE and LSHADE-cnEpSin on 2 test functions and is also better than EaDE on 1 test function. Compared with LSHADE-RSP, MSDE-ASS wins on F21 and F27, and loses on F28 and F29; besides, as shown in Figure 5g,h, MSDE-ASS reaches the smallest BE value among all algorithms on F21 by maintaining the diversity and, then, converges quickly like on F5, F7, and F8, but on F27, MSDE-ASS rapidly searches to small BE value early and, then, always maintains an advantage over other state-of-the-art algorithms.

Therefore, MSDE-ASS can maintain its performance advantage and algorithm competitiveness on different test functions with diverse mathematics characteristics.

Further, the overall performance ranking of state-of-the-art DEs and MSDE-ASS can be obtained by the Friedman test, which is given in Table 8. It is confirmed that MSDE-

ASS achieves the overall best efficiency among all state-of-the-art DE algorithms with the smallest ranking value, of 3.05, followed by LSHADE-RSP (3.16), jSO (3.43), and EaDE (3.78).

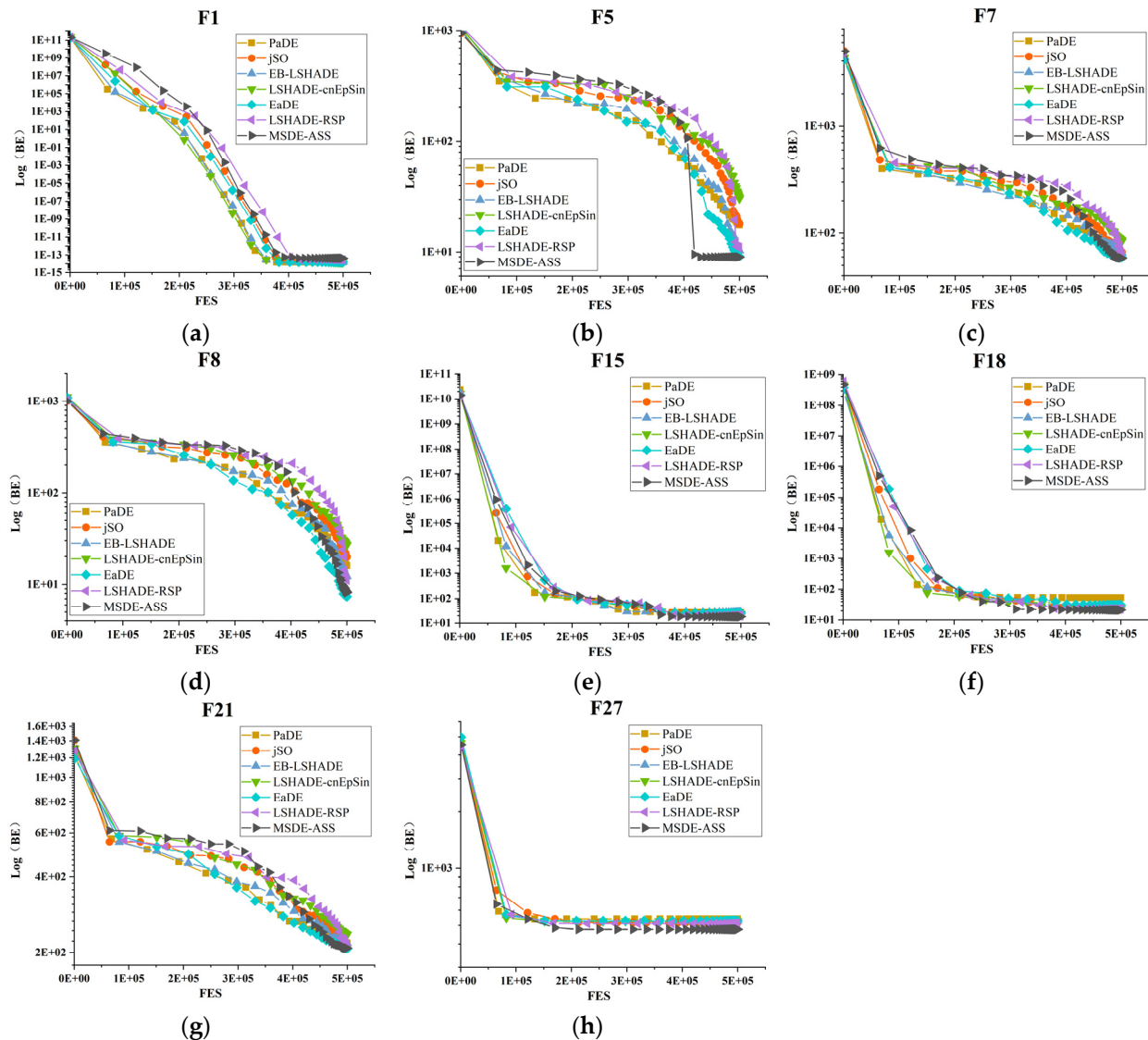


Figure 5. Convergence graphics of the best error value (BE) achieved by the compared DEs on the 50-D CEC2017 functions: (a) F1; (b) F5; (c) F7; (d) F8; (e) F15; (f) F18; (g) F21; (h) F27.

Table 8. Overall performance ranking of state-of-the-art DEs and MSDE-ASS on 29 50-D CEC2017 benchmark problems.

	PaDE	jSO	EB-LSHADE	LSHADE-cnEpSin	EaDE	LSHADE-RSP	MSDE-ASS
Ranking	5.38	3.43	5.00	4.21	3.78	3.16	3.05

To further show the statistical significance, Figure 6 plots the critical difference diagram [60], where algorithms with no significant difference are connected. It is indicated that MSDE-ASS is statistically better than PaDE and EB-LSHADE, but it is not statistically better than the other four state-of-the-art DEs.

