Information Sciences 514 (2020) 203–233

Contents lists available at ScienceDirect

Information Sciences

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

Insights into the effects of control parameters and mutation strategy on self-adaptive ensemble-based differential evolution

Shir Li Wang[a], Farid Morsidi[a], Theam Foo Ng[b,*], Haldi Budiman[a,d], Siew Chin Neoh[c]

[a] Faculty of Art, Computing and Creative Industry, Universiti Pendidikan Sultan Idris, 35900 Tanjong Malim, Perak, Malaysia
[b] Centre for Global Sustainability Studies, Universiti Sains Malaysia, Penang 11800, Malaysia
[c] School of Computer Science, Faculty Science, University Notthingham Malaysia, Jln Broga, Semenyih, Selangor 43500, Malaysia
[d] Fakultas Teknologi Informasi, Universitas Islam Kalimantan, Muhammad Arsyad AL-Banjar, Jl. Adhiyaksa No. 2 Kayutangi Banjarmasin, Kalimantan Selatan 70123, Indonesia

ARTICLE INFO

Article history:
Received 19 April 2019
Revised 20 November 2019
Accepted 25 November 2019
Available online 26 November 2019

Keywords:
Differential evolution
Self-adaptive
Evolutionary algorithm
Ensemble
Control parameters
Mutation strategies

ABSTRACT

This work explores the challenges in identifying appropriate and significant parameter configurations in differential evolution (DE) under the influence of population diversity and dimension size. For most DE algorithms, the configuration of control parameters is a vital prerequisite for balancing exploration and exploitation within the confinement of a search space. This study investigates the implementation of various adaptive parameter setting configurations on benchmark functions via the proposal of an algorithmic scheme called self-adaptive ensemble-based DE (SAEDE). This algorithm uses self-adaptive and ensemble mechanisms to set the relevant parameters for each generation. SAEDE is compared with two other ensemble-based DEs, and their performance is evaluated using 34 benchmark functions consisting of 20 low dimensions and 14 high dimensions. Furthermore, the convergence of these DEs is tested by using *Q*-measure. Experimental results indicate that SAEDE achieves the highest frequency of maximum success rate in 28 out of the 34 benchmark functions. SAEDE also achieves the lowest *Q*-measure of 4237318. These findings show the competitiveness and efficiency of SAEDE in locating optimal solutions while avoiding exhaustive searches of suitable parameters by users in terms of achieving optimization while minimizing the dependency on user setting.

1. Introduction

Artificial intelligence (AI) has been widely applied in various domains owing to its capability to solve uncertain and complex problems. Previously, AI was used to obtain solutions for stationary optimization and static environments, but this purpose has changed. The computational paradigm in AIs capability has shifted. Instead of solving static problems, AI solves dynamic and complex problems, such as complexity science. In-depth understanding on complexity science, especially problems related to dynamic human populations, is beneficial for risk mitigation in various scenarios, such as crowd disasters,

* Corresponding author.
E-mail addresses: shirli_wang@fskik.upsi.edu.my (S.L. Wang), farid.mors90@gmail.com (F. Morsidi), tfng@usm.my (T.F. Ng), haldibudiman01@gmail.com, haldibudiman@uniska-bjm.ac.id (H. Budiman), SiewChin.Neoh@nottingham.edu.my (S.C. Neoh).

https://doi.org/10.1016/j.ins.2019.11.046

crime, terrorism, wars, and disease epidemics [1]. Improved understanding of these problems leads to superior counterattack and prevention strategies, thus minimizing possible casualties.

Classical approaches are no longer sufficient for studying problems related to complexity science. Instead, AI, specifically evolutionary algorithm (EA), has considerable potential for complexity science because its characteristics share a similarity with the nature of complex system behavior. Approaches adopted for complex systems need to be able to self-organize with minimum interference, deal with the unpredictable dynamics of complex systems, and allow the integration of interaction and institutional settings [1]. An evolutionary-based framework known as multiobjective evolutionary-based risk assessment, which was introduced by Abbass et al. [2], potentially offers solutions along these lines. The framework was proposed to explore and evaluate different strategies under risk for the considered scenarios. With increased problem complexity and decreased computation time, EAs should be simple, robust, and computationally efficient. An EA that meets these requirements is differential evolution (DE).

From the introduction of its development, DE has been constituted with several baseline framework branches, such as initialization, crossover, mutation, and selection. The minimized number of steps required to compute a complete solution in terms of convergence speed and robustness makes DE a preferable option for achieving approximate solutions in a limited computational time. Self-adaptive and ensemble approaches have been developed in recent years to enable the dynamical tuning of control parameters in the search process. The performance of DE is significantly dependent on core parameter variables, such as scale factor, *F*, and crossover rate, *CR* [3]. However, the proper setting of these parameters requires prior knowledge. Trial-and-error parameter tuning has been implemented as an alternative to supplement optimum parameter tuning. However, this method is computationally expensive and sacrifices key features in exploration or exploitation; thus, it is not a permanent solution. Although DE is a popular method among optimization algorithms, it tends to suffer from slow convergence speed when encountering complicated problems such as an increase in the dimensionality of optimization problems [4]. The dimensionality of problems refers to the number of attributes in a solution vector for the underlying problem. By representing the solution vector as $\mathbf{x} = \{x^1, x^2, \ldots, x^D\}$, the dimensionality of the problem is denoted as *D*.

Dynamic mechanisms, such as self-adaptation and ensemble, have been proposed to tackle the robustness and overall performance of fundamental DE algorithms. Despite the promising results from the incorporation of self-adaptive features with traditional EA frameworks, such as DE, most of the implementations are not viable for overcoming multiple optimization problems and are limited to the execution of single evolution algorithm approaches [3]. The motive of using self-adaptive mechanisms is to allow adaptability to various strategies at different stages for the optimization stages depending on the output generated from mutation strategies. Recent research has indicated that different control parameter configurations and mutation strategies are required in various stages. Another issue to be addressed is the effectiveness of adaptive measures in improving DE optimization performance. Four categorization aspects reflecting the importance of parameter setting methods imposed during mutation and crossover strategies were highlighted in [5]. These aspects are scrutinized in the current work, wherein the classification of each of the research strategies relating to parameter variables is represented by the following segments:

1. Modifications on DE parameters: This study reviews the observations on DE parameters with mutation strategy (*M*), *F*, *CR*, and population size (*NP*) preferred to achieve optimal results and cater to different problems. The preference of *NP* is investigated in the scale of the dimensionality of problems.
2. How or where alterations are performed: Self-adaptive features proposed in several related studies are further explored in context. Parameter values that could be adapted to augment the further optimization of each algorithm execution is tested as suggested by the selected frameworks. Minute details of each progression are noted, especially propagation on the next generations and convergence. The motive is to determine the extent to which adaptive elements can affect stagnation.
3. Modification range: The self-adaptive mechanism is prioritized, as each main parameter variable is adaptive on the basis of a certain range or the dimensionality of problems. This aspect helps in determining whether stagnation would be in effect if parameter variables are not fixed and dynamically alter on the basis of internal catalysts.
4. Results that support the changes: End output is observed intently after the self-adaptive mechanism is imposed. Evaluation is performed in relation to the degree of function evaluations, fitness achievement, success rate, and stagnation.

The remainder of this paper is organized as follows: Section 2 describes works related to DE. Section 3 presents the methodology of our proposed DE model, and Section 4 provides the experimental setup. Section 5 discusses the experimental results through the analysis methodology. Finally, Section 6 presents the conclusions and future work.

The remainder of this paper is organized as follows: Section 2 describes works related to DE. Section 3 presents the methodology of our proposed DE model, and Section 4 describes the experimental setup. Section 5 discusses the experimental results through the analysis methodology. Finally, Section 6 presents the conclusions and future work.

2. Literature review

A myriad of variable generation strategies, together with adaptive machinations, has been imposed on DE to augment its existing advantageous traits, particularly in terms of simplicity, robustness, or computational efficacy. DE exhibits remarkable performance while optimizing a wide variety of objective functions in terms of final accuracy, computational speed, and robustness [6]. DE and its variants, such as self-adaptive DE (SaDE) [7], ensemble of mutation and crossover strategies and parameters in DE (EPSDE) [8] and ensemble of multiple DE variants (EDEV) [9], produce better global optimal results than genetic algorithms and particle swarm optimization algorithms [10]. However, DE remains plagued by several issues related to the configuration of parameter settings, such as mutation strategy, scale factor, crossover rate, population size, and (especially) problem dimensionality. These problems severely affect the solution steps that involve multiple and singular objectives.

2.1. Influence of parameter setting

The setting of parameters is heavily problem dependent. Improper selection of mutation and crossover strategies and the associated parameters may lead to premature convergence, stagnation, or wastage of computational resources [4]. No single fixed value exists for each parameter for solving all types of problems with a reasonable solution quality [3]. The optimization capability of DE remains insufficient, given that DE performance may deteriorate quickly with increased search space dimensionality [11]. An increase in dimension number exponentially decreases the volume of initialization space compared with that of symmetric initialization. On complex problems, the performance of DE becomes highly sensitive to the mutation strategy and associated parameter values.

DE behaves linearly in solving numerous optimization issues, although the efficacy relies highly upon predetermined offspring generation strategies, such as mutation and crossover strategies, and the subsequent control parameters mentioned in this work, which mainly consist of F, CR, and NP. The performance of DE is highly dependent on the parent selection strategies, which are also known as mutation strategies. The performance of mutation strategies depends on the participation of multiple parents and affects the convergence and function-solving capability. Alternatives for improving the performance of DE through mutation strategies include pool selection for offspring generation strategies, together with the improvisation of control parameters depending on the success rates of the generated trial vectors [6]. Problems arise when multiple generation strategies need to be produced to induce solution steps for a pool of mutation strategies within predictable ranges. Future generations rely upon the mutation scheme for the previous generated trial [6]. The use of a global mutation operator along the evolution is unsuitable for multimodal optimization problems, which require parallel and distributed convergence to distinct optimum positions [3]. Only after analyzing trial vector generating strategies can good values for the control parameters, such as F and CR, be acquired [12]. The effects of these control parameters are discussed in the following paragraph.

The efficiency of DE is sensitive to the setting of F and CR. F is randomly generated within the range [0.5, 1.0] depending on CR, whether binomial, exponential, or a hybridization of both [4]. The selection of a crossover variant over the other is difficult, given that no results establish the superiority of any one variant for a given type of problem. Control parameter F should be set above a certain critical value to avoid premature convergence to a suboptimal solution [3]. F can also be altered between trials to cater toward each individual algorithm [3,13]. In this regard, the generation strategy should be decided before optimization problems are resolved.

F and CR affect the convergence velocity and robustness of the search process. Setting CR exactly to 1.0 cannot be recommended because potential solutions will be dramatically decreased. A large value of CR increases the population diversity and improves the convergence speed, whereas a small value increases the possibilities of stagnation and slows down the search process [3]. Completely eliminating the crossover leads to poor performance of DE if the underlying problems are multimodal [14]. In reality, we rarely know in advance whether the given problems are unimodal or multimodal. Therefore, the setting of CR becomes increasingly challenging. If CR is too small, then DE needs additional generations to identify the minimum; it might even fail to find the minimum [15].

The optimization capability of DE remains insufficient because DE performance may deteriorate quickly with increased search space dimensionality. The limitation is related closely to population diversity. When the best individual in a population with low diversity is far from the global optimum or located in a local minimum, other individuals may easily be attracted to the local optimum region, leading to premature convergence. Therefore, the convergence can be accelerated but with the trade-off of the entire population retracting to a local optimum.

Insufficient population size may cause premature convergence and local saturation [12]. Most DE algorithms perform well with small population sizes because a low NP reduces explorative and greedy behavior. A large NP can extend the exploring space and the chance of discovering possible good solutions. However, the use of large NP suffers from high computational cost and slow convergence velocity because it requires a large number of function evaluations [16]. In other words, a large NP can sustain diversity, but the search is compromised with the drawback in convergence rate.

The rule of thumb in setting a reasonable NP lies in the dimensionality of problems, i.e., within [$5D$ $10D$], as suggested by Storn and Price [17]. According to [15], NP should be within [$3D$ $8D$] instead for a specific value of F and range of CR. However, population diversity cannot be guaranteed with this setting because of a lack of investigation of the relationship between NP settings and problem dimensionality D. According to [7], NP does not require considerable tuning and can be set on the basis of a few typical values that can be pre-estimated depending on the complexity of the problem. This conclusion contradicts that in [7,17], which weighted the effect of NP differently. Furthermore, problem characteristics are rarely known beforehand.

The conclusion drawn with regard to NP is that its setting causes a trade-off in achieving low computation cost and high diversity. By considering such trade-off and the dimensionality of problems, efforts for optimizing and determining NP should be lefted on the population size itself in the scale of D. A trade-off should be balanced in optimizing NP (whether to sacrifice by having a high computational cost for improved objective function outcomes). Setting appropriate *NP* to balance convergence velocity and convergence maturity remains challenging.

2.2. Methods of parameter setting

Various DE parameter tuning methods have been suggested, and the related work has been summarized in [8,18]. Even though some ranges for the control parameters and their pre-conditions have been suggested by the researchers, the setting has no clear and consistent justification. The only conclusion that can be obtained is that *F* needs to be large, whereas *CR* needs to be small in exploration, and vice versa in exploitation.

Instead, an increasing number of studies have been focusing on the use of self-adaption as the popular approach in setting control parameters and mutation strategies. Self-adaptation enables DE to determine the setting automatically on the basis of rules or schemes with minimal manual tuning from users. The use of self-adaptation in parameter setting enables a model to update its control parameters to appropriate values with minimum prior knowledge and experience about the algorithm and the characteristics of the problem. However, the complexity of parameter tuning increases when adaptation is implemented to DE models, which experience different levels of modification, i.e., genetic operator, evolutionary structure, and hybrid. The following paragraphs describe the research related to parameter setting in modified DEs.

2.2.1. Modification in genetic operator

Modification at the level of genetic operator is restricted to changes occurring to the genetic operator of mutation, crossover, and selection, but the evolutionary structure remains the same. For example, in JADE, a new mutation strategy DE/current-to-*p*best with optional external archive and an adaptive method of tuning control parameters is introduced. For the adaptive tuning of control parameters, two new parameters are introduced, namely, *c* and *p*. The *c* parameter controls the rate of adaptation, whereas the *p* parameter determines the greediness of the mutation strategy [19]. Brest et al. proposed jDE, which introduces two new parameters, namely, τ_1 and τ_2 into DE [20]. These new parameters are responsible for the probability of maintaining or changing the values of *F* and *CR*. With the existing parameters that need to be determined, namely, *CR, F*, and *NP*, a user should determine the appropriate values for τ_1 and τ_2. The jDE model is further improved with the introduction of jDE-2. The jDE-2 model is better than the former because it uses two DE strategies, as opposed to the single strategy in jDE [21]. For research related to modification at the genetic operator level, some parameter values change adaptively, but at least one parameter is held constant. Therefore, a user still needs to determine the constant parameters value. Given that the classical DEs parameter setting lacks appropriate guidelines, introducing new parameters or strategies into DE decreases a users confidence in setting the values.

2.2.2. Modification in structure

The evolutionary structure is modified when the construction of the evolutionary process changes, e.g., the split of a single population to multiple populations. Two SaDE algorithms incorporate a heuristic mix of mutation and crossover operators through an adaptive learning process to determine the parameters, i.e., DE/HMO/1 [22] and DE/HMO/2 [22]. The subpopulation sizes are determined adaptively on the basis of a learning strategy, and the subpopulations are combined to become a single population. Both variants split the population into subpopulations, and local search procedures are used to expedite the convergence of the solutions. Following the concept of "divide and conquer", multiple-subpopulation DE (MPADE) divides a single population into three subpopulations in which three new mutation strategies are adjusted adaptively [11]. Contrary to the subpopulation sizes in "DE/HMO/1" and "DE/HMO/2", that in MPADE is constant. The effect of subpopulation size on the performance of MPADE may require further investigation. Zaharie [23] proposed a parameter adaptation, namely, γ for a multipopulation-operated DE. The parameter γ is responsible for balancing population diversity and convergence. The individuals in each subpopulation are allowed to migrate to other subpopulations on the basis of migration probability p_m. For the proposed DE, a user should determine additional parameters, such as the number of subpopulations, subpopulation size, γ and p_m, in addition to the common parameters. The above mentioned DEs have multiple subpopulations instead of a single population in the evolution. These DEs still lack investigations into the effects of the number and size of subpopulations.

2.2.3. Hybrid

Modification at the hybrid level refers to changes that happen to a DE model when another algorithm, which can be statistical techniques or machine learning, is used to manipulate its evolutionary process. An example of the use of a statistical technique can be found in the research by Omran et al. [24] and Abbass [25], where the normal distributions of mean and standard deviation (std dev) are involved in the adaptations of *F* and *CR*. Both DE models were used to solve multiobjective optimization problems. However, these studies lacked justification for selecting mean and std dev. Liu and Lampinen [26] proposed fuzzy logic for tuning the values of *F* and *CR*. However, the use of fuzzy logic is user dependent because the method requires the users knowledge and prior experience to construct fuzzy rules and membership functions. Besides fuzzy logic, artificial neural networks (ANNs) are used as regression methods to predict the best strategy from an ensemble of mutation strategies in generating better trial vectors within original DE and jDE, namely, nnDE and nnjDE, respectively [27]. In this study, nnDE and nnjDE outperformed their predecessors but were influenced by the ANN regression probability, p_r. The setting of p_r depends on the problem dimensions. Furthermore, several common parameters for original DE and jDE still require user specification in the implementation of their successors.

