Archives of Computational Methods in Engineering
https://doi.org/10.1007/s11831-024-10136-0

SURVEY ARTICLE

Differential Evolution: A Survey on Their Operators and Variants

Elivier Reyes-Davila[1] · Eduardo H. Haro[1] · Angel Casas-Ordaz[1] · Diego Oliva[1] · Omar Avalos[1]

Received: 13 January 2024 / Accepted: 22 April 2024

Abstract
The Differential Evolution (DE) algorithm is one of the most popular and studied approaches in Evolutionary Computation (EC). Its simple but efficient design, such as its competitive performance for many real-world optimization problems, has positioned it as the standard comparison scheme for any proposal in the field. Precisely, its simplicity has allowed the publication of a great number of variants and improvements since its inception in 1997. Moreover, several DE variants are recognized as well-founded and highly competitive algorithms in the literature. In addition, the multiple DE applications and their proposed modifications in the state-of-the-art have propitiated the drafting of many review and survey works. However, none of the DE compilation work has studied the different variants of DE operators exclusively, which would benefit future DE enhancements and other topics. Therefore, in this work, a survey analysis of the variants of DE operators is presented. This study focuses on the proposed DE operators and their impact on the EC literature over the years. The analysis allows understanding of each year's trends, the improvements that marked a milestone in the DE research, and the feasible future directions of the algorithm. Finally, the results show a downward trend for mutation or crossover variants while readers are increasingly interested in initialization and selection enhancements.

1 Introduction

Differential Evolution (DE) is one of the most popular and used algorithms in Evolutionary Computation (EC); it was developed by Rainer Storn and Kenneth V. Price around 1995 [1]. Like many other optimization techniques, the DE was created to solve real-world engineering problems as the Chebyshev polynomial fitting problem and the optimization of digital filter coefficients [2, 3]. By using the Genetic Annealing algorithm [4], Kenneth found the solution to the five-dimensional Chebyshev problem. However, he concludes that this approach does not meet the performance requirements of a competitive optimization scheme due to its slow convergence and the fact that adjusting the efficient control parameters was an arduous task. After this initial research, he experimented with modifications to the Genetic Annealing into the encoding and arithmetic operations instead of logic ones. Eventually, he discovered the differential mutation operator on which DE is based [5]. After Rainer suggested architecture design and creating a separate parent and child population, the DE was conceived as we know it now.

Due to its simple design and optimization flexibility, the DE has been applied to many scientific and engineering optimization problems such as image processing [6, 7], energy systems [8, 9], healthcare [10, 11], among several others. This flexibility feature of the DE has allowed that nowadays, its published researches in the state-of-the-art correspond to an enormous amount of related works that are almost impossible to gather in a compendium. Actually, since the inception of the DE, its rate of related published works has grown exponentially as is shown in Fig. 1. Therefore, this has propitiated the existence of many works that compile all publications related to the DE and group them according to specific topics. Of course, they are relevant works

Elivier Reyes-Davila, Eduardo H. Haro, Angel Casas-Ordaz, Diego Oliva and Omar Avalos have contributed equally to this work.

✉ Diego Oliva
diego.oliva@cucei.udg.mx

Elivier Reyes-Davila
elivier.reyes8810@alumnos.udg.mx

Eduardo H. Haro
eduardo.hernandezh@academicos.udg.mx

Angel Casas-Ordaz
angel.casas5699@alumnos.udg.mx

Omar Avalos
omar.avalos@academicos.udg.mx

[1] Depto. de Ingeniería Electro-Fotónica, Universidad de Guadalajara, CUCEI, 44430 Guadalajara, Jalisco, Mexico

Published online: 23 May 2024

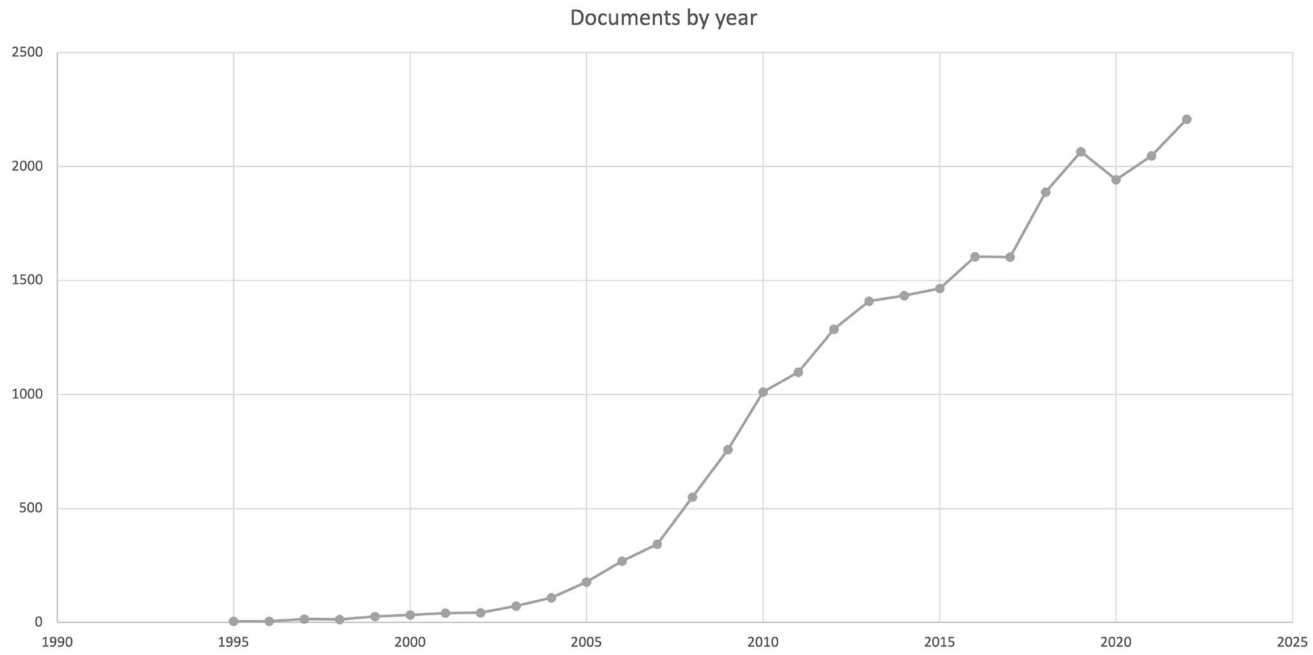

Fig. 1 Amount of works related to the DE published per year since 1995

that contribute priceless value to their respective fields of study [12]. These publications usually present a brief general review of the DE and its history, like the survey of Opara et al. [13] where the main DE variants are discussed through the years. Also, Parouha presented a recent DE overview where its most important variants are chronologically compared against the Particle Swarm Optimization (PSO) variants [14, 15]. Finally, Mashwani discusses the different evolutionary strategies inspired by the DE [16], to mention a few related works. Regarding the concept of multiple ways to expose a DE review work, another remarkable way to expose the state-of-the-art of the DE is according to its optimization field. Chakraborty published a priceless recent review of the different applications of DE in image processing problems [17], while Qing presented a book explaining in detail all the electrical engineering applications of the DE [18]. In other engineering fields like chemistry and communications, some important summarize works have been published, like the review of Dragoi et al. [19] who exposed the use of DE in chemical optimization problems over the years, in the same way, Okagbue developed a precise analysis of the impact of DE in wireless communications through the years [20], just for mention some review applications. On the other hand, a classical way to perform a review work of the DE is according to specific algorithm features; however, it is worth mentioning that these types of works are usually the minority among the DE reviews. One of the most recent examples was presented by Piotrowski et al. [21], who resumed the relevance of different population sizes in DE variants. Xin presented the essential hybridizations of DE and PSO algorithms to determine the best design optimizers [22]. A related recent and relevant summary work was presented by Tanabe et al. [23], where the most critical ways to tune the control parameters of DE (mutation and crossover rates) are analyzed. Finally, a classical feature of any optimization approach is its capability to be competitive in multi-objective optimization problems (MOPs); in that sense, the most recent related work was proposed by Ayaz, who exposed the most remarkable advances in the state-of-the-art for the improvement of DE in MOPs [24]. All these feature information reviews of DE are just a few of the many related works.

As noted, the DE has been summarized in many ways through the years, and these works must keep existing since the published advances in DE are just growing in terms of amount and complexity [25]. However, the review (or surveys) works of the DE correspond only to 1% of the DE published works; the rest of the topics are shown in Fig. 2. As can be seen, most of the works correspond to journal articles from different scientific fields. Moreover, after 28 years of existence of the DE, there is still a lack of information that has been vaguely broached in some review works of the DE history, the DE operators, and its variants. The importance of discussing the DE operators lies in the fact that nowadays, these operators number in the dozens, and to the best of the author's knowledge, there is no work in the literature that summarizes all of them. Moreover, as explained before, it is well documented that the accuracy and optimization of the DE and its variants are directly related to the quality of its respective operators. Thus, knowing the advances and

limitations of all DE operators in the literature is crucial to proposing a real DE improvement. Regarding the review and survey works reported in Fig. 2, in this work, only the reviews and survey works directly related to the DE itself have been considered. This is quite remarkable since DE is one of the algorithms most analyzed in most research fields. Another important data is that this work only considers compilation works unrelated to Open Access journals. Thus, the literature reports many compilation works about the DE applied to said fields. In that sense, Table 1 presents all the compilation works used for this article, the editorial they belong to, and their publication year. The highlight is that all the compilation works with the aforementioned specifications started to be published in 2010. Also, it is seen that Springer used to bet on these types of works in the first half of the decade, while Elsevier led the rest of the decade. This last editorial continues to publish this type of work.