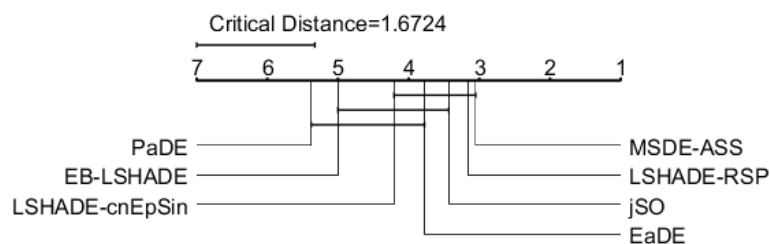


Figure 6. Comparison of DEs on 29 50-D CEC2017 benchmark problems with critical difference value.

Through the results of the above series of experiments, it is concluded that, compared with 6 state-of-the-art DEs, MSDE-ASS does not statistically significant advantage on multiple problem tests but performs better on single problem tests.

Finally, the performance of MSDE-ASS has also been investigated on 4 real-world problems (RP1–RP4) [61], which are the parameter estimation for frequency modulated (FM) sound waves problem (RP1), the spread spectrum radar polly phase code design problem (RP2), the circular antenna array design problem (RP3), and the “Cassini 2” spacecraft trajectory optimization problem (RP4), respectively. From Table 9, MSDE-ASS is prominent on RP1 and RP3 because it achieves better performance than all competitors except for LSHADE-RSP on RP1 and PaDE on RP3; besides, it performs significantly better than the EB-LSHADE and LSHADE-RSP on RP2 and is significantly similar to other compared algorithms. On RP4, PaDE and LSHADE-cnEpSin are better than other algorithms, including MSDE-ASS, while MSDE-ASS is not worse than the other 4 competitors. Overall, on these 4 real-world problems, MSDE-ASS has advantages in performance against all algorithms excluding PaDE and it is still competitive compared to PaDE.

Table 9. Performance comparisons of MSDE-ASS with state-of-the-art DEs on 4 real-world problems.