Nobakhti and Wang used co-evolutionary to determine *F* and balance exploration and exploitation [28]. The concept of co-evolutionary was introduced to create an arm race between the solutions for *F*. These parameters are user specified and set at certain values. The effect of different values for these parameters on DE was not examined in the research.

The work in [29] showed a variant of DE that integrates three algorithms. SaDE [7] was enhanced by JADEs mutation strategy [19] and hybridized with the modified multitrajectory search (MMTS) algorithm; this variant of DE is known as SADE-MMTS. The use of JADEs mutation strategy, i.e., “DE/current-to-*p*best”, means that a user still needs to specify the value of *p*, which represents the top *p*% of solutions that can be randomly selected for differential mutation. Moreover, *NP* for SADE-MMTS was held constant in the research, which could have affected the populations diversity. The involvement of such a large number of algorithms in SADE-MMTS requires an in-depth understanding of the interaction between the algorithms parameters.

An artificial immune system, namely, ISDE, was used to determine the parameters *F* and *CR* in DE [30]. ISDE is inspired by immune functions, principles, and mechanisms in solving problems. In [30], DE is responsible for the search of problem solutions, whereas the immune system is responsible for the search of DEs control parameters. A user must determine the values of three parameters, namely, α, β and vaccination probability (P_v) in this immune system. *NP* in DE was also user specified in this work.

The problem with parameter setting in hybrid DE is that a user needs to have prior experience and knowledge about algorithms aside from DE. Other algorithms usually have their own sets of parameters to be optimized aside from DEs common parameters. Most studies have not further investigated the effect of other algorithms parameters at different values. Thus, their approaches are insufficiently objective, and the complexity of parameter setting in DE is increased.

2.3. Adaptive parameter settings in DE

EAs have emerged as a popular approach for solving various optimization problems in the real world because these algorithms can cope not only with nondifferentiable functions but also with a great number of local minima [31]. Therefore, DE, which is one type of EAs, has been widely applied to solve optimization problems in various fields. However, in most situations, users are interested in solving the underlying optimization problems rather than how to operate the optimization algorithms. This situation is worsened when users are confused with various conflicting and unclear conclusions drawn regarding the selection of strategy and control parameters [7].

The use of EAs, such as DE, could help in finding diversified sets of solutions for not only single-objective but also multi-objective problems. The approach provides added options in decision making. However, the user could be interested only in solving the underlying optimization problems rather than how to operate the optimization algorithms. Finding appropriate parameters for optimization algorithms requires multiple runs and thus consumes time [4]. Time consumption will increase if a user lacks knowledge or experience in using any DE. The conflicting and confusing conclusions on how to set parameters deteriorate a users confidence in using DE. In most studies related to DE parameter setting, at least one of the parameters is user specified. Therefore, the budding interest toward adaptive approaches for DE stems from the intention of decreasing reliance on user-specified parameters to solve optimization problems.

Given that DE can adaptively search the optimal solutions of the underlying problem through evolution, it can also use the same evolutionary process to adapt its own optimal parameters. The evolution paths in solving a particular problem can vary during different runs even if the same algorithm is used due to the stochastic nature of DE. Instead of fixing global parameters for the entire evolutionary process, DE could have flexibility to determine its own parameters. The following section discusses DE models on the basis of their different combinations of adaptive parameters.

2.3.1. Adaptive NP

Two algorithms, namely, absolute encoding (DE-Abs) and relative encoding (DE-Rel), based on self-adaptive population size were developed in [32]. DE-Abs and DE-Rel self-adapt the population size by encoding the absolute value and growth rate of the population size respectively into each individual to evolve with other parameters simultaneously. However, a user needs to determine the values of *F* and *CR* for both DE models.

2.3.2. Adaptive CR

Different from the conventional crossover operators in DE, a new rotating crossover operator (RCO) was created by utilizing multiangle searching strategy-based RCO [33]. The development of DE based on RCO is known as DE-RCO. In DE-RCO, the generation of trial vectors in the control of the self-adaptive crossover parameter and rotation control vectors is based on Lévy distribution. DE-RCO is a variant of DE that experiences modification at the genetic operator level.

2.3.3. Adaptive F and CR

A review focusing on the parameter control methods (PCMs) of adjusting *F* and *CR* only was performed [34]. PCMs are further divided into three main categories, namely, deterministic PCMs (DPCMs), adaptive PCMs (APCMs), and self-adaptive PCMs (SPCMs). The categorization depends on the involvement of information from the search process and genetic operation. APCMs use the information from the search process to determine *F* and *CR*, whereas DPCMs do not use such information. SPCMs involve genetic operators to evolve the control parameters. Another variant of DE that operates on the basis of adaptive *F* and *CR* is success history-based adaptive DE (SHADE). The variants of SHADE have shown promising results in CEC competition for several years. These successes should be credited to the original SHADE developed by Tanabe and Fukunaga [35]. Its operation includes initialization, mutation, crossover, selection, and historical memory updates of successful *F* and *CR*. To determine the appropriate pairs of *F* and *CR* adaptively, association rule mining (ARM) was used in [36]. The model uses Apriori algorithm and greedy operator in the selection of improved solutions. However, the use of other algorithms, such as ARM, Apriori algorithm, and greedy operator, requires user knowledge about the algorithms. The work in [37] proposed a DE algorithm with self-adaptation strategy and control parameters based on the symmetric Latin hypercube design. This strategy can adaptively adjust to match different evolution stages on the basis of prior experience, and it can solve constraint and multiobjective optimization problems.

2.3.4. *Adaptive* F, CR *and* NP

The work in [38] extended the study on DE-Abs and DE-Rel by simultaneously implementing self-adaptation on parameters *F, CR*, and *NP*. This work led to the development of two models based on self-adaptation and ensemble, namely, self-adaptive differential evolution absolute coding (SADE-Abs) and self-adaptive differential evolution relative coding (SADE-Rel) [38]. The ensemble approach, which has been adopted recently for EA to resolve optimization problems, selects an optimal parameter setting from multiple parameter settings on the basis of consensus. This work showed that SADE-Rel is better than SADE-Abs in terms of effectiveness, but SADE-Abs has better efficiency than SADE-Rel. The two models performed as well as classical DE and do not require lengthy trial-and-error effort for parameter setting. However, *M* is held constant in both models.

SHADE is improved with a linear decrease in population size in L-SHADE [39]. The capability of L-SHADE is further enhanced with ensemble parameter sinusoidal adaptation, which leads to the development of LSHADE-EpSin [40]. A mixture of two sinusoidal formulas and a Cauchy distribution is used in the model to balance the exploration and the exploitation of the search. For further improvement, a framework of adaptive *F* and *CR* that is applicable for all SHADE variants was proposed in [41]. The main drawback for SHADE variants is the involvement of parameters aside from the classical ones. Similar to the success of EPSDE, that of SHADE variants is contributed by the probability of successful control parameters.

2.3.5. *Adaptive* F, CR *and* M

Instead of using a constant global setting, the work in [7] showed that a dynamic setting for mutation strategies and control parameters can be effective. This study developed a SaDE wherein the trial vector generation strategies and their associated control parameters are adaptively determined on the basis of their probabilities of generating promising solutions [7]. SaDE outperforms other adaptive DE variants and provides viable solutions in terms of minimum mean, best objective, and std dev [42]. However, the main limitation in this model is that *NP* remains a user-specified parameter. The work in [8] used ensemble to establish the configurations of *CR, F*, and *M* in different stages of evolution, but the value of *NP* was held constant. This model, which is known as EPSDE, uses an ensemble to decide the appropriate configurations of *F, CR*, and *M* for each individual. The configurations of parameters for the individuals in EPSDE differ from the common practice of applying a single set of configuration to the entire population. The implementation of EPSDE in [8,16] showed that dynamic parameter settings may be required at different stages of evolution for effective performance. The ensemble strategy is integrated with compact DE (cDE) to create ensemble cDE (ENcDE) [4]. The strength of cDE is its use of a small amount of computational resources, but it suffers from premature convergence. Therefore, an ensemble is used to overcome its limitation. cDE consists of a matrix of probability vectors with the size of $2 \times D$; in this matrix, each row of vectors corresponds to the mean and std dev of a variable in the solution. In comparison with cDE, ENcDE consists of N_c matrices of probability vectors, but the effect of N_c on performance has not been investigated further. A novel parameter-adaptive DE (PaDE) based on adaptation schemes for *F, CR*, and *M* was developed to overcome slow convergence [43]. In addition, a novel parabolic population size reduction scheme was proposed in PaDE. The use of adaptation schemes for *F* and *CR* is based on statistical distributions. However, the proposed adaptation schemes involve the introduction of parameters in addition to the conventional ones.

2.3.6. *Adaptive* F, CR, M *and* NP

The work in [44] proposed multirole-based DE (MRDE), which divides the population into g_n small populations, each of which consisting of g_s groups. In each group, the individuals are assigned different roles on the basis of their fitness. Three configurations exist, namely, *F, CR*, and *M*, which vary in exploration and exploitation capabilities. The selection of configurations for the individuals depends on their roles. Moreover, MRDE can adaptively increase and decrease its populations. However, the development of MRDE involves structural modification and several newly introduced parameters. The values of the parameters may not be optimally determined, and their interaction effects lack sufficient investigation. Self-adaptive ensemble-based DE (SAEDE), which self-adapts *F, CR, M*, and *NP* simultaneously, was introduced in our previous work [45]. However, this study did not investigate the effects of control parameters on SAEDE. The detailed operation of SAEDE is explained in a subsequent section.

2.4. *Overview on parameter setting*

For an overview of the adaptive parameter setting in DE, Table 1 summarizes the reviewed articles in this research area. The table shows the degrees of adaptive level in different works and briefly states whether the work involves con-

Table 1
List of articles being reviewed for the implementation of adaptive parameter setting in DE.

Article	Adaptive				Constant
	F	*CR*	*M*	*NP*	Parameter
Nobakhti and Wang [28]	√				*CR, M, NP*
Deng et al. [33]		√			*F, M, NP*
Cui et al. [11]			√		*F, CR, NP*
Fister et al. [27]			√		*F, CR, NP*
Teng et al. [32]				√	*F, CR, M*
Liu and Lampinen [26]	√	√			*M, NP*
Zhang and Sanderson [19]	√	√			
Brest at al. [20]	√	√			*M, NP*
Brest [21]	√	√			*M, NP*
Omran et al. [24]	√	√			*M, NP*
Abbass [25]	√	√			*M, NP*
Hu and Yan [30]	√	√			*M, NP*
Tanabe and Fukunaga [34]	√	√			*M, NP*, extra parameter(s)
Tanabe and Fukunaga [35]	√	√			*M, NP*, extra parameter(s)
Wang et al. [36]	√	√			*M, NP*
Zhao et al. [37]	√	√			*M*, extra parameter(s)
Qin et al. [7]	√	√	√		*NP*
Mallipeddi et al. [8]	√	√	√		*NP*
Wu et al. [9]	√	√	√		*NP*
Mallipeddi et al. [4]	√	√	√		*NP*, extra parameter(s)
Kadhar et al. [42]	√	√	√		*NP*, extra parameter(s)
Mallipeddi et al. [16]	√	√	√		*NP*
Zhao et al. [29]	√	√	√		*NP*
Meng et al. [43]	√	√	√		*NP*, extra parameter(s)
Elsayed et al. [22]	√	√		√	*M*
Wang et al. [38]	√	√		√	*M*
Tanabe and Fukunaga [39]	√	√		√	*NP*, extra parameter(s)
Awad et al. [40]	√	√		√	*M*, extra parameter(s)
Viktorin et al. [41]	√	√		√	*M*
Gui et al. [44]	√	√	√	√	
Wang et al. [45]	√	√	√	√	

stant classical parameters and additional new parameter(s). The conclusion that can be obtained from the research related to parameter settings is that their settings are related to problem characteristics, but problem characteristics are usually unknown to users in reality. Moreover, the guidelines suggested in most research with regard to parameter settings are restricted not only to problem characteristics but also to mutation strategies and the control parameters values considered in the research. Most studies related to parameter tuning in DE have focused either on a single parameter or on a partial combination of parameters, and only a few have focused on all of them. In our research, we propose an approach that simultaneously self-adapts all parameters, and we evaluate its performance relative to that of other self-adaptive approaches. In introducing self-adaptation in DE, we aim to minimize confusion and uncertainty in setting DE parameters and mutation strategies. Therefore, self-adaption is performed without modifying DEs genetic operation or structure and without using another algorithm. We are also interested in investigating the effect of dynamic *NP* on the scale of *D*, given that it is always used as a reference for setting *NP*.

3. Methodology

We call our DE model SAEDE owing to its use of self-adaptive and ensemble mechanisms to set the relevant parameters for each generation. Self-adaptation refers to the evolution of solutions encoded into chromosomes, whereas an ensemble refers to the method of constructing the final solution from a set of solutions on the basis of consensus. In our model, self-adaptation is used to determine the value of *NP*, and ensemble is used to determine the values of *F, CR*, and *M* at each generation. The concept of using these mechanisms comes from DE models DE-Rel [32] and EPSDE [8]. DE-Rel can adaptively determine *NP* for each generation, but the values of *CR* and *F* are held constant in the evolution. EPSDE uses ensemble to determine the combinations of *CR, F*, and *M* in different stages of evolution, but *NP* is held constant.

Similar to the chromosome in DE-Rel, that in SAEDE is represented by a parameter vector, $\mathbf{z}_{i,g} = \{\mathbf{x}_{i,g}, y_{i,g}\}$ as shown in Fig. 1, with $i = 1, 2, \ldots, NP_g$. Here, NP_g refers to the population size at generation-g and it varies across generations. $\mathbf{x}_{i,g}$ consists of a D-dimensional vector that represents the solution-i for an optimization problem at generation-g, i.e., $\mathbf{x}_{i,g} = \{x^1_{i,g}, \cdots, x^D_{i,g}\}$. The variable $y_{i,g}$ refers to the population size's growth rate associated with individual-i at generation-g.

The initial jth decision variable of the ith solution at generation, $g = 0$ is generated within the search space constrained by the prescribed minimum and maximum decision variable's bounds $\mathbf{x}_{\min} = \{x^1_{\min}, \ldots, x^D_{\min}\}$ and $\mathbf{x}_{\max} = \{x^1_{\max}, \ldots, x^D_{\max}\}$ as shown in Eq. (3). The initial NP_g at generation $g = 0$ is generated within the range of the prescribed minimum and maximum bounds of $[10D, \ 100D]$.

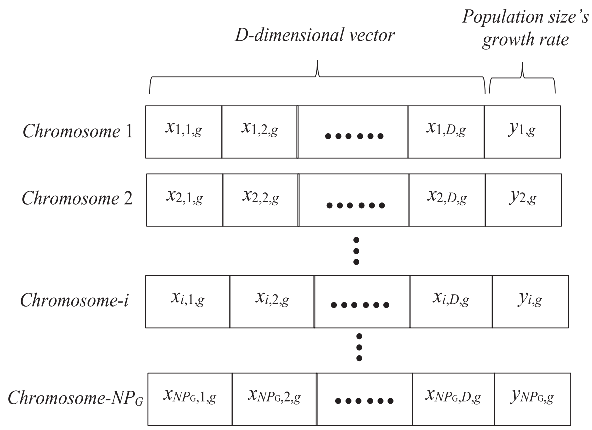

Fig. 1. Chromosome representations for SAEDE.

After initialization, the population of $y_{i,g}$ of generation-g determines the population size of the next generation, NP_{g+1}. The population size's growth rate for generation-g+1, y_{g+1} is determined by obtaining the average of the population $y_{i,g}$ as shown in Eq. (1). Then, y_{g+1} is used to calculate the value of NP_{g+1} for SAEDE on the basis of Eq. (1). If NP_{g+1} is larger than NP_g, then a number of solutions are randomly generated within their respective bounds to fulfill the population size NP_{g+1}. However, if NP_{g+1} is smaller than NP_g, then a number of least fit solutions are deleted to meet the population size NP_{g+1}.

$$y_{g+1} = (int)\left(\frac{y_{1,g} + y_{2,g} + \ldots + y_{NP_g,g}}{y_g} + 0.5\right) \tag{1}$$

$$NP_{g+1} = (int)(y_{g+1} * NP_g) + NP_g + 0.5 \tag{2}$$

$$\mathbf{x}_{i,0} = x^j_{min} + rand(0,1) \cdot (x^j_{max} - x^j_{min}) \quad j = 1, 2, \ldots, D \tag{3}$$

Similar to the population of individuals in original DE, that in SAEDE evolves over generations through mutation, crossover, and selection. However, SAEDE does not use constant *CR*, *F*, and *M*, as commonly practiced in original DE. Instead, SAEDE dynamically changes the control parameters and mutation strategy, as demonstrated in EPSDE. Thus, the term "control parameters" refers to *CR* and *F* only.

For EPSDE, *CR* is regarded as in the range of 0.1 - 0.9 in steps of 0.1, whereas *F* is regarded as in the range 0.4 0.9 in steps of 0.1, as suggested in [8]. Given the nine and five options for the setting of *CR* and *F*, respectively, the total combination of *CR* and *F* is 45 (9 × 5). For crossover schemes, EPSDE has three mutation strategies, as suggested in [8]. The same mutation strategies are included in SAEDE, namely, {DE/rand/1/bin, DE/best/2/bin, DE/current-to-rand/1/bin}. Given the 45 combinations of *CR* and F and the three mutation strategies used by EPSDE, EPSDE contains 135 (45 × 3) configurations of mutation strategy and control parameters in the pool. Settings of the mutation strategy and control parameters across the generations are selected from the pool of 135 configurations.