To solve this lack of information, a survey of all the DE operators exposed in the literature is presented in this article. The four main stages of the algorithm (initialization, mutation, crossover, and selection) are exposed, analyzed, and discussed. Moreover, each stage is separately broached; their

Table 1 Summarize the reported review works of DE in the literature

No	Refs.	Title	Editorial	Year
1	[26]	Differential evolution: A recent review based on state-of-the-art works	Elsevier	2022
2	[27]	Differential Evolution: A review of more than two decades of research	Elsevier	2020
3	[23]	Reviewing and Benchmarking Parameter Control Methods in Differential Evolution	IEEE	2020
4	[13]	Differential Evolution: A survey of theoretical analyses	Elsevier	2019
5	[28]	A review of the recent use of Differential Evolution for Large-Scale Global Optimization: An analysis of selected algorithms on the CEC 2013 LSGO benchmark suite	Elsevier	2019
6	[29]	Review of Differential Evolution population size	Elsevier	2017
7	[30]	Recent advances in differential evolution – An updated survey	Elsevier	2016
8	[31]	Parameter control and hybridization techniques in differential evolution: a survey	Springer	2016
9	[32]	Differential Evolution: An Overview	Springer	2016
10	[33]	Parameter control mechanisms in differential evolution: A tutorial review and taxonomy	IEEE	2013
11	[34]	Differential Evolution Algorithm: Recent Advances	Springer	2012
12	[35]	Recent advances in differential evolution: a survey and experimental analysis	Springer	2010
13	[36]	Differential evolution: A survey of the state-of-the-art	IEEE	2010

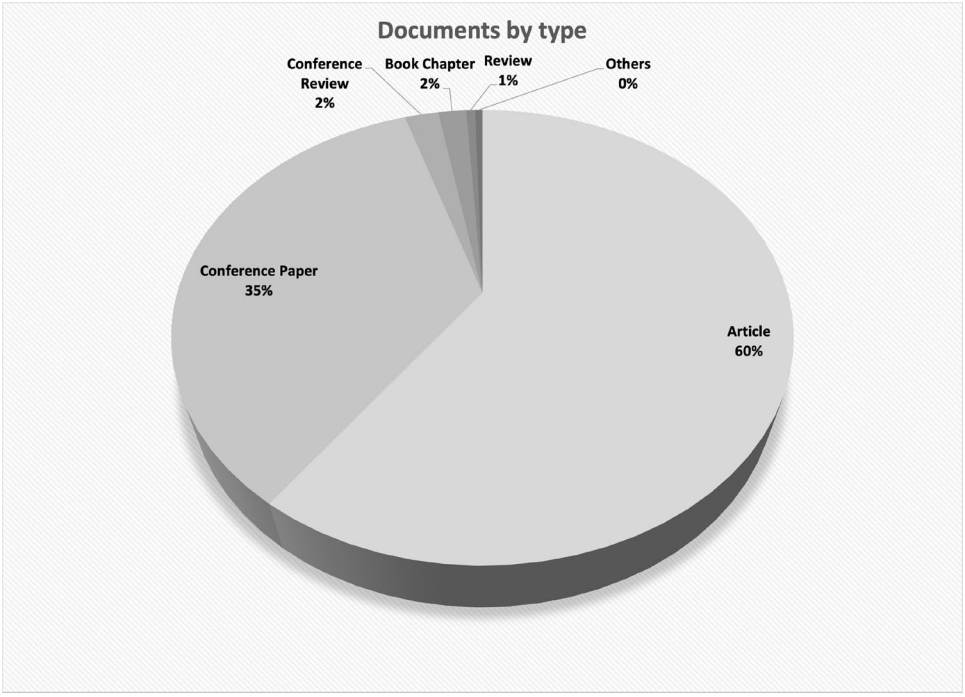

Fig. 2 Amount of articles related to DE per publication type

operator variants are analyzed individually so the reader can easily find those operators that may be considered more relevant, instead of a single long paragraph where the information can be lost. Additionally, at the end of its respective subsection, a table summarizes the advantages and limitations of each operator variant so the reader can judge those techniques that seem more attractive for future research. On the other hand, once all the operators are presented, the survey ends with a discussion subsection for each stage of the DE, trying to offer an objective perspective of the schemes, as mentioned earlier.

In the same way, these discussions allow us to compare all the operators simultaneously to expose the best ones in terms of optimization or give a general view of future opportunities for each operator. Finally, it is worth mentioning that this survey work pretends to be a valuable source of information for future DE improvements. Therefore, gathering information on the state-of-the-art covers 113 works published up to the date of development of this work, demonstrating that the presented information is vast and treated through a rigorous scientific gathering process. Summarizing the given information in this section, this novel review work can be synthesized in the following five features:

- This work compiles a considerable amount of the DE operator and variants presented in the literature.
- The broached literature processes scientifically 113 publications regardless of the fields of study.
- The exposed information is orderly presented for easier reading to the reader.
- All the presented operators are analyzed and discussed deeply.
- This work corresponds to a compendium of priceless information for future DE improvements.

The rest of the article is as follows: in Sect. 2, the basics of DE are explained, Sect. 3 presents a study of the DE operator variants in their respective stages, the discussions of the stages of the DE are exposed in Sect. 4, finally Sect. 5 reports the conclusions of the authors.

2 Differential Evolution

As is known, Differential Evolution (DE) is a population-based metaheuristic algorithm that uses an iterative process to find solutions to a problem. This search for solutions is achieved by generating improved candidate solutions through an evolutionary process. This process consists of four stages: initialization, mutation, crossover, and selection. Figure 3 shows the flowchart of the general scheme of the DE algorithm. These stages are briefly described in the following subsections.

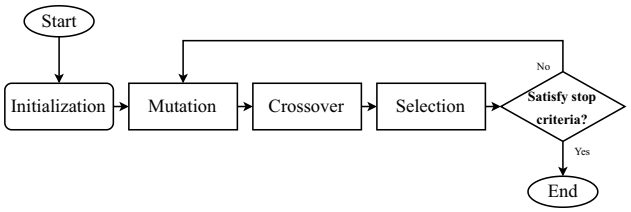

Fig. 3 Flowchart of DE algorithm

2.1 Initialization Stage

The initialization stage is the first step in the DE algorithm process to find the global optimal solutions within the search space. This process stage randomly generates an initial population $x^d_{i,G}$. The initial population of size Np consists of i-th individuals with d-dimensionality for each G-th generation, represented as $x_{i,G} = \left\{x^1_{i,G}, x^2_{i,G} \ldots, x^d_{i,G}\right\}$ and is generated by a random process with the following equation:

$$x^d_{i,0} = x^d_{min} + rand(0,1) * (x^d_{max} - x^d_{min}) \tag{1}$$

where $i = 1, \ldots, Np$, $rand(0, 1)$ represents a uniformly distributed random variable within the range [0,1], x_{min} and x_{max} are the lower and upper bounds for each dimension of the search space. Once the initial population has been generated, the steps involved in the iterative process are carried out.

2.2 Mutation Stage

DE algorithm implements a perturbation process performed on the elements of each candidate solution to produce a modified version of each individual called mutant vector $v^d_{i,G}$. The most common mutation operator used in the DE algorithm is the DE/rand/1, defined in Eq. 2.

$$v^d_{i,G+1} = x_{r3,G} + F(x_{r1,G} - x_{r2,G}) \tag{2}$$

where x_{r1}, x_{r2}, x_{r3} are population vectors which are randomly selected considering that $r1 \neq r2 \neq r3 \neq i$, and F is the scale factor that is a fixed value by the user in the range of [0, 2].

2.3 Crossover Stage

In this stage, the target individual is combined with a randomly chosen individual, generating a trial individual. This process is guided by the crossover probability (Cr) (a constant value in the [0, 1] range). To generate a new vector, also called trial vector $u^d_{i,G}$, the crossover is performed between target vector $x^d_{i,G}$, and mutant vector $v^d_{i,G}$ as follows:

$$u^d_{i,G+1} = u^d_G \begin{cases} v^d_{i,G} & \text{if } d = d_{rand} \text{ or } rand(0,1) \leq Cr \\ x^d_{i,G} & \text{otherwise} \end{cases} \tag{3}$$

where d_{rand} is a randomly chosen index $\in 1, 2, ..., d$ which ensures that $u_{i,G}$ includes at least one parameter of the mutant vector $v_{i,G}$.

2.4 Selection Stage

The selection process enables DE to determine the survival of the target or trial solution in the next generation through a comparison according to their fitness values. The vector with the fittest values continues in the next generation's population. This operation is performed as follows:

$$x_{i,G+1} = \begin{cases} u^j_{i,G} & \text{if } f(u_{i,G+1}) \leq f(x_{i,G}), \\ x^j_{i,G} & \text{otherwise,} \end{cases} \quad (4)$$

Once the individual in the population is replaced by a new one, the mutation, crossover, and selection process is repeated until the optimum is located or the specific termination criterion is satisfied.

Each stage in the DE algorithm process will be examined and expanded upon in the following sections.

3 A Study of the Differential Evolution Operators and Their Variants

This section analyzes the operators proposed in the literature related to DE. It explains the variants that are possible to find for each of the DE stages (initialization, crossover, mutation, and selection). Such modified operators were taken from previously published articles and explained. Besides, some advantages and limitations of the operators are discussed.

3.1 Initialization in DE

In general terms, an initialization approach is always used in different algorithms to define their initial parameters and establish the initial population. Different studies have demonstrated that the quality of the search of an algorithm is directly related to the quality of the initial population [37, 38]. In the same way, a correctly initialized population can be understood as a population of individuals where diversity is the most essential feature. Based on the above, different initialization processes in the literature have been used in the DE. However, many related approaches use an initialization already established in the literature. That is why, in this section, only the proposed techniques in which the initialization phase is one of the novelties of its proposal will be analyzed.

