Information Sciences 619 (2023) 439–456

Contents lists available at ScienceDirect

Information Sciences

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

An improved differential evolution by hybridizing with estimation-of-distribution algorithm

Yintong Li *, Tong Han, Shangqin Tang, Changqiang Huang, Huan Zhou, Yuan Wang

Aviation Engineering School, Air Force Engineering University, Xi'an 710038, China

ARTICLE INFO

Article history:
Received 9 May 2022
Received in revised form 21 October 2022
Accepted 10 November 2022
Available online 17 November 2022

Keywords:
Artificial intelligence
Differential evolution
Hybridization
Estimation distribution algorithm

ABSTRACT

To fully exploit the strong exploitation of differential evolution (DE) and the strong exploration of the estimation-of-distribution algorithm (EDA), an improved differential evolution by hybridizing the estimation-of-distribution algorithm named IDE-EDA is proposed in the study. Firstly, a novel cooperative evolutionary framework is proposed to hybridize LSHADE-RSP, a state-of-the-art DE variant incorporating DE-based effective improvement strategies, with EDA. Secondly, the dominant individuals generated by LSHADE-RSP are used to establish the probability distribution model for EDA to enhance its exploitation in each generation, and a new control parameter is introduced to balance exploitation and exploration. Then, the use of greed strategy works via EDA to fully retain high-quality solutions to the next generation to improve the convergence speed. Finally, the greedy strategy is used to shrink the external archive when its size decreases due to the reduction of the population size. A comparison of IDE-EDA with cutting-edge DE-based and EDA-based variants, including AAVS-EDA, EB-LSHADE, ELSHADE-SPACMA, jSO, LSHADE-RSP, RWGEDA, HSES, and APGSK-IMODE, was implemented to verify its efficiency. The statistical test results on the IEEE CEC 2018 and IEEE CEC 2021 test suites demonstrate that IDE-EDA is an excellent hybrid algorithm. The MATLAB source code of IDE-EDA can be downloaded from https://github.com/Yintong-Li/IDE-EDA.

1. Introduction

Compared with traditional gradient-based optimization algorithms, evolutionary algorithms (EAs) [1] have been proven to exhibit more promising performance in real-world multimodal, nonconvex, disconnected, and oscillated problems [2]. Thus, EAs has become a hot topic for scholars and has been widely used to solve complex real parameter problems over the past decade [3]. In addition, the IEEE Congress on Evolutionary Computation (CEC) conference series [4–7] have attracted many top scholars from all over the world and greatly promoted the development of EAs. Owing to the simplicity and efficiency of differential evolution (DE) [8], it is regarded as one of the most representative EAs, and has been widely chosen by researchers as the basis for proposing new algorithms and DE-based variants. According to the improvement mechanism, DE-based variants can be divided into three main categories.

(a) DE variants with parameter adaptation. Some DE variants with parameter adaptation were proposed before 2009 [9,10]. However, the most representative adaptive parameter control strategy for the scaling factor F and crossover rate

* Corresponding author.
E-mail address: yintongli0007@163.com (Y. Li).

https://doi.org/10.1016/j.ins.2022.11.029
0020-0255/

Cr in DE was proposed by Zhang and Sanderson [11] in 2009, and the novel DE variant was named JADE in which the parameter adaptation dynamically updated the *F* and *Cr* for each individual at each generation according to the feedback from the evolutionary search. Since 2009, the adaptive parameter control strategy in JADE for *F* and *Cr* has been adopted in the most promising DE-based variants. In 2013, the success-history-based adaptive DE (SHADE) [12] was proposed by Tanabe and Fukunaga to improve the robustness of JADE, in which a different parameter adaptation mechanism based on a historical record of successful parameter settings was used. In 2014, a modification of the SHADE was proposed by Tanabe and Fukunaga [13] by using linear population size reduction (L-SHADE) to balance exploration and exploitation, and it was ranked top in the IEEE CEC 2014 competition on single-objective real-valued optimization. In 2016, an improved version of the L-SHADE (iL-SHADE) [14], ranked third in the IEEE CEC2016 competition, was proposed by Brest et al. The main difference between L-SHADE and iL-SHADE is that the historical memory values of the previous generation were used to calculate the historical memory values for the next generation, and the *p*-value of DE/*current-to-pbest*/1 in iL-SHADE was gradually reduced as a linear function. In 2016, the LSHADE-EpSin [15] based on L-SHADE was proposed by Awad et al., in which a combination of two sinusoidal waves for tuning the scaling factor *F* in an adaptive manner was adopted to balance exploration and exploitation. In 2020 and 2021, Ochoa et al. [16,17] introduced the concept of shadowed and general type 2 fuzzy systems into DE to dynamically adapt parameters, effectively improving the processing speed of Type-2 fuzzy systems for dynamic parameter adaptation in DE. In 2021, Sun et al. [18] introduced a novel adaptive parameter control approach based on a reinforcement learning algorithm called policy gradient into DE to propose a learning adaptive differential evolution algorithm (LDE).

(b) DE variants with novel mutation strategies. The most representative mutation strategy "*DE/current-to-pbest*" with an external archive of inferior solutions was introduced into JADE [11], that also includes the most representative adaptive parameter control strategy and is still regarded as the basis for proposing novel DE-based variants. In 2016, a variant of SHADE, named SHADE4 [19], was proposed by Bujok et al., in which one of four mutation strategies selected randomly with a probability proportional to its success in the previous search was used to create a new trial point. In 2017, jSO [20], ranked second in the IEEE CEC 2017 competition, was proposed by introducing a novel weighted mutation strategy DE/*current-to-pbest-w*/1 to iL-SHADE [14]. In jSO, a lower (higher) scaling factor at the early (late) stage of the iterative optimization process was adopted to balance the exploration ability and the exploitation ability. In 2017, LSHADE-RSP [21], ranked second in the IEEE CEC 2018 competition, was proposed by Stanovov et al., by adopting a rank-based selective pressure strategy instead of the randomly selected individuals used in the mutation strategy. In 2019, EB-LSHADE was proposed by Mohamed et al. [22], who introduced a novel mutation strategy DE/current-to-ord_pbest/ into LSHADE to enhance its performance. In 2021, Xia et al.[23] proposed a new DE-based variant named NFDDE by introducing a novelty-hybrid-fitness driving force to balance the exploration ability and the exploitation ability, in which the fitness and novelty were considered for selecting individuals to generate mutant individuals. In 2021, Stanovov et al. [24] proposed the NL-SHADE-RSP algorithm, which is a further development of the LSHADE-RSP algorithm by adopting nonlinear population size reduction, adaptive archive set usage, and control crossover rate. In 2022, Kumar et al. [25] developed a novel variant of DE, named OLSHADE-CS, by implementing the orthogonal array and neighborhood search-based initialization to construct the initial population. In addition, a conservative selection scheme and a combination of multiple mutation strategies were also utilized in OLSHADE-CS. In 2022, Zeng et al., proposed a new selection method by using discarded individuals to improve the performance of DE [26]. According to the selection framework, the target individual is replaced with the best target individual or a random individual selected from the discarded trial individuals. In 2022, Cao et al. [27] proposed an adaptive differential evolution algorithm by using the standard deviation of fitness value and the sum of standard deviation of each dimension of population to fully extract and efficiently utilize the population feature information.

(c) Hybrid DE with other EAs. In 2017, Awad et al. [28] proposed LSHADE-cnEpSin based on LSHADE-EpSin [15] by adopting an adaptation scheme based on earlier success for the ensemble of sinusoidal approaches, and a new crossover operator based on covariance matrix adaptation with Euclidean neighborhood for non-variance dealing capability. Besides, Awad et al.[29] also proposed a hybrid algorithm named LSHADE-SPACMA, ranked fourth place in the IEEE CEC 2017 competition, by combining LSHADE-SPA [29] and the modified CMA-ES to make full use of each advantage and character of the two algorithms. In 2018, Hadi et al. [30] proposed an improved version of LSHADE-SPACMA named ELSHADE-SPACMA, ranked third in the IEEE CEC 2018 competition, in which *p* value that controls the greediness of the mutation strategy is dynamic to balance exploration and exploitation for different stages of the evolutionary process. Additionally, the ELSHADE-SPACMA was further enhanced by integrating another directed mutation strategy within the hybridization framework [31] in 2019. Zhang et al. [32] proposed an hybrid sampling-evolution strategy, named HS-ES that is the the winner of the IEEE CEC 2018 competition, by combining the covariance matrix adaptation-evolution strategy (CMA-ES) and univariate sampling method in 2018. In 2021, Mohamed et al. [33] proposed a hybrid algorithm, named APGSK-IMODE that is the winner of the IEEE CEC 2021 competition, by combining gaining sharing knowledge based algorithm with adaptive parameters (APGSK) and improved multi-operator differential evolution algorithm (IMODE) to enhance the performance of APGSK. In 2022, Zhao et al. [34] proposed a hybrid cooperative differential evolution variants named jSO_CMA-ES_LBFGS by introducing the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) with the local search of Limited-Memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) mechanism to solve complex continuous problems.

Estimation of distribution algorithm (EDA) [35] is a special class of EAs because there are no crossover and mutation operators as other kinds of EAs during the evolutionary process. There is no crossover operator or mutation operator in EDA, and the new solutions in EDA are sampled based on a probability distribution, which is established using the dominant individuals obtained at previous generations. During the last decade, many researchers have been attracted by EDA and its variants

has achieved great success in both combinatorial and continuous domains [36,37]. The main efforts for EDA are focused on the construction of excellent probability distribution models, which can reflect the main structural characteristics of the problem, and an excellent probability distribution model will guide EDA to explore more promising regions. In 2018, a novel variant of Gaussian estimation of distribution algorithms (GEDA, called EDA for simplicity), called AAVS-EDA [38] was developed by adopting an anisotropic adaptive variance scaling (AAVS) technique to address the drawbacks of the variable variances decrease fast, and the main search tends to perpend in particular direction. In 2019, Wang et al. [39] proposed a novel GEDA variant with random walk strategies namely RW-GEDA in which the weighted maximum likelihood estimation method is used to overcome the premature convergence, and the Gaussian random walk and Lévy walk are activated to enhance the exploration performance. In 2020, Liang et al. [40] developed a new GEDA variant named EDA^2 by using an archive that preserves a certain number of high-quality solutions generated in the previous generations, rather than only utilizing the high-quality solutions in current generation to estimate the probability distribution model to exploit the evolution direction. In 2021, Tang et al. [41] proposed a new EDA variant, named EDA-FL, by introducing the idea of Kalman filtering to enhance the accuracy of modeling and a learning strategy to improve the effectiveness of sampling. In 2022, Wang et al. [42] proposed a novel permutation-based multi-factorial EDA variant, called PMFEA-EDA, by developing a new sampling mechanism. Moreover, PMFEA-EDA has been employed to efficiently deal with multiple composition requests in web service composition.