Initially, each target vector $\mathbf{z}_{i,g}$ in the initial population is randomly assigned a configuration of mutation strategy and control parameters. Therefore, the mutation and crossover operations experienced by the target vectors depend on the assigned configurations. The *CR*, *F* and *M* associated with the *i*-solution at generation-*g* are denoted as $CR_{i,g}$, $F_{i,g}$ and $M_{i,g}$, respectively. At generation-g, the individuals in the current population $P_{\mathbf{z}}$ are known as target vectors and correspond to $\{\mathbf{x}_{i,g}, y_{i,g}\}$. Corresponding to each target vector $\mathbf{x}_{i,g}$, a mutant vector, $\mathbf{v}_{i,g} = \{v^1_{i,g}, \cdots, v^D_{i,g}\}$ is produced through the differential mutation operation. The same operation is extended to the last gene in each chromosome corresponding to $y_{i,g}$ to produce a mutant scalar $\dot{y}_{i,g}$. The differential mutation operation adds a scaled, randomly sampled, vector difference to a third vector, as shown in Eqs. (4)–(6). At this stage, the population is called intermediary population, $P_{\dot{\mathbf{z}}} = \dot{\mathbf{z}}i,\ g = \{\mathbf{v}_{i,g}, \dot{y}_{i,g}\}$.

The associated $F_{i,g}$ is used in the differential mutation operation to control the evolution rate of $\mathbf{z}_{i,g}$. Three mutation strategies used in SAEDE are shown in Eqs. (4)–(6). Each vector in the current population, $P_{\mathbf{z}}$ is then combined with a mutant vector in accordance with the setting of $CR_{i,g}$ to produce trial vector $\mathbf{u}_{i,g} = \{u^1_{i,g}, \cdots, u^D_{i,g}\}$ on the basis of Eq. (7) and trial scalar $\ddot{r}_{i,g}$ on the basis of Eq. (7). As a result, a trial population $P_{\ddot{\mathbf{z}}} = \ddot{\mathbf{z}}i,\ g = \{\mathbf{u}_{i,g}, \ddot{y}_{i,g}\}$ is produced. During mutation and crossover operations, any decision variable in the newly generated trial vectors whose value exceeds its corresponding upper or lower bounds is re-initialized randomly and uniformly within the pre-specified range until it falls within the bounds.

DE/best/2 [46]: $$\dot{\mathbf{z}}_{i,g} = \mathbf{z}_{best,g} + F(\mathbf{z}_{r^i_1,g} - \mathbf{z}_{r^i_2,g}) + F(\mathbf{z}_{r^i_3,g} - \mathbf{z}_{r^i_4,g}) \tag{4}$$

DE/rand/1 [46]: $$\dot{\mathbf{z}}_{i,g} = \mathbf{z}_{r^i_1,g} + F(\mathbf{z}_{r^i_2,g} - \mathbf{z}_{r^i_3,g}) \tag{5}$$

DE/current-to-rand/1 [47]: $$\dot{\mathbf{z}}_{i,g} = \mathbf{z}_{i,g} + K(\mathbf{z}_{r^i_1,g} - \mathbf{z}_{i,g}) + F(\mathbf{z}_{r^i_2,g} - \mathbf{z}_{r^i_3,g}) \tag{6}$$

$$u_{i,g}^{j} = \begin{cases} v_{i,g}^{j} & \text{if } (rand_j[0,1] \le CR_{i,g}) \text{ or } (j = j_{rand}) \\ & \qquad j = 1,2,\dots,D \\ x_{i,g}^{j} & \text{otherwise} \end{cases} \tag{7}$$

$$\ddot{y}_{i,g} = \begin{cases} \dot{y}_{i,g} & \text{if } (rand_j[0,1] \le CR_{i,g}) \\ & \qquad j = D+1 \\ y_{i,g} & \text{otherwise} \end{cases} \tag{8}$$

During selection, the objective function of all trial vectors $\mathbf{u}_{i,g}$ is evaluated. In a minimization problem, when the objective function of a trial vector $f(\mathbf{u}_{i,g}) \le f(\mathbf{x}_{i,g})$, it replaces the target vector in the next generation; otherwise, the target vector retains its place in the population, as shown in Eqs. (9) and (10). The configurations associated in the production of trial vectors are evaluated. If the generated trial vector produces a better objective function than its target vector, then the associated configuration of mutation strategy and control parameters will be retained in the next generation and stored in a successful archive, *A*. Otherwise, if the target vector has better objective function than its trial vector, the associated configuration of mutation strategy and control parameters will be randomly re-initialized with a new configuration or randomly selected from the configurations available in the successful archive. The genetic operations of mutation, crossover and selection are repeated generation after generation until the stopping criterion is satisfied. Table 2 shows the pseudocode of SAEDE. Different equations in the pseudocode of SAEDE, covering Eqs. (1)–(10), are outlined to highlight their implementation in the algorithm. Therefore, SAEDE combines the autonomous procedures of setting control parameters, mutation strategies from EPSDE, and population size from DE-Rel. Therefore, the settings of *F, CR, NP* and *M* will be automatically adjusted across the generation with minimal intervention from users.

We introduced SAEDE in [45], but we have not further investigated the effects of parameters on SAEDE and the selected comparative DEs in our previous work (DE-Rel and EPSDE). Two types of parameters were identified in our investigation, namely, adaptive and constant parameters. Adaptive parameters are parameters that can be adjusted dynamically by the algorithm itself on the basis of some rules or schemes with minimal user intervention during evolution. Constant parameters are parameters that are user specified and normally held constant during evolution. In our previous work, the DE-Rel model dynamically adjusted its *NP*, but its *F, CR*, and *M* were held constant. The adaptive parameter of DE-Rel is *NP*, whereas the constant parameters are *F, CR* and *M*. The constant parameters of DE-Rel were set as $F = 0.5$, $CR = 0.5$ and $M =$ DE/rand/1/bin. The adaptive parameters of EPSDE are *F, CR* and *M*. The only constant parameter of EPSDE is *NP*, and the value was set as 50 in our previous work. Furthermore, in our previous work, the constant parameters for DE-Rel and EPSDE were tested on the basis of a single value. In the current study, the constant control parameters for DE-Rel and EPSDE are varied, and their effects on performance are investigated. The effects of adaptive parameters on SAEDE are considered in this work but not in our previous work. In the present study, the performance of the involved DEs is investigated thoroughly by involving additional alternative measurements and analyses, such as stagnation and convergence. Numerous complex benchmark functions of high dimensions are included as well to evaluate the DEs performance.

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} & \text{if } f(\mathbf{u}_{i,g}) \le f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g} & \text{otherwise} \end{cases} \tag{9}$$

$$y_{i,g+1} = \begin{cases} \ddot{y}_{i,g} & \text{if } f(\mathbf{u}_{i,g}) \le f(\mathbf{x}_{i,g}) \\ y_{i,g} & \text{otherwise} \end{cases} \tag{10}$$

4. Experimental setup

The experimental setup can be divided into two main parts; one part involves benchmark functions of low dimensions, and the other involves benchmark functions of high dimensions. The details of the experimental setup are explained in the following sections.

4.1. Experimental setup i : Functions of low dimensions

In the investigation of parameter setting, the performances of different models of DEs are compared. Twenty benchmark functions of low dimensions in [32] are adopted to evaluate the DEs. Table 3 summarizes the 20 benchmark functions (F1 to F20). All functions are minimization of problems with their best minimum solution denoted as $f_{\min}$. The number of variables involved (*D*) varies from 2 to 10, and the ranges of variable search are represented by *S*.

The different DE models are compared under similar conditions. Added heuristic strategies may provide different results. Therefore, experimental setups similar to those used by DE-Rel in [32] and EPSDE in [8] are considered in the implementation of DEs in our experiments to minimize the nuisance factors affecting the results and to enable the same analyses to be conducted for them. For ease of understanding, Table 4 summarizes the settings of the mutation strategies and control parameters for the DE models.

The summary in Table 4 shows 1 experimental setup for SAEDE, 3 experimental setups for EPSDE, and 15 experimental setups for DE-Rel. Each experimental setup is run 30 times. For SAEDE, the settings of *F, CR, M* and *NP* dynamically change

Table 2
The pseudocode SAEDE.

The pseudocode of our proposed model SAEDE
Step 1 Initialization
Step1.1 Set g = 0. Randomly initialize population size, NP_g.
Step 1.2 Randomly initialize a population of NP_g individuals, $P_{\mathbf{z}} = \{\mathbf{z}_{i,g}, ..., \mathbf{z}_{g,g}\}$ with $\mathbf{z}_{i,\,g} = \{\mathbf{x}_{i,\,g},\ y_{i,\,g}\}$. The generation of $\{\mathbf{x}_{0,\,g}$ is constrained within the search space of Equation 3.
Step1.3 Initialize a pool of configurations of control parameters and mutation strategies.
Step 2 Randomly assign each population individual with a configuration of control parameters and mutation strategies.
Step 3 Evolution
while stopping criterion is not met **do**
Step 3.1 Mutation Step
for $i = 0$ to N_g **do**
Generate a mutated vector $\mathbf{v}_{i,g} = \{v_{i,g}^1, v_{i,g}^2, ..., v_{i,g}^D\}$ and mutated scalar $\dot{y}_{i,g}$ corresponding to target vector $\mathbf{z}_{i,g}$ by using the configuration of control parameters and mutation strategy with the target vector.
end for
Step 3.2 Crossover
for $i = 0$ to N_g **do**
$j_{rand} = [rand[0,1] \cdot D]$
for $j = 0$ to D **do**
Generate a trial vector $\mathbf{u}_{i,g} = \{u_{i,g}^1, u_{i,g}^2, ..., u_{i,g}^D\}$ for each target vector $\mathbf{x}_{i,g}$ associated with $\mathbf{z}_{i,g}$ based on Equation 7.
end for
for $j = D + 1$ **do**
Generate a trial vector $\ddot{y}_{i,g}$ for each target scalar $y_{i,g}$ associated **with** $\mathbf{z}_{i,g}$ based on Equation 8.
end for
end for

(continued on next page)

Table 2 (*continued*)

Step 3.3 Selection
for $i = 0$ to N_g **do**
  if $f(\mathbf{u}_{i,g}) \leq f(\mathbf{x}_{i,g})$ **then**
    $\mathbf{x}_{i,g+1} = \mathbf{u}_{i,g}$ Equation 9, $y_{i,g+1} = \ddot{y}_{i,g}$ Equation 10
    Copy the configuration of control parameters and mutation strategy associated with the trial vector to the successful archive, A
    if $f(\mathbf{u}_{i,g}) \leq f(\mathbf{x}_{best,g})$ **then**
      $\mathbf{x}_{best,g} = \mathbf{u}_{i,g}, y_{best} = \ddot{y}_{i,g},$
    end if
  else
    $\mathbf{x}_{i,g+1} = \mathbf{x}_{i,g}$ Equation 9, $y_{i,g+1} = y_{i,g}$ Equation 10
  end if
end for
Step 3.4 Update configurations
for $i = 1$ to N_g **do**
  if $f(\mathbf{u}_{i,g}) > f(\mathbf{x}_{i,g})$ **then**
    Randomly select a new configuration of control parameters and mutation strategy from the pool or from the successful archive. The mutation strategy can be based on either Equation 4, Equation 5 or Equation 6.
  end if
end for
Step 3.4 Population size adaptation
for $i = 1$ to N_g **do**
  Calculate y_{g+1} based on Equation 1
  Calculate NP_{g+1} based on Equation 2
end for
end while

Table 3
Details on the benchmark functions.

Denotation	Test Function	D	S	$f_{\min}$	X^*	U/M
2D to 10D						
F1	De Jong's function 1 (Sphere Model)	10	$[-100,\ -90]^D$	0	(0, 0,..., 0)	U
F2	De Jong's function 2 (Rosenbrock's Saddle	2	$[-2.048,\ 2.048]^D$	0	(1, 1)	U
F3	Camel back – three humo problem	2	$[-5.0,\ -4.5]^D$	0	(0, 0)	M
F4	Becker and Lago's problem	2	$[-10,\ -9]^D$	0	(± 5, ± 5)	M
F5	Schwefel's problem 2.22	10	$[-10,\ -9]^D$	0	(0, 0,..., 0)	U
F6	Rastragin's function	10	$[-5.120,\ -4.608]^D$	0	(0, 0,..., 0)	M
F7	Modified Rosenbrock's problem	2	$[-5.0,\ -4.5]^D$	0	(1, 1), (0.3412, 0.1164)	M
F8	Griewangk's problem	10	$[-600,\ -540]^D$	0	(0, 0,..., 0)	M
F9	Ackley's path function	10	$[-32.0,\ -28.8]^D$	0	(0, 0,..., 0)	M
F10	Bohachevsky's problem 2	2	$[-50,\ -48]^D$	0	(0, 0)	M
F11	Rotate hyper-ellipsoid function	10	$[-65.5360,\ -58.9824]^D$	0	(0, 0,..., 0)	U
F12	Sum of different power problem	10	$[-1.0,\ -0.9]^D$	0	(0, 0,..., 0)	U
F13	Miele and Cantrell's problem	4	$[-1.0,\ -0.9]^D$	0	(0, 1, 1, 1)	M
F14	Schaffer's problem 1	2	$[-100,\ -90]^D$	0	(0, 0)	M
F15	Moved axis parallel hyper-ellipsoid function	10	$[-5.120,\ -4.608]^D$	0	(5^*N, $5{*}N$, ..., 5^*N)	U
F16	Helical valley problem	3	$[-10,\ -9]^D$	0	(1, 0, 0)	U
F17	Salomon's problem	10	$[-100,\ -90]^D$	0	(0, 0,..., 0)	M
F18	Powell's quadratic problem	4	$[-10,\ -9]^D$	0	(0, 0, 0, 0)	U
F19	Bohachevsky's problem 1	2	$[-50,\ -45]^D$	0	(0, 0)	M
F20	Wood's function	4	$[-10,\ -9]^D$	0	(1, 1, 1, 1)	U
30D						
F21	Shifted Sphere function	30	$[-100,\ 100]^D$	0	$\boldsymbol{o}$	U
F22	Shifted Schwefel's Problem 1.2	30	$[-100,\ 100]^D$	0	$\boldsymbol{o}$	U
F23	Rosenbrock function	30	$[-100,\ 100]^D$	0	$\boldsymbol{o}$	U
F24	Shifted Schwefel's Problem 1.2 with noise in fitness	30	$[-32,\ 32]^D$	0	$\boldsymbol{o}$	U
F25	Shifted Ackley's function	30	$[-32,\ 32]^D$	0	$\boldsymbol{o}$	U
F26	Shifted rotated Ackley's function	30	$[0,\ 600]^D$	0	$\boldsymbol{o}$	U
F27	Shifted Griewank's function	30	$[0,\ 600]^D$	0	$\boldsymbol{o}$	U
F28	Shifted rotated Griewank's function	30	$[-5,\ 5]^D$	0	$\boldsymbol{o}$	U
F29	Shifted Rastrigin's function	30	$[-5,\ 5]^D$	0	$\boldsymbol{o}$	U
F30	Shifted rotated Rastrigin's function	30	$[-5,\ 5]^D$	0	$\boldsymbol{o}$	U
F31	Shifted noncontinuous Rastrigin's function	30	$[-500,\ 500]^D$	0	$\boldsymbol{o}$	U
F32	Schewel's function	30	$[-500,\ 500]^D$	0	$\boldsymbol{o}$	U
F33	Composition function 1	30	$[-5,\ 5]^D$	0	$\boldsymbol{o}_1$	U
F34	Composition function 6	30	$[-5,\ 5]^D$	0	$\boldsymbol{o}_1$	U

$\boldsymbol{o}$ is shifted vector.
$\boldsymbol{o}_1$ is shifted vector for the firs basic function in the Composition function.

within their respective options during evolution. SAEDEs possible values for F range within [0.4, 0.9], those for CR range within [0.1, 0.9], and those for NP range within [10D, 100D]. The M strategy for SAEDE can dynamically interchange within the options [DE/rand/1/bin, DE/best/2/bin, DE/current-to-rand/1/bin] during evolution. Similar to the settings of SAEDE, those of F, CR, and M for EPSDE interchange within their respective options, but its NP is held constant throughout the evolution process. As for each experimental setup of DE-Rel, only NP changes; the remaining parameters (F, CR, and M) are held constant during evolution.

The original settings of F and CR, which were fixed at 0.5 in [32], are adopted for DE-Rel. Instead of using only the same settings of F and CR, we include the same ranges suggested in [8] in DE-Rel to investigate the effects of F and CR on performance. Table 4 shows the combinations of F and CR for DE-Rel. The justification of selecting the combinations is based on our aim of understanding their effects on exploration and exploitation when their values are set with the following conditions: (i) both values are high, (ii) both values are low, and (iii) one of them is high, whereas the other is low. The combination of $CR = 0.1$ and $F = 0.4$ indicates that both parameters have low values, whereas the combination of $CR =$ 0.9 and $F = 0.9$ indicates that both parameters are set at high levels. Five combinations of F and CR exist. The same five combinations of F and CR are adopted for each M.

The evolutions for all three models of DEs are run for 100,000 generations ($G =$ 100,000) and repeated by using the k-th seed numbers, $k = 1, 2, \ldots, K$. The same 30 seed numbers are used to evaluate different DEs, $K = 30$. The evolutionary processes are terminated if the best-fitness, $f_{best} < f_{target}$ is reached, otherwise the processes continue until they reach G. For functions of low dimensions, $f_{target} = f(x^*)$ + 1.0E-20.