- Micro-opposition-based differential evolution
  Rahnamayan [39] proposed an initialization based on micro opposition (micro-ODE) with tiny population size. This algorithm was used for sixteen image threshold tasks. The results were compared against the well-known Kittler approach, and the micro-ODE demonstrated its remarkable benefits against the Kittler method [40].
- Quadratic interpolation differential evolution
  Pant [41] presented a novel work in which quadratic interpolation is applied in the DE to improve the algorithm's convergence rate. The QIDE was tested on 10 benchmark problems and compared against the DE and the Opposition-based-learning DE (ODE). The results proved a significant improvement to the classical DE.
- Nonlinear simplex differential evolution
  Ali [42] used three different initialization schemes in DE to determine the best one in computational terms. The novelty of his work is the implementation of the Nonlinear Simplex Method (NSM) as an initialization technique [43]. A comparison was made between the DE, ODE, and QIDE, demonstrating the competitiveness of NSDE.
- Smart sampling differential evolution
  De Melo [44] used machine learning techniques to find promising regions in a continuous search space. Through the Smart Sampling method (SS), the proposed SSDE was tested in a set of benchmark functions against the ODE, QODE, and UQODE algorithms.
- Improved self-adaptive differential evolution with multiple strategies
  Deng [45] proposed a novel variation of DE based on a dynamical population divided according to the fitness of the individuals. Deng also used an adjustment method to tune the F and Cr factors. The ISDEMS was compared with the DE, ACDE, and SACDE on different optimization tests, proving the good performance of the ISDEMS.
- Adaptive population topology differential algorithm
  Sun [46] structured a new DE variation for unconstrained optimization problems. The APTDE bases its performance on actualizing population topology to avoid premature convergences. The results showed the algorithm's remarkable efficiency.
- Adaptive population tuning scheme
  Zhu [47] based his work on the dynamic population concept, where the redundant individuals are removed according to their ranking order. This APTS allows for the improvement of diversity in the population, which is effective, as the results demonstrated, for various evolutionary strategies.
- Symbiosis co-evolutionary model based on the population topology differential evolution
  Sun [48] proposed a DE variant similar to the APTDE, where the population is divided to improve diversity. The SCoPTDE was compared with other related approaches, and remarkable results were obtained.
- Adaptive multi-population differential evolution

Wang [49] presented the concept of dynamical population but applied it to multi-objective systems. His work is a hybrid MOEA that allows the improvement of the exploitation of the DE. The results showed important benefits to DE regarding multi-objective search strategies.

- Cumulative differential evolution
  Aalto in [50] presented a modified DE where the population is adapted based on the probability mass function (Cumu-DE). This was designed to auto-tune the three-parameter settings of the DE. The approach proved to be faster than other related schemes and computationally effective.
- Ensemble sinusoidal differential evolution with niching reduction
  Awad [51] published a variant of LSHADE [52] (EsDEr-NR), where the exploration and exploitation phases are improved through a mixture of two sinusoidal formulas of the already best solutions. The proposal outperformed other related approaches, including CMAES variants [53].
- Reinitialization mid-point check
  Du Plessis [54] proposed a novel technique for Dynamic DE (DynDE) [55]. This approach allows populations to compete for function evaluations based on their performance. The results show that all DynDE approaches have improved significantly.
- Self-adaptive strategy differential evolution with symmetric Latin hypercube design
  Zhao [56] developed a technique where the population is initialized by a symmetric Latin hypercube (SLADE-SLHD), which increases the population's diversity.
- Chaotically initialized differential evolution
  Ozer [57] presented a novel DE initialized by seven different chaotic maps (CIDE), demonstrating that the sinusoidal and circle chaotic maps significantly outperform the DE in diverse scenarios.
- Cluster-based population initialization for differential evolution frameworks
  Poikolainen [58] developed an initialization of three stages (CBPI), where a randomly generated population is clustered by a K-means approach and grouped by the Euclidean distances of the individuals to improve the quality of the initial population.
- Differential evolution based on clustering
  Bajer [59] developed a novel initialization technique (DEc) where promising regions are searched through a clustering process, and then, a simple Cauchy mutation method is used to generate new individuals in the region [60].
- Adaptive multi-population differential evolution with dynamic population reduction
  Ali [61] presented a new DE variant (sTDE-dR), which clusters the population in multiple tribes, and then the algorithm utilizes different mutation and crossover schemes for each tribe to reduce dynamically the population according to the success of the tribes.
- Heuristic for initialization of cluster centers
  Mustafi [62] proposed a new initialization for the k-means algorithm through evolutionary strategies. Essentially, the DE generates a requisite number of clusters at each iteration; by doing this, the empty clusters are avoided, and the k-means are improved.
- Chaotic oppositional differential evolution
  Ahmad [63] presented a novel DE variant that combines the concept of chaotic maps and Opposition-Based Learning strategies [64] to improve the quality of the proposed initialized solutions of the algorithm. The scheme is applied to different classical DE variants over several benchmark functions, demonstrating the viability of this new initialization process (Table 2).

3.2 Mutation in DE

DE has proven to be one of the most popular and successful evolutionary algorithms for solving optimization problems. However, due to the problems that continually emerge from modern application areas, there is still a need for improvement in the search performance of this algorithm [65]. The performance and effectiveness of DE heavily rely on the selected mutation operator and its associated control parameter value. This has led several researchers to focus their work on the ensemble of new approaches or improvement of these strategies to make the performance of DE more robust. A Mutation in biology can be defined as an instant change in a chromosome's gene composition. In contrast, in the context of evolutionary computation, it can be referred to as a random perturbation process performed on selected decision variables. In the DE algorithm, an individual is taken as a donor and perturbed with a scaled vector differential from the other two individuals to produce a mutated individual. Generally, the DE mutation strategies can be represented in the format 'DE/α/β', where DE stands for Differential Evolution, α specifies the base vector to be perturbed, and β refers to the number of difference vectors involved in the mutation process.

Notably, no single mutation strategy satisfactorily solves all the problems that arise [30] because an operator may be effective for specific problems but perform poorly for others. It is similar to what the "No free lunch theorem [66]" establishes. For this reason, different mutation strategies with different control parameter settings may better solve a particular problem than a single mutation strategy with

Table 2 Summary of the DE initialization proposed schemes: advantages and limitations

Technique	Advantages	Limitations
micro-ODE *micro-Opposition-based differential evolution* [39]	The micro-ODE requires a lower NFC, also by using small population size, the computational time is lower	The performance of the proposal depends on the designed fitness function
QIDE *Quadratic interpolation differential evolution* [41]	The QIDE is computationally less expensive than the ODE	The algorithm has not been tested on constrained real-life problems
NSDE *Nonlinear simplex differential evolution* [42]	The NSDE is computationally similar to QIDE in almost all aspects	The tests probed that NSDE is unsuitable for hybrid and composition functions
SSDE *Smart sampling differential evolution* [44]	The SSDE probed to be highly competitive in continuous search spaces	The SSDE did not perform well in all functions, especially in non-continuous search spaces
ISDEMS *Improved self-adaptive differential evolution with multiple strategies* [45]	The ISDEMS is highly competitive in terms of search precision and convergence performance	The dynamically divided population makes the ISDEMS computationally expensive
APTDE *Adaptive population topology differential algorithm* [46]	The APTDE is more efficient in terms of solution quality and function evaluations	The algorithm is designed exclusively for unconstrained optimization problems
APTS *Adaptive population tuning scheme* [47]	The APTS demonstrated to improve significantly to all EAs, especially to JADE	The APTS has not been tested for constrained or multi-objective problems
SCoPTDE *Symbiosis co-evolutionary model based on the population topology differential evolution* [48]	The experiments show that SCoPTDE improves the performance of DE, obtaining higher solution quality and stronger robustness	The algorithm still needs to be treated for constrained optimization problems
AMPDE *Adaptive multi-population differential evolution* [49]	The proposal outperforms the rest of related multi-objective approaches in computational terms in the literature	The algorithms still need much more experiments on real industry applications
Cumu-DE *Cumulative differential evolution* [50]	The Cumu-DE significantly outperforms the DE in almost all experiments	The Cumu-DE probed to be inconsistent for constrained optimization problems
EsDEr-NR *Ensemble sinusoidal differential evolution with niching reduction* [51]	The EsDEr-NR outperformed other CMAES and LSHADE variants and probe to be highly competitive	The EsDEr-NR has not been applied for engineering nor real optimization problems
RCM *Reinitialization midpoint check* [54]	The proposed scheme probed to have a good performance in high dimensional problems	The RCM has not been tested on dynamical nor multi-objective problems
SLADE-SLHD *Self-adaptive strategy differential evolution with symmetric Latin hypercube design* [56]	The SLADE-SLHD was computationally competitive against the rest of the approaches in almost all functions	The algorithms still need experimentation in constrained problems
CIDE *Chaotically initialized differential evolution* [57]	The results show that applying deterministic chaotic maps instead of random sequences can improve the performance of DE	The CIDE has not been tested on complex benchmark functions; thus, it is a relatively unexplored approach
CBPI *Cluster-based population initialization for differential evolution frameworks* [58]	The CBPI was adapted to different DE variants, obtaining important improvements in all benchmark functions and engineering problems	The computational cost of the DE variants increased significantly, and the CBPI reported difficulties in multi-modality tests
DEc *Differential evolution based on clustering* [59]	The DEc benefits from all the evolutionary strategies utilized since their convergence rates were improved	The approach was not competitive in constrained optimization problems. Therefore, it still needs much more analysis
sTDE-dR *Adaptive multi-population differential evolution with dynamic population reduction* [61]	The sTDE-dR demonstrated its robustness against other 10 DE variants in the CEC2014 benchmark dataset for different dimensions	Nowadays, the approach has not been tested in constrained optimization problems, which indicates that said performance is not optimal
k-means DE *Heuristic for initialization of cluster centers* [62]	The approach was improved by the DE since the method does not require the evaluation of the k-Means algorithm itself to find the fitness value for the chromosomes	On the negative side, the proposal is computationally noncompetitive since it is much slower than the basic k-means

Table 2 (continued)

Technique	Advantages	Limitations
CODE *Chaotic oppositional differential evolution* [63]	The CODE outperforms its competitors through a better trade-off between a fast convergence rate and the diversity preservation in population	The proposal has not demonstrated a feasible scheme for constrained optimization problems, which can be understood as a current limitation

The bold in Table 2 refers to the names of the techniques that are analyzed

fixed parameter settings, as is the case with the original version of DE.