In summary, DE and its variants have been extensively studied on above-mentioned aspects and shown promising performance. However, DE and its variants also tend to trap into local optima resulting from the decrease in population diversity in late stage of evolution process. Additionally, EDA has indeed shown a strong exploration performance with a simple structure that is completely different from the DE framework. The motivation of this work is to hybridize state-of-the-art DE variant with EDA to further exploit their respective advantages. In our work, a hybridization algorithm by combining LSHADE-RSP and EDA, named IDE-EDA, is proposed to fully exploit the advantages of LSHADE-RSP and EDA in different optimization stages. The contributions of our work are three points. The first contribution is that a novel hybrid framework for DE-based variants and EDA is proposed. Secondly, a control parameter τ is adopted to control the number of offspring sampled by EDA to balance exploitation and exploration. The third contribution is that the use of greed strategy works via EDA in the hybrid framework to fully retain high-quality solutions. Moreover, the greedy strategy is used to shrink the external archive $\boldsymbol{A}$ to accelerate rate of convergence.

The work is structured as follows. The basic principles of LSHADE-RSP and EDA are presented in Section 2. Details of IDE-EDA are described in Section 3. In Section 4, the numerical experimental environment setup is briefly described. In Section 5, the numerical experimental results and discussion are analyzed in detail. Finally, the conclusions and future work are discussed in Section 6.

2. Principles of LSHADE-RSP and EDA

In the section, the state-of-the-art DE variant incorporating DE-based effective improvement strategies, called LSHADE-RSP, and EDA are briefly presented.

2.1. LSHADE-RSP

LSHADE-RSP [21], one of the winners of the IEEE CEC 2018 competition, is a very excellent DE variant that was proposed by Stanovov, et al., in 2018 by incorporating DE-based effective improvement strategies. The mathematical models of initialization strategy, mutation strategy, crossover strategy, selection strategy, linear population size reduction strategy, and adaptive parameter control strategy are described as follows.

2.1.1. Initialization strategy

The initialization strategy in LSHADE-RSP is the same as that in the canonical DE [8]. At the beginning of the evolutionary process, each candidate is randomly initialized as follows.

$$x_i^j = x_{\min}^j + rand \cdot (x_{\max}^j - x_{\min}^j) \tag{1}$$

where $i \in \{1, 2, 3, \cdots, NP\}$ is the index of the individual in population, and NP denotes the population size. $j \in \{1, 2, 3, \cdots, D\}$ is the index of the dimension of the individual, and D denotes the dimensionality. *rand* denotes a random number with uniform distribution between 0 and 1. $\boldsymbol{x}_{\min} = (x_{\min}^1, x_{\min}^2, \cdots, x_{\min}^D)$ and $\boldsymbol{x}_{\max} = (x_{\max}^1, x_{\max}^2, \cdots, x_{\max}^D)$ denote the problem-specific lower and upper search boundaries.

2.1.2. Mutation strategy

In LSHADE-RSP, the mutation strategy with rank-based selective pressure named "DE/*current-to-pbest-w*/*r*" [21] is utilized to produce the mutant vectors $\boldsymbol{v}_i$ as follows.

$$\boldsymbol{v}_i = \boldsymbol{x}_i + Fw_i \cdot (\boldsymbol{x}_{pbest} - \boldsymbol{x}_i) + F_i \cdot (\boldsymbol{x}_{pr_1} - \boldsymbol{x}_{pr_2}) \tag{2}$$

$$Fw_i = \begin{cases} 0.7 \cdot F_i, & FEs < 0.2 \cdot FEs_{max} \\ 0.8 \cdot F_i, & 0.2 \cdot FEs_{max} \leqslant FEs < 0.4 \cdot FEs_{max} \\ 1.2 \cdot F_i, & FEs \geqslant 0.4 \cdot FEs_{max} \end{cases} \tag{3}$$

where F_i is the scaling factor in range (0,1] and it will be discussed in detail in Section 2.1.6. $\boldsymbol{x}_{\text{pbest}}$ is a dominant individual randomly selected from the top $NP \times p$ individuals of the current population [12], and p is the greediness factor used to balance exploitation and exploration. $\boldsymbol{x}_{pr1}$ and $\boldsymbol{x}_{pr2}$ are different individuals selected based on rank-based probabilities from the current population. Besides, $\boldsymbol{x}_{pr2}$ may also be randomly selected from the external archive $\boldsymbol{A}$ [21]. The selected probability of i^{th} individual in the current population is computed as Eq. (4).

$$Pr_i = Rank_i/(Rank_1 + Rank_2 + \cdots + Rank_{NP}) \tag{4}$$

$$Rank_i = k(NP - i) + 1 \tag{5}$$

In Eq. (5), the best individual will be assigned the largest rank, and the worst individual will be assigned the smallest rank according to the fitness value, i.e., here i taken from the range [1, NP] is the index in a sorted fitness array [21]. k is used to control the greediness of the rank selection, and it is set to 3 in LSHADE-RSP. In addition, the mutant vector $\boldsymbol{v}_i$ will be adjusted using Eq. (6) if it is beyond the search boundary.

$$v_i^j = \begin{cases} (x_{\min}^j + x_i^j)/2, & v_i^j < x_{\min}^j \\ (x_{\max}^j + x_i^j)/2, & v_i^j > x_{\max}^j \end{cases} \tag{6}$$

2.1.3. Crossover strategy

The crossover operation will be implemented to generate trial vectors after the mutation operation is finished. In the LSHADE-RSP, the binomial crossover [8] is used to generate trial vector $\boldsymbol{u}_i$ as Eq. (7).

$$u_i^j = \begin{cases} v_i^j, & \text{rand}_i^j < Cr_i \quad \text{or} \quad j = j_{rand} \\ x_i^j, & \text{otherwise} \end{cases} \tag{7}$$

where j_{rand} is the randomly generated dimension within the range [1, D], that can ensure at least a dimension of $\boldsymbol{u}_i$ coming from the mutant individual $\boldsymbol{v}_i$. Cr_i is the crossover rate that will be discussed in detail in Section 2.1.6.

2.1.4. Selection strategy

The selection strategy will be implemented to determine the survivor for next generation according to the fitness values of $\boldsymbol{u}_i$ and $\boldsymbol{x}_i$. In LSHADE-RSP, the greedy selection strategy [8] is used as follows.

$$\boldsymbol{x}_i = \begin{cases} \boldsymbol{u}_i, & f(\boldsymbol{u}_i) \leqslant f(\boldsymbol{x}_i) \\ x_i, & otherwise \end{cases} \tag{8}$$

where f is the objective function. In addition, the external archive $\boldsymbol{A}$ [12] is used in LSHADE-RSP to enhance the diversity of population. The target individual $\boldsymbol{x}_i$ will be preserved for the next generation if it is better than the trial individual $\boldsymbol{u}_i$ according to the fitness values. Otherwise, it will be discarded from the current population and saved into the external archive $\boldsymbol{A}$. As for the external archive update mechanism, an individual randomly selected in external archive $\boldsymbol{A}$ will be replaced by the newly inserted individual if the external archive is full.

2.1.5. Linear population size reduction

The linear population size reduction (LPSR) proposed in L-SHADE [13] is also utilized in LSHADE-RSP to balance the exploitation and the exploration. According to the LPSR, the population size decreases during the evolutionary process as follows.

$$NP = \text{round}\left[\frac{FEs}{FEs_{max}}(NP_{\min} - NP_{\text{init}}) + NP_{\text{init}}\right] \tag{9}$$

where FEs and $FEs_{\max}$ are the current and maximum number of function evaluations, respectively. NP_{init} and NP_{min} denote the maximum and minimum size of the population, respectively. And round $(\cdot)$ is an integer function. Thus, the worst individuals will be removed from the population at each generation when NP is updated. Moreover, the external archive size $|\boldsymbol{A}| = round(NP \times r^{arc})$ will be resized according to the NP at the end of each generation.

2.1.6. Adaptive parameter control

In LSHADE-RSP, the history-based parameter adaptation scheme as in iL-SHADE [14] is used to adjust the scaling factor F_i and crossover rate Cr_i for every individual at each generation. The successful pairs of F_i and Cr_i, the trail vector $\boldsymbol{u}_i$ generated using them is better than the target vectors $\boldsymbol{x}_i$, will be respectively saved in $\boldsymbol{S}_{\text{CR}}$ and $\boldsymbol{S}_F$ to update the historical memories $\boldsymbol{M}_{CR,k}$ and $\boldsymbol{M}_{F,k}$ based on the weighted Lehmer mean as follows at the end of each generation.

$$M_{F,k} = \begin{cases} M_{F,k}, & \boldsymbol{S}_F = \varnothing \\ \frac{(\text{mean}_{WL}(\boldsymbol{S}_F)+M_{F,k})}{2}, & \text{otherwise} \end{cases} \tag{10}$$

$$M_{CR,k} = \begin{cases} 0, & M_{CR,k} = \perp \quad or \quad \max(\boldsymbol{S}_{CR} = 0) \\ \frac{(\text{mean}_{WL}(\boldsymbol{S}_{CR})+M_{CR,k})}{2}, & \text{otherwise} \end{cases} \tag{11}$$

$$\text{mean}_{WL}(\boldsymbol{S}) = \frac{\sum_{n=1}^{|\boldsymbol{S}|} \omega_n \cdot S_n^2}{\sum_{n=1}^{|\boldsymbol{S}|} \omega_n \cdot S_n} \tag{12}$$

$$\omega_n = \frac{|f(\boldsymbol{u}_n) - f(\boldsymbol{x}_n)|}{\sum_{n=1}^{|\boldsymbol{S}|} |f(\boldsymbol{u}_n) - f(\boldsymbol{x}_n)|} \tag{13}$$

In Eq. (12), $\boldsymbol{S}$ refers to either $\boldsymbol{S}_{CR}$ or $\boldsymbol{S}_F$. For each generation, only k^{th} cell of $\boldsymbol{M}_{CR}$ and $\boldsymbol{M}_F$ is updated. The value of k starts at 1 and increases by 1 in one generation. And k will be reset to 1 once k exceeds the predefined memory size H [14]. All cell values in $\boldsymbol{M}_F$ and are $\boldsymbol{M}_{CR}$ initialized to 0.3 and 0.8, respectively. Moreover, one historical memory entry contains fixed values during the evolutionary process, i.e., $\boldsymbol{M}_{CR,\,H}$ and $\boldsymbol{M}_{F,\,H}$ are always set to 0.9.