4.2. Experimental setup II : Functions of high dimensions

In this experiment, 14 benchmark 30D functions f_1–f_{14} [7] are adopted to evaluate the performance of DEs further; they are renamed as F21–F34 in Table 3. The functions are unimodal, but they are shifted and rotated. The global optimal point is

Table 4
The effects of changes in control parameters and mutation strategies on DEs' performances.

Model	Configuration Notation	Fixed Settings			Average measures for 20 functions				
DE-Rel	D	M	CR	F	SC	SR	$\bar{q}_{mean}$	$\bar{q}_{best}$	NP(× D)
	D1	best/2/bin	0.1	0.4	**15**	**0.83**	**13,376**	**11,277**	**56**
	D2	best/2/bin	0.1	0.9	13	0.75	9402	9045	58
	D3	best/2/bin	0.5	0.5	**15**	**0.83**	**12,517**	**11,100**	**49**
	D4	best/2/bin	0.9	0.4	10	0.69	25,583	24462	42
	D5	best/2/bin	0.9	0.9	14	0.74	11,630	10,313	70
	D6	rand/1/bin	0.1	0.4	13	0.79	13,827	11,982	49
	D7	rand/1/bin	0.1	0.9	14	0.75	10,511	9895	49
	D8	rand/1/bin	0.5	0.5	**15**	**0.83**	**10888**	**10001**	**64**
	D9	rand/1/bin	0.9	0.4	12	0.80	17,621	17,483	58
	D10	rand/1/bin	0.9	0.9	13	0.74	14,581	13,053	61
	D11	current-to-rand/1/bin	0.1	0.4	13	0.73	16,378	11,809	53
	D12	current-to-rand/1/bin	0.1	0.9	0	0.10	219	188	62
	D13	current-to-rand/1/bin	0.5	0.5	14	0.84	11,733	11,194	65
	D14	current-to-rand/1/bin	0.9	0.4	4	0.61	20,471	18,887	58
	D15	current-to-rand/1/bin	0.9	0.9	11	0.65	8124	7771	66
EPSDE	E	NP (in D)			SC	SR	$\bar{q}_{mean}$	$\bar{q}_{best}$	NP(× D)
	E1	10			15	0.82	12,960	13,110	10
	E2	50			16	0.84	11277	11,054	50
	E3	100			**16**	**0.85**	**10,977**	**10,696**	**100**
SAEDE	S	-			SC	SR	$\bar{q}_{mean}$	$\bar{q}_{best}$	NP(× D)
	S1				**16**	**0.84**	**11,308**	**11,195**	**75**

Results in bold refer the configurations associated with the highest SR for each DE model.

shifted to a random position to prevent the global optimal point from being at the center of the search range, and rotation is performed to avoid local optima from being along the coordinate axes. The details on performing shifting and rotation on the functions were explained in [48].

Only the best performers are selected from DE-Rel and EPSDE to evaluate the performance of DEs on the benchmark functions of high dimensions. With the inclusion of SAEDE, three DEs are evaluated in this experiment. The evolutions for the selected DEs are run for 100,000 generations ($G = 100{,}000$) and repeated by using the k-th seed numbers, $k = 1, 2, \ldots, K$. The same 30 seed numbers are used to evaluate different DEs, $K = 30$. The evolutionary processes are terminated if f_{best} is not worse than the pre-specified optimal values, i.e., $f(x^*)$ + 1.0E-5, as suggested in [7]. Otherwise the processes continue until they reach G, whereby $G = 100{,}000$ used in this experiment is more stringent than $G = 300{,}000$ used for 30D functions in [7].

5. Result and analysis

This section is further divided into three sections, namely, Sections 5.1–5.3. Sections 5.1 and 5.2 discuss the performance of DEs on the functions of low and high dimensions, respectively. Section 5.3 analyzes the convergence of the representative DEs on the functions of low and high dimensions. To investigate the performance of different DEs, several measurements are used in the investigation, namely, the average of best fitness, $\bar{f}_{best}$, frequency of *successful cases* (SC), *success rate* (SR), stagnation level, hypothesis test, and convergence analysis. Details of some of the measurements are explained in the following section.

For the benchmark functions of low dimensions (2D to 10D), when the $f_{best} < f_{target}$, the evolution for this run is called a successful run. SR computes the number of successful runs against the number of independent runs. Therefore, SR measures the average success probability of the DEs, and SC measures the number of cases where the DEs achieve 100% probability. For benchmark functions of low dimensions (2D to 10D), $f_{target} = f(x*)$ + 1.0E-20. For those with high dimensions (30D), $f_{target} = f(x*)$ + 1.0E-5.

In this work, we are interested in the stability guarantee of convergence to the minimum fitness of the DEs. Given that the DEs may stop at different g values when the stopping criterion is met, we compute the difference of f_{best} at g and at 50 generations before g (g - 50). We call the difference of f_{best} between the two numbers of generations as f_{dif} and use it to analyze the convergence stability. f_{dif} is computed for 30 runs across different benchmark functions. Furthermore, we use the Q-measure (Q_m) from [49] to investigate convergence for the representative DEs. Q_m is a measurement parameter that balances convergence speed and probability convergence. The calculation of Q_m is shown in Eq. (11). Convergence speed is calculated by convergence measure C_m on the basis of Eq. (12), whereas the calculation of probability convergence P_c is shown in Eq. (13).

$$Q_m = \frac{C_m}{P_c} \tag{11}$$

$$C_m = \frac{fesr_{sum}}{nsr} \tag{12}$$

$$P_c = \frac{nsr}{ntr} \tag{13}$$

where *nsr* refers to the number of successful runs for all tested functions, *ntr* refers to the total number of runs for all tested functions, and $fesr_{sum}$ refers to the summation of the numbers of function evaluations for the successful runs.

We use the second property of stagnation from [50] to measure the inability of the algorithm to find efficient solutions with the assigned control parameters and mutation strategy. This phenomenon can be shown by evaluating the number of recent consecutive unsuccessful updates $q_{i,g}$ in the gth generation with the assigned control parameters and mutation strategy.

$$q_{i,\,g+1} = \begin{cases} 0 & \text{if } f(\mathbf{u}_{i,\,g}) \le f(\mathbf{x}_{i,\,g}) \\ q_{i,\,g} + 1 & \text{otherwise} \end{cases} \tag{14}$$

for $i = 1, 2, \ldots,\ NP$

Initially, $q_{i,0} = 0$. An infinite increase in the sequence $\{q_{i,g}\}$ means that DE cannot generate any successful solutions for the *i*th target vector with the assigned control parameters and mutation strategy. Indirectly, $q_{i,g}$ measures how long the capability of particular evolutionary procedures causes the stagnation of the *i*th solution. The higher the value of $q_{i,g}$ the longer the *i*th solution has been stagnant. Moreover, determining how long the best solution has been stagnant is interesting. This parameter can be measured by the value of $q_{i,g}$ associated with the best solution when the stopping criterion is met, and it is denoted as q_{best}.

$$q_{mean,\,g} = \frac{\sum_{i=1}^{N_g} q_{i,\,g}}{N_g} \tag{15}$$

Overall, we are interested to determine the average stagnation level for a population of solutions at the *g*-th generation. We define the mean of the number of recent consecutive unsuccessful updates for a population of solutions at the *g*-th generation as $q_{mean,g}$ in Eq. (15). Given that the number of generation *g* varies when the stopping criterion is met for different numbers of runs, the performance measures are based on the averages of 30 runs when evolution stops.

5.1. Functions of low dimensions

The measurements in Tables 4–7 are based on the averages of 30 runs when evolution stops. The results summarized in Table 4 indicate that the DE-Rel models, namely, D1, D3, and D8, achieve the highest $SC = 15$. The D1 model has the highest $\bar{q}_{mean}$ and moderate *NP*; D3 has moderate $\bar{q}_{mean}$ and the lowest *NP*; D8 has the lowest $\bar{q}_{mean}$ and highest *NP*. Among

Table 5
Performance measures based on mean ± std dev for D3 on benchmark functions of 2*D* to 10*D*.

Function	$\bar{f}_{best}$	*SR*	$\bar{q}_{mean}$	$\bar{q}_{best}$	*NP*(× *D*)
F1	8.1E-21 ± 1.7E-21	1.00 ± 0.00	2 ± 1	0 ± 0	54 ± 40
F2	5.4E-21 ± 2.4E-21	1.00 ± 0.00	4 ± 3	0 ± 0	55 ± 38
F3	4.9E-21 ± 2.8E-21	1.00 ± 0.00	1 ± 0	0 ± 0	55 ± 41
F4	4.7E-21 ± 2.8E-21	1.00 ± 0.00	2 ± 1	0 ± 0	52 ± 42
F5	8.8E-21 ± 9.0E-22	1.00 ± 0.00	3 ± 2	0 ± 0	63 ± 42
F6	0.00 ± 0.00	1.00 ± 0.00	3 ± 4	0 ± 0	57 ± 42
F7	**3.5E-03 ± 3.7E-03**	**0.53 ± 0.50**	**42,929 ± 46,768**	**42,866 ± 46,731**	**58 ± 42**
F8	**8.2E-04 ± 4.4E-03**	**0.97 ± 0.18**	**3249 ± 17,485**	**3242 ± 17,456**	**71 ± 40**
F9	**8.9E-16 ± 2.0E-31**	**0.00 ± 0.00**	**60,652 ± 30,416**	**36,061 ± 27,196**	**76 ± 40**
F10	0.00 ± 0.00	1.00 ± 0.00	2 ± 2	0 ± 0	57 ± 43
F11	7.8E-21 ± 1.6E-21	1.00 ± 0.00	12 ± 4	0 ± 0	50 ± 42
F12	5.6E-21 ± 2.8E-21	1.00 ± 0.00	2 ± 2	0 ± 0	72 ± 40
F13	**1.0E-08 ± 1.7E-24**	**0.00 ± 0.00**	**97,793 ± 1184**	**97,694 ± 1438**	**55 ± 45**
F14	0.00 ± 0.00	1.00 ± 0.00	2 ± 1	0 ± 0	66 ± 39
F15	8.0E-21 ± 1.5E-21	1.00 ± 0.00	2 ± 2	0 ± 0	67 ± 39
F16	5.8E-21 ± 3.0E-21	1.00 ± 0.00	1 ± 1	0 ± 0	50 ± 40
F17	**1.0E-01 ± 6.7E-15**	**0.00 ± 0.00**	**45,655 ± 6514**	**42,134 ± 30,759**	**73 ± 41**
F18	6.2E-21 ± 2.7E-21	1.00 ± 0.00	5 ± 2	0 ± 0	60 ± 44
F19	0.00 ± 0.00	1.00 ± 0.00	2 ± 1	0 ± 0	53 ± 41
F20	7.1E-21 ± 2.3E-21	1.00 ± 0.00	14 ± 15	0 ± 0	65 ± 42

Results in bold refer to functions associated with SR < 1.

Table 6
Performance measures based on mean ± standard deviation for E3 on benchmark functions of 2D to 10D.

Function	$\bar{f}_{best}$	SR	$\bar{q}_{mean}$	$\bar{q}_{best}$	$NP(\times D)$
F1	8.2E-21 ± 1.6E-21	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F2	5.1E-21 ± 3.0E-21	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F3	4.6E-21 ± 3.1E-21	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F4	4.4E-21 ± 3.0E-21	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F5	9.0E-21 ± 9.5E-22	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F6	0.00 ± 0.00	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F7	**2.5E-04 ± 1.3E-03**	**0.97 ± 0.18**	**2316 ± 12,462**	**2514 ± 13,537**	**100 ± 0**
F8	0.00 ± 0.00	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F9	**8.9E-16 ± 2.0E-31**	**0.00 ± 0.00**	**71,852 ± 3156**	**71,021 ± 3229**	**100 ± 0**
F10	0.00 ± 0.00	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F11	8.7E-21 ± 1.2E-21	1.00 ± 0.00	6 ± 0	0 ± 0	100 ± 0
F12	5.6E-21 ± 2.7E-21	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F13	**1.0E-08 ± 1.7E-24**	**0.00 ± 0.00**	**96,788 ± 199**	**96,713 ± 2010**	**100 ± 0**
F14	0.00 ± 0.00	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F15	8.2E-21 ± 1.5E-21	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F16	5.5E-21 ± 2.9E-21	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F17	**1.0E-01 ± 2.4E-16**	**0.00 ± 0.00**	**48,948 ± 947**	**43,674 ± 29,576**	**100 ± 0**
F18	6.3E-21 ± 2.7E-21	1.00 ± 0.00	3 ± 0	0 ± 0	100 ± 0
F19	0.00 ± 0.00	1.00 ± 0.00	2 ± 0	0 ± 0	100 ± 0
F20	6.4E-21 ± 2.0E-21	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0

Results in bold refer to functions associated with SR < 1.

Table 7
Performance measures based on mean ± std dev for S1 on benchmark functions of 2D to 10D.

Function	$\bar{f}_{best}$	SR	$\bar{q}_{mean}$	$\bar{q}_{best}$	$NP(\times D)$
F1	8.2E-21 ± 1.3E-21	1.00 ± 0.00	4 ± 3	0 ± 0	84 ± 24
F2	5.5E-21 ± 3.0E-21	1.00 ± 0.00	3 ± 1	0 ± 0	65 ± 35
F3	4.5E-21 ± 3.2E-21	1.00 ± 0.00	2 ± 1	0 ± 0	78 ± 30
F4	4.3E-21 ± 3.1E-21	1.00 ± 0.00	2 ± 1	0 ± 0	68 ± 31
F5	9.2E-21 ± 9.7E-22	1.00 ± 0.00	3 ± 1	0 ± 0	70 ± 37
F6	0.00 ± 0.00	1.00 ± 0.00	3 ± 1	0 ± 0	78 ± 30
F7	**1.7E-03 ± 3.1E-03**	**0.77 ± 0.42**	**17127 ± 31344**	**17025 ± 31504**	**84 ± 26**
F8	0.00 ± 0.00	1.00 ± 0.00	3 ± 1	0 ± 0	72 ± 33
F9	**8.9E-16 ± 2.0E-31**	**0.00 ± 0.00**	**68493 ± 7835**	**68000 ± 8231**	**79 ± 38**
F10	0.00 ± 0.00	1.00 ± 0.00	2 ± 2	0 ± 0	57 ± 33
F11	8.4E-21 ± 1.4E-21	1.00 ± 0.00	7 ± 7	0 ± 0	86 ± 24
F12	6.7E-21 ± 2.3E-21	1.00 ± 0.00	3 ± 1	0 ± 0	68 ± 36
F13	**1.0E-08 ± 1.7E-24**	**0.00 ± 0.00**	**96011 ± 1357**	**95896 ± 2098**	**88 ± 31**
F14	0.00 ± 0.00	1.00 ± 0.00	2 ± 0	0 ± 0	67 ± 34
F15	7.8E-21 ± 1.8E-21	1.00 ± 0.00	3 ± 1	0 ± 0	77 ± 30
F16	6.0E-21 ± 2.8E-21	1.00 ± 0.00	2 ± 2	0 ± 0	70 ± 33
F17	**1.0E-01 ± 2.4E-14**	**0.00 ± 0.00**	**44468 ± 9159**	**42973 ± 25386**	**88 ± 31**
F18	5.7E-21 ± 2.7E-21	1.00 ± 0.00	4 ± 3	0 ± 0	75 ± 28
F19	0.00 ± 0.00	1.00 ± 0.00	2 ± 1	0 ± 0	72 ± 34
F20	6.7E-21 ± 2.0E-21	1.00 ± 0.00	5 ± 1	0 ± 0	73 ± 34

Results in bold refer to functions associated with SR < 1.

the 15 models of DE-Rel, D3 shows the best performance overall owing to its highest SC, moderate $\bar{q}_{mean}$ and lowest NP. However, the main drawback for DE-Rel is that a user needs to decide the appropriate configurations of F, CR and M on the basis of trial and error. For our case, we should execute 450 (15 × 30) runs in to determine that the DE-Rel model with the configuration consisting of $CR = 0.5$, $F = 0.5$, and M = DE/best/2/bin achieves the best results. Three types of EPSDE models are associated with different NP values, namely, *10D*, 50D and 100D and they are denoted as E1, E2, and E3, respectively. An interesting finding can be found from the measurements of SR, $\bar{q}_{mean}$ and $\bar{q}_{best}$ for the E1 - E3 models, whereby the measurements improve with an increase in NP. However, the main drawback for EPSDE is its reliance on a user to determine the appropriate NP. Without explicitly relying on a user to set the involved parameters, SAEDE achieves high values of SC and SR. For further comparison, we use D3 and E3 to represent DE-Rel and EPSDE, respectively, for comparison with S1. By having the same ranges of parameters F, CR, NP, and M, we find that the SC and SR of the models are compatible. On the basis of a comparison of $\bar{q}_{best}$ and $\bar{q}_{mean}$, D3 achieves the highest values, followed by S1 and E3. In other words, the performance of D3 has been stagnant for a longer time than that of other DE models.

The results in Table 4 show interesting findings between measurements in the different DEs. Further investigations are required for the measurements, but the length of this paper is restricted. Therefore, Tables 5–7 show the detailed results of

Table 8
Correlation analysis and significance tests for $q_{best,g}$ and $q_{mean,g}$.