Each individual in the population is an n-dimensional vector representing a candidate solution, so the idea behind the original mutation strategy is to take the difference vector between two individuals, scale it by the mutation factor, and then added to a third individual to create a new candidate solution. Figure 4 describes this process in a two-dimensional search space. Two individuals x_{r1} and x_{r2} are randomly chosen with $r1 \neq r2$, and the scaled version of the difference between these two individuals is added to the individual randomly selected $r3 \notin \{r1, 2\}$. This results in a mutant vector $v_{i,G}$, which might be accepted into the population as a new candidate solution depending on its fitness value. At every generation G, this operator creates a mutant vector ($v_{i,G}$) corresponding to each population member, also known as the target vector ($x_{i,G}$). Listed below are some of the mutations that have been proposed in the last two decades.

- DE/rand/1
  In this strategy (see Eq. 5), three vectors are selected from the population, where one will be the base vector and disturbed by the difference between the other two. The indices of these vectors $r1$, $r2$, and $r3$ must be mutually exclusive integers randomly chosen from the range $[1, Np]$ and different from the index i of the target vector. This randomness when selecting the individuals that participate in creating a mutant vector makes the algorithm able to maintain population diversity and global search capability with no bias to any specific search direction. Nevertheless, it might slow down the convergence speed [67].

$$v_{i,G} = x_{r3,G} + F(x_{r1,G} - x_{r2,G}) \tag{5}$$

  Brest. *et al*. employing this mutation operator developed the JDE algorithm [68], which is a method that seeks to adapt the control parameters F and CR automatically and get rid of the trial and error practices when establishing these values.
- DE/rand/2
  Eq. 6 can be considered an extension of DE/rand/1 since a new pair of vectors ($r4$ and $r5$) is added, which might lead to a better perturbation than the strategies with only one difference vector. Some researchers [69] have stated that two difference vector strategies are better than DE/rand/1 due to their ability to improve diversity by producing more trial vectors while increasing their exploration ability of the search space.

$$v_{i,G} = x_{r5,G} + F(x_{r1,G} - x_{r2,G}) + F(x_{r3,G} - x_{r4,G}) \tag{6}$$

  The self-adaptive DE algorithm with improved mutation strategy (IMSaDE) [70] takes this operator along with an

elite archive strategy to generate mutant vectors. In this way, they seek to obtain the good exploration capacity of this operator and, at the same time, a better convergence rate through the inclusion of individuals with good fitness.

- DE/best/1
  The difference between this strategy (Eq. 7) and DE/rand/1 is that instead of randomly selecting the base vector, it utilizes the information of the individual with the best fitness found so far ($x_{best,G}$), attracting all the individuals toward the same best position. In fact, according to Mezura et al. [71] this scheme is one of the most competitive for unimodal, separable, and non-separable problems, but due to its exploitative tendency, may cause premature convergence, diversity loss, and more chances of falling into a local optimum when dealing with multimodal problems [72, 73].

$$v_{i,G} = x_{best,G} + F(x_{r1,G} - x_{r2,G}) \tag{7}$$

  It is difficult to find an algorithm that employs only this mutation strategy since, in most cases, it is used to compensate for the weak exploitation of other strategies like in the original DE algorithm. For example, the Modified Mutation Strategy DE (MMDE) algorithm [74] uses this operator in conjunction with DE/rand/1, where a random number and a threshold determine which one is used.
- DE/best/2
  In this strategy, similarly to DE/rand/2, an additional pair of vectors is included, which adds an extra component of random variation in each mutation, also seen as an enhancement of the exploration capability [75]. The strategy is defined as follows:

$$v_{i,G} = x_{best,G} + F(x_{r1,G} - x_{r2,G}) + F(x_{r3,G} - x_{r4,G}) \tag{8}$$

  Like in the previous schemes (and practically in most mutation operators, unless otherwise stated) $r1 \neq r2 \neq r3 \neq r4 \neq i \neq best$. Ho-Huu et al. [76] included this operator in their proposal alongside DE/rand/1, DE/rand/2, and DE/best/1, where at each generation, two of them are selected based on an adaptive scheme based on the absolute deviation between the best and the mean of the objective function in the previous generation.
- DE/current-to-rand/1
  In this mutation operator (Eq. 9), the current vector is the base vector that will be perturbed by the difference between two pairs of individuals. The effect of this method presented in [77] can be seen as a local search because the current vector can be attracted to a surrounding point determined by the difference of four vectors selected randomly.

$$v_{i,G} = x_{i,G} + F(x_{r1,G} - x_{r2,G}) + F(x_{r3,G} - x_{r4,G}) \tag{9}$$

  Its efficacy has been demonstrated when used to address multi-objective optimization problems in [78]. In the Fitness Landscape DE (FLDE) algorithm [79], Z. Tan et al. used this operator along with DE/rand/1 and DE/current-to-best/1 to train a random forest model with CEC2014 and CEC2015 functions to predict the mutation strategy when solving new problems.
- DE/current-to-best/1
  Eq. 10, also known as DE/target-to-best/1 [2, p. 140] takes into consideration the information of the best solution so far in the population to the generation of a mutant vector, which can be interpreted as an attraction from the current vector to the best vector.

$$v_{i,G} = x_{i,G} + F(x_{best,G} - x_{i,G}) + F(x_{r1,G} - x_{r2,G}) \tag{10}$$

  In an attempt to balance the effects of this mutation operator, S. Das et al. proposed two neighborhood models in the local and global neighborhood-based mutations DE (DEGL) algorithm [80]. The first one is known as the local neighborhood model, where each individual is mutated using the best position of a small zone. The second one, referred to as the global mutation model, employs the best position of the entire population, just like DE/current-to-best/1 does.
- DE/rand-to-best/1
  According to the DE literature, Qin. et al. [72] developed this operator that combines the techniques that rely on incorporating the current best individual and the subtraction of two pairs of vectors. The mutant vector is generated as follows:

$$v_{i,G} = x_{r3,G} + F(x_{best,G} - x_{3,G}) + F(x_{r1,G} - x_{r2,G}) \tag{11}$$

  This operator's effect on the population gradually guides the population toward the best solution. However, the attraction is not uniformly among the population at each iteration as it would with the DE/current-to-best/1 operator since an individual can be selected and perturbed multiple times in the same iteration.
- DE/current-to-pbest/1
  In the self-adaptive differential evolution with fast and reliable convergence performance (JADE) algorithm [81], the authors introduced Eq. 12 as a generalized version of the operator DE/current-to-best/1 to diversify the population and improve the convergence performance of the proposal by randomly choosing a vector from the top-ranked set of individuals of the current population to play the role of the single best solution in DE/current-to-best, thus guiding guide the search process not only toward one single point but to $100p\%$ possibilities. The

recommended size (p) for this set of good individuals is between 5% to 20% of Np.

$$v_{i,G} = x_{i,G} + F(x^{p}_{best,G} - x_{i,G}) + F(x_{r1,G} - x_{r2,G}) \quad (12)$$

However, to improve the diversity in choosing the individuals involved in generating a new solution, the authors proposed a variation that includes the information of an external archive formed by recently explored solutions that have been replaced by their offspring in the selection phase. The operator with the optional archive is shown in Eq. 13

$$v_{i,G} = x_{i,G} + F_i(x^{p}_{best,G} - x_{i,G}) + F_i(x_{r1,G} - \tilde{x}_{r2,G}) \quad (13)$$

where $\tilde{x}_{r2,g}$ is an individual randomly picked from the union of the current population and the archive ($P \cup A$). This mutation strategy is also employed in the Success-History based Adaptive DE (SHADE) algorithm [82] where Tanabe and Fukunaga proposed a technique for parameter adaptation using a historical memory of successful control parameter configurations to guide the selection of future control parameter values.

Furthermore, a similar approach was proposed in [83] where the authors presented two mutation schemes named DE/e-rand/2 and DE/e-best/2. The letter e refers to the fact that the involved individuals in generating a mutant vector come from the "elite" part of the top-ranked population (usually between 30% to 50%).