In every generation, each individual $\boldsymbol{x}_i$ will be assigned a scaling factor F_i for mutation operation and crossover rate Cr_i for crossover operation. F_i and Cr_i are generated using Cauchy distribution and normal distribution, respectively, as follows.

$$F_i = randc(M_{F,R_i}, 0.1) \tag{14}$$

$$F_i = \begin{cases} \text{Apply Eq.}(14), & F_i \leqslant 0 \\ \min(F_i, 0.7), & FEs < 0.6 \cdot FEs_{\max}, \quad F_i > 0 \\ \min(F_i, 1), & 0.6 \cdot FEs_{\max} \leqslant FEs, \quad F_i > 0 \end{cases} \tag{15}$$

$$Cr_i = randn(M_{CR,R_i}, 0.1) \tag{16}$$

$$Cr_i = \begin{cases} 0, & Cr_i < 0 \\ \max(Cr_i, 0.7), & FEs < 0.25 \cdot FEs_{\max}, \ Cr_i > 0 \\ \max(Cr_i, 0.6), & 0.25 \cdot FEs_{\max} \leqslant FEs < 0.5 \cdot FEs_{\max}, \ Cr_i > 0 \\ \min(Cr_i, 1), & 0.5 \cdot FEs_{\max} \leqslant FEs, \ Cr_i > 0 \end{cases} \tag{17}$$

where $R_i \in \{1, 2, 3, \cdots H\}$ is the index of the cell selected from $\boldsymbol{M}_{CR}$.

Moreover, the greediness factor p in LSHADE-RSP increases during the evolutionary process to prevent premature convergence as follows.

$$p = 0.085 + 0.085 \cdot \frac{FEs}{FEs_{\max}} \tag{18}$$

2.2. EDA

In this part, the basic knowledge of EDA is briefly presented. The framework of basic EDA is simple and the general procedure of EDA is presented as follows.

Step 1. Initializing parameters and population $\boldsymbol{P}$.
Step 2. Evaluating solutions in $\boldsymbol{P}$ and updating the best solution $\boldsymbol{x}_{\text{best}}$.
Step 3. Outputting $\boldsymbol{x}_{\text{best}}$ if algorithm terminates, otherwise executing **Step 4.**
Step 4. Selecting dominant population $\boldsymbol{P}_{\text{d}}$ from $\boldsymbol{P}$ to construct Gaussian probability distribution model.
Step 5. Generating new population $\boldsymbol{P}$ by sampling from the probability distribution model.
Step 6. Checking bound-constraints, then execute **Step 2.**

In continuous EDA and its variants, the Gaussian model is the most widely adopted probability model [40]. The joint Gaussian probability density function (PDF) for a random vector $\boldsymbol{x}$ with D dimensions can be parameterized by the mean $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{C}$ as follows [38].

$$G(\boldsymbol{x})_{(\boldsymbol{\mu},\boldsymbol{C})} = \sqrt{\frac{1}{(2 \cdot \pi)^D \det(\boldsymbol{C})}} \cdot \exp\left(-(\boldsymbol{x} - \boldsymbol{\mu})^T \cdot \boldsymbol{C}^{-1} \cdot (\boldsymbol{x} - \boldsymbol{\mu})/2\right) \tag{19}$$

$$\boldsymbol{\mu} = \frac{1}{|\boldsymbol{P}_d|} \cdot \sum_{i=1}^{|\boldsymbol{P}_d|} \boldsymbol{x}_i, \qquad \boldsymbol{x}_i \in \boldsymbol{P}_d \tag{20}$$

$$\boldsymbol{C} = \frac{1}{|\boldsymbol{P}_d|} \cdot \sum_{i=1}^{|\boldsymbol{P}_d|} (\boldsymbol{x}_i - \boldsymbol{\mu}) \cdot (\boldsymbol{x}_i - \boldsymbol{\mu})^T, \qquad \boldsymbol{x}_i \in \boldsymbol{P}_d \tag{21}$$

where $|\boldsymbol{P}_\mathrm{d}|$ denotes the cardinality of $\boldsymbol{P}_\mathrm{d}$, i.e., the number of dominant individuals included in $\boldsymbol{P}_\mathrm{d}$.

The estimated Gaussian probability model can be depicted as shown in Fig. 1 using the probability density ellipsoid (PDE) in the hyperspace. In Fig. 1, the center of the probability density ellipsoid denotes the mean $\boldsymbol{\mu}$, that is the search center of EDA, and the eigendirections and the corresponding eigenvalues of the covariance matrix $\boldsymbol{C}$ determine the axis direction and length of the probability density ellipsoid, respectively [40]. Therefore, the covariance matrix $\boldsymbol{C}$ represents the search direction and scope of EDA.

When $\boldsymbol{\mu}$ and $\boldsymbol{C}$ of the estimated Gaussian probability distribution model is obtained according to the dominant population, new population will be sampled using the mean $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{C}$ as follows.

$$\boldsymbol{x}_i = \boldsymbol{\mu} + \ \boldsymbol{g}_i, \quad \boldsymbol{g}_i \sim N(0, \boldsymbol{C}) \tag{22}$$

In addition, the vector $\boldsymbol{x}_i$ will be adjusted using Eq. (23) if it is beyond the search boundary.

$$x_i^j = x_{\min}^j + rand \cdot (x_{\max}^j - x_{\min}^j), if \ \ x_i^j < x_{\min}^j \ \text{ or } \ x_i^j > x_{\max}^j \tag{23}$$

3. The proposed IDE-EDA

This section describes the novel hybrid algorithm, IDE-EDA, that is proposed by hybridizing LSHADE-RSP with EDA. Firstly, the reasons why LSHADE-RSP is selected as the basis for hybridization are that: (1) DE has been regarded as one of most representative algorithms of EAs due to its simplicity and efficiency. (2) DE-based algorithms have shown promising performance in IEEE CEC conference series. (3) LSHADE-RSP is regarded as the most competitive DE-based variant without changing the basic framework of DE because it ranks second in the IEEE CEC 2018 competition. Then, the reason why EDA is selected as a component of the hybrid algorithm is due to its strong exploration resulting from not depending on a particular individual, but only on the information included in dominant individuals in the evolutionary process, which can effectively alleviate the reduction of population diversity in LSHADE-RSP.

3.1. Cooperative evolutionary framework

Research on hybrid algorithms is an important direction of evolutionary algorithms. The focus is on how to efficiently combine two or more EAs and give full play to their respective advantages to develop better algorithms. A novel cooperative evolutionary framework is utilized in the proposed IDE-EDA to fully exploit the strong exploitation of LSHADE-RSP and the strong exploration of EDA in different optimization stages. Specifically, LSHADE-RSP tends to trap into the local optima due

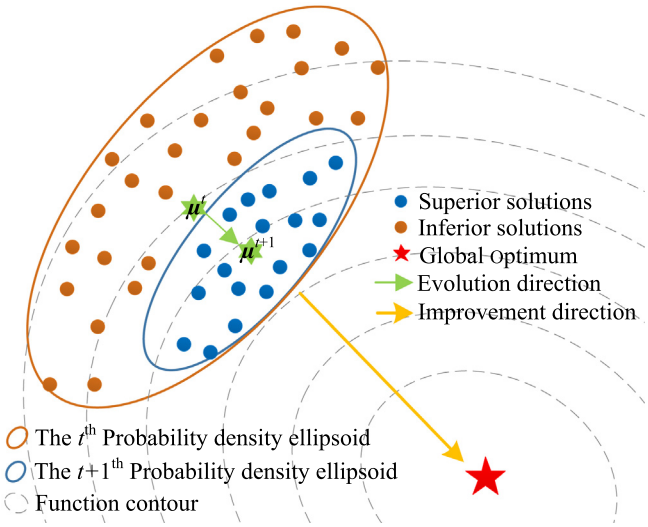

Fig. 1. Change of probability density ellipsoid in EDA.

to the decrease of population diversity during the evolution process. Also, as shown in Fig. 1, the long axis of the semicircular ellipsoid tends to be parallel to the contour of the function, therefore the long axis of the new probability density ellipsoid estimated using the dominant individuals within the semicircular ellipsoid also tends to be parallel to the function contour [40]. In other words, the main search direction of EDA is often orthogonal to the improvement direction during the evolutionary process. EDA has indeed demonstrated a strong exploration performance, although the above-mentioned defect exists.

In IDE-EDA, two stages in each generation are designed based on the characteristics of LSHADE-RSP and EDA, and they are applied independently at each stage. The novelty of the cooperative evolutionary framework is mainly reflected in the population interaction between the two algorithms. The procedure of IDE-EDA is presented in Fig. 2 and described in **Algorithm 1**. Firstly, LSHADE-RSP is used to search the solution space to fully utilize its strong exploitation resulting from the implementation of $\boldsymbol{x}_{\text{pbest}}$ and the rank-based selective pressure in mutation. Then, the dominant individuals selected from the population $\boldsymbol{P}$, generated by LSHADE-RSP, are used to establish the probability distribution model for EDA to fully utilize its strong exploration in the evolutionary process. Because the dominant individuals selected to establish the probability distribution model are not all generated by EDA, the defect of orthogonal improvement direction evolution direction of EDA could be alleviated. In the cooperative evolutionary framework, the population updating strategy during the implementation of LSHADE-RSP is consistent with the original algorithm, as shown in lines 17 to 23 of **Algorithm 1**. During the implementation of EDA, the greed search strategy is utilized to preserve better individuals into the next generation as shown in lines 33 to 36 of **Algorithm 1** to improve the convergence speed of IDE-EDA, in which the individuals generated by LSHADE-RSP and those sampled by EDA are combined, and the best *NP* individuals will form the new population $\boldsymbol{P}$. Moreover, the greedy strategy is used to shrink the external archive $\boldsymbol{A}$ during the evolutionary process, i.e., the worst individuals will be discarded from the external archive when its size $|\boldsymbol{A}|$ decreases due to the reduction of *NP*.

Algorithm 1: The procedure of IDE-EDA

Input: f, $[\boldsymbol{x}_{\min}, \boldsymbol{x}_{\max}]$, $FEs_{\max}$.
Output: $\boldsymbol{x}_{\text{best}}$.

// Initialization.
1. $NP_{\text{init}} = 75 \cdot D^{(2/3)}$, $k = 3$, $H = 5$, $NP_{\min} = 4$. $r^{\text{arc}} = 1$, $\tau = 0.9$, $FEs = 0$, $\boldsymbol{A}=\varnothing$, $N_{\text{A}} = 0$, $\boldsymbol{M}_{\text{CR}} = 0.8$, $\boldsymbol{M}_{\text{F}} = 0.3$.
2. Initialize population $\boldsymbol{P}$ by using Eq. (1).
3. Evaluate $\boldsymbol{P}$ to determine their fitness value by using $f(\boldsymbol{P})$.
4. $FEs = FEs + NP$.