DE model	$q_{best,g}$ and $q_{mean,g}$	p-value
D3	0.906	0.00
E3	0.969	0.00
S1	0.978	0.00

Table 9
Correlation analysis and significance tests for the paired comparisons between stagnation level–f_{best} and stagnation level–SR.

DE model	$q_{best,g}$–f_{best}	p-value	$q_{best,g}$–SR	p-value
D3	0.279	0.00	-0.847	0.00
E3	0.277	0.00	-0.921	0.00
S1	0.274	0.00	-0.926	0.00

each representative model, instead of all results. Given that the number of generations for the runs could have varied for the DEs, Tables 5–7 show the mean and std dev of the involved measurements to evaluate performance consistency.

The results show that the DE models associated with low $\overline{q}_{best}$ and $\overline{q}_{mean}$ achieve a high SR. Similar findings are found for the remaining DE models of DE-Rel and EPSDE (results are not shown). This finding may indicate that the DE models that produce better offspring than their target vectors are likely to identify the optimal solutions without being stagnant for an excessive length of time. Moreover, we are interested in the scenarios in which D3 and E1 achieve $SR < 1$. For function F8, the stagnation level for D3 is within a small scale of value ($< 10^4$) when its SR approximates 1. For functions F7, F9, F13, and F17, which are associated with very low SR, the stagnation levels of D3 are within a large scale of value ($> 10^4$). Similar findings can be obtained for E3. Therefore, a high stagnation level leads to a low SR. The functions F7, F9, F13, and F17 are multimodal. DE models that are stagnant for prolonged periods have a reduced tendency to obtain optimal solutions in dealing with multimodal problems. Therefore, the measure of stagnation can be used as a guide in searching for optimal solutions. High values of stagnation may indicate that the search has been trapped in local minima, and some efforts are required to release the search. For these four multimodal functions associated with low SR, the mean and std dev of $\overline{q}_{best}$ and $\overline{q}_{mean}$ for all representative DEs are high. High std devs in both measurements can reveal that the stagnation across the runs can be different, and this phenomenon may be due to the search trapped in different local minimal positions.

On the basis of an observation of f_{best} based on mean and std dev across Tables 5–7, all representative DEs can consistently achieve low f_{best} for all functions except F7 and F8 even though some of the values are not below f_{target} ($< 1 \times 10^{-20}$). The std dev of f_{best} is small, which means that the DEs consistently generate similar solution qualities regardless of the number of runs. For function F7, std devs are high regardless of the DE type. For function F8, D3 is the only model among the three representative DEs wherein $f_{best} > 1 \times 10^{-20}$ and with high std dev. Indirectly, the constant settings of F, CR, and M in D3 can influence the identification of the global optimum of F8. In general, the performance of the representative DEs is consistent.

The results in Tables 5–7 show an evidently strong correlation between $\overline{q}_{best}$ and $\overline{q}_{mean}$ regardless of the DE models, given that both correspond to the same scale of values in different DE models. When $\overline{q}_{best}$ and $\overline{q}_{mean}$ correspond to low values, the best individual is better than its corresponding parental vector (target vector) and the entire population of solutions. In other words, the entire population is better than the parental population. Correlation analyses between $q_{best,g}$ and $q_{mean,g}$ are performed for D3, E3, and S1 to confirm this relationship. We set a significance level of 0.05, $\alpha = 0.05$; the aim is to evaluate whether the correlation between the measurements is significantly different from zero, i.e., the p-value is less than α. The correlation analyses presented in Table 8 show a strong positive correlation between $q_{best,g}$ and $q_{mean,g}$. Given that the p-values are less than α, the relationship is significant. Therefore, we may use $q_{best,g}$ only to measure the stagnation of the population. Therefore, we decide to use $q_{best,g}$ to investigate the relationship of stagnation with f_{best} and SR for the three representative DE models.

Table 9 shows the correlation analysis between the paired $q_{best,g}$–f_{best} and $q_{best,g}$–SR. We set a significance level of 0.05, $\alpha = 0.05$; the aim is to evaluate whether the correlation between the paired measurements is significantly different from zero, i.e., the p-value is less than α. The results in Table 9 indicate a strong negative correlation between $q_{best,g}$ and SR, and the relationship is significant. For the paired measurements of $q_{best,g}$–f_{best}, the positive correlation is fairly strong, and the relationship is significant. The same findings can be found in all three DE models. The correlation results show a relationship between stagnation level and f_{best} and SR. Therefore, a well-performing DE can find optimal appropriate solutions without being stagnant for prolonged periods. Moreover, we may be able to use stagnation to guard the exploration process of DE in identifying optimal solutions.

The results in Tables 5 to 7 show 16 cases in which all representative DE models achieve $SR \approx 1$. Figs. 2 to 10 show the subplots of f_{best} and stagnation level against generations for the 16 cases. Figs. 2–10 show the plots for the unimodal functions, and Figs. 11 to 17 show the plots for the multimodal functions. Each figure consists of two subplots, with the upper subplot referring to f_{best} and the lower subplot referring to stagnation. Here, we use $q_{best,g}$ to represent stagnation.

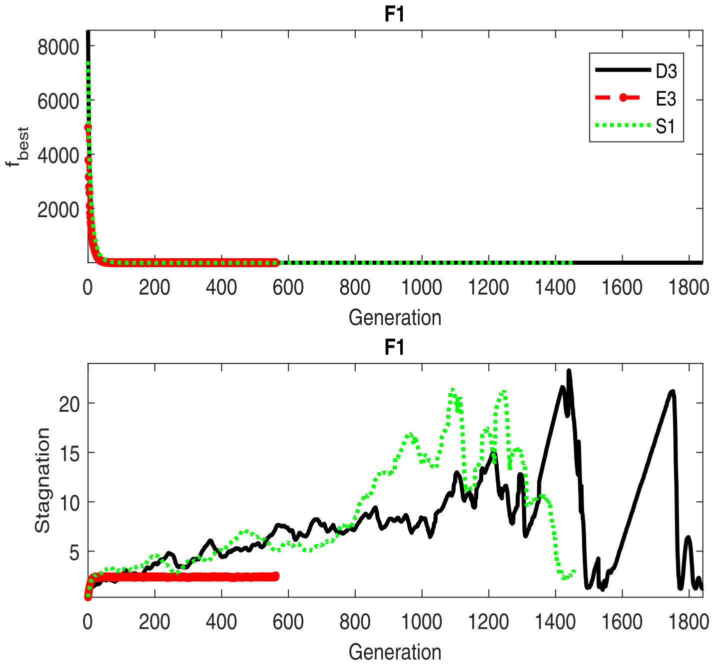

Fig. 2. Uni-modal F1 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 & S1 → E3.

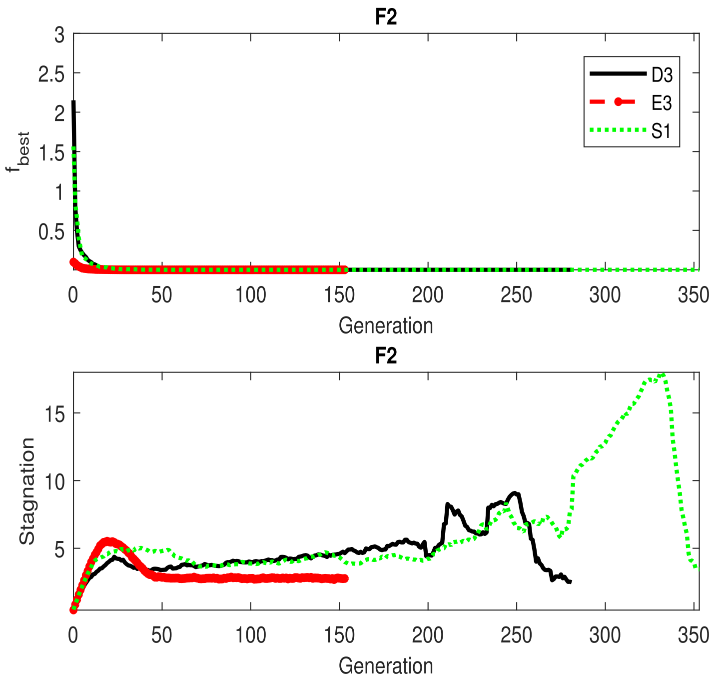

Fig. 3. Uni-modal F2 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): S1 → D3 → E3.

Each figure has a caption that briefly describes the achievement of f_{target} and stagnation level for the representative DEs. The subplots of the 16 cases have similarities regardless of the modality of the functions.

The comparisons show that E3 is usually associated with the lowest stagnation level and shortest generation among the 16 cases. E3 is followed by S1, which is associated with a moderate stagnation level and moderate generation in 7 out of the 16 cases. The last ranking model refers to D3, which is associated with the highest stagnation level and the most number

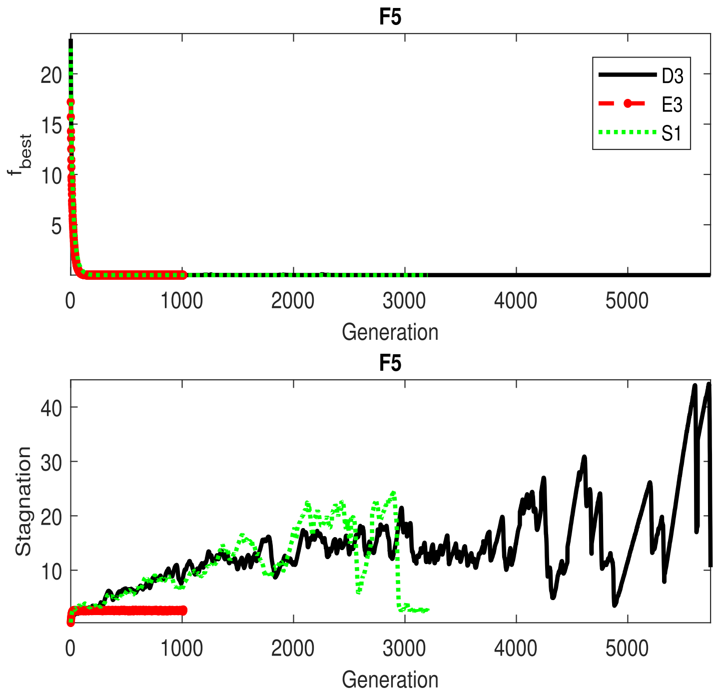

Fig. 4. Uni-modal F5 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 → E3.

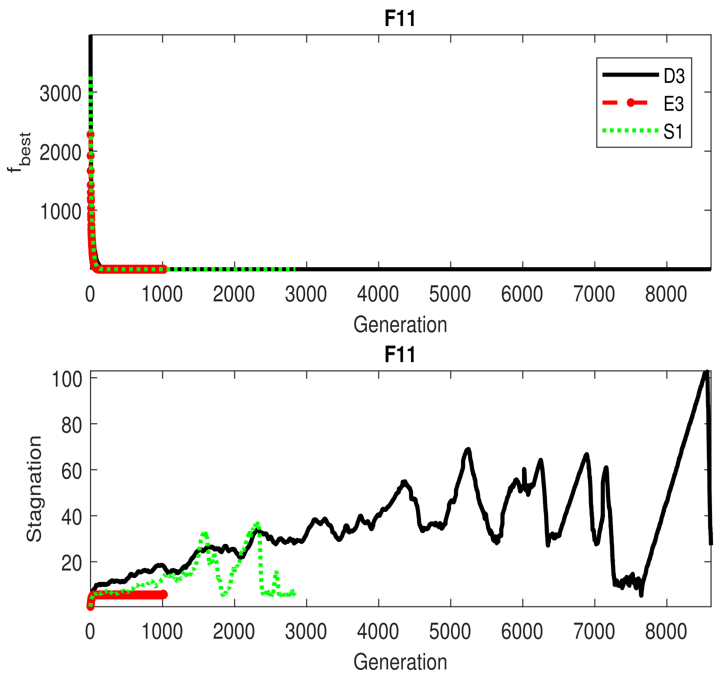

Fig. 5. Uni-modal F11 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 → E3.

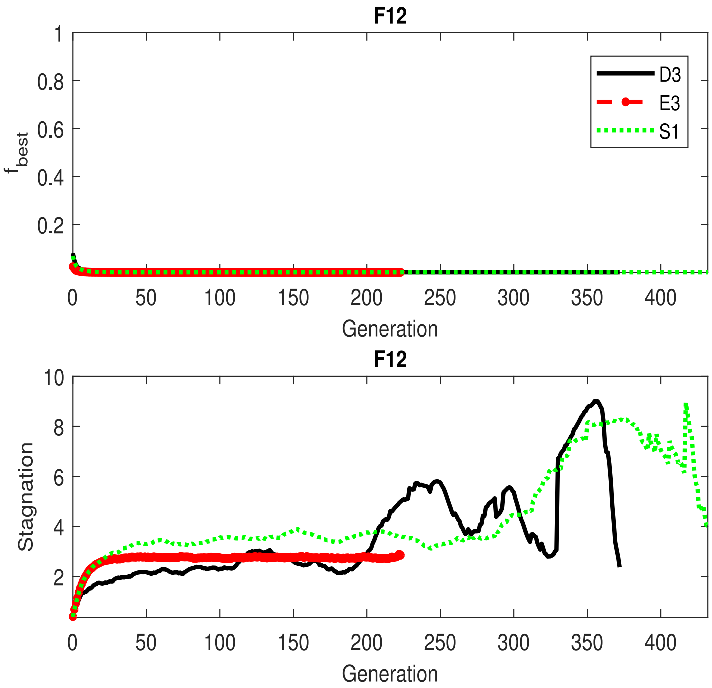

Fig. 6. Uni-modal F12 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): D3 & S1 → E3.

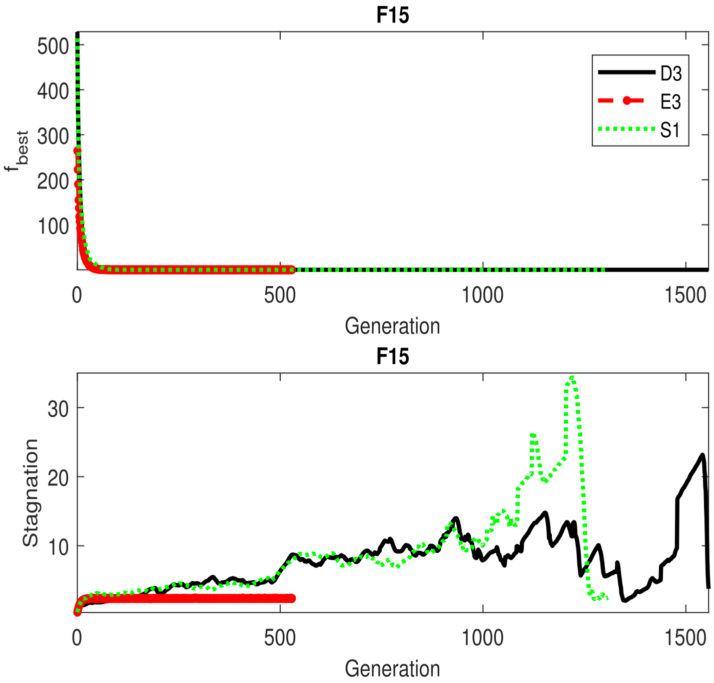

Fig. 7. Uni-modal F15 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): S1 → D3 → E3.

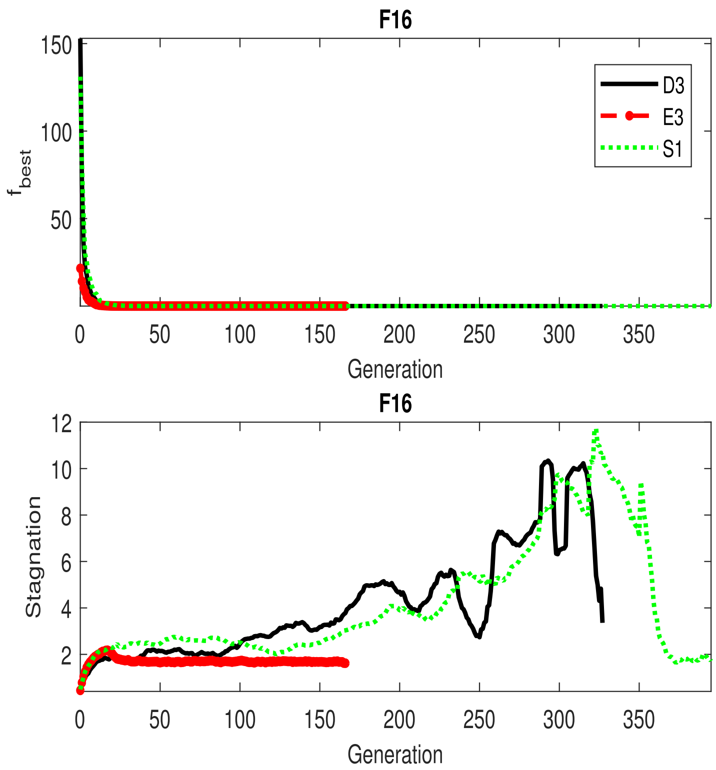

Fig. 8. Uni-modal F16 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): D3 & S1 → E3.