- DE/current-to-gr_best/1
  Another variant of DE/current-to-best/1, less greedy and more explorative, is proposed in [84]. This mutation strategy (Eq. 14) utilizes $x_{gr\_best,G}$ that refers to the best vector of a dynamic group of q vectors randomly selected from the current population to replace the $x_{best,G}$ of the Eq. 10. This feature helps to prevent premature convergence at local optima since it ensures that the target solutions are not always attracted to the same best position of the entire population.

$$v_{i,G} = x_{i,G} + F(x_{gr\_best,G} - x_{i,G}) + F(x_{r1,G} - x_{r2,G}) \quad (14)$$

  The authors found that a group size of 15% of Np provides good results on most tested benchmarks. It might seem that DE/current-to-pbest/1 and DE/current-to-gr_best/1 are the same, but while the first one takes $x_{best,G}$ from the best group of the entire population, the second one takes it from a group whose members are randomly chosen. Consequently, it has more chances of escaping from local optima, but the convergence rate can be slower.
- DE/rand-to-best &current/1
  In [85], the authors claimed that including information from the current individual and the best solution will improve the DE algorithm's exploration and exploitation capabilities. This combination of information is carried out as shown in the following equation:

$$v_{i,G} = x_{r3,G} + F_\alpha(x_{best,G} - x_{r2,G}) + F_\beta(x_{i,G} - x_{r1,G}) \quad (15)$$

  Moreover, in Eq. 15, it is observed that two different scaling factors intend to control the information's contribution of the best and the current vectors, achieving a similar effect of having two strategies (DE/rand-to-best and DE/rand-to-current) in a single one.
- Triangular mutation
  A. Mohamed [86] introduced this mutation intending to enhance the global exploration and local exploitation abilities and improve the algorithm's convergence rate. With this adjustment, the convex combination vector $\bar{x}_{c,G}$ will be used to replace the random base vector $x_{r3,G}$ of the Eq. 5. The remaining two vectors will be substituted for the best and worst of the three randomly chosen vectors to produce the difference vector. The triangular mutation is defined in Eq. 16.

$$v_{i,G} = \bar{x}_{c,G} + 2F(x_{best,G} - x_{worst,G}) \quad (16)$$

  From Eq. 16 the convex combination vector $\bar{x}_{c,G}$ of the triangle is a weighted sum of the three randomly selected vectors where the best vector has the highest contribution, and is defined as follows:

$$\bar{x}_{c,G} = w_1 \cdot x_{rbest,G} + w_2 \cdot x_{rbetter,G} + w_3 \cdot x_{rworst,G} \quad (17)$$

  where the real weights are given by $w_i = p_i / \sum_{i=1}^{3} p_i$, i = 1, 2, 3 and $p_1 = 1$, $p_2 = rand(0.75, 1)$ and $p_3 = rand(0.5, p_2)$. This mutation process exploits the nearby region of each $\bar{x}_{c,G}$ in the direction of each $(x_{best,G} - x_{worst,G})$ for each mutated vector. It focuses on exploiting some sub-regions of the search space, improving the local search tendency, and accelerating the proposed algorithm's convergence speed.
- Trigonometric mutation
  In this approach [87], instead of taking the best vector, the current one, or a vector randomly chosen as the base vector, it will be the one that corresponds to the central point of the hyper-geometric triangle formed by three randomly chosen vectors. Moreover, the perturbation imposed on this base vector comprises a sum of three weighted vector differentials as shown in Eq. 18.

$$\begin{aligned} v_{i,G} =& (x_{r1,G} + x_{r2,G} + x_{r3,G})/3 + (p_2 - p_1)(x_{r1,G} - x_{r2,G}) + \dots \\ & \dots (p_3 - p_2)(x_{r2,G} - x_{r3,G}) + (p_1 - p_3)(x_{r3,G} - x_{r1,G}) \end{aligned} \quad (18)$$

Where:

$$p_1 = \frac{|f(x_{r1,G})|}{|f(x_{r1,G})|+|f(x_{r2,G})|+|f(x_{r3,G})|}$$
$$p_2 = \frac{|f(x_{r2,G})|}{|f(x_{r1,G})|+|f(x_{r2,G})|+|f(x_{r3,G})|} \quad (19)$$
$$p_3 = \frac{|f(x_{r3,G})|}{|f(x_{r1,G})|+|f(x_{r2,G})|+|f(x_{r3,G})|}$$

These weight values scale the contribution magnitudes of each differential vector to the perturbation applied to the base vector. It is easy to see that the mutant vector is strongly biased to the best of the three individuals, so this mutation can be considered a greedy local search operator.

- Historical population-based mutation
  Meng and Yang [88] developed this strategy (Eq. 20) so that the mutant vector contains information not only from the current population but from a historical population, which reflects the landscape of the target and the knowledge extracted over past generations.

$$v_{i,G} = x_{i,G} + F(x^p_{best,G} - x_{i,G}) + F(x_{r1,G} - \widehat{x}_{r2,G}) \quad (20)$$

  This way, $x^p_{best,G}$ corresponds to a randomly selected individual from the current generation's top $p\%$ of the current population, $\widehat{x}_{r2,G}$ denotes a randomly selected vector from the union of the current population and the historical individuals in the external archive.
- Reflection-based mutation
  The authors of [89] based their strategy on the Nelder-Mead method, a traditional direct algorithm for unconstrained nonlinear optimization problems that construct a polyhedron of $n+1$ vertices for n-dimensional optimization. They found that an approximate optimal solution can be obtained through reflection, expansion, contraction, and shrink operations over this polyhedron known as simplex. First, they randomly select four individuals from the current population, sorting them from best to worst according to their object function values, $f(x_{r1,G}) < f(x_{r2,G}) < f(x_{r3,G}) < f(x_{r4,G})$. Then, the mutant vector is generated according to the Eq. 21

$$v_{i,G} = x_{o,G} + F(x_{r1,G} - x_{r4,G}) \quad (21)$$

  where:

$$x_{o,G} = w_1 \cdot x_{r1,G} + w_2 \cdot x_{r2,G} + w_3 \cdot x_{r3,G} \quad (22)$$

  and the respective weights of $x_{r1,G}$, $x_{r2,G}$, and $x_{r3,G}$ are defined as follows:

$$w_1 = \frac{f(x_{r1,G})}{f(x_{r1,G})+f(x_{r2,G})+f(x_{r3,G})}$$
$$w_2 = \frac{f(x_{r2,G})}{f(x_{r1,G})+f(x_{r2,G})+f(x_{r3,G})} \quad (23)$$
$$w_3 = \frac{f(x_{r3,G})}{f(x_{r1,G})+f(x_{r2,G})+f(x_{r3,G})}$$

  Although this mutation operation can balance exploration and exploitation better than other basic mutation strategies, it is still susceptible to premature convergence when solving complex multimodal optimization problems due to its tendency towards the best individuals. Thus, the authors combined this strategy with DE/rand/1 and DE/current-to-rand/1.
- Hemostasis-based mutation
  Prabha and Yadav [90] took inspiration from the biological phenomenon called Hemostasis, which regulates the flow of blood vessels during injury. Hence, the operator in Eq. 24 aims to maintain the internal environment so it does not get stuck in the local optimum. To do so, this procedure introduces the Hemostasis vectors (x_{HeV1} and x_{HeV2}), defined as one pair of "good" vectors selected randomly from the best half of the sorted population and one pair of bad vectors chosen randomly from the worst half of the population. Then, two trial vectors are generated and later compared with the current vector of their respective population.

$$v_{i,G} = x_{best,G} + F(x_{HeV1,G} - x_{HeV2,G}) \quad (24)$$

- Union differential evolution mutation
  Sharifi-Noghabi et al. [91] presented a mutation strategy that can be seen as a modification of the DE/rand/2 operator, where the mechanism to select the vectors involved in the generation of a mutant vector, takes into consideration the advantages of both design and fitness spaces criteria. The mutant vector is generated as follows:

$$v_{i,G} = x_{FS1,G} + F(x_{FS2,G} - x_{r1,G}) + F(x_{DS,G} - x_{r2,G}) \quad (25)$$

  where $x_{r1,G}$, $x_{r2,G}$ are randomly chosen from the population, $x_{FS1,G}$, $x_{FS2,G}$ are the parent vectors selected by fitness space criterion, to obtain them it is necessary to:

  – Sort the population from best to worst fitness
  – Calculate the selection probability for each individual according to:

$$P_i = \frac{Np - i}{Np} \quad (26)$$

  – Select the two individuals using the roulette wheel.

$x_{DS,G}$ is the vector selected by design space criterion, and to obtain it the next steps are required:

- Construct the distance matrix (DM) based on the Euclidean distance between all the individuals in the population

$$DM = \begin{pmatrix} \|x_1 - x_1\| & \cdots & \|x_1 - x_{NP}\| \\ \vdots & \ddots & \vdots \\ \|x_{NP} - x_1\| & \cdots & \|x_{NP} - x_{NP}\| \end{pmatrix} \tag{27}$$

- Calculate the probability matrix PM based on DM

$$PM = 1 - \frac{DM(i,j)}{\sum_{\forall k} DM(i,k)} \tag{28}$$

- Finally, roulette wheel selection without replacement is carried out on every row of PM matrix (for each member of the population).
  After seeing how x_{FSi} are obtained, this mutation seems more similar to DE/best/2 since vectors with better fitness are more likely to be chosen. However, due to Eq. 28, it can also be considered a local search operator because it exploits the regions around a predefined member by assigning a higher probability to closer individuals.
- Gaussian-based mutation
  Huiwang et al.[92] presented an almost parameter-free algorithm to determine the optimal control parameter in the differential evolution algorithm. The strategy mutation of this proposal (Eq. 29) generates a new vector by a Gaussian distribution, sampling the search space based on the current position.

$$V_{i,G} = N(\mu, \sigma) \tag{29}$$

  where $\mu = (x_{best,G} + x_{i,G})/2$ and $\sigma = \mid x_{best,G} - x_{i,G} \mid$. In the early evolutionary phases, the search process focuses on exploration due to the large deviation (initially, the distance between individuals is large). As the generations increase, the deviation becomes smaller, and the search process will focus on exploitation.
- Cauchy-based mutation
  Although this operator was originally introduced by A. Stacey et al. [93] as an improvement of the Particle Swarm Optimization (PSO) algorithm, M. Ali and M.Pant [94] implemented this mutation (with some variations) as a mechanism to help individuals escape the local basin by allowing them to jump to a new region. Each element of the vector has a probability of 90% of being perturbed by a random number generated from a Cauchy distribution, which is similar to the Gaussian distribution but with more of its probability in the tails, increasing the probability of large values being generated. This operator is shown in Eq. 30

$$u_{j,i.G+1} = \begin{cases} x_{j,\,\text{best}\,,G} + \mathrm{C}(\gamma, 0) & \text{if rand}(0,1) \le 0.9 \\ x_{j,i.G} & \text{otherwise} \end{cases} \tag{30}$$

  where j corresponds to the jth element of the best solution (j=1,2..., Dims) at generation G, $\mathrm{C}(\gamma, 0)$ stands for a random number generated by Cauchy probability distribution with scale parameter and centered at the origin.