// Main loop.
5. **While** $FEs < FEs_{\max}$ **do**

Applying LSHADE-RSP.
6. Determine p as per Eq. (18).
7. Determine $\boldsymbol{Pr}$ as per Eq. (4).
8. $M_{CR,\,H} = 0.9$, $M_{F,\,H} = 0.9$. $\boldsymbol{S}_{CR} = \varnothing$, $\boldsymbol{S}_{F} = \varnothing$.
9. **For** $i = 1$ **to** NP **do**
10. Determine F_i, as per Eq. (14) and (15)
11. Determine Fw_i as per Eq. (3).
12. Determine Cr_i as per Eq. (16) and (17)
13. Generate mutant vector $\boldsymbol{v}_i$ using Eq. (2).
14. Check bound-constraints for $\boldsymbol{v}_i$ using Eq. (6).
15. Generate a trial individual $\boldsymbol{u}_i$ using Eq. (7).
16. Evaluate $\boldsymbol{u}_i$ to determine its fitness value by using $f(\boldsymbol{u}_i)$.
17. **If** $f(\boldsymbol{u}_i) < f(\boldsymbol{x}_i)$ **then**
18. Update external archive $\boldsymbol{A}$.
19. $\boldsymbol{x}_i = \boldsymbol{u}_i$.
20. $f(\boldsymbol{x}_i) = f(\boldsymbol{u}_i)$.
21. $\boldsymbol{S}_{CR} = Cr_i \cup \boldsymbol{S}_{CR}$.
22. $\boldsymbol{S}_{F} = F_i \cup \boldsymbol{S}_{F}$.
23. **End if**
24. **End for**
25. $FEs = FEs + NP$.
26. Update $\boldsymbol{M}_{F}$ and $\boldsymbol{M}_{CR}$ as per Eq. (10) and Eq. (11).

Selecting the dominant individuals for EDA.
27. Select NP_{d} dominant individuals from $\boldsymbol{P}$ to form $\boldsymbol{P}_{\text{d}}$ for EDA.

Applying EDA.
28. Estimate $G(\boldsymbol{x})_{(\boldsymbol{\mu},\,\boldsymbol{C})}$ by using Eq. (20) and Eq. (21).

(*continued on next page*)

* (*continued*)

Algorithm 1: The procedure of IDE-EDA

```
Sample P_E using Eq. (22).
Check bound-constraints by using Eq. (23).
Evaluate P_E by using f(P_E).
FEs = FEs + NP_E.
P=P ∪ P_E.
f(P) = f(P) ∪ f(P_E).
// Reducing population size using LPSR.
Calculate NP according to Eq. (9).
// Updating population and the external archive using greedy strategy.
Shrink P by discarding the worst individuals.
Shrink A by discarding the worst individuals.
End while
```

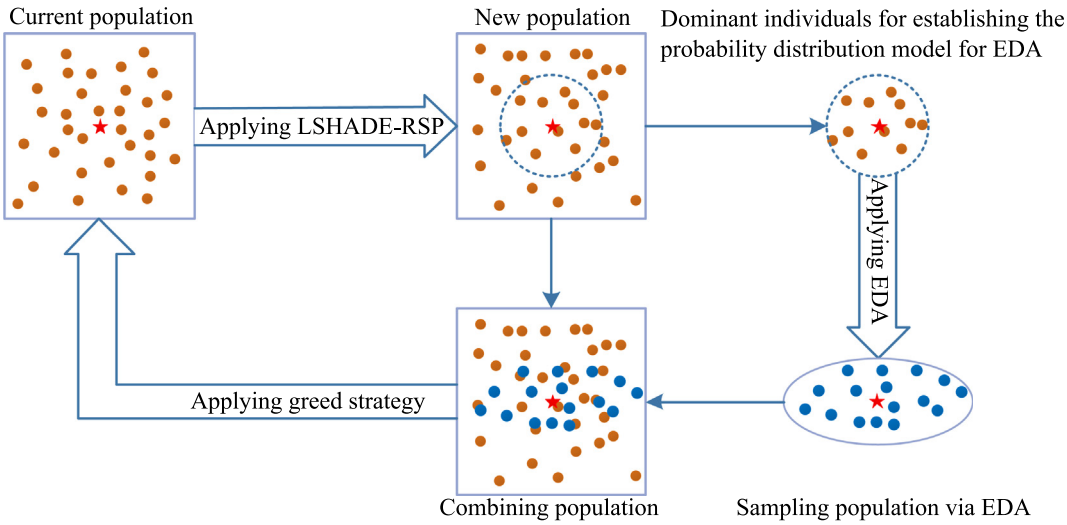

Fig. 2. Sketch for procedure of the IDE-EDA.

3.2. Parameters of EDA in IDE-EDA

In the framework of IDE-EDA, there are two parameters of EDA that need to be considered.

The first parameter is NP_d that represents the number of dominant individuals selected from $\boldsymbol{P}$ generated by LSHADE-RSP. In order to estimate the Gaussian probability distribution model as accurately as possible, half of the individuals in population $\boldsymbol{P}$ with better fitness values are selected as dominant individuals. Moreover, the population size NP will decrease to a small value later in the evolutionary process, resulting in low population diversity, as LPSR is adopted in IDE-EDA. In this case, half of the individuals in population $\boldsymbol{P}$ are not enough to correctly estimate the Gaussian distribution model of the promising region, and the entire population $\boldsymbol{P}$ will be used as dominant individuals to enhance the diversity. Thus, the value of NP_d can be computed as follows.

$$NP_d = \begin{cases} 0.5 \cdot NP, & NP \geqslant 4 \cdot D \\ NP, & NP < 4 \cdot D \end{cases} \tag{24}$$

The second parameter is NP_E that represents the number of individuals sampled using EDA. However, if NP_E is set to be a large value, that does not generate better solutions, but certainly wastes more function evaluations. Conversely, a small value for NP_E will not give full play to EDA in the proposed algorithm. Thus, the value of NP_E is adjusted according to changes in greediness factor p and population size NP during evolutionary process for adaptive as follows.

$$NP_E = \tau \cdot p \cdot NP \tag{25}$$

where $\tau \in (0,1]$ is utilized to control the number of individuals generated by EDA in the second stage of the cooperative evolutionary framework, and its influence will be qualitatively analyzed in Section 5.1.

4. Experimental environment setting

4.1. Experimental platform and configuration

All experiments in this paper are implemented on Windows system in a computer with 16 GB of memory and an Inter(R) Core (TM) i7-10750H CPU@2.60 GHz processor. IDE-EDA and all competitors were developed in MATLAB R2019b with 64 bits.

4.2. Test functions and performance metrics

To comprehensively investigate the performance of IDE-EDA by comparing it with competitors experimentally, the IEEE CEC 2018 test suite [4] in 10, 30, and 50 dimensions on single objective bound constrained real-parameter optimization is used. Details of the IEEE CEC 2018 test suite can refer to [4], in which the maximum function evaluation ($FEs_{\max}$) is set to $D \times 10000$ for all benchmark functions, and the search boundaries are set to $[-100, 100]^D$. Moreover, all the experimental results were obtained by 51 runs independently for statistical analysis. The IEEE CEC 2021 test suite in 20 dimensions is also conducted to measure the performance of IDE-EDA against shifted, rotated, and biased functions. Details of the IEEE CEC 2021 test suite can refer to [7], in which the $FEs_{\max}$ is set to 1,000,000 for all benchmark functions in 20 dimensions, and the search boundaries are set to $[-100, 100]^D$. Moreover, all the experimental results were obtained by 30 runs independently for statistical analysis. The function error measure $f(\boldsymbol{x}_{Best})$-$f(\boldsymbol{x}^*)$ is recorded as the result, where $\boldsymbol{x}_{Best}$ and $\boldsymbol{x}^*$ denote the optimal solution derived from the algorithm and the global optimum solution of the test function, respectively. Note that the results will be taken as 0 if the error measure is smaller than 10^{-8} [4,7].

Two non-parametric statistical hypothesis tests, including the Wilcoxon signed-rank test and the Friedman test [43], are used to analyze the results derived from IDE-EDA and competitors. Specifically, the Wilcoxon signed-rank test is used to test whether there is a significant difference in the performance of each function between IDE-EDA and its competitors, and then to judge IDE-EDA is better or worse than the competitor [44]. Furthermore, the final rankings obtained by the Friedman test for algorithms on all functions can be used to evaluate the significant differences among algorithms in overall performance.

4.3. Test algorithms

There are eight algorithms used in this paper for the comparative analysis to evaluate the performance of IDE-EDA. The brief descriptions of the eight competitors are summarized as follows.

1) IDE-EDA: the proposed algorithm.
2) AAVS-EDA: the state-of-the-art EDA variant [38].
3) EB-LSHADE: the state-of-the-art DE variant [22].
4) ELSHADE-SPACMA: the state-of-the-art hybrid algorithm by combining DE-based variants and CMA-ES, ranked third in the IEEE CEC 2018 competition [30].
5) jSO: the state-of-the-art DE variant ranked second in the IEEE CEC 2017 competition [20].
6) LSHADE-RSP: the state-of-the-art DE variant, the winner of the IEEE CEC 2018 competition [21].
7) RW-GEDA: the state-of-the-art EDA variant [39].
8) HSES: the winner of the IEEE CEC 2018 competition [32].
9) APGSK-IMODE: the winner of the IEEE CEC 2021 competition [33].

Regarding the parameters associated with the competitors, the recommendations by the authors of each paper are adopted in this paper and tabulated in Table 1.

5. Results and discussions

The results and discussions for the IEEE CEC 2018 test suite (D = 10, 30, and 50) and the IEEE CEC 2021 test suite (D = 20) are presented in this section. Firstly, the sensitivity of IDE-EDA to parameter τ is discussed. Then, the comparison results between IDE-EDA and the state-of-the-art competitors are discussed.

5.1. Sensitivity of IDE-EDA to parameter τ

To investigate the sensitivity of IDE-EDA to control parameter τ, the IEEE CEC 2018 test suite is used to evaluate the performance of IDE-EDA with 10 different parameter settings. For brevity, only four benchmark functions with 30D including

Table 1
Parameter settings for seven algorithms.