	PaDE	jSO	EB-LSHADE	MSDE-ASS
RP1	6.00E-03 (1.60E-02) +	8.23E-01 (2.88E+00) +	1.96E-03 (9.94E-03) +	1.45E+00 (3.68E+00)
RP2	1.00E+00 (1.85E-01) =	9.83E-01 (1.37E-01) =	1.14E+00 (9.00E-02) +	1.03E+00 (9.73E-02)
RP3	−2.23E+01 (4.12E-02) =	−2.16E+01 (7.42E-02) +	−2.16E+01 (9.54E-02) +	−2.18E+01 (1.44E+00)
RP4	1.20E+01 (1.02E+00) −	1.52E+01 (2.61E+00) =	1.49E+01 (1.13E+00) =	1.42E+01 (2.60E+00)
+ / = / −	1 / 2 / 1	2 / 2 / 0	3 / 1 / 0	
	LSHADE-cnEpSin	EaDE	LSHADE-RSP	MSDE-ASS
RP1	2.40E+00 (4.48E+00) +	4.92E-01 (2.46E+00) +	5.98E-01 (2.42E+00) =	1.45E+00 (3.68E+00)
RP2	1.01E+00 (7.76E-02) =	1.02E+00 (1.43E-01) =	1.08E+00 (1.02E-01) +	1.03E+00 (9.73E-02)
RP3	−2.13E+01 (1.60E-01) +	−2.16E+01 (7.49E-02) +	−2.17E+01 (3.96E-02) +	−2.18E+01 (1.44E+00)
RP4	1.30E+01 (1.33E+00) −	1.31E+01 (2.80E+00) =	1.47E+01 (2.29E+00) =	1.42E+01 (2.60E+00)
+ / = / −	2 / 1 / 1	2 / 2 / 0	2 / 2 / 0	

5. Conclusions

To deal with the weakness of the over-approximation of MCG and the fixed *GD* parameter setting of the SS rule in SCSS, this paper proposes multiple candidates generation with multi-strategy (MCG-MS) and the adaptive similarity selection (ASS) rule, respectively. Further, based on jSO, a state-of-the-art DE algorithm, namely, multi-strategy differential evolution algorithm with the adaptive similarity selection rule (MSDE-ASS), is proposed by combining MCG-MS and the ASS rule. There are two symmetrical mutation strategies “DE/current-to-cbest-w/1” and “DE/current-to-pbest-w/1” to generate the candidates in MCG-MS and an appropriate individual selected from candidates as the offspring controlled by adaptive *GD* in the ASS rule. So, it combines the advantages of two symmetric strategies and has an efficient individual selection mechanism without parameter adjustment. We have described the design motivation (Section 3.1), the details of the algorithm MSDE-ASS (Section 3.2), and discussed its time complexity (Section 3.3), compared the multi-strategy to the single strategy (Section 4.1), analyzed the working

mechanism (Section 4.2), and also compared the MSDE-ASS to the state-of-the-art DEs (Section 4.3). Comprehensive experiments show that: (1) two proposed components MCG-MS and ASS rule are both effective to enhance the performance algorithm; and (2) of the total 174 cases in the performance comparison results, MSDE-ASS wins in 81 cases and loses in 30 cases; besides, it has the smallest performance ranking value, of 3.05. As a state-of-the-art DE, MSDE-ASS stands out among the all competitors with its best overall performance.

Author Contributions: Conceptualization, L.Z.; methodology, Y.W.; software, Y.W.; validation, Y.W.; funding acquisition, L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: The work described in this paper was supported by the National Special Project Number for International Cooperation (under grant number 2015DFR11050) and the Applied Science and Technology Research and Development Special Fund Project of Guangdong Province (under grant number 2016B010126004).

Data Availability Statement: Not applicable.

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

Appendix A

Table A1. Performance comparisons (mean (std.)) of state-of-the-art DEs with MSDE-ASS on 29 50-D CEC2017 benchmark problems.