Table 3 presents an overview of all the aforementioned mutation strategies, their classification and the type of search they perform.

3.3 Crossover in DE

The basic algorithmic framework of the DE algorithm conforms to four phases: initialization, mutation, crossover, and selection. This section centers on the crossover operator phase. Since its introduction in 1995, the crossover operator has come to enhance the potential diversity of the population, in which the mutant vector and current vector cross their components in a probabilistic form to produce a trial vector (also called offspring). The crossover process allows the current solution to inherit features from the donor or mutant vector. This combination of elements is controlled by a parameter called Crossover Rate (CR), which is set as 0 to 1. This trial vector contends with the respective element of the current population in the current generation to know who is the best, with its respective objective function, and is transferred into the next generation. Two commonly used crossover operators are binomial (uniform) crossover and exponential (modular two-point) crossover [2, 26, 30]. The principal difference between binomial and exponential crossover is that while in the binomial case, the components inherited from the mutant vector are arbitrarily selected, they form one or two compact sub-sequences in the case of exponential crossover. An interesting comparative study is proposed by [95, 96]. Another interesting comparative study is shown by [97]. Other crossover variants are analyzed in this section. In [98] proposed a crossover rate repair technique for adaptive DE algorithms. The parent-Centric Crossover approach was proposed by [99] in which multiple parents recombine to produce the child and showed that the proposed algorithm works better regarding convergence rate and robustness. Epistatic arithmetic Crossover was proposed by [100]. A crossover rule called preferential crossover rule [101] was proposed to reduce the drawbacks of scaling parameters. Self-Adaptive Differential Evolution [102] was proposed with adaptive crossover strategies. [103] introduced an orthogonal crossover scheme applied in different variants of DE. A locality-based crossover scheme for dynamic optimization problems is introduced by [104]. Eigenvector-based crossover is proposed by [105] and demonstrates that this scheme can be applied to any crossover

Table 3 Summary of the DE mutation proposed schemes

Classification	Type	Mutation	Description
Exploration	Random	DE/rand/1	It uses the difference of two solutions to modify a third. They are all randomly chosen and mutually exclusive
		DE/rand/2	It employs the difference of four individuals to modify a fifth. They are all randomly chosen and mutually exclusive
	Local	DE/current-to-rand/1	It uses the difference of two randomly selected vectors to modify the current vector, also known as the target vector
	Probabilistic	Cauchy based mut	Each element of the best solution has a 90% of the probability of being modified by a value from a Cauchy distribution
Exploitation		Gaussian-based mut	To create a mutant vector, half the sum of the best vector and the target vector is used as the mean of the normal distribution. At the same time, the deviation is given by the absolute value of the difference between these two vectors
	Fully guided	DE/best/1	The mutant vector is the result of the perturbation of the best individual with the difference of two individuals randomly selected
		DE/best/2	The mutant vector is the result of the perturbation of the best individual with the difference of four individuals randomly selected
		DE/current-to-best/1	The difference between the best vector and the current vector, in addition to the difference of two individuals randomly selected, are used to perturb the current vector
	Partially guided	DE/current-to-pbest/1	Similar process of the above mutation but instead of using the best vector of the population, it employs one random vector of the top 100p% individuals
		DE/current-to-gr-best/1	The vectors involved in creating a mutant vector are the current vector, two random vectors, and the best vector of a group formed by randomly selected individuals
		DE/rand to best/1	A random vector is modified by the difference of four vectors, where one is the best vector and the others are randomly picked
		DE/rand-to-best &current/1	This strategy is similar to the one above; however, in the second difference, the target vector is included instead of a random vector
		Triangular Mutation	The randomly selected weighted sum of three solutions is modified by the difference between the best and worst solutions
		Trigonometric Mutation	Here, the average of 3 randomly selected vectors is perturbed by the sum of three weighted differences of these vectors
		Hist. pop. based Mut	The vectors involved in the perturbation of the target vector are one of the top p% of the current population, a random one, the target vector itself, and a randomly chosen vector from the union of the current population and an external archive
		Reflection based Mut	The weighted sum of three random solutions is disturbed by the difference of two. Here, the solutions with the best fitness are the ones that influence the result the most
		Hemostasis based Mut	This strategy generates two trial vectors, both perturb the best solution, but the first is through the differential of two vectors selected randomly from the best half of the sorted population. In contrast, the other perturbation comes from the differential of a pair of bad vectors chosen randomly from the worst half of the population
		UDE Mut	Two individuals involved in this strategy are randomly selected, fitness space criteria select another two, and the design space criteria select another one

strategy. Using a linear increasing strategy, [106] modify the crossover rate. [107] proposed a Self-Adaptive Differential Evolution (SaDE) improving the generation of trial vectors and control parameters values are self-adaptive based on previous experiences. This work aims to analyze and compare the impact of crossover operators and their variants on the Differential Evolution algorithm.

- Exponential crossover
  The exponential crossover is proposed in the original work of Storn and Price in [1]. Since its introduction, the crossover operator allows the construction of a new trial element starting from the current and mutant elements. The exponential (Modular two-point) crossover is similar to the 1 to 2-point crossover in GA's. One

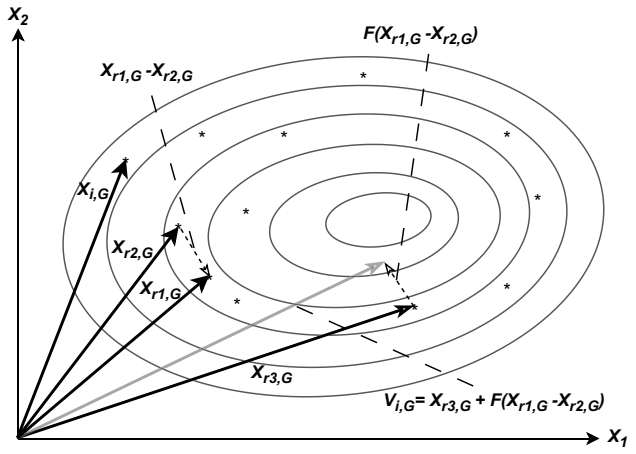

Fig. 4 Graphical representation of Mutation scheme of DE algorithm

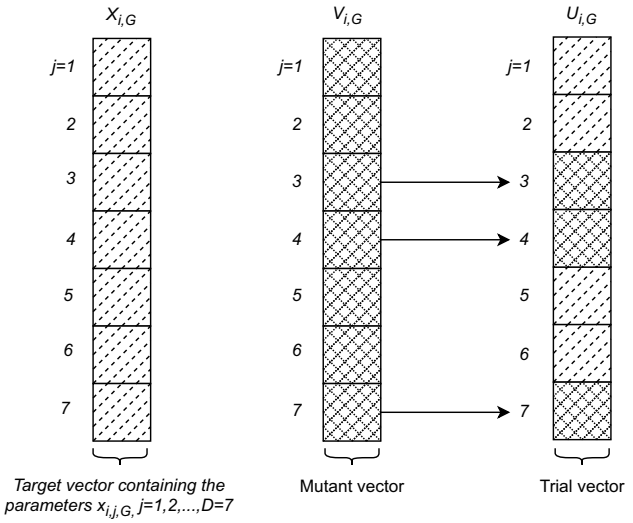

Fig. 5 Crossover process

of the first appearances of the crossover operator is in [1], and the implementation of this is to increase the diversity of the perturbed parameter vector described in the Sect. 3.2. For the exponential crossover, the trial vector is generated as follows:

$$u_{i,G} = u_{j,G} \begin{cases} v^j_{i,G} & \text{for } j = \langle n \rangle_D, \langle n+1 \rangle_D, \ldots, \langle n+L-1 \rangle_D \\ x^j_{i,G} & \text{for all other} j \in [1, D] \end{cases} \tag{31}$$

where the angular brackets $\langle\rangle_D$ denote a module function with modulus D. The integer L is drawn from [1, D]. In Fig. 5 an example of the mechanism of the crossover operator for 7-dimensional vectors is shown.

- Binomial crossover
  DE, like GA, is a simple real-coded evolutionary algorithm and uses the concept of fitness in the same sense as in genetic algorithms. The most significant difference between DE and GA is that in DE, some of the parents are generated through a mutation process before crossover is performed. In contrast, GA usually selects parents from the current population, performs crossover, and then mutates the offspring. In the DE algorithm, the role of the crossover operator is to enhance the diversity of the population. The crossover is executed after generating the donor control vector through mutation. In the vast majority of research on the DE algorithm, it is typically used the binomial crossover (or continuous crossover) that is performed on each of the D variables whenever a random number between 0 to 1 is less or equal to the Cr (the crossover rate value), in this case for the binomial crossover the trial vector is created in the following way:

$$u_{i,G} = u_{j,G} \begin{cases} v^j_{i,G} & \text{if } j = j_{rand} \text{ or } rand(0,1) \leq CR \\ x^j_{i,G} & \text{otherwise} \end{cases} \tag{32}$$

where $rand(0, 1)$ is a uniformly distributed random number, $j_{rand} \in [1, 2, \ldots, D]$.