Algorithm	Parameter settings
IDE-EDA	$NP_{\text{init}} = 75{\cdot}D^{(2/3)}$, $k = 3$, $H = 5$, $NP_{\min} = 4$, $r^{\text{arc}} = 1$, $\tau = 0.9$.
AAVS-EDA	$NP = 1000$, $\tau = 0.35$, $\alpha = 0.17$, $\beta = 1/\alpha$ as in [38].
EB-LSHADE	$NP_{init} = 18{\cdot}D$, $NP_{min} = 4$, $p = 0.11$, $\|\boldsymbol{A}\|=1.4{\cdot}NP$, $H = 5$, $c = 0.8$ as in [22].
ELSHADE-SPACMA	$NP_{init} = 18{\cdot}D$, $NP_{min} = 4$, $\|\boldsymbol{A}\|=1.4{\cdot}NP$, $H = 5$, $F_{CP} = 0.5$, $c = 0.8$, $p_{AGDE} = 0.1$, $p_{init} = 0.3$, $p_{min} = 0.15$ *threshold* = *MaxFEs*/2, as in [30].
jSO	$NP_{init}=25\log(D)\sqrt{D}$, $NP_{min} = 4$, $H = 5$, $\|\boldsymbol{A}\|=2.6{\cdot}NP$, $p_{max} = 0.25$, $p_{min} = p_{max}/2$ as in [20].
LSHADE-RSP	$NP_{\text{init}} = 75{\cdot}D^{(2/3)}$, $k = 3$, $H = 5$, $NP_{\min} = 4$. $r^{\text{arc}} = 1$ as in [21].
RW-GEDA	$NP = 12{\cdot}D$, $\tau = 0.5$ as in [39].
HSES	λ = round(3·ln D) + 80, at first step of HS-ES: $M = 200$, $N = 100$, $I = 20$ and $cc = 0.96$, at fourth step of HS-ES: $M = 200$, $N = 160$ ($D = 10, 30$); $M = 450$, $N = 360$ ($D = 50$) as in [32].
APGSK-IMODE	$PS_2 = 7.5{\cdot}D$, $PS_2^{\min} = 12$, $p = 0.05$, Kw_P = [0.85,0.05,0.05,0.05], c = 0.05, $Arc\_rate = 1.4$, $H = 15{\cdot}D$, $PS_1 = 30{\cdot}D$- PS_2, $NP_1^{\min} = 4$, $p^{\text{init}} = 0.3$, $p^{\min} = 0.15$, $CS = 50$ as in [33].

the simple multimodal function F7, hybrid function F16, and composition functions F26 and F30 are used in the experiment. And each problem is solved 30 times independently.

The value of control parameter τ in this experiment includes $\tau\in\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The influence of the parameter τ on the performance of IDE-EDA is tabulated in Table 2, and the rankings differences are shown in Fig. 3. According to the results, there is no significant linear correlation between the performance of IDE-EDA and the value of τ. Specifically, IDE-EDA obtained the best results on hybrid function F16, and composition functions F26 and F30 when the parameter τ is set to 0.9. As for the simple multimodal function F7, IDE-EDA obtained the best results when the control parameter τ is set to 0.5. Thus, $\tau = 0.9$ is recommended and adopted for all experiments in this paper.

5.2. Comparison with state-of-the-art competitors using the IEEE CEC 2018 test suite

In the section, the IEEE CEC 2018 test suite is used to comprehensively evaluate IDE-EDA. The detailed results obtained by IDE-EDA, AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA for the IEEE CEC 2018 test suite ($D = 10, 30, 50$) are illustrated in Table S1-S3 provided in the **supplementary file**, in which "Mean" and "SD" denote the mean value and standard deviation of the results. The best solutions among IDE-EDA and six competitors are shown in **bold**.

5.2.1. Analysis of the results derived by the Wilcoxon signed-rank test

The Wilcoxon signed-rank test results between IDE-EDA and six competitors with a significance level $\alpha = 0.05$ for the IEEE CEC 2018 test suite are summarized in Table 3, and the detailed results are listed in Table S1-S3 provided in the **supplementary file**, in which the symbol '+' denotes that IDE-EDA outperforms the competitor, while the symbol '−' represents that IDE-EDA is worse than the competitor. And the symbol '=' represents that IDE-EDA has no significant difference with the competitor.

According to Table 3, it can be concluded that IDE-EDA outperforms all six competitors in 10*D* and 30*D* for the IEEE CEC 2018 test suite, and outperforms six competitors except LSHADE-RSP in 50*D* for the IEEE CEC 2018 test suite. Details are given below.

a) For 10*D*, IDE-EDA is superior (inferior) to AAVS-EDA on 20(3) functions, EB-LSHADE on 9(4) functions, ELSHADE-SPACMA on 9(4) functions, jSO on 12(3) functions, LSHADE-RSP on 3(1) functions, and RW-GEDA on 20(0) functions on 10*D*.
b) For 30*D*, IDE-EDA is superior (inferior) to AAVS-EDA on 13(10) functions, EB-LSHADE on 11(6) functions, ELSHADE-SPACMA on 15(3) functions, jSO on 18(3) functions, LSHADE-RSP on 6(1) functions, and RW-GEDA on 20(1) functions on 30*D*.

Table 2
Influence of the parameter τ on the performance of IDE-EDA. ($FES_{\max}$ = 300000).

No.		τ = 0.1	τ = 0.2	τ = 0.3	τ = 0.4	τ = 0.5	τ = 0.6	τ = 0.7	τ = 0.8	τ = 0.9	τ = 1.0
F7	Mean	3.82E + 01	3.86E + 01	3.88E + 01	3.87E + 01	3.77E + 01	3.84E + 01	3.88E + 01	3.85E + 01	3.88E + 01	3.85E + 01
	Rank	2	6	9	7	**1**	3	10	4	8	5
F16	Mean	2.93E + 01	4.09E + 01	2.52E + 01	3.52E + 01	3.76E + 01	1.73E + 01	2.29E + 01	2.02E + 01	1.69E + 01	1.86E + 01
	Rank	7	10	6	8	9	2	5	4	**1**	3
F26	Mean	9.22E + 02	9.23E + 02	9.15E + 02	9.03E + 02	9.19E + 02	9.11E + 02	8.96E + 02	8.99E + 02	8.92E + 02	8.92E + 02
	Rank	9	10	7	5	8	6	3	4	**1**	2
F30	Mean	1.97E + 03	1.97E + 03	1.97E + 03	1.97E + 03	1.96E + 03	1.97E + 03	1.97E + 03	1.97E + 03	1.96E + 03	1.97E + 03
	Rank	8	3	5	6	2	10	9	7	**1**	4
Mean Rank		6.5	7.25	6.75	6.5	5	5.25	6.75	4.75	**2.75**	3.5

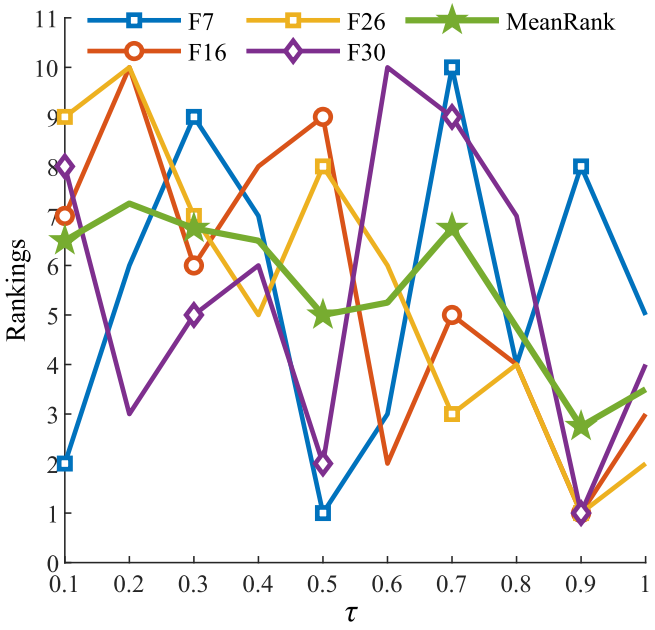

Fig. 3. Rankings of the different parameter τ for performance.

Table 3
The Wilcoxon signed-rank test results for IDE-EDA and six competitors (α = 0.05).

R (+/-/=)	10*D*	30*D*	50*D*	Total
IDE-EDA vs AAVS-EDA	20/3/6	13/10/6	13/9/7	46/22/19
IDE-EDA vs EB-LSHADE	9/4/16	11/6/12	13/8/8	33/18/36
IDE-EDA vs ELSHADE-SPACMA	9/4/16	15/3/11	12/6/11	36/13/38
IDE-EDA vs jSO	12/3/14	18/3/8	15/5/9	45/11/31
IDE-EDA vs LSHADE-RSP	3/1/25	6/1/22	6/8/15	15/10/62
IDE-EDA vs RW-GEDA	22/0/7	20/1/8	21/3/5	63/4/20

c) For 50*D*, IDE-EDA is superior (inferior) to AAVS-EDA on 13(9) functions, EB-LSHADE on 13(8) functions, ELSHADE-SPACMA on 12(6) functions, jSO on 15(5) functions, LSHADE-RSP on 6(8) functions, and RW-GEDA on 21(3) functions on 30*D*.

Moreover, "Total" in the last column of Table 3 is the sum of all dimensions results, that can reflect the comprehensive performance of IDE-EDA and six competitors on the IEEE CEC 2018 test suite. Based on the results of "Total", IDE-EDA outperforms AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA because IDE-EDA get more '+' than '−'. Therefore, the conclusion supported by Wilcoxon signed-rank test results is that IDE-EDA outperforms six competitors on the IEEE CEC 2018 test suite.

Table 4
Rankings derived by the Friedman test for the results on the IEEE CEC 2018 test suite (α = 0.05).

Algorithm	10*D*	30*D*	50*D*	Mean Ranking	Rank
IDE-EDA	3.03	**3.09**	**3.14**	**3.09**	**1**
AAVS-EDA	5.00	3.86	4.07	4.31	6
EB-LSHADE	3.64	3.86	4.31	3.94	5
ELSHADE-SPACMA	3.69	4.16	3.76	3.87	3
jSO	3.71	4.29	4.14	4.05	4
LSHADE-RSP	**2.88**	3.38	3.21	3.16	2
RW-GEDA	6.05	5.36	5.38	5.60	7
Friedman-p-value	2.00E-10	8.79E-04	6.85E-04	N/A	N/A

5.2.2. Analysis of the results derived by the Friedman test

The Friedman test results with a significance level α = 0.05 for the results on the IEEE CEC 2018 test suite are listed in Table 4 to illustrate differences among IDE-EDA, AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA. In Table 4, "Mean Ranking" is the average value of the rankings of algorithms on 10*D*, 30*D*, and 50*D* derived by the Friedman test, and "Rank" is the ranking value of "Mean Ranking" results.

According to Table 4, all *Friedman-p-values* are less than the significance level α, which represents that IDE-EDA and six competitors show significant difference in terms of performance in the IEEE CEC 2018 test suite on all dimensions. To exhibit the Friedman test results more visually, the rankings of IDE-EDA and six competitors obtained by the Friedman test are depicted in Fig. 4. In Fig. 4, IDE-EDA outperforms all competitors except LSHADE-RSP on 10*D* of the IEEE CEC 2018 test suite. For 30*D* and 50*D*, IDE-EDA shows significantly better performance than six competitors. In addition, IDE-EDA gets the first ranking in terms of "Mean Ranking". Thus, the conclusion supported by the Friedman test results is that IDE-EDA is superior to AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA on the IEEE CEC 2018 test suite.