		PaDE	jSO	EB-LSHADE	LSHADE-cnEpSin	EaDE	LSHADE-RSP	MSDE-ASS
F1	Mean	2.26E-14	0.00E+00	2.01E-14	1.81E-14	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	7.06E-15	0.00E+00	7.06E-15	6.40E-15	0.00E+00	0.00E+00	0.00E+00
	Sig.	=	=	=	=	=	=	=
F3	Mean	1.77E-13	0.00E+00	1.93E-13	1.18E-13	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	4.78E-14	0.00E+00	5.69E-14	3.57E-14	0.00E+00	0.00E+00	0.00E+00
	Sig.	=	=	=	=	=	=	=
F4	Mean	7.44E+01	6.50E+01	7.09E+01	4.81E+01	6.98E+01	5.49E+01	5.72E+01
	Std. Dev.	4.77E+01	4.59E+01	4.86E+01	4.54E+01	5.22E+01	4.85E+01	4.55E+01
	Sig.	=	=	+	—	=	=	=
F5	Mean	1.70E+01	1.63E+01	1.23E+01	2.74E+01	7.92E+00	1.30E+01	8.40E+00
	Std. Dev.	1.96E+00	3.35E+00	1.93E+00	6.82E+00	3.15E+00	3.09E+00	2.65E+00
	Sig.	+	+	+	+	=	+	=
F6	Mean	3.92E-04	3.73E-07	2.34E-05	7.70E-07	2.41E-08	1.71E-07	4.62E-06
	Std. Dev.	1.78E-03	5.80E-07	1.15E-04	6.46E-07	9.85E-08	4.91E-07	1.04E-05
	Sig.	—	—	—	—	—	—	—
F7	Mean	6.51E+01	6.65E+01	6.29E+01	7.75E+01	6.11E+01	6.60E+01	6.11E+01
	Std. Dev.	2.15E+00	2.94E+00	1.90E+00	5.98E+00	1.97E+00	3.70E+00	1.73E+00
	Sig.	+	+	+	+	=	+	=
F8	Mean	1.73E+01	1.73E+01	1.27E+01	2.65E+01	8.28E+00	1.23E+01	9.53E+00
	Std. Dev.	2.33E+00	2.75E+00	2.30E+00	6.52E+00	3.12E+00	3.71E+00	2.81E+00
	Sig.	+	+	+	+	—	+	=
F9	Mean	4.90E-14	0.00E+00	1.76E-03	9.14E-14	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	5.69E-14	0.00E+00	1.25E-02	4.56E-14	0.00E+00	0.00E+00	0.00E+00
	Sig.	=	=	+	=	=	=	=
F10	Mean	3.15E+03	3.12E+03	3.16E+03	3.20E+03	3.01E+03	3.51E+03	3.12E+03
	Std. Dev.	3.62E+02	3.74E+02	3.46E+02	2.60E+02	4.47E+02	4.45E+02	3.47E+02
	Sig.	=	=	=	=	=	+	=
F11	Mean	6.25E+01	2.69E+01	4.50E+01	2.15E+01	3.27E+01	2.36E+01	2.29E+01
	Std. Dev.	9.99E+00	3.95E+00	7.18E+00	1.72E+00	4.16E+00	3.91E+00	3.64E+00
	Sig.	+	+	+	=	+	=	=
	Mean	2.26E+03	1.62E+03	2.12E+03	1.33E+03	2.01E+03	1.56E+03	1.70E+03

Table A1. Cont.