- Crossover rate repair technique
  In this scheme, the crossover rate in DE is repaired by its corresponding binary strings using the average number of components taken from the mutant vector. The mean value of the binary string is the replacement for the original crossover rate. This proposal analyzes the behavior of the crossover in DE, and it proposes a methodology in which it is considered that the trial vector is directly related to its binary strings but not directly related to the crossover rate [98]. This work mainly focuses on improving the adaptive DE algorithm based on the proposed crossover rate repair technique to improve its performance. The analysis of the proposed method is next: Let b_i be a binary string generated for each target vector x_i as follows:

$$b_{i,j} = \begin{cases} 1, & \text{if } rndreal(0,1) \leq CR \text{ or } j = j_{rand} \\ 0, & \text{otherwise} \end{cases} \tag{33}$$

Consequently, the binomial crossover of DE can be reformulated as:

$$u_{i,j} = b_{i,j} \cdot v_{i,j} + (1 - b_{i,j}) \cdot x_{i,j} \tag{34}$$

where $i = 1, \ldots, NP$ and $j = 1, \ldots, D$. It can see that the binary string b_i is stochastically related to CR; Nevertheless, the trial vector u_i is directly related to its binary string b_i, but not directly related to its crossover rate CR. In addition, this work proposes to update the CR based on the successful parameter, calculating the CR' as:

$$CR'_i = \frac{\sum_{j=1}^{D} b_{i,j}}{D} \tag{35}$$

- Parent-centric crossover
  The proposed modification of the DE algorithm uses parent centric approach [99]. The name of the improvement is called DEPCX. This modification measures the balance between the convergence rate and the algorithm's robustness. Parent-Centric Crossover (PCX) was first introduced by [108, 109]. In [108, 109], PCX is a normal crossover operator, while in in DEPCX, it is treated as a mutant vector and is applied to each candidate solution vector, showing better results regarding convergence rate and algorithm robustness.
- Epistatic arithmetic crossover
  The epistatic arithmetic crossover operator [100], unlike the ordinary arithmetic crossover, considered the impact of epistatic genes in the context of evolutionary computation by expressing the epistatic as a graph product of two linear graphs represented by the candidate solution that is involved in the crossover process. This operator has embedded into a DE variant called eXEDE.
- Preferential crossover
  In [101] is proposed a crossover rule called the preferential crossover rule. The modified DE algorithm is called DEPC. This algorithm uses two population sets called S_1 and S_2. Each set contains N elements (total of the population). S_2 is called the auxiliary population set and is used to record the trial points discarded from the original DE process.
- Crossover strategies adaptation
  A self-adaptive Differential Evolution algorithm with crossover strategies adaptation (CSA-SADE) is proposed in [102]. In this proposal, each individual has its mutation strategy, crossover strategy, and control parameters, which can be adaptively adjusted through the entire search process. In the case of the crossover strategy, this methodology involves binomial and exponential crossover. For a detailed and longer explanation of the implementation of the methodology, please refer to [102].
- Orthogonal crossover
  Orthogonal crossover (OX) operators are based on orthogonal design [110]. These operators can perform a systematic and rotational search in the region the parent solutions define. In [103], a framework is suggested using an OX in DE called OXDE, a combination of *DE*/*rand*/1/*bin* and a version of OX (QOX). In [111] proposes a quantization technique into OX and proposes a version of OX, called QOX, for dealing with numerical optimization. This version of OX is used in this improved version.
- Locality-based crossover
  This article proposes an algorithm called Adaptive Differential Evolution with Locality-based Crossover (ADE-LbX) [104]. The mutation operation is guided by a locality-based scheme that preserves the features of the nearest individuals based on Euclidean distance around a potential solution. One of the most prominent features of this implementation is the use of the L-best crossover technique. L-best crossover uses the concept of blending rate, which determines the rate at which traits interbreed to produce successful offspring, as opposed to conventional crossover techniques that specify a Cr parameter of crossover probability.
- Eigenvector-based crossover
  Eigenvector-based Crossover, known as a rotationally invariant operator, was proposed by [105] to address optimization problems with rotated fitness landscapes more effectively. This consists of a rotated coordinate system that was first constructed by referring to the eigenvector information of the population's covariance matrix. The offspring solutions can then be generated by the randomly selected parents from standard or rotated coordinate systems to prevent the rapid diversity loss of the population. The eigenvectors form a basis of the population and let the crossover operator exchange information between the target and donor vectors with a basis of eigenvectors rather than a natural basis. The eigenvector-based crossover operator is fully explained in [105].
- Hybrid linkage crossover
  The hybrid linkage crossover (HLX) was proposed by [112] to leverage the problem-specific linkage information between pairs of variables for more effective guidance of the search process. An improved differential grouping technique was first incorporated into HLX to adaptively extract groups of tightly interactive variables known as building blocks (BBx). Two group-wise crossover operators, GbinX and GorthX, were then proposed to guide the crossover process without disrupting the tight linkage structures of BBs. The proposed HXL scheme can be easily adapted into the existing DE variants to achieve more promising optimization performance.
- Superior–inferior and superior-superior crossover
  Superior-inferior (SI) Crossover and Superior-Superior (SS) Crossover strategies were proposed by [113] to improve the diversity of the population in the DE algorithm. The SS scheme is triggered to enhance the exploration strength of an algorithm if the population diversity is too low. The SS scheme promotes the exploitation search if the population is diversified. The SS and SI schemes are adaptable to typical binomial and exponential crossover operators and can be incorporated into various DE frameworks. The choice that determines which of the two implementations is used is given by the calculation of the diversity of the population given in [113]. This diversity decides whether the implementation should be used (SI) or (SS).
- Multiple exponential crossover

Multiple exponential crossovers enable the formation of a new solution by combining multiple segments of target and mutant vectors [114]. This semi-consecutive crossover operator showed that it is not only prepared with the strengths of conventional binomial and exponential crossover operators but also demonstrates better capability in handling the subset of tightly interactive variables.

- Rotating crossover operator
  A new DE variant was proposed by [115] and consists of a rotating crossover operator (RCO) with a multi-angle searching strategy, aiming to reduce the likelihood of generating inferior offspring solutions by expanding the search space tactically. Unlike the conventional binomial crossover scheme, the trial vector of RCO can be generated diversely within the circle regions around the donor and target vector by referring to the self-adaptive crossover parameter and rotation control vectors followed by the Levy distribution. This scheme uses a rotation control vector and a binary parameter to generate trial vectors around target vectors or mutant vectors at certain angles and distances. The RCO crossover strategy can generate trial vectors by rotating around donor vectors or target vectors controlled by rotating vectors, whose direction and angles can be guided to flexible search space and diverse offspring.
- Optional blending crossover
  In [116] is proposed a switched parameter DE with success-based mutation and modified BLX crossover (SWDE_Success_mBLX) to solve scalable optimization problems without sacrificing the simplicity of its algorithmic framework. A simple control parameter selection strategy that enables the random and uniform switching of mutational scale factors and crossover rates within their feasible ranges was first proposed. A success-based switching strategy was incorporated to determine the mutation schemes of each solution based on its search performance history. The crossover of each target-donor pair was performed using a binomial crossover or a modified BLX (mBLX) crossover via a probability-based selection scheme. The latter mBLX crossover scheme facilitated the search in the region between and beyond the dimensional bounds established by the target-donor pairs to balance the exploration and exploitation searches. The mBLX strategy follows a similar philosophy of BLX-α-β [117, 118] but avoids the extra fitness evaluation.
- Linear increasing crossover strategy
  Zuo Dexuan and Gao Liqun designed an Efficient Improved Differential Evolution (EIDE). Its method modifies the scalar factor and improves the crossover rate [106]. The IEDE is different from DE in two principal aspects: 1.- For the scale factor (F) it was adopted a random number from a uniform distribution in the range $[0-0.6]$, that is $F \sim U(0-0.6)$. 2.- For crossover rate CR it was adopted a linear increasing strategy as follows:

$$CR = CR_{min} + (CR_{max} + CR_{min}) \times k/K \tag{36}$$

  Where CR_{min} and CR_{max} are the minimal and maximal crossover rates, respectively. k is the current number of iterations, and K is the maximal number of iterations.

In the same way as the last subsections, this crossover subsection ends with a summary of the aforementioned proposed approaches. Table 4 shows said information in a similar structure as the one presented in the initialization subsection, where the respective scheme and its reference, such as its advantages and limitations, are reported in three columns.

3.4 Selection in DE

As mentioned before, the selection stage is usually ignored when a new DE modification is proposed because the selection stage is not as relevant for the DE algorithm process as in a GA. However, over the years, different studies have demonstrated that DE improves significantly by adding an enhanced selection approach [27, 120]. Next, the different selection approaches presented in the literature are explained in detail, such as their respective obtained results and the essential part of their methodologies. It is worth mentioning that DE selection stage modifications related to a multi-objective optimization process will not be considered in this work; this is because said process is often analyzed as a different field in optimization research, which does not compete with this article.