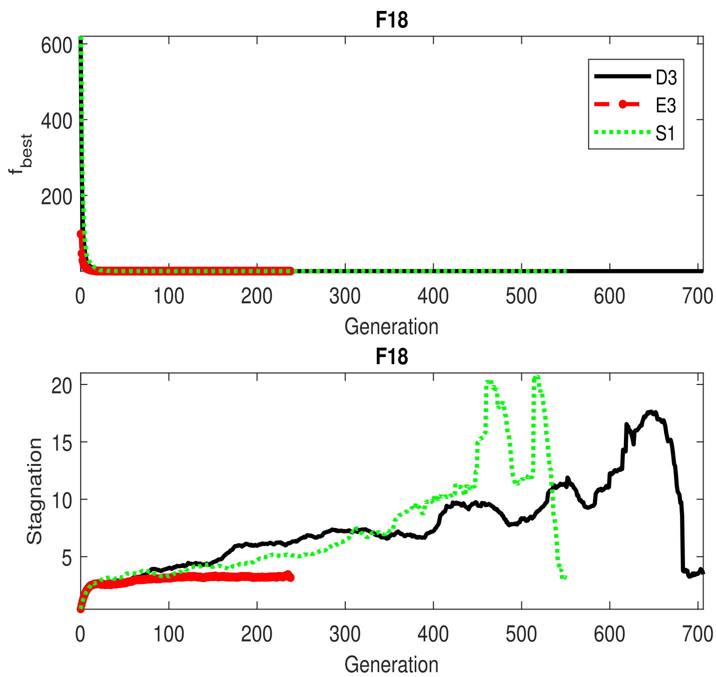

Fig. 9. Uni-modal F18 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 & S1 → E3.

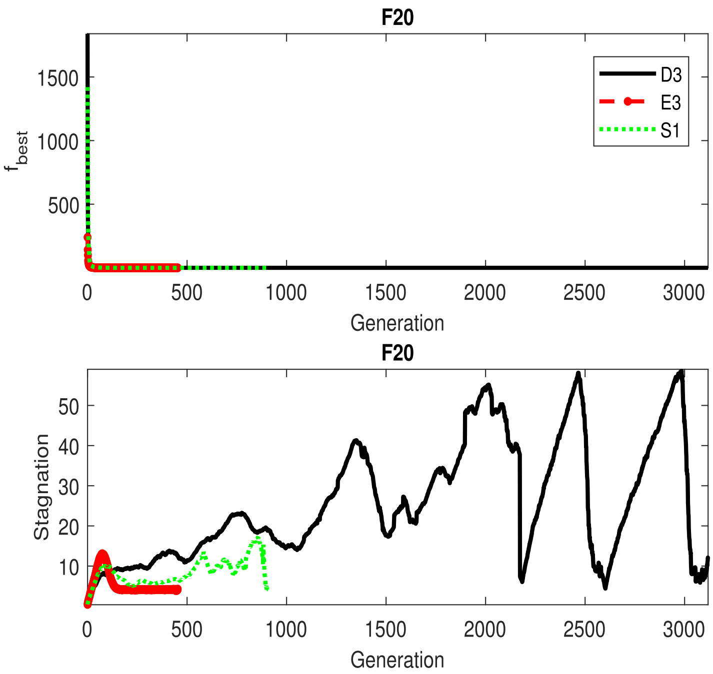

Fig. 10. Uni-modal F20 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 → E3.

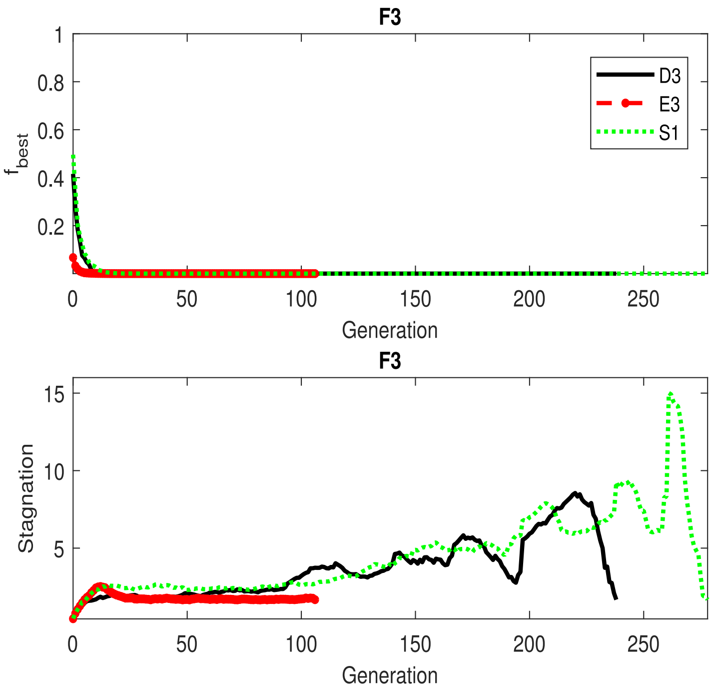

Fig. 11. Multi-modal F3 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): S1 → D3 → E3.

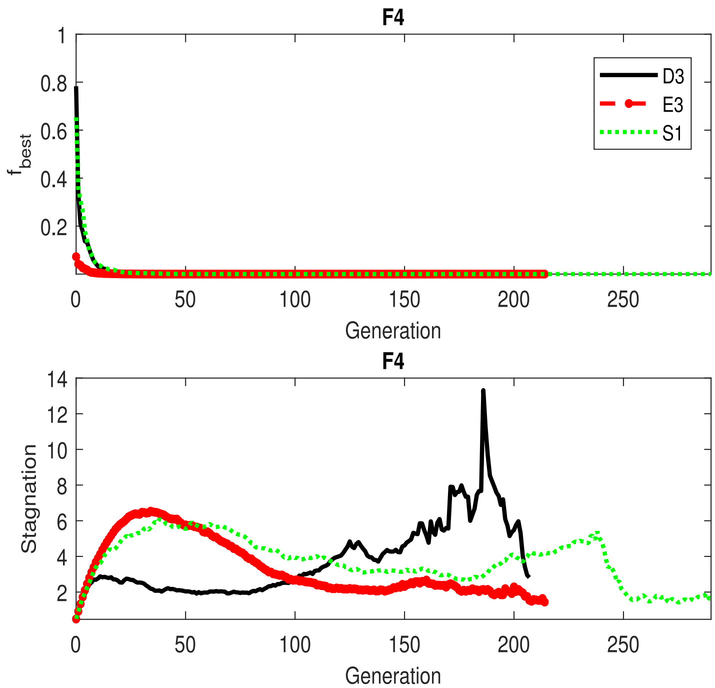

Fig. 12. Multi-modal F4 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → E3 → D3. Overall stagnation level (high → low): D3 → S1 & E3.

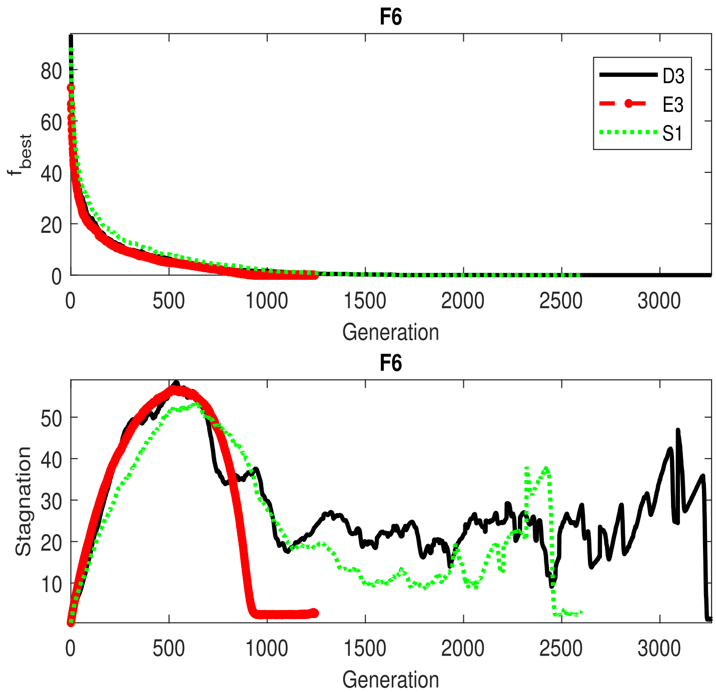

Fig. 13. Multi-modal F6 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 → E3.

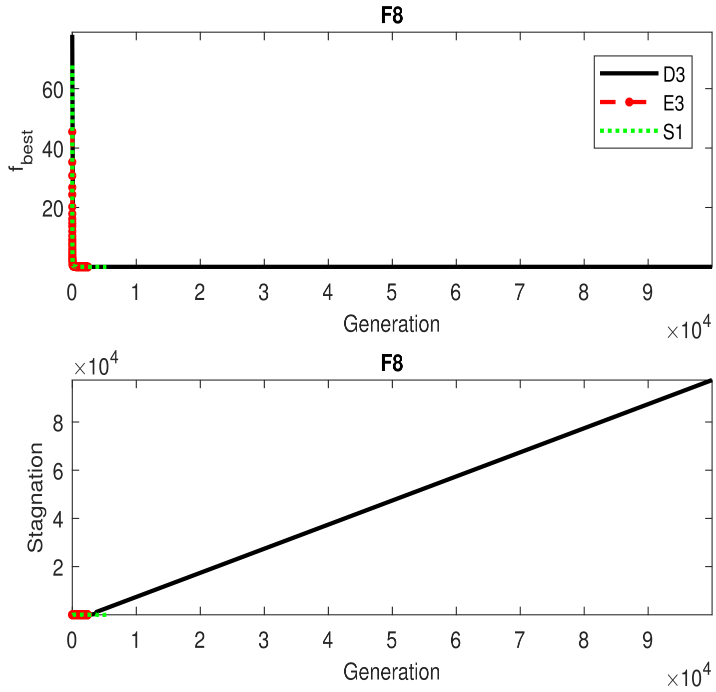

Fig. 14. Multi-modal F8 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 & E3.

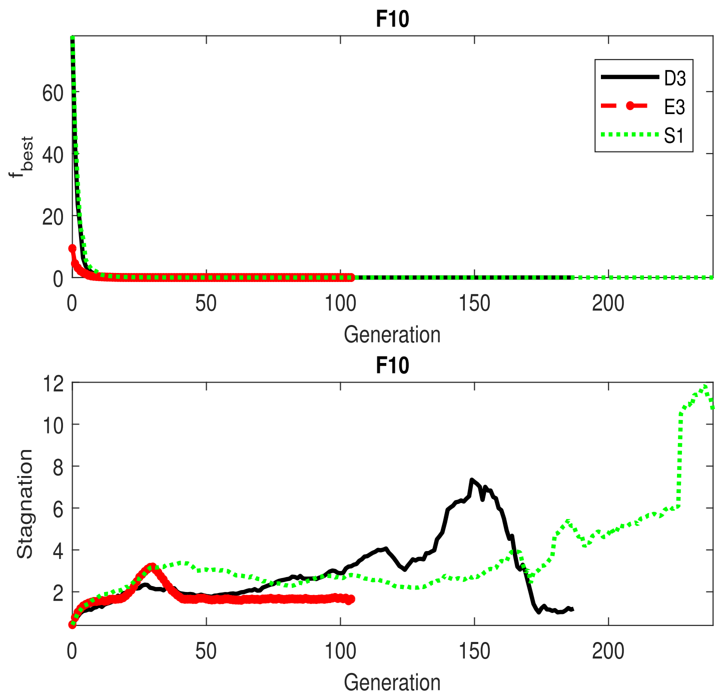

Fig. 15. Multi-modal F10 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): S1 → D3 → E3.

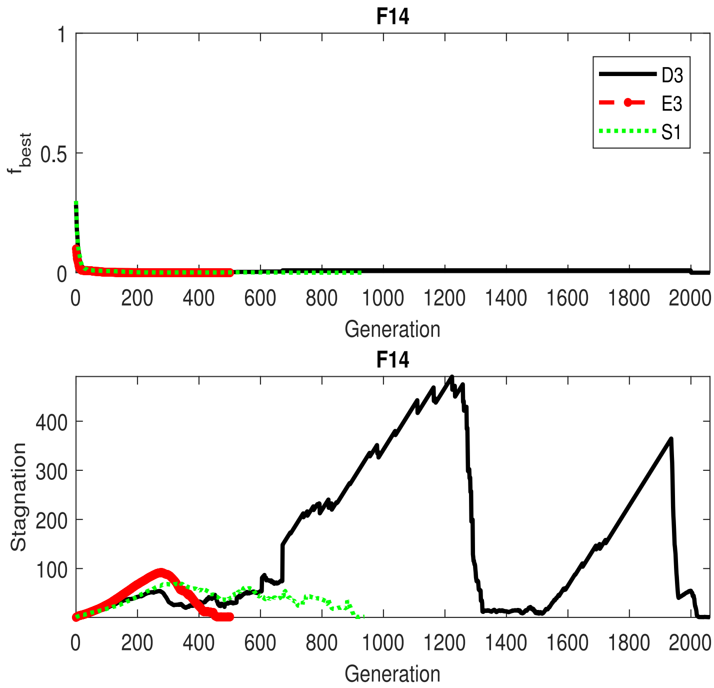

Fig. 16. Multi-modal F14 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): D3 → S1 → E3. Overall stagnation level (high → low): D3 → S1 & E3.

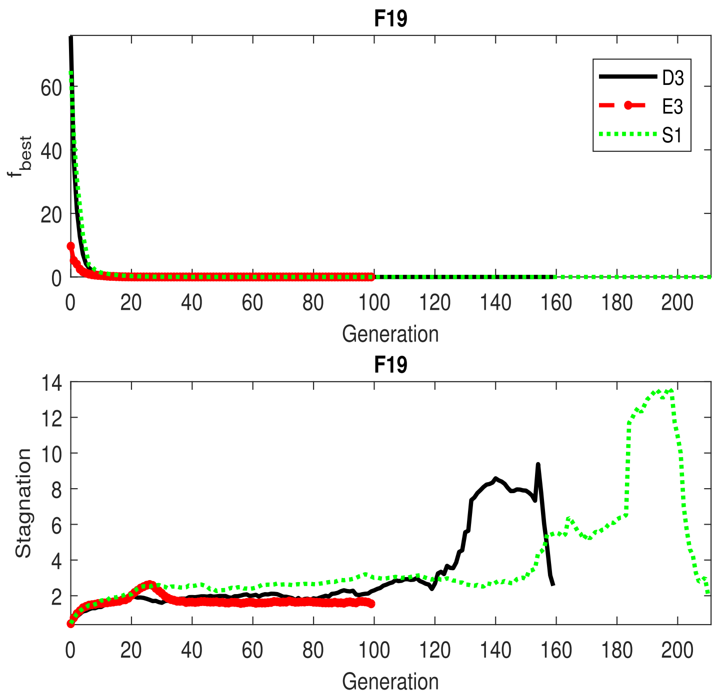

Fig. 17. Multi-modal F19 is associated with $SR = 1$ for all representative DEs. Achievement of f_{target} (slow → fast): S1 → D3 → E3. Overall stagnation level (high → low): S1 → D3 → E3.

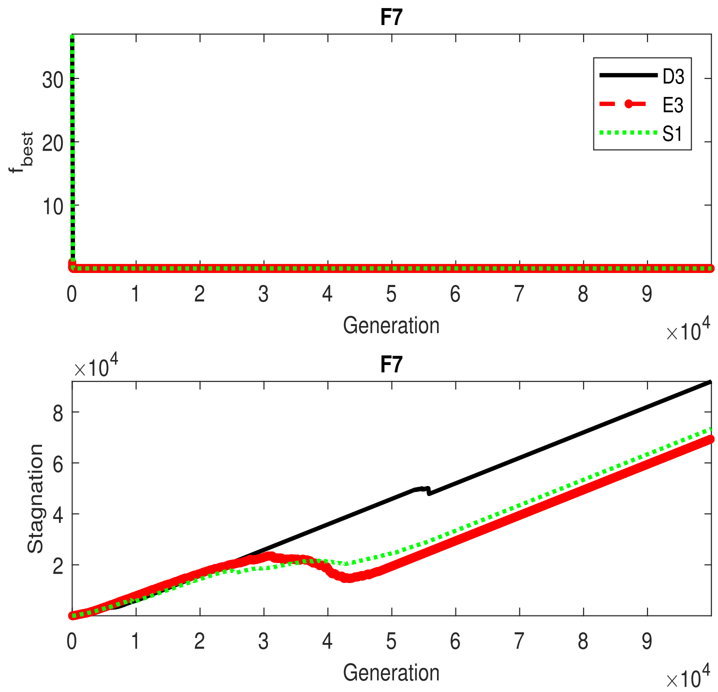

Fig. 18. Multi-modal F7 is associated with $SR < 1$ for all representative DEs. Overall stagnation level (high → low): D3 → S1 & E3.

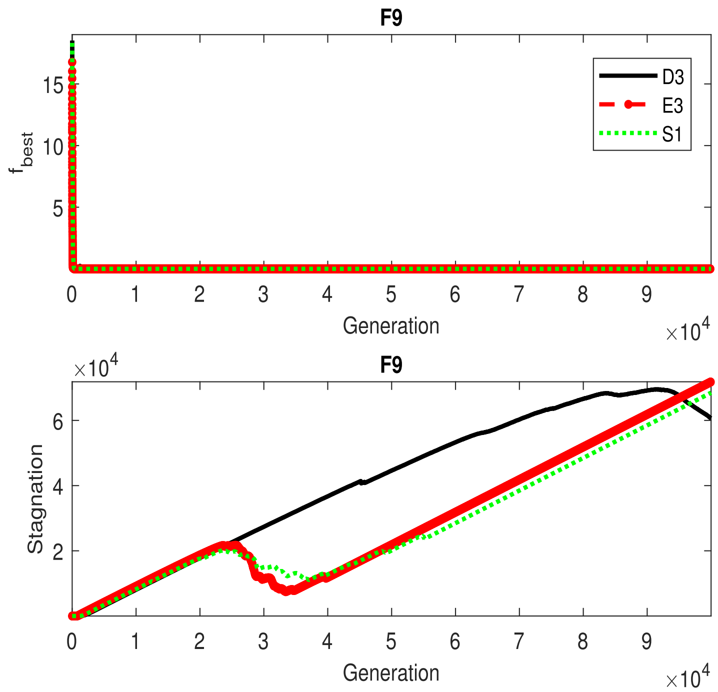

Fig. 19. Multi-modal F9 is associated with $SR < 1$ for all representative DEs. Overall stagnation level (high → low): D3 → S1 & E3.

of generations. Nine cases belong to such conditions. Given that E3 utilizes the maximum *NP*, E3 has more diverse solutions than D3 and S1. Therefore, E3 could have used a small number of generations to find the optimal solutions.