		PaDE	jSO	EB-LSHADE	LSHADE-cnEpSin	EaDE	LSHADE-RSP	MSDE-ASS
F12	Mean	2.26E-14	0.00E+00	2.01E-14	1.81E-14	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	4.86E+02	4.48E+02	5.08E+02	3.98E+02	5.76E+02	4.41E+02	4.54E+02
	Sig.	+	=	+	—	+	=	
F13	Mean	5.53E+01	3.28E+01	5.31E+01	7.68E+01	4.53E+01	3.56E+01	3.10E+01
	Std. Dev.	2.61E+01	2.65E+01	2.95E+01	3.70E+01	2.46E+01	2.10E+01	1.69E+01
	Sig.	+	=	+	+	+	=	
F14	Mean	2.98E+01	2.42E+01	2.76E+01	2.67E+01	2.61E+01	2.33E+01	2.44E+01
	Std. Dev.	3.31E+00	1.89E+00	2.93E+00	2.68E+00	2.31E+00	1.99E+00	2.54E+00
	Sig.	+	=	+	+	+	—	
F15	Mean	4.17E+01	2.32E+01	3.35E+01	2.57E+01	2.80E+01	2.15E+01	2.03E+01
	Std. Dev.	1.09E+01	2.33E+00	7.14E+00	3.63E+00	4.00E+00	1.95E+00	1.61E+00
	Sig.	+	+	+	+	+	+	
F16	Mean	3.65E+02	4.41E+02	3.74E+02	3.02E+02	3.59E+02	3.46E+02	3.16E+02
	Std. Dev.	9.85E+01	1.56E+02	1.17E+02	1.10E+02	1.39E+02	1.45E+02	1.37E+02
	Sig.	+	+	+	=	=	=	
F17	Mean	2.76E+02	2.70E+02	2.27E+02	2.15E+02	2.88E+02	2.34E+02	2.25E+02
	Std. Dev.	6.76E+01	9.54E+01	6.91E+01	6.38E+01	1.12E+02	9.43E+01	9.32E+01
	Sig.	+	+	=	=	+	=	
F18	Mean	4.69E+01	2.43E+01	3.29E+01	2.42E+01	2.73E+01	2.28E+01	2.22E+01
	Std. Dev.	2.12E+01	1.96E+00	6.63E+00	2.19E+00	3.63E+00	1.22E+00	1.22E+00
	Sig.	+	+	+	+	+	+	
F19	Mean	2.69E+01	1.38E+01	1.89E+01	1.76E+01	1.61E+01	1.10E+01	1.18E+01
	Std. Dev.	9.71E+00	2.77E+00	3.73E+00	2.62E+00	3.23E+00	2.84E+00	2.89E+00
	Sig.	+	+	+	+	+	=	
F20	Mean	1.74E+02	1.24E+02	1.61E+02	1.01E+02	1.94E+02	1.35E+02	1.40E+02
	Std. Dev.	7.73E+01	6.99E+01	5.49E+01	1.51E+01	1.39E+02	7.53E+01	6.17E+01
	Sig.	+	—	+	—	=	=	
F21	Mean	2.18E+02	2.17E+02	2.13E+02	2.26E+02	2.09E+02	2.14E+02	2.10E+02
	Std. Dev.	2.98E+00	3.77E+00	2.34E+00	6.36E+00	4.02E+00	3.98E+00	2.49E+00
	Sig.	+	+	+	+	=	+	
F22	Mean	1.77E+02	1.45E+03	2.60E+03	1.56E+03	2.69E+03	2.46E+03	2.67E+03
	Std. Dev.	4.28E+02	1.79E+03	1.59E+03	1.70E+03	1.62E+03	1.87E+03	1.55E+03
	Sig.	—	=	=	—	=	=	
F23	Mean	4.26E+02	4.29E+02	4.28E+02	4.39E+02	4.28E+02	4.31E+02	4.32E+02
	Std. Dev.	5.50E+00	5.40E+00	4.14E+00	7.87E+00	4.49E+00	6.58E+00	9.82E+00
	Sig.	—	=	=	+	=	=	
F24	Mean	5.04E+02	5.06E+02	5.06E+02	5.13E+02	5.06E+02	5.09E+02	5.09E+02
	Std. Dev.	5.79E+00	4.30E+00	2.99E+00	7.20E+00	3.02E+00	4.17E+00	4.28E+00
	Sig.	—	—	—	+	—	=	
F25	Mean	4.97E+02	4.81E+02	4.86E+02	4.81E+02	4.82E+02	4.81E+02	4.81E+02
	Std. Dev.	2.94E+01	3.39E+00	1.18E+01	2.51E+00	3.86E+00	3.19E+00	2.76E+00
	Sig.	+	+	+	—	+	=	
F26	Mean	1.14E+03	1.13E+03	1.11E+03	1.22E+03	1.14E+03	1.13E+03	1.15E+03
	Std. Dev.	7.10E+01	4.29E+01	5.68E+01	9.06E+01	6.09E+01	4.35E+01	6.66E+01
	Sig.	=	=	—	+	=	=	
F27	Mean	5.37E+02	5.14E+02	5.29E+02	5.26E+02	5.26E+02	5.12E+02	5.05E+02
	Std. Dev.	1.05E+01	1.08E+01	1.82E+01	9.20E+00	1.46E+01	1.21E+01	1.14E+01
	Sig.	+	+	+	+	+	+	
F28	Mean	4.97E+02	4.59E+02	4.77E+02	4.58E+02	4.66E+02	4.59E+02	4.59E+02
	Std. Dev.	2.03E+01	1.60E-13	2.39E+01	9.74E+00	1.76E+01	3.98E-13	4.51E-13
	Sig.	+	—	+	—	=	—	
	Mean	3.51E+02	3.61E+02	3.53E+02	3.52E+02	3.61E+02	3.65E+02	3.77E+02

Table A1. Cont.