Furthermore, the post-hoc Iman-Davenport test [45], a statistic distributed based on the F-distribution, is used to further analyze the magnitude of significant differences.

$$F_F^2 = \frac{(N-1)\cdot\chi_F^2}{N\cdot(K-1)-\chi_F^2} \tag{26}$$

where K and N denote the number of algorithms and functions, respectively. In this section, K is set to 7 and N is set to 29. The critical difference value (*CDV*), calculated by using Eq. (27), is utilized to analyze the magnitude of significant differences among IDE-EDA and six competitors based on the Friedman test results.

$$CDV = q_a \cdot \sqrt{\frac{K\cdot(K+1)}{6N}} \tag{27}$$

where q_a is obtained from the F-distribution. q_a is equal to 2.45 and *CDV* is equal to 1.39 in this section.

Multiple comparisons reflecting the magnitude of significant differences between IDE-EDA and six competitors are depicted in Fig. 5, in which algorithms that can be connected using *CDV* do not show significant differences. As shown in Fig. 5, IDE-EDA is significantly better than AAVS-EDA and RW-GEDA, and does not show significant differences with EB-LSHADE, jSO, ELSHADE-SPACMA, and LSHADE-RSP on 10*D*. As for 30*D* and 50*D*, there are no significant differences between IDE-EDA, AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, and LSHADE-RSP, but IDE-EDA significantly outperforms RW-GEDA.

Therefore, the conclusion supported by the results of the Wilcoxon signed-rank test and Friedman test is that the proposed IDE-EDA shows promising performance in the IEEE CEC 2018 test suite.

5.2.3. Analysis of convergence speed

The mean convergence graphs (the mean convergence value is taken as 0 if it is smaller than 1e-08) of 51 independent results derived by IDE-EDA and six competitors are utilized to analyze the convergence speed of IDE-EDA. For brevity, the

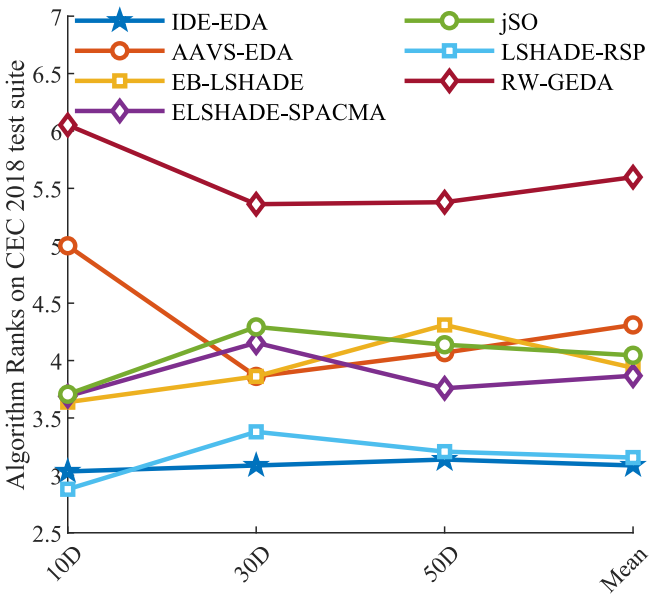

Fig. 4. Rankings derived by the Friedman test for the results.

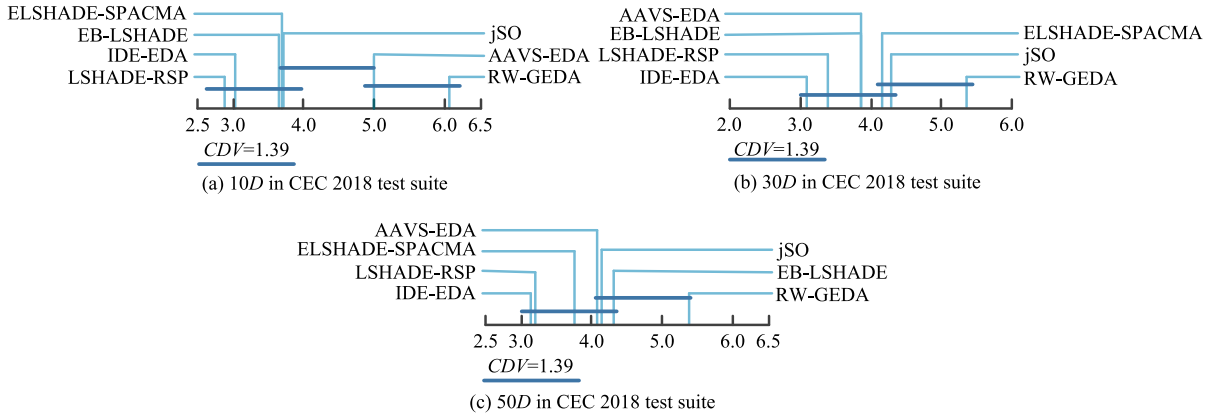

Fig. 5. Multiple comparisons using the post-hoc Iman-Davenport test.

mean convergence graphs only on six functions (unimodal function F01, simple multimodal function F06, hybrid functions F13 and F16, and composition functions F24 and F29) sampled from the IEEE CEC 2018 test suite (D = 30) are depicted in Fig. 6. The rest of convergence graphs on all dimensions of the IEEE CEC 2018 test suite are shown as Figs. S1–S4 provided in the **supplementary file**. Fig. 6 shows that IDE-EDA is slightly better than AAVS-EDA, ELSHADE-SPACMA, LSHADE-RSP, jSO, and RW-GEDA, slightly worse than EB-LSHADE on F1 and F6 in terms of convergence performance. As for F13, F16, F24, and F29, the convergence speed of AAVS-EDA is worse than the other six algorithms, which do not show significant differences in terms of convergence speed. In conclusion, IDE-EDA provides an excellent performance in terms of convergence performance on the IEEE CEC 2018 test suite.

5.2.4. Analysis of algorithm complexity

In this section, the algorithm complexity is evaluated according to the instructions in the IEEE CEC 2018 test suite [4] in which the algorithm complexity is reflected by $(\bar{T}_2 - T_1)/T_0$. T_0 denotes the time running the code listed in Table 5, and the

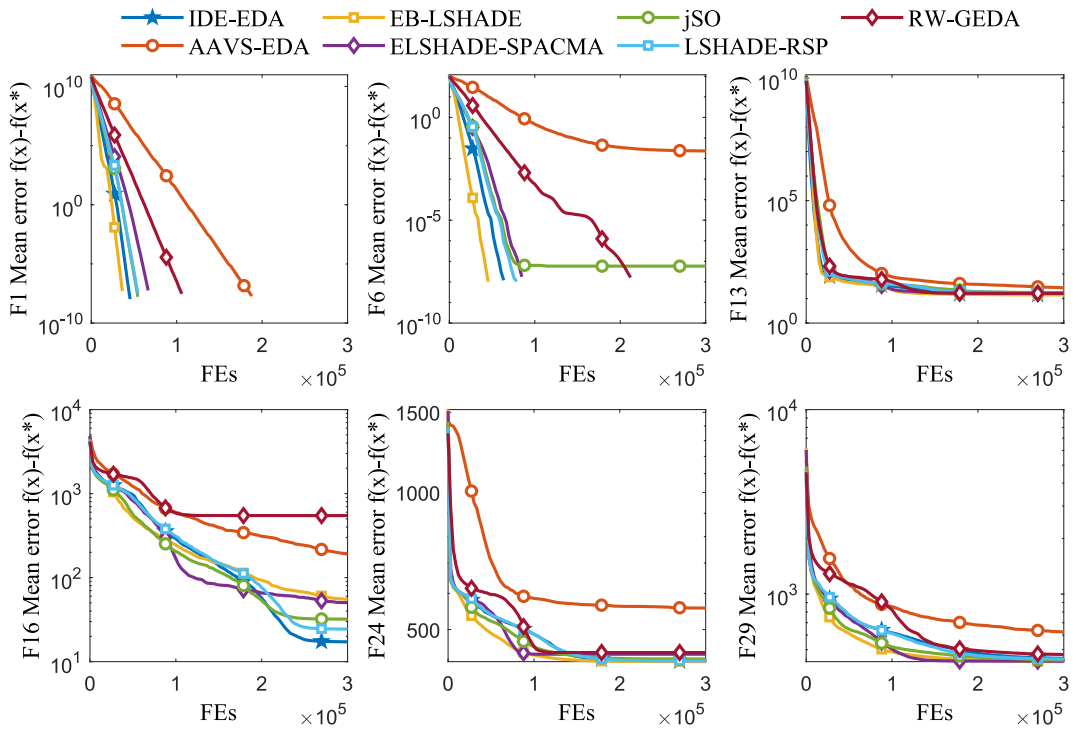

Fig. 6. Convergence curves of IDE-EDA and six competitors on six functions sampled from the IEEE CEC 2018 test suite (30D).

Table 5
Code for calculating the time T_0.

Input: *iteration* = 1,000,000
Output: T_0
Step 1. tic;
Step 2. y = 0.55;
Step 3. for t = 1: *iteration*
Step 4. $y = y + y$; $y = y/2$;
Step 5. $y = y^2$; y = sqrt(y);
Step 6. y = log(y); y = exp(y);
Step 7. $y = y/(y + 2)$;
Step 8. end
Step 9. T_0 = toc;

meaning of T_1, T_2 and $\bar{T}_2$ can refer to [44]. In this section, all parameters except the initial (maximum) population size NP_{init} (NP) of IDE-EDA and six competitors are set as listed in Table 1, and the initial (maximum) population size NP_{init} (NP) for all algorithms are always set to 500 on all dimensions in the IEEE CEC 2018 test suit to make a fair and rational comparison of computational complexity.

The computational complexities of IDE-EDA, AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA are listed in Table 6. For a more intuitive comparison, the results listed in Table 6 are depicted in Fig. 7. From Fig. 7, IDE-EDA is more time-consuming than AAVS-EDA, LSHADE-RSP, and RW-GEDA, but less time-consuming than ELSHADE-SPACMA on all dimensions. In addition, the computational complexity of IDE-EDA increases linearly with the dimension. The main reason for the computational complexity of IDE-EDA being slightly higher than those of LSHADE-RSP and AAVS-EDA is that there are two evolutionary stages in each generation. In particular, the calculation of covariance matrix $\boldsymbol{C}$ in EDA adds extra time consumption. Currently, complexity increases widely in hybrid algorithms, which requires further study.

5.3. Comparison with state-of-the-art algorithms using the IEEE CEC 2021 test suite

In order to further measure the performance of IDE-EDA against shifted, rotated, and biased functions, the IEEE CEC 2021 test suite is conducted. The detailed results derived by IDE-EDA, AAVS-EDA, APGSK-IMODE, EB-LSHADE, HSES, ELSHADE-SPACMA, and RW-GEDA for the IEEE CEC 2021 test suite (D = 20) are shown in Table S4 provided in the **supplementary file**. In the IEEE CEC 2021 test suite, each benchmark function has five different transformations that are parameterized by bias, rotation, and shift [7].