Figs. 18 to 21 show the subplots of f_{best} and the stagnation level against generations for the representative DE models, which are associated with $SR \leq 0.77$ for the benchmark functions F7, F9, F13, and F17. For these benchmark functions, the stagnation for the representative DE models increases with the number of generations, and their values are relatively similar. For benchmark functions F7 and F9, the stagnation levels for D3 are considerably higher than those of E3 and S1.

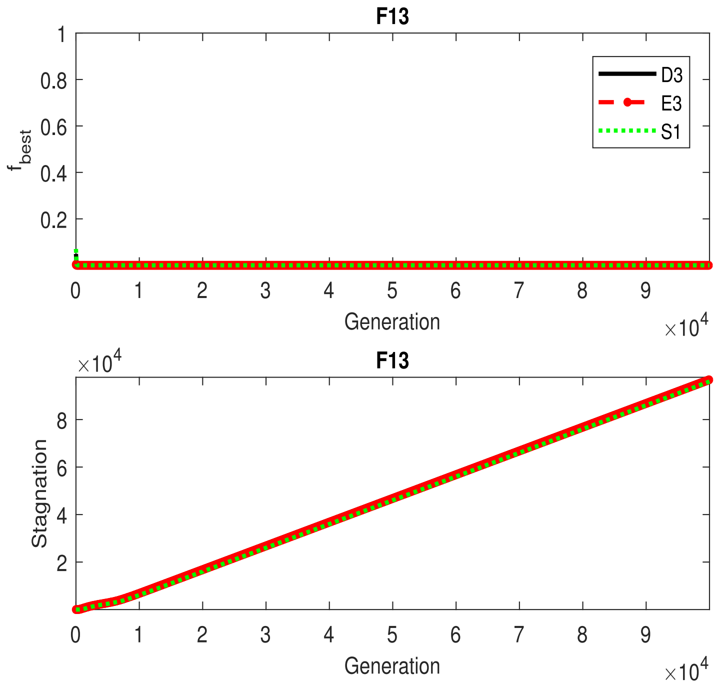

Fig. 20. Multi-modal F13 is associated with *SR* < 1 for all representative DEs. Overall stagnation level (high → low): D3 ≈ S1 ≈ E3.

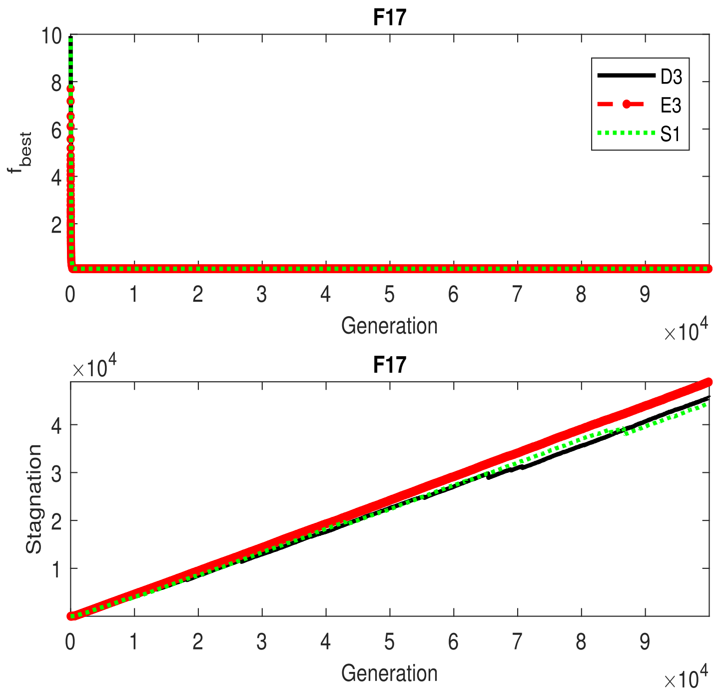

Fig. 21. Multi-modal F17 is associated with *SR* < 1 for all representative DEs. Overall stagnation level (high → low): D3 ≈ S1 ≈ E3.

Given that these functions are multimodal, the representative models have been stagnant for prolonged periods but still fail to find the optimal solutions. The DE models could have been trapped in local minima in solving the multimodal problems.

From the comparison of the three representative DE models, the smallest *NP* is associated with DE-Rel, followed by SAEDE and then is EPSDE. Therefore, the DE model associated with a large *NP* will require prolonged processing time in evolution. However, DE-Rel and EPSDE require user setting for the parameters and execution of a number of trial runs for setting evaluation. Therefore, both models consume a long time in setting up their operations. In comparison with DE-Rel and EPSDE, SAEDE is exempted from this inconvenience because it can adaptively change the setting of the parameters during evolution while still obtaining the optimal solutions. Moreover, the number of generations required by SAEDE to achieve adequate performance is moderate relative to those of the other models. Hence, SAEDE can perform well with moderate processing time and minimum interference from users. In reality, most users do not pay considerable attention to the optimization of the underlying algorithms; they may use the default parameter setting because they are interested in only solving problems. An autonomously operating DE, such as SAEDE, can emerge as a preferable option for users instead of requiring them to optimize the parameter settings.

5.2. Functions of high dimensions

The performance of the representative DEs is further evaluated on benchmark functions of high dimensions on the basis of the same measurements. Tables 10–12 show the performance measures based on mean and std dev for 30 runs.

Out of 14 benchmark functions of 30*D*, E3 and S1 achieve maximum *SR* in 12 functions, except F30 and F34. D1 achieves maximum *SR* in 8 out of the 14 functions. By focusing only on F30 and F34, we observe that the difference between the performance measures between E3 and S1 in terms of f_{best}, $\bar{q}_{mean}$ and $\bar{q}_{best}$ is not considerable. The measures for these DEs are in the same scale of values. The main strength of S1 over E3 is that it can dynamically adjust its own configurations of parameters with minimum guidance from an ordinary or expert user. Given that the performance of E3 and S1 is comparable, S1 may become a preferable option because E3 still requires the user to specify *NP*. As shown in [7], SaDE achieves the maximum SR in 8 out of the 14 benchmark functions. E3 and S1 perform better than SaDE because they are associated with $SR = 1$ in four functions more than those of SaDE even though the maximum generation used in this experiment is more stringent than the one in [7].

For S1, the adjustment of *CR* is set in the range 0.1-0.9 in steps of 0.1, and *F* is adjusted in the range 0.4-0.9 in steps of 0.1. Therefore, the changes in *CR* and *F* are in steps of 0.1. Given that S1 is the preferable option, we further investigate the effects of the control parameters on SAEDE by varying the changes in steps for these parameters. Therefore, we perform another experiment by increasing the steps for these parameters on SAEDE. With the same ranges maintained for *CR* and *F*,

Table 10
The performance measures based on mean ± std dev for D3 on benchmark functions of 30*D*.

Function	$\bar{f}_{best}$	*SR*	$\bar{q}_{mean}$	$\bar{q}_{best}$	*NP*(× *D*)
F21	9.1E-07 ± 8.8E-08	1.00 ± 0.00	4 ± 4	0 ± 0	63 ± 40
F22	9.2E-07 ± 7.4E-08	1.00 ± 0.00	332 ± 55	0 ± 0	55 ± 45
F23	**1.3E-01 ± 7.2E-01**	**0.97 ± 0.18**	**1489 ± 7904**	**634 ± 3415**	**73 ± 38**
F24	9.1E-07 ± 7.0E-08	1.00 ± 0.00	507 ± 144	0 ± 0	61 ± 44
F25	9.5E-07 ± 3.6E-08	1.00 ± 0.00	4 ± 3	0 ± 0	40 ± 38
F26	9.3E-07 ± 6.2E-08	1.00 ± 0.00	26 ± 11	0 ± 0	56 ± 43
F27	9.3E-07 ± 6.3E-08	1.00 ± 0.00	4 ± 3	0 ± 0	51 ± 39
F28	9.2E-07 ± 8.3E-08	1.00 ± 0.00	7 ± 2	0 ± 0	51 ± 42
F29	**2.8E+01 ± 1.1E+01**	**0.10 ± 0.30**	**10,546 ± 7034**	**8847 ± 9501**	**70 ± 41**
F30	**1.4E+02 ± 8.7E+00**	**0.00 ± 0.00**	**44,410 ± 14619**	**40,994 ± 29,527**	**52 ± 45**
F31	**1.8E+00 ± 1.4E+00**	**0.20 ± 0.40**	**11,042 ± 6728**	**6391 ± 7666**	**50 ± 44**
F32	6.5E-04 ± 3.5E-05	1.00 ± 0.00	4 ± 4	0 ± 0	48 ± 42
F33	**6.7E+00 ± 2.5E+01**	**0.93 ± 0.25**	**6483 ± 24,245**	**6408 ± 23,976**	**70 ± 39**
F34	**9.0E+02 ± 1.8E+01**	**0.00 ± 0.00**	**78,138 ± 6329**	**33141 ± 29078**	**68 ± 42**

Results in bold refer to functions associated with SR < 1.

Table 11
Performance measures based on mean ± std dev for E3 on benchmark functions of 30*D*.

Function	$\bar{f}_{best}$	*SR*	$\bar{q}_{mean}$	$\bar{q}_{best}$	*NP*(× *D*)
F21	9.2E-07 ± 5.9E-08	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F22	9.1E-07 ± 1.0E-07	1.00 ± 0.00	11 ± 0	0 ± 0	100 ± 0
F23	8.9E-07 ± 6.9E-08	1.00 ± 0.00	9 ± 0	0 ± 0	100 ± 0
F24	8.9E-07 ± 1.2E-07	1.00 ± 0.00	15 ± 0	0 ± 0	100 ± 0
F25	9.6E-07 ± 4.5E-08	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F26	9.3E-07 ± 8.0E-08	1.00 ± 0.00	9 ± 0	0 ± 0	100 ± 0
F27	9.0E-07 ± 9.5E-08	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F28	9.0E-07 ± 1.0E-07	1.00 ± 0.00	7 ± 0	0 ± 0	100 ± 0
F29	9.0E-07 ± 6.4E-08	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F30	**4.8E+01 ± 4.8E+00**	**0.00 ± 0.00**	**6496 ± 134**	**4334 ± 4049**	**100 ± 0**
F31	0.00 ± 0.00	1.00 ± 0.00	561 ± 41	0 ± 0	100 ± 0
F32	6.5E-04 ± 2.8E-05	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F33	9.2E-07 ± 6.9E-08	1.00 ± 0.00	4 ± 0	0 ± 0	100 ± 0
F34	**8.2E+02 ± 1.5E+02**	**0.00 ± 0.00**	**97,919 ± 265**	**97,568 ± 269**	**100 ± 0**

Results in bold refer to functions associated with SR < 1.

Table 12
Performance measures based on mean ± std dev for S1 on benchmark functions of 30*D*.

Function	$\bar{f}_{best}$	*SR*	$\bar{q}_{mean}$	$\bar{q}_{best}$	*NP*(× *D*)
F21	8.7E-07 ± 1.1E-07	1.00 ± 0.00	5 ± 2	0 ± 0	69 ± 34
F22	9.4E-07 ± 5.7E-08	1.00 ± 0.00	11 ± 1	0 ± 0	56 ± 38
F23	9.1E-07 ± 6.7E-08	1.00 ± 0.00	9 ± 2	0 ± 0	53 ± 39
F24	8.8E-07 ± 1.0E-07	1.00 ± 0.00	15 ± 3	0 ± 0	59 ± 40
F25	9.4E-07 ± 4.7E-08	1.00 ± 0.00	5 ± 3	0 ± 0	56 ± 42
F26	9.0E-07 ± 8.0E-08	1.00 ± 0.00	10 ± 3	0 ± 0	54 ± 36
F27	8.9E-07 ± 9.4E-08	1.00 ± 0.00	5 ± 1	0 ± 0	57 ± 40
F28	9.2E-07 ± 7.0E-08	1.00 ± 0.00	8 ± 1	0 ± 0	56 ± 38
F29	8.9E-07 ± 8.9E-08	1.00 ± 0.00	5 ± 3	0 ± 0	48 ± 35
F30	**5.1E+01 ± 1.1.E+01**	**0.00 ± 0.00**	**5803 ± 1192**	**3708 ± 3057**	**52 ± 44**
F31	4.1E-09 ± 2.2E-08	1.00 ± 0.00	604 ± 167	0 ± 0	46 ± 43
F32	6.5E-04 ± 3.2E-05	1.00 ± 0.00	5 ± 2	0 ± 0	53 ± 38
F33	9.1E-07 ± 7.6E-08	1.00 ± 0.00	5 ± 1	0 ± 0	56 ± 41
F34	**8.8E+02 ± 7.5E+01**	**0.00 ± 0.00**	**97,360 ± 331**	**97,078 ± 340**	**58 ± 44**

Results in bold refer to functions associated with SR < 1.

Table 13
The effects of control parameters on the performance measures based on mean ± std dev for SAEDE.

Function	$\bar{f}_{best}$		*SR*		$\bar{q}_{mean}$		$\bar{q}_{best}$		*NP*(× *D*)	
	S1	S2	S1	S2	S1	S2	S1	S2	S1	S2
F21	8.7E-07 ± 1.1E-07	9.0E-07 ± 7.2E-08	1.00 ± 0.00	1.00 ± 0.00	5 ± 2	4 ± 1	0 ± 0	0 ± 0	69 ± 34	40 ± 38
F22	9.4E-07 ± 5.7E-08	9.0E-07 ± 8.4E-08	1.00 ± 0.00	1.00 ± 0.00	11 ± 1	11 ± 3	0 ± 0	0 ± 0	56 ± 38	55 ± 43
F23	9.1E-07 ± 6.7E-08	9.0E-07 ± 9.0E-08	1.00 ± 0.00	1.00 ± 0.00	9 ± 2	8 ± 5	0 ± 0	0 ± 0	53 ± 39	59 ± 40
F24	8.8E-07 ± 1.0E-07	8.8E-07 ± 8.6E-08	1.00 ± 0.00	1.00 ± 0.00	15 ± 3	14 ± 4	0 ± 0	0 ± 0	59 ± 40	60 ± 41
F25	9.4E-07 ± 4.7E-08	9.5E-07 ± 4.9E-08	1.00 ± 0.00	1.00 ± 0.00	5 ± 3	5 ± 5	0 ± 0	0 ± 0	56 ± 42	52 ± 40
F26	9.0E-07 ± 8.0E-08	8.9E-07 ± 9.6E-08	1.00 ± 0.00	1.00 ± 0.00	10 ± 3	8 ± 3	0 ± 0	0 ± 0	54 ± 36	61 ± 40
F27	8.9E-07 ± 9.4E-08	9.0E-07 ± 8.1E-08	1.00 ± 0.00	1.00 ± 0.00	5 ± 1	4 ± 0	0 ± 0	0 ± 0	57 ± 40	50 ± 40
F28	9.2E-07 ± 7.0E-08	9.2E-07 ± 7.4E-08	1.00 ± 0.00	1.00 ± 0.00	8 ± 1	6 ± 0	0 ± 0	0 ± 0	56 ± 38	46 ± 39
F29	8.9E-07 ± 8.9E-08	9.0E-07 ± 7.6E-08	1.00 ± 0.00	1.00 ± 0.00	5 ± 3	4 ± 1	0 ± 0	0 ± 0	48 ± 35	42 ± 39
F31	4.1E-09 ± 2.2E-08	5.6E-09 ± 3.0E-08	1.00 ± 0.00	1.00 ± 0.00	604 ± 167	385 ± 32	0 ± 0	0 ± 0	46 ± 43	71 ± 41
F32	6.5E-04 ± 3.2E-05	6.5E-04 ± 2.6E-05	1.00 ± 0.00	1.00 ± 0.00	5 ± 2	4 ± 0	0 ± 0	0 ± 0	53 ± 38	47 ± 42
F33	9.1E-07 ± 7.6E-08	9.1E-07 ± 7.1E-08	1.00 ± 0.00	1.00 ± 0.00	5 ± 1	4 ± 1	0 ± 0	0 ± 0	56 ± 41	62 ± 39

CR is varied in steps of 0.2 (0.1:0.2:0.9), whereas *F* is varied in steps of 0.25 (0.4:0.25:0.90). We call SAEDE with large steps of *CR* and *F* as S2. A total of 15 (5 × 3) configurations of *F* and *CR* exist. Given that S1 has $SR = 0$ for F30 and F34, both functions are excluded in S2.