		PaDE	jSO	EB-LSHADE	LSHADE-cnEpSin	EaDE	LSHADE-RSP	MSDE-ASS
F29	Mean	2.26E-14	0.00E+00	2.01E-14	1.81E-14	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	8.58E+00	1.50E+01	9.94E+00	1.06E+01	1.15E+01	1.76E+01	1.69E+01
	Sig.	—	—	—	—	—	—	—
F30	Mean	6.16E+05	6.11E+05	6.70E+05	6.63E+05	6.50E+05	6.16E+05	6.22E+05
	Std. Dev.	3.66E+04	3.59E+04	9.14E+04	7.00E+04	7.39E+04	3.90E+04	5.33E+04
	Sig.	=	=	+	+	+	=	=
+ / = / —		17/7/5	12/12/5	19/6/4	14/7/8	11/14/4	8/17/4	

References

- Storn, R. On the usage of differential evolution for function optimization. In Proceedings of the North American Fuzzy Information Processing, Berkeley, CA, USA, 19–22 June 1996.
- Holland, J.H. *Adaptation in Natural and Artificial Systems*; University of Michigan Press: Ann Arbor, MI, USA, 1975.
- Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 372–373.
- Zhou, S.; Xing, L.; Zheng, X.; Du, N.; Wang, L.; Zhang, Q. A self-adaptive differential evolution algorithm for scheduling a single batch-processing machine with arbitrary job sizes and release times. *IEEE Trans. Cybern.* **2019**, *51*, 1430–1442. [\[CrossRef\]](#) [\[PubMed\]](#)
- Zhou, X.G.; Peng, C.X.; Liu, J.; Zhang, Y.; Zhang, G.J. Underestimation-assisted global-local cooperative differential evolution and the application to protein structure prediction. *IEEE Trans. Evol. Comput.* **2019**, *24*, 536–550. [\[CrossRef\]](#) [\[PubMed\]](#)
- Baatar, N.; Zhang, D.; Koh, C. An improved differential evolution algorithm adopting λ -best mutation strategy for global optimization of electromagnetic devices. *IEEE Trans. Magn.* **2013**, *49*, 2097–2100. [\[CrossRef\]](#)
- Michalak, K. Low-dimensional euclidean embedding for visualization of search spaces in combinatorial optimization. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Prague, Czech Republic, 13–17 July 2019.
- Yu, K.; Liang, J.; Qu, B.; Luo, Y.; Yue, C. Dynamic selection preference-assisted constrained multiobjective differential evolution. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *52*, 2954–2965. [\[CrossRef\]](#)
- Qin, A.K.; Suganthan, P.N. Self-adaptive differential evolution algorithm for numerical optimization. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Edinburgh, UK, 2–5 September 2005.
- Montes, E.M.; Velázquez-Reyes, J.; Coello, C.A.C. A comparative study of differential evolution variants for global optimization. In Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, Seattle, WA, USA, 8–12 July 2006.
- Zhang, J.; Sanderson, A.C. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [\[CrossRef\]](#)
- Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [\[CrossRef\]](#)
- Liu, J.; Lampinen, J. On setting the control parameter of the differential evolution method. In Proceedings of the 8th International Conference on Soft Computing, MENDEL 2002, Brno, Czech Republic, 5–7 June 2002.
- Liu, J.; Lampinen, J. A fuzzy adaptive differential evolution algorithm. *Soft Comput.* **2005**, *9*, 448–462. [\[CrossRef\]](#)
- 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. [\[CrossRef\]](#)
- Qin, A.K.; Huang, V.L.; Suganthan, P.N. Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization. *IEEE Trans. Evol. Comput.* **2009**, *13*, 398–417. [\[CrossRef\]](#)
- Yu, W.J.; Shen, M.; Chen, W.N.; Zhan, Z.H.; Gong, Y.J.; Lin, Y.; Liu, O.; Zhang, J. Differential Evolution with Two-Level Parameter Adaptation. *IEEE Trans. Cybern.* **2014**, *44*, 1080–1099. [\[CrossRef\]](#)
- Tanabe, R.; Fukunaga, A.S. Success-history based parameter adaptation for Differential Evolution. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation (CEC), Cancun, Mexico, 20–23 June 2013.
- 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.
- Poláková, R.; Tvrdík, J.; Bujok, P. Differential evolution with adaptive mechanism of population size according to current population diversity. *Swarm Evol. Comput.* **2019**, *50*, 100519. [\[CrossRef\]](#)
- Meng, Z.; Pan, J.S.; Kong, L. Parameters with Adaptive Learning Mechanism (PALM) for the enhancement of Differential Evolution. *Knowl.-Based Syst.* **2018**, *141*, 92–112. [\[CrossRef\]](#)
- 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. [\[CrossRef\]](#)
- Stanovov, V.; Akhmedova, S.; Semenkin, E. Biased parameter adaptation in differential evolution. *Inf. Sci.* **2021**, *566*, 215–238. [\[CrossRef\]](#)