5.3.1. Analysis of the results derived by the Wilcoxon signed-rank test

The Wilcoxon signed-rank test results with a significance level α = 0.05 for five transformations of the IEEE CEC 2021 test suite are summarized in Table 7. The detailed results of the Wilcoxon signed-rank test are presented in Table S4 provided in the **supplementary file**. Details are given below.

a) For biased functions, IDE-EDA is superior (inferior) to AAVS-EDA on 7(0) functions, APGSK-IMODE on 0(5) functions, EB-LSHADE on 0(3) functions, ELSHADE-SPACMA on 1(4) functions, HSES on 5(1) functions, and RW-GEDA on 0(5) functions. That is to say that IDE-EDA is superior to AAVS-EDA and HSES, but inferior to the other competitors in terms of biased functions in the IEEE CEC 2021 test suite.

Table 6
Computational complexities of IDE-EDA and six competitors.

Algorithm	$\bar{T}_2$/s			$(\bar{T}_2 - T_1)/T_0$		
	10D	30D	50D	10D	30D	50D
IDE-EDA	0.54282	1.03610	1.65923	43.93008	69.82731	97.20672
AAVS-EDA	0.16668	0.52491	1.04715	8.76097	22.03061	39.97706
EB-LSHADE	0.88823	1.25433	1.80810	76.22601	90.23242	111.12588
ELSHADE-SPACMA	0.66203	1.08634	1.78693	55.07628	74.52514	109.14714
jSO	0.61642	0.92187	1.42058	50.81212	59.14642	74.89273
LSHADE-RSP	0.39432	0.66194	1.12168	30.04565	34.84339	46.94486
RW-GEDA	0.37563	0.87060	1.48904	28.29825	54.35258	81.29417

T_0 = 0.01070 s, T_1 = 0.07298 s (10D), T_1 = 0.28929 s(30D), T_1 = 0.61960 s (50D).

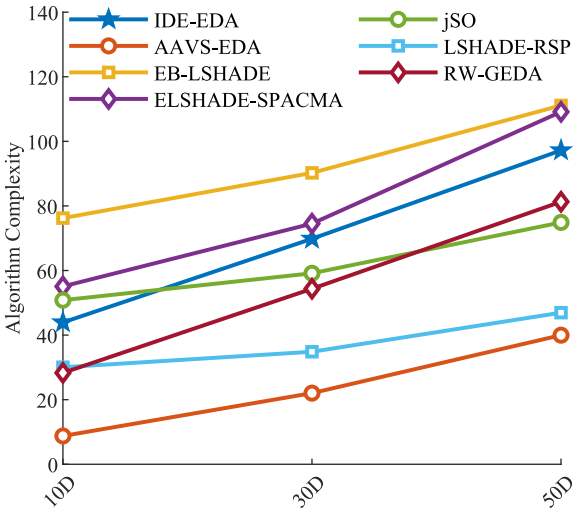

Fig. 7. Algorithm complexities of IDE-EDA and six competitors.

b) For shifted functions, IDE-EDA is superior (inferior) to AAVS-EDA on 8(1) functions, APGSK-IMODE on 1(4) functions, EB-LSHADE on 1(4) functions, ELSHADE-SPACMA on 0(5) functions, HSES on 7(2) functions, and RW-GEDA on 7(0) functions. That is to say that IDE-EDA is superior to AAVS-EDA, HSES, and RW-GEDA, but inferior to the other competitors on shifted functions in the IEEE CEC 2021 test suite.
c) For biased and shifted functions, IDE-EDA is superior (inferior) to AAVS-EDA on 6(2) functions, APGSK-IMODE on 2(4) functions, EB-LSHADE on 1(5) functions, ELSHADE-SPACMA on 0(5) functions, HSES on 7(2) functions, and RW-GEDA on 7(0) functions. That is to say that IDE-EDA is superior to AAVS-EDA, HSES, and RW-GEDA, but inferior to the other competitors on biased and shifted functions in the IEEE CEC 2021 test suite.
d) For shifted and rotated functions, IDE-EDA is superior (inferior) to AAVS-EDA on 5(3) functions, APGSK-IMODE on 5(3) functions, EB-LSHADE on 3(2) functions, ELSHADE-SPACMA on 2(2) functions, HSES on 8(1) functions, and RW-GEDA on 9(0) functions. That is to say that IDE-EDA is superior to AAVS-EDA, APGSK-IMODE, EB-LSHADE, HSES, and RW-GEDA, and shows the same performance as ELSHADE-SPACMA on shifted and rotated functions in the IEEE CEC 2021 test suite.
e) For biased, shifted, and rotated functions, IDE-EDA is superior (inferior) to AAVS-EDA on 4(4) functions, APGSK-IMODE on 5(2) functions, EB-LSHADE on 3(2) functions, ELSHADE-SPACMA on 2(2) functions, HSES on 7(1) functions, and RW-GEDA on 9(0) functions. That is to say that IDE-EDA is superior to APGSK-IMODE, EB-LSHADE, HSES, and RW-GEDA, and shows the same performance as AAVS-EDA and ELSHADE-SPACMA on biased, shifted, and rotated functions in the IEEE CEC 2021 test suite.

The conclusion supported by the Wilcoxon signed-rank test results is that IDE-EDA outperforms AAVS-EDA, HSES, and RW-GEDA, but is worse than ELSHADE-SPACMA on the IEEE CEC 2021 test suite, and APGSK-IMODE, EB-LSHADE, and IDE-EDA show different advantages in different transformation for functions in the IEEE CEC 2021 test suite.

5.3.2. Analysis of the results derived by the Friedman test

The Friedman test results with a significance level $\alpha = 0.05$ for five transformations of the IEEE CEC 2021 test suite to illustrate the differences between IDE-EDA, AAVS-EDA, APGSK-IMODE, EB-LSHADE, ELSHADE-SPACMA, HSES, and RW-GEDA are listed in Table 8. All Friedman-*p-values* listed in Table 8 are less than α, that is, IDE-EDA and six competitors show significant

Table 7
Results derived by the Wilcoxon signed-rank test between IDE-EDA and six competitors ($\alpha = 0.05$).

R (+/−/=)	Bias (000)	Shift (010)	Bias and shift (110)	Shift and rotation (011)	Bias, shift, and rotation (111)
IDE-EDA vs AAVS-EDA	7/0/3	8/1/1	6/2/2	5/3/2	4/4/2
IDE-EDA vs APGSK-IMODE	0/5/5	1/4/5	2/4/4	5/3/2	5/2/3
IDE-EDA vs EB-LSHADE	0/3/7	1/4/5	1/5/4	3/2/5	3/2/5
IDE-EDA vs ELSHADE-SPACMA	1/4/5	0/5/5	0/5/5	2/2/6	2/2/6
IDE-EDA vs HSES	5/1/4	7/2/1	7/2/1	8/1/1	7/1/2
IDE-EDA vs RW-GEDA	0/5/5	7/0/3	7/0/3	9/0/1	9/0/1

Table 8
Rankings of IDE-EDA and six competitors in the IEEE CEC 2021 test suite (20D) obtained by the Friedman test (α = 0.05).

Algorithms	Bias	Shift	Bias and shift	Shift and rotation	Bias, shift, and rotation
IDE-EDA	4.20	3.75	3.80	2.65	2.80
AAVS-EDA	5.75	4.90	4.35	3.70	3.35
APGSK-IMODE	2.30	2.20	2.50	3.25	3.30
EB-LSHADE	3.65	2.90	3.10	2.95	3.20
ELSHADE-SPACMA	3.45	2.95	2.95	3.45	3.35
HSES	5.90	5.10	5.10	5.60	5.60
RW-GEDA	2.75	6.20	6.20	6.40	6.40
Friedman-p-value	3.21E-06	2.69E-05	1.52E-04	4.61E-05	9.41E-05

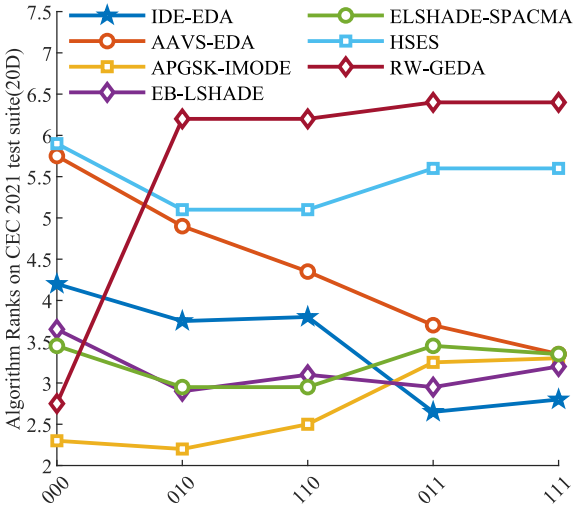

Fig. 8. Rankings of algorithms derived by the Friedman test.

performance difference in the IEEE CEC 2021 test suite. To show the results more visually, the rankings of IDE-EDA and six competitors derived by the Friedman test are shown in Fig. 8.

From Fig. 8, IDE-EDA performs better than AAVS-EDA and HSES in all transformations of the IEEE CEC 2021 test suite. IDE-EDA performs better than RW-GEDA in all transformations except Bias (000) of the IEEE CEC 2021 test suite. Moreover, IDE-EDA outperforms APGSK-IMODE, ELSHADE-SPACMA, and EB-LSHADE on the transformations of shift and rotation (011) and bias, shift, and rotation (111) of the IEEE CEC 2021 test suite, but inferior to them on the transformations of bias (000), shift (010), bias and shift (110) of the IEEE CEC 2021 test suite.

6. Conclusions and future work

In the work, the IDE-EDA, a novel hybrid algorithm by combining LSHADE-RSP and EDA, is proposed by using a novel cooperative evolutionary framework to enhance its performance. The excellent performance of IDE-EDA has been validated by comparing with cutting-edge DE-based and EDA-based variants. The statistical test results on the IEEE CEC 2018 test suite (10D, 30D, and 50D) demonstrate that IDE-EDA outperforms the state-of-the-art DE-based and EDA-based variants including AAVS-EDA, EB-LSHADE, jSO, ELSHADE-SPACMA, LSHADE-RSP, and RW-GEDA. The statistical test results on the IEEE CEC 2021 test suite (20D) demonstrate that IDE-EDA also shows excellent performance against shifted, rotated, and biased functions by comparing with state-of-the-art algorithms. In summary, it can be concluded that IDE-EDA proposed in this work is an excellent hybrid algorithm.

In future work, there are two possible directions that can be addressed. Firstly, the algorithm complexity of IDE-EDA may be reduced by optimizing the co-evolutionary framework and adopting more efficient programming language. Secondly, IDE-EDA will be adopted to solve some challenging real-world problems, such as cooperative trajectory planning and target assignment of UAV swarm operations.