Table 13 shows the effects of the control parameters on SAEDE based on the comparison between S1 and S2. The ranges of *CR* and *F* for S1 and S2 are the same. Their difference lies in the changes in steps for the parameters. S1 has smaller steps than S2 for *CR* and *F*. From the results shown in Table 13, five performance measures, namely, f_{best}, *SR*, q_{mean}, q_{best}, and *NP*, between S1 and S2 are compatible for the involved functions. Given that S2 has larger changes in steps than S1, it has less configurations than S1. However, the condition does not prevent S2 from performing as well as S1. This result may mean that the self-adaption process enables S2 to find the appropriate control parameters even though its number of configurations is less than that of S1. In other words, SAEDE can self-adapt the control parameters on the basis of the options availability to optimize performance.

5.3. Convergence analysis

The representative DEs are selected for convergence analysis, and Table 14 shows the results. The results of f_{dif} are measured in means and std devs. The means and std devs of f_{dif} for the same benchmark functions are within a similar scale of values. This finding may show that the representative DEs have similar stabilities of convergence. Most of the f_{dif} of representative DEs show negative values, which means that improvement (albeit small) always exists in the search for efficient solutions. By assuming that the DEs have found the best solution they could possibly find, the search tends to find similar solution qualities. Therefore, f_{dif} does not differ considerably. In other words, the solutions quality is similar. Only the absolute value is considered, and the smallest mean of f_{dif} is emphasized in bold. The results in Table 14 show four functions associated with $f_{dif} = 0.00 \pm 0.00$ for all representative DEs, namely, F9, F13, F17, and F34. For these functions, the representative DEs cannot find their global optimal solutions. Given that three of these functions are multimodal, the representative DEs may be easily trapped in the local minima of the multimodal functions. As shown by the results, S1 achieves the highest frequency of the smallest mean of f_{dif} in 19 out of 34 functions. D3 achieves the smallest mean f_{dif} in 10 functions, whereas E3 achieves the same in 2 functions only. Given that S1's f_{dif} fluctuates less in most functions, it has a more stable

Table 14
Convergence analysis based on mean ± std dev of f_{dif} for the representative DEs.

Function (2D to 10D)	D3	E3	S1	Function (30D)	D3	E3	S1
F1	-4.9E-18 ± 7.4E-18	-1.2E-18 ± 4.9E-19	**-5.6E-19 ± 3.8E-19**	F21	-5.3E-06 ± 4.1E-06	-4.4E-06 ± 1.1E-06	**-3.1E-06 ± 1.6E-06**
F2	-1.0E-12 ± 4.7E-12	-1.1E-12 ± 4.6E-12	**-4.3E-13 ± 1.7E-12**	F22	**-2.2E-07 ± 1.8E-07**	-1.3E-06 ± 3.6E-07	-1.5E-06 ± 6.8E-07
F3	-4.9E-10 ± 1.7E-09	-5.2E-11 ± 1.2E-10	**-9.6E-12 ± 3.4E-11**	F23	**-5.5E-07 ± 4.1E-07**	-1.8E-06 ± 5.0E-07	-1.9E-06 ± 8.1E-07
F4	-6.6E-09 ± 3.1E-08	-3.3E-11 ± 4.3E-11	**-1.8E-11 ± 8.1E-11**	F24	**-1.8E-07 ± 1.5E-07**	-1.1E-06 ± 4.0E-07	-1.2E-06 ± 5.5E-07
F5	-9.6E-20 ± 1.1E-19	-1.0E-19 ± 2.5E-20	**-6.2E-20 ± 3.0E-20**	F25	-1.3E-06 ± 8.1E-07	-1.3E-06 ± 1.9E-07	**-1.1E-06 ± 4.1E-07**
F6	-7.1E-12 ± 7.9E-12	-2.5E-12 ± 1.1E-12	**-1.4E-12 ± 1.3E-12**	F26	-5.4E-06 ± 2.8E-05	-1.8E-06 ± 5.6E-07	**-1.7E-06 ± 6.6E-07**
F7	-5.5E-13 ± 1.1.E-12	-1.5E-11 ± 5.3E-11	**-4.7E-13 ± 1.3E-12**	F27	-5.1E-06 ± 4.1E-06	-4.4E-06 ± 9.6E-07	**-3.5E-06 ± 1.3E-06**
F8	-5.5E-14 ± 9.1E-14	-2.1E-14 ± 6.1E-15	**-9.0E-15 ± 6.8E-15**	F28	**-1.7E-06 ± 1.0E-06**	-2.1E-06 ± 4.3E-07	-2.0E-06 ± 9.2E-07
F9	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	F29	**-1.2E-07 ± 5.9E-07**	-4.2E-06 ± 9.2E-07	-3.7E-06 ± 1.7E-06
F10	-5.8E-06 ± 1.7E-05	-1.9E-07 ± 3.1E-07	**-1.7E-08 ± 3.8E-08**	F30	**0.00 ± 0.00**	**0.00 ± 0.00**	-3.0E-02 ± 1.6E-01
F11	**-2.9E-20 ± 2.0E-20**	-1.4E-19 ± 8.5E-20	-1.1E-19 ± 4.9E-20	F31	**-2.9E-03 ± 1.1E-02**	-8.9E-02 ± 2.2E-01	-1.1E-01 ± 1.7E-01
F12	-6.0E-15 ± 1.2E-14	-1.9E-16 ± 2.3E-16	**-4.0E-17 ± 4.7E-17**	F32	-1.7E-03 ± 1.3E-03	-1.2E-03 ± 3.1E-04	**-1.1E-03 ± 5.0E-04**
F13	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	F33	**-3.5E-06 ± 3.6E-06**	-4.5E-06 ± 9.1E-07	-4.0E-06 ± 1.9E-06
F14	-1.9E-06 ± 3.8E-06	-1.2E-07 ± 1.1E-07	**-3.5E-08 ± 8.0E-08**	F34	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
F15	-3.1E-18 ± 5.5E-18	-1.2E-18 ± 3.7E-19	**-6.2E-19 ± 4.7E-19**				
F16	-3.4E-12 ± 1.0E-11	-8.9E-14 ± 1.2E-13	**-1.0E-14 ± 3.1E-14**				
F17	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00				
F18	-1.2E-16 ± 2.4E-16	-4.3E-16 ± 5.8E-16	**-8.2E-17 ± 1.0E-16**				
F19	-9.7E-05 ± 3.1E-04	**-5.6E-07 ± 9.0E-07**	-1.2E-06 ± 6.4E-06				
F20	**-4.1E-19 ± 4.1E-19**	-9.7E-18 ± 8.3E-18	-1.6E-17 ± 6.4E-17				

Results in bold refer to the DE model associated with the lowest $|f_{dif}|$ for the same function.

Table 15
Convergence analysis based on Q_m for the representative DEs.

Representative DE	nsr	ntr	P_c	$fesr_{sum}$	C_m	Q_m
D3	801	1020	0.7853	1.3E+10	15,563,533	19,818,732
E3	869	1020	**0.8520**	3.4E+09	3,860,255	4,531,024
S1	863	1020	0.8461	3.1E+09	**3,585,104**	**4,237,318**

Results in bold show the best values of the respective measurements.

convergence than E3 and D3. The convergence of the representative DEs is further investigated on the basis of Q_m. On the basis of the values of P_c and C_m in Table 15, S1 has moderate probability convergence and the fastest convergence speed, which causes S1 to have the lowest Q_m. S1 is followed by E3 and then by D3. Therefore, S1 is more competitive than E3 and D3.

6. Conclusion and future work

In this study, we propose the SAEDE algorithm, which minimizes the reliance on user setting and exhausting trial-and-error efforts for appropriately configuring mutation strategy and control parameters *F, CR*, and *NP*. We also investigate the relationship between the parameter configurations and several performance measures. We compare the performance of SAEDE with that of two adaptive DE variants over 20 benchmark functions of low dimensions and 14 benchmark functions of high dimensions. The experimental results show that SAEDE is more efficient in obtaining good-quality solutions, with moderate population sizes, high success rates, and further stable convergence. Another interesting finding is the relationship between stagnation and success rate. A strong negative correlation is found between the measures, which can indicate that a high stagnation level leads to a tendency of being trapped in local minima. These findings can provide insights into how stagnation can be used to guide DE exploration, thus improving SAEDEs computation time and success rates in the future.

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.

CRediT authorship contribution statement

Shir Li Wang: Conceptualization, Data curation, Writing - original draft. **Farid Morsidi:** Conceptualization, Data curation, Writing - original draft. **Theam Foo Ng:** Conceptualization, Data curation, Writing - original draft. **Haldi Budiman:** Conceptualization, Data curation, Writing - original draft. **Siew Chin Neoh:** Conceptualization, Data curation, Writing - original draft.

Acknowledgment

This work is funded by the Ministry of Education of Malaysias Fundamental Research Grant Scheme (Grant No. 203/PCGSS/6711510, FRGS/1/2017/ICT02/UPSI/02/3) and Bridging-Intensif Universiti Sains Malaysia (Grant No. 304.PCGSS.6316349).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ins.2019.11.046

References

[1] D. Helbing, D. Brockmann, T. Chadefaux, K. Donnay, U. Blanke, O. Woolley-Meza, M. Moussaid, A. Johansson, J. Krause, S. Schutte, et al., Saving human lives: what complexity science and information systems can contribute, J. Stat. Phys. 158 (3) (2015) 735–781.
[2] H.A. Abbass, S. Alam, A. Bender, MEBRA: Multiobjective evolutionary-based risk assessment, IEEE Comput. Intell. Mag. 4 (3) (2009) 29–36.
[3] Z. Guo, G. Liu, D. Li, S. Wang, Self-adaptive differential evolution with global neighborhood search, Soft Comput. 21 (13) (2017) 3759–3768.
[4] R. Mallipeddi, G. Iacca, P.N. Suganthan, F. Neri, E. Mininno, Ensemble strategies in compact differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2011, pp. 1972–1977.
[5] A. Zamuda, J. Brest, Self-adaptive control parameters randomization frequency and propagations in differential evolution, Swarm Evolut. Comput. 25 (2015) 72–99.
[6] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution–an updated survey, Swarm Evolut. Comput. 27 (2016) 1–30.
[7] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evolut. Comput. 13 (2) (2009) 398–417.
[8] R. Mallipeddi, P.N. Suganthan, Q.-K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679–1696.
[9] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, P. Suganthan, Ensemble of differential evolution variants, Inf. Sci. 423 (2018) 172–186.
[10] L. Arya, P. Singh, L. Titare, Optimum load shedding based on sensitivity to enhance static voltage stability using DE, Swarm Evolut. Comput. 6 (2012) 25–38.
[11] L. Cui, G. Li, Q. Lin, J. Chen, N. Lu, Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations, Comput. Operat. Res. 67 (2016) 155–173.
[12] W. Zhu, Y. Tang, J.-A. Fang, W. Zhang, Adaptive population tuning scheme for differential evolution, Inf. Sci. 223 (2013) 164–191.
[13] Y. Wang, Z. Cai, Q. Zhang, Enhancing the search ability of differential evolution through orthogonal crossover, Inf. Sci. 185 (1) (2012) 153–177.
[14] D. Zaharie, Influence of crossover on the behavior of differential evolution algorithms, Appl. Soft Comput. 9 (3) (2009) 1126–1138.
[15] R. Gämperle, S.D. Müller, P. Koumoutsakos, A parameter study for differential evolution, Adv. Intell. Syst. Fuzzy Syst. Evolut. Comput. 10 (10) (2002) 293–298.
[16] R. Mallipeddi, P.N. Suganthan, Ensemble differential evolution algorithm for CEC2011 problems, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2011, pp. 1557–1564.
[17] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[18] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evolut. Comput. 15 (1) (2011) 4–31.
[19] J. Zhang, A.C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Trans. Evolut. Comput. 13 (5) (2009) 945–958.
[20] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evolut. Comput. 10 (6) (2006) 646–657.
[21] J. Brest, Constrained real-parameter optimization with ε-self-adaptive differential evolution, in: Constraint-Handling in Evolutionary Optimization, Springer, 2009, pp. 73–93.
[22] S.M. Elsayed, R.A. Sarker, D.L. Essam, Self-adaptive differential evolution incorporating a heuristic mixing of operators, Comput. Optim. Appl. 54 (3) (2013) 771–790.
[23] D. Zaharie, Control of population diversity and adaptation in differential evolution algorithms, in: Proceedings of MENDEL, 9, 2003, pp. 41–46.
[24] M.G. Omran, A. Salman, A.P. Engelbrecht, Self-adaptive differential evolution, in: Proceedings of the International Conference on Computational and Information Science, Springer, 2005, pp. 192–199.
[25] H.A. Abbass, The self-adaptive Pareto differential evolution algorithm, in: Proceedings of the Congress on Evolutionary Computation. CEC'02, 1, IEEE, 2002, pp. 831–836.
[26] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, Soft Comput. 9 (6) (2005) 448–462.
[27] I. Fister, P.N. Suganthan, S.M. Kamal, F.M. Al-Marzouki, M. Perc, D. Strnad, Artificial neural network regression as a local search heuristic for ensemble strategies in differential evolution, Nonlinear Dyn. 84 (2) (2016) 895–914.
[28] A. Nobakhti, H. Wang, A simple self-adaptive differential evolution algorithm with application on the alstom gasifier, Appl. Soft Comput. 8 (1) (2008) 350–370.
[29] S. Zhao, P.N. Suganthan, S. Das, Self-adaptive differential evolution with multi-trajectory search for large-scale optimization, Soft Comput. 15 (11) (2011) 2175–2185.
[30] C. Hu, X. Yan, An immune self-adaptive differential evolution algorithm with application to estimate kinetic parameters for homogeneous mercury oxidation, Chinese J. Chem. Eng. 17 (2) (2009) 232–240.
[31] A.P. Piotrowski, P.M. Rowinski, J.J. Napiorkowski, Comparison of evolutionary computation techniques for noise injected neural network training to estimate longitudinal dispersion coefficients in rivers, Expert Syst. Appl. 39 (1) (2012) 1354–1361.
[32] N.S. Teng, J. Teo, M.H.A. Hijazi, Self-adaptive population sizing for a tune-free differential evolution, Soft Comput. 13 (7) (2009) 709–724.
[33] L.-B. Deng, S. Wang, L.-Y. Qiao, B.-Q. Zhang, DE-RCO: Rotating crossover operator with multiangle searching strategy for adaptive differential evolution, IEEE Access 6 (2018) 2970–2983.
[34] R. Tanabe, A. Fukunaga, Reviewing and benchmarking parameter control methods in differential evolution, IEEE Trans. Cybern. (2019) 1–15.
[35] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2013, pp. 71–78.
[36] C. Wang, Y. Liu, Q. Zhang, H. Guo, X. Liang, Y. Chen, M. Xu, Y. Wei, Association rule mining based parameter adaptive strategy for differential evolution algorithms, Expert Syst. Appl. 123 (2019) 54–69.
[37] Z. Zhao, J. Yang, Z. Hu, H. Che, A differential evolution algorithm with self-adaptive strategy and control parameters based on symmetric latin hypercube design for unconstrained optimization problems, Eur. J. Operat. Res. 250 (1) (2016) 30–45.
[38] S.L. Wang, T.F. Ng, N.A. Jamil, S.M. Samuri, R. Mailok, B. Rahmatullah, Self-adapting approach in parameter tuning for differential evolution, in: Proceedings of the Conference on Technologies and Applications of Artificial Intelligence (TAAI), IEEE, 2015, pp. 113–119.
[39] R. Tanabe, A.S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: Proceedings of the IEEE congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1658–1665.

[40] N.H. Awad, M.Z. Ali, P.N. Suganthan, Ensemble of parameters in a sinusoidal differential evolution with niching-based population reduction, Swarm Evolut. Comput. 39 (2018) 141–156.
[41] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, A. Zamuda, Distance based parameter adaptation for success-history based differential evolution, Swarm Evolut. Comput. 50 (2019) 100462.
[42] K.M.A. Kadhar, S. Baskar, S.M.J. Amali, Diversity controlled self adaptive differential evolution based design of non-fragile multivariable PI controller, Eng. Appl. Artif. Intell. 46 (2015) 209–222.
[43] Z. Meng, J.-S. Pan, K.-K. Tseng, Pade: an enhanced differential evolution algorithm with novel control parameter adaptation schemes for numerical optimization, Knowl. Based Syst. 168 (2019) 80–99.
[44] L. Gui, X. Xia, F. Yu, H. Wu, R. Wu, B. Wei, Y. Zhang, X. Li, G. He, A multi-role based differential evolution, Swarm Evolut. Comput. 50 (-) (2019) 100508.
[45] S.L. Wang, T.F. Ng, F. Morsidi, Self-adaptive ensemble based differential evolution, Int. J. Mach. Learn. Comput. 8 (3) (2018) 286–293.
[46] K.V. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, Berlin, Heidelberg, 2005.
[47] A.W. Iorio, X. Li, Solving rotated multi-objective optimization problems using differential evolution, in: Proceedings of the Australasian Joint Conference on Artificial Intelligence, Springer, 2004, pp. 861–872.
[48] J.J. Liang, P.N. Suganthan, K. Deb, Novel composition test functions for numerical global optimization, in: Proceedings of the IEEE Swarm Intelligence Symposium, 2005 (SIS 2005), IEEE, 2005, pp. 68–75.
[49] G. Jeyakumar, C. Shanmugavelayutham, Convergence analysis of differential evolution variants on unconstrained global optimization functions, Int. J. Artif. Intell. Appl. 2 (2) (2011) 116–127.
[50] S.-M. Guo, C.-C. Yang, P.-H. Hsu, J.S.-H. Tsai, Improving differential evolution with a successful-parent-selecting framework, IEEE Trans. Evolut. Comput. 19 (5) (2015) 717–730.