24. Ghosh, A.; Das, S.; Das, A.K.; Senkerik, R.; Viktorin, A.; Zelinka, I.; Masegosa, A.D. Using spatial neighborhoods for parameter adaptation: An improved success history based differential evolution. *Swarm Evol. Comput.* **2022**, *71*, 101057. [\[CrossRef\]](#)
25. Gong, W.; Cai, Z.; Ling, C.X.; Li, H. Enhanced Differential Evolution with Adaptive Strategies for Numerical Optimization. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2011**, *41*, 397–413. [\[CrossRef\]](#)
26. Mallipeddi, R.; Suganthan, P.N.; Pan, Q.K.; Tasgetiren, M.F. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Appl. Soft Comput.* **2011**, *11*, 1679–1696. [\[CrossRef\]](#)
27. 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. [\[CrossRef\]](#)
28. Zhang, S.X.; Chan, W.S.; Peng, Z.K.; Zheng, S.Y.; Tang, K.S. Selective-candidate framework with similarity selection rule for evolutionary optimization. *Swarm Evol. Comput.* **2020**, *56*, 100696. [\[CrossRef\]](#)
29. 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), Donostia, Spain, 5–8 June 2017.
30. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M. Opposition-based differential evolution. *IEEE Trans. Evol. Comput.* **2008**, *12*, 64–79. [\[CrossRef\]](#)
31. 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.
32. 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.)* **2011**, *42*, 482–500.
33. Fu, C.M.; Jiang, C.; Chen, G.S.; Liu, Q.M. An adaptive differential evolution algorithm with an aging leader and challengers mechanism. *Appl. Soft Comput.* **2017**, *57*, 60–73.
34. Zhou, Y.Z.; Yi, W.C.; Gao, L.; Li, X.Y. Adaptive Differential Evolution with Sorting Crossover Rate for Continuous Optimization Problems. *IEEE Trans. Cybern.* **2017**, *47*, 2742–2753. [\[PubMed\]](#)
35. Zheng, L.M.; Zhang, S.X.; Tang, K.S.; Zheng, S.Y. Differential evolution powered by collective information. *Inf. Sci.* **2017**, *399*, 13–29.
36. Deng, W.; Shang, S.; Cai, X.; Zhao, H.; Song, Y.; Xu, J. An improved differential evolution algorithm and its application in optimization problem. *Soft Comput.* **2021**, *25*, 5277–5298.
37. Meng, Z.; Yang, C. Hip-DE: Historical population based mutation strategy in differential evolution with parameter adaptive mechanism. *Inf. Sci.* **2021**, *562*, 44–77.
38. Ali, M.Z.; Awad, N.H.; Suganthan, P.N.; Reynolds, R.G. An Adaptive Multipopulation Differential Evolution with Dynamic Population Reduction. *IEEE Trans. Cybern.* **2017**, *47*, 2768–2779.
39. Tang, L.; Dong, Y.; Liu, J. Differential Evolution with an Individual-Dependent Mechanism. *IEEE Trans. Evol. Comput.* **2015**, *19*, 560–574.
40. Cui, L.; Li, G.; Lin, Q.; Chen, J.; Lu, N. Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations. *Comput. Oper. Res.* **2016**, *67*, 155–173.
41. Fan, Q.; Yan, X. Self-Adaptive Differential Evolution Algorithm with Zoning Evolution of Control Parameters and Adaptive Mutation Strategies. *IEEE Trans. Cybern.* **2016**, *46*, 219–232.
42. Liu, X.F.; Zhan, Z.H.; Lin, Y.; Chen, W.N.; Gong, Y.J.; Gu, T.L.; Yuan, H.Q.; Zhang, J. Historical and Heuristic-Based Adaptive Differential Evolution. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *49*, 2623–2635.
43. Tian, M.; Gao, X. Differential evolution with neighborhood-based adaptive evolution mechanism for numerical optimization. *Inf. Sci.* **2018**, *478*, 422–448.
44. Zhou, X.G.; Zhang, G.J. Differential evolution with underestimation-based multimutation strategy. *IEEE Trans. Cybern.* **2018**, *49*, 1353–1364.
45. Mohamed, A.W.; Suganthan, P.N. Real-parameter unconstrained optimization based on enhanced fitnessadaptive differential evolution algorithm with novel mutation. *Soft. Comput.* **2018**, *22*, 3215–3235.
46. Sun, G.; Yang, B.; Yang, Z.; Xu, G. An adaptive differential evolution with combined strategy for global numerical optimization. *Soft Comput.* **2019**, *24*, 6277–6296.
47. Attia, M.; Arafa, M.; Sallam, E.; Fahmy, M. An enhanced differential evolution algorithm with multi-mutation strategies and self-adapting control parameters. *Int. J. Intell. Syst. Appl.* **2019**, *11*, 26–38.
48. Sun, G.; Lan, Y.; Zhao, R. Differential evolution with Gaussian mutation and dynamic parameter adjustment. *Soft. Comput.* **2019**, *23*, 1615–1642.
49. Wang, S.; Li, Y.; Yang, H. Self-adaptive mutation differential evolution algorithm based on particle swarm optimization. *Appl. Soft Comput.* **2019**, *81*, 105496.
50. Deng, L.; Zhang, L.; Sun, H.; Qiao, L. DSM-DE: A differential evolution with dynamic speciation-based mutation for singleobjective optimization. *Memetic Comput.* **2020**, *12*, 73–86.
51. Xia, X.; Gui, L.; Zhang, Y.; Xu, X.; Yu, F.; Wu, H.; Wei, B.; He, G.; Li, Y.; Li, K. A fitness-based adaptive differential evolution algorithm. *Inf. Sci.* **2020**, *549*, 116–141.
52. Yan, X.; Tian, M. Differential evolution with two-level adaptive mechanism for numerical optimization. *Knowl.-Based Syst.* **2022**, *241*, 108209.