CRediT authorship contribution statement

Yintong Li: Conceptualization, Methodology, Software, Data curation, Writing – original draft. **Tong Han:** Data curation, Investigation, Validation. **Shangqin Tang:** Visualization, Investigation, Writing – review & editing. **Changqiang Huang:**

Supervision, Methodology. **Huan Zhou:** Software, Validation, Funding acquisition. **Yuan Wang:** Validation, Writing – review & editing.

Data availability

Data will be made available on request. The MATLAB source code of IDE-EDA can be downloaded from https://github.com/Yintong-Li/IDE-EDA.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China [grant numbers 62101590]; the Natural Science Foundation of Shaanxi Province [grant numbers 2021JM-224, 2021JM-223, 2022JQ-584].

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ins.2022.11.029.

References

[1] K. De Jong, Evolutionary computation, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, ACM, New York, NY, USA, 2020: pp. 327–342. 10.1145/3377929.3389871.
[2] A.W. Mohamed, A.A. Hadi, A.K. Mohamed, Differential evolution mutations: taxonomy, comparison and convergence analysis, IEEE Access. 9 (2021) 68629–68662, https://doi.org/10.1109/ACCESS.2021.3077242.
[3] K.M. Sallam, M.A. Hossain, R.K. Chakrabortty, M.J. Ryan, An improved gaining-sharing knowledge algorithm for parameter extraction of photovoltaic models, Energy Convers. Manage. 237 (2021), https://doi.org/10.1016/j.enconman.2021.114030 114030.
[4] N.H. Awad, M.Z. Ali, J. Liang, B.Y. Qu, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2017 special session and competition on real-parameter optimization, 2016.
[5] N.H. Awad, M.Z. Ali, J. Liang, B.Y. Qu, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization, 2013.
[6] P.N.S.G.W. R. Mallipeddi, Problem definitions and evaluation criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization, 2010. http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC10-Const/TR-April-2010.pdf.
[7] A.W. Mohamed, A.A. Hadi, A.K. Mohamed, P. Agrawal, A. Kumar, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2021 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization, 2020.
[8] R. Storn, K. Price, Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optimiz. 11 (1997) 341–359, https://doi.org/10.1023/A:1008202821328.
[9] J. Teo, Exploring dynamic self-adaptive populations in differential evolution, Soft Comput. 10 (2006) 673–686, https://doi.org/10.1007/s00500-005-0537-1.
[10] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (2006) 646–657, https://doi.org/10.1109/TEVC.2006.872133.
[11] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (2009) 945–958, https://doi.org/10.1109/TEVC.2009.2014613.
[12] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for Differential Evolution, in: 2013 IEEE Congress on Evolutionary Computation, CEC 2013, 2013: pp. 71–78. 10.1109/CEC.2013.6557555.
[13] R. Tanabe, A.S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014: pp. 1658–1665. 10.1109/CEC.2014.6900380.
[14] J. Brest, M.S. Maučec, B. Bošković, IL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization, in: 2016 IEEE Congress on Evolutionary Computation, CEC 2016, 2016: pp. 1188–1195. 10.1109/CEC.2016.7743922.
[15] N.H. Awad, M.Z. Ali, P.N. Suganthan, R.G. Reynolds, An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems, in: 2016 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2016: pp. 2958–2965. 10.1109/CEC.2016.7744163.
[16] P. Ochoa, O. Castillo, J. Soria, High-speed interval type-2 fuzzy system for dynamic crossover parameter adaptation in differential evolution and its application to controller optimization, Int. J. Fuzzy Syst. 22 (2020) 414–427, https://doi.org/10.1007/s40815-019-00723-w.
[17] P. Ochoa, O. Castillo, P. Melin, J. Soria, Differential evolution with shadowed and general type-2 fuzzy systems for dynamic parameter adaptation in optimal design of fuzzy controllers, Axioms 10 (2021) 194, https://doi.org/10.3390/axioms10030194.
[18] J. Sun, X. Liu, T. Back, Z. Xu, Learning adaptive differential evolution algorithm from optimization experiences by policy gradient, IEEE Trans. Evol. Comput. 25 (2021) 666–680, https://doi.org/10.1109/TEVC.2021.3060811.
[19] P. Bujok, J. Tvrdik, R. Polakova, Evaluating the performance of SHADE with competing strategies on CEC 2014 single-parameter test suite, in: 2016 IEEE Congress on Evolutionary Computation, CEC 2016, IEEE, 2016: pp. 5002–5009. 10.1109/CEC.2016.7748322.
[20] J. Brest, M.S. Maucec, B. Boskovic, Single objective real-parameter optimization: Algorithm jSO, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017: pp. 1311–1318. 10.1109/CEC.2017.7969456.
[21] V. Stanovov, S. Akhmedova, E. Semenkin, LSHADE Algorithm with Rank-Based Selective Pressure Strategy for Solving CEC 2017 Benchmark Problems, in: 2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings, IEEE, 2018: pp. 1–8. 10.1109/CEC.2018.8477977.
[22] A.W. Mohamed, A.A. Hadi, K.M. Jambi, Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization, Swarm Evol. Comput. 50 (2019), https://doi.org/10.1016/j.swevo.2018.10.006 100455.
[23] X. Xia, L. Tong, Y. Zhang, X. Xu, H. Yang, L. Gui, Y. Li, K. Li, NFDDE: a novelty-hybrid-fitness driving differential evolution algorithm, Inform. Sci. 579 (2021) 33–54, https://doi.org/10.1016/j.ins.2021.07.082.
[24] V. Stanovov, S. Akhmedova, E. Semenkin, NL-SHADE-RSP Algorithm with Adaptive Archive and Selective Pressure for CEC 2021 Numerical Optimization, in: 2021 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2021: pp. 809–816. 10.1109/CEC45853.2021.9504959.

[25] A. Kumar, P.P. Biswas, P.N. Suganthan, Differential evolution with orthogonal array-based initialization and a novel selection strategy, Swarm Evol. Comput. 68 (2022), https://doi.org/10.1016/j.swevo.2021.101010 101010.
[26] Z. Zeng, Z. Hong, H. Zhang, M. Zhang, C. Chen, Improving differential evolution using a best discarded vector selection strategy, Inform. Sci. 609 (2022) 353–375, https://doi.org/10.1016/j.ins.2022.07.075.
[27] Z. Cao, Z. Wang, Y. Fu, H. Jia, F. Tian, An adaptive differential evolution framework based on population feature information, Inform. Sci. 608 (2022) 1416–1440, https://doi.org/10.1016/j.ins.2022.07.043.
[28] N.H. Awad, M.Z. Ali, P.N. Suganthan, Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems, in: 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings, 2017: pp. 372–379. 10.1109/CEC.2017.7969336.
[29] A.W. Mohamed, A.A. Hadi, A.M. Fattouh, K.M. Jambi, LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017: pp. 145–152. 10.1109/CEC.2017.7969307.
[30] A.A. Hadi, A.W. Mohamed, K.M. Jambi, Single-Objective Real-Parameter Optimization: Enhanced LSHADE-SPACMA Algorithm, in: Studies in Computational Intelligence, 2021: pp. 103–121. 10.1007/978-3-030-58930-1_7.
[31] A.W. Mohamed, A.K. Mohamed, Adaptive guided differential evolution algorithm with novel mutation for numerical optimization, Int. J. Mach. Learn. Cybernet. 10 (2019) 253–277, https://doi.org/10.1007/s13042-017-0711-7.
[32] G. Zhang, Y. Shi, H.S.E. Strategy, for Solving Single Objective Bound Constrained Problems, in,, IEEE Congress on Evolutionary Computation (CEC), IEEE 2018 (2018) 1–7, https://doi.org/10.1109/CEC.2018.8477908.
[33] A.W. Mohamed, A.A. Hadi, P. Agrawal, K.M. Sallam, A.K. Mohamed, Gaining-Sharing Knowledge Based Algorithm with Adaptive Parameters Hybrid with IMODE Algorithm for Solving CEC 2021 Benchmark Problems, in: 2021 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2021: pp. 841–848. 10.1109/CEC45853.2021.9504814.
[34] F. Zhao, H. Bao, L. Wang, X. He, Jonrinaldi, A hybrid cooperative differential evolution assisted by CMA-ES with local search mechanism, Neural Computing and Applications. 34 (2022) 7173–7197. 10.1007/s00521-021-06849-z.
[35] P. Larrañaga, J.A. Lozano, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, 2002. http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0792374665.
[36] A. Zhou, J. Sun, Q. Zhang, An estimation of distribution algorithm with cheap and expensive local search methods, IEEE Trans. Evol. Comput. 19 (2015) 807–822, https://doi.org/10.1109/TEVC.2014.2387433.
[37] S. Pang, W. Li, H. He, Z. Shan, X. Wang, An EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing, IEEE Access. 7 (2019) 146379–146389, https://doi.org/10.1109/ACCESS.2019.2946216.
[38] Z. Ren, Y. Liang, L. Wang, A. Zhang, B. Pang, B. Li, Anisotropic adaptive variance scaling for Gaussian estimation of distribution algorithm, Knowl. Based Syst. 146 (2018) 142–151, https://doi.org/10.1016/j.knosys.2018.02.001.
[39] X. Wang, H. Zhao, T. Han, Z. Wei, Y. Liang, Y. Li, A gaussian estimation of distribution algorithm with random walk strategies and its application in optimal missile guidance handover for multi-UCAV in over-the-horizon air combat, IEEE Access. 7 (2019) 43298–43317, https://doi.org/10.1109/ACCESS.2019.2908262.
[40] Y. Liang, Z. Ren, X. Yao, Z. Feng, A. Chen, W. Guo, Enhancing gaussian estimation of distribution algorithm by exploiting evolution direction with archive, IEEE Trans. Cybernet. 50 (2020) 140–152, https://doi.org/10.1109/TCYB.2018.2869567.
[41] L. Tang, X. Song, J. Liu, C. Liu, An estimation of distribution algorithm with filtering and learning, IEEE Trans. Automat. Sci. Eng. 18 (2021) 1478–1491, https://doi.org/10.1109/TASE.2020.3019694.
[42] C. Wang, H. Ma, G. Chen, S. Hartmann, Using an estimation of distribution algorithm to achieve multitasking semantic web service composition, IEEE Trans. Evol. Comput. (2022) 1–15, https://doi.org/10.1109/TEVC.2022.3170899.
[43] M. Hollander, D.A. Wolfe, E. Chicken, Nonparametric statistical methods, Wiley (2015), https://doi.org/10.1002/9781119196037.
[44] Y. Li, T. Han, H. Zhou, S. Tang, H. Zhao, A novel adaptive L-SHADE algorithm and its application in UAV swarm resource configuration problem, Inform. Sci. 606 (2022) 350–367, https://doi.org/10.1016/j.ins.2022.05.058.
[45] J.M. Davenport, Approximations of the critical region of the friedman statistic, Commun. Statist. Theory Methods 9 (1980) 571–595, https://doi.org/10.1080/03610928008827904.