53. Wang, M.; Ma, Y.; Wang, P. Parameter and strategy adaptive differential evolution algorithm based on accompanying evolution. *Inf. Sci.* **2022**, *607*, 1136–1157.
54. Awad, N.H.; Ali, M.Z.; Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization*; Nanyang Technological University: Singapore, 2016.
55. Meng, Z.; Pan, J.S.; Tseng, K.K. PaDE: An enhanced Differential Evolution algorithm with novel control parameter adaptation schemes for numerical optimization. *Knowl.-Based Syst.* **2019**, *168*, 80–99.
56. Mohamed, A.W.; Hadi, A.A.; Jambi, K.M. Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization. *Swarm Evol. Comput.* **2019**, *50*, 100455.
57. 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), Donostia, Spain, 5–8 June 2017.
58. Zhang, S.X.; Chan, W.S.; Tang, K.S.; Zheng, S.Y. Adaptive strategy in differential evolution via explicit exploitation and exploration controls. *Appl. Soft Comput.* **2021**, *107*, 107494.
59. Stanovov, V.; Akhmedova, S.; Semnkin, 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.
60. Demiar, J.; Schuurmans, D. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
61. Das, S.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems*; Jadavpur University: Calcutta, India; Nanyang Technological University: Singapore, 2010.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.