- Roulette wheel selection-elitist-differential evolution
  Ho-Huu et al. [121] presented an elitist selection technique instead of the common greedy selection. Ho-Huu utilized the elitist selection technique developed in [122], combining the children and parent populations. Certain individuals are selected from this new population to create the next generation. This allows the algorithm to accelerate its convergence rate, among other important benefits.
- Landscape-based adaptive operator selection mechanism for differential evolution
  Sallam et al. [123] presented a DE multi-modification approach, including the selection stage. This new algorithm considered the landscape information and the performance histories of the operators' problems. About the selection stage, a landscape-based adaptive operator was proposed; essentially, this technique is

Table 4 Summary of the DE crossover proposed schemes

Technique	Advantages	Limitations
Rcr *Crossover Rate Repair Technique* [98]	The Rcr outperformed the evolutionary and non-evolutionary schemes in terms of quality and convergence speed	The approach did not demonstrate to enhance the EA's in the constrained optimization problems; thus, it is a focus for future studies
DEPCX *Differential Evolution with Parent Centric Crossover* [99]	The DEPCX improves the convergence rate of the DE compared against other related schemes without compromising the quality of the solutions	The proposed approach is computationally non-competitive, and its performance in constrained problems has been shown
eXEDE *epistatic differential crossover based on Cartesian graph product in ensemble DE* [100]	The eXEDE significantly outperformed the rest of the approaches on the CEC2014 tests regarding fitness accuracy and computational costs at 30 and 50 dimensions	Applying the Cartesian products of graphs requires expert knowledge of the topic to manipulate this algorithm
DEPC *Differential Evolution with Preferential Crossover* [101]	Through many computational experiments, the DEPC probed to reduce the number of function evaluations and the computational cost	The DEPC presents optimization troubles when it is applied to high-dimensional problems
CSA-SADE *Self-Adaptive Differential Evolution with Crossover Strategies Adaptation* [102]	The CSA-SADE demonstrated a competitive approach for classical benchmark functions and estimating optimization problems	The algorithm is computationally expensive, and it has not been tested in multi-objective or constrained optimization problems
OXDE *Orthogonal Crossover Differential Evolution* [103]	The OX allows the DE to maximize its search in the region of the parents' solution, accelerating its convergence	The OXDE is limited to non-rotated optimization problems since said process changes the set of the sample points
ADE-LbX *Adaptive Differential Evolution with Locality based Crossover* [104]	The algorithm outperforms the rest of state-of-the-art schemes for DOPs, maintaining the population diversity due to the LbX	The approach is limited to dynamic optimization problems; therefore, trying to adapt it to other OP's is a hard task
EBC *Eigenvector-Based Crossover* [105]	The EBC was tested on multiple DE variants on many benchmark datasets, demonstrating that it significantly improves the performance of any EA, especially on unimodal functions	The EBC structure does not allow it to modify its internal parameters; thus, its performance is linked to the DE variant
HLXDE *Hybrid Linkage Crossover Differential Evolution* [112]	The HLXDE was tested on six DE variants, demonstrating that it improves said EAs in optimization terms	Implementing HLX is a hard task and does not necessarily benefit the algorithm's convergence rate, especially for solving hybrid composite functions
JADE-SI *Adaptive Differential Evolution with Superior-Inferior Crossover* [113]	The SI crossover is highly superior to binomial and exponential crossover. Also, the JADE-SI is competitive against its DE variants	The SI demonstrates to be highly competitive against the classical crossovers, but the SI method to improve the performance of DE variants could be questionable for large-scale and complex problems
CRm *Multiple Exponential Recombination Crossover* [114]	The CRm was applied mainly to SADE and tested on different benchmark problems, demonstrating its good performance against the binomial crossover	The approach has not shown good performance in constrained and real-world optimization problems
RCO *Rotating Crossover Operator* [115]	The RCO was applied to six DE variants, demonstrating to improve the convergence performance of the algorithms over the 50%	The approach's computational cost is high, making the scheme a non-ideal operator for high dimensional problems
WDE_Success_mBLX *Switched Parameter Differential Evolution with Optional Blending Crossover* [116]	The proposed scheme improves the classical BLX in all benchmark functions, enhancing the algorithm	Implementing the WDE_Success_mBLX in any DE variant is a complex task, which can be considered a relative limitation of the scheme
EIDE *Efficient Improved Differential Evolution* [106]	The EIDE probed a better stability and convergence rate performance than its other DE variant competitors	The high dimensional problems are the main challenge of the EIDE since it is non-competitive in said cases

Table 4 (continued)

Technique	Advantages	Limitations
Binomial Crossover *Binomial crossover or (uniform crossover)* [1]	In The binomial crossover tuning, the value of CR between [0, 1] is appropriate to strong convergence	The main disadvantage is that it is not a rotationally invariant process, making it less effective for rotated functions
Exponential Crossover *Exponential Crossover or (Two points modular)* [119]	In the Exponential Crossover, the CR is used to decide how many components will be mutated and allows to mutate just consecutive, in a circular manner	In the Exponential Crossover, there is a small range of CR values (usually [0,9- 1] to which the DE is sensitive. It needs a larger value for the scaling parameter, F, to avoid premature convergence

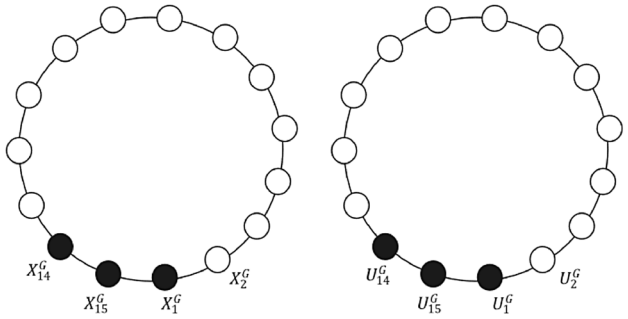

Fig. 6 STS selection process described by Guo et al. [127]

based on the difference between the information landscape vector of the problem and a spherical function defined as a reference landscape [124]. The mentioned technique is defined as follows:

$$f_{ref}(\vec{x}) = \sum_{j=1}^{D} \left(x_j - x_j^b\right)^2 \tag{37}$$

where $\vec{x}^b$ is the best individual in the iteration or sample.

- Improved differential evolution with individual-based selection
  Tian et al. [125] presented an Improved Differential Evolution with Individual-based parameter setting and selection strategy (IDEI) [126]. The author developed a diversity-based selection stage established by a mix of the well-known greedy selection and defining a new weighted fitness value according to the fitness values and positions of the target.
- Subset-to-subset
  Guo et al. [127] presented a modification to the DE to improve the premature convergence. The author developed a novel Subset-To-Subset (STS) selection operator, which divides target and trial populations into subsets to rank them later. An example is given in Fig. 6, representing a population of 15 individuals divided into 3 subsets. In Fig. 6, the left and right figures show the target and trial populations, respectively, while the black circles indicate the first pair of respective subsets.
- Noisy-objective-optimization-problem DE
  Rakshit [128] presented an improved DE algorithm to optimize a single noisy objective problem (NOP). About the selection stage, it is modified to allow competition among individuals in the population; this is developed by adding a probabilistic crowding based niching (local optima) method [129].
- $pbest_{rr}$ Joint approximate diagonalization of eigenmatrices
  Yi et al. [130] developed a DE variant based on the well-known JADE algorithm [81]. In this case, the selection stage was modified by implementing a *pbest* wheel

operator, which works according to the function value of the top vectors chosen in the mutation stage for each individual, Eq. 38 summarizes the model as follows:

$$pbest_i = \begin{cases} pbest_i & \text{if } f(u_{i,G+1}) \leq f(X_{i,G}) \\ elite_j & \text{if } f(u_{i,G+1}) > f(X_{i,G}) and cpro_{j-1} \leq r and < cpro_j \end{cases} \quad (38)$$

where NP and N are the N-dimensional individuals and the iterations, respectively. thus, $i = 1, 2, ..., NP; j = 1, 2, ..., N$. $u_{i,G}$ and $X_{i,G}$ are the fitness value of the target vector.

- New selection operator
  Zeng et al. [131] proposed one of the most recent selection stage modifications in the literature. Zeng proposed that if the DE is in a stage of stagnation, then three candidate vectors will survive and be chosen in the next generation. Nonetheless, these candidate vectors cannot be randomly chosen since it does not guarantee that the population will be improved; that is why discarded trial vectors are considered to replace the parent vector.
- Ensemble differential evolution for a distributed blocking flowshop scheduling problem
  Zhao et al. [132] proposed some modifications to an Ensemble Differential Evolution (EDE) for a Distributed Blocking Flowshop Scheduling Problem (DBFSP), which is an optimization problem related to the operations research field [133]. A biased selection operator replaces the original greedy selection operator in the selection stage. For the optimization problem, the new target individual $X_{i,g+1}$ is defined as follows:

$$X_{i,g+1} = \begin{cases} U & \text{if } \omega < 0 \text{ or } rand < max\{\eta - \omega, 0\} \\ X & Otherwise \end{cases} \quad (39)$$

$$\omega = \frac{f(U) - f(X)}{f(X)} \quad (40)$$

where ω is the error of U according to X, while η is defined as the selection scale factor, and it belongs to the space $0 - 1$.

- DE orthogonal array-based initialization and new selection strategy
  Kumar et al. [134] presented a conservative selection scheme for selecting the solutions from current and trial populations for processing in the next iteration. Basically, a randomly created neighborhood $N_{s,i}$ of size ns is defined for each trial solution. This neighborhood comprises iterative trial solutions under the next three conditions: 1. The created trial solution must be better than the current population solution. 2. Each trial solution must comprise 25% of its neighborhood's solutions. 3. This entire selection process can only be done for the first 60% of the optimization process; the classical greedy selection process defines the selection of the 40% spare process.
- Multi-objective differential evolution with ensemble selection methods
  Qu et al. [135] utilized a Multi-Objective Differential Evolution (MODE) for solving the extended Dynamic Economic Emission Dispatch (DEED) [136]. To solve the said model, the author proposed an ensemble of selection methods in the MODE algorithm [137]. This selection approach combines the newly generated offspring and their parents according to one condition: if a uniformly generated random number between 0 and 1 is ≥ 0.5, then the classical non-domination sorting technique is used to sort the population. Otherwise, the summation-based sorting method will be utilized for the same purpose.

Finally, to summarize all the given information about the modified selection methods for the Differential Evolution, Table 5 is presented. For as in the rest of the operators, Table 5 exposes the information structurally with the results, metrics, advantages, and limitations of each proposed algorithm.

4 Discussion

In this section, the four stages of the DE will be analyzed and discussed to expose the advantages and disadvantages of each method in comparison with the rest of its respective approaches. In the same way, as far as possible, the best and worst variations will be described in terms of computational costs, applied experimentation, improvement rates, and complexity development.

4.1 Initialization

As explained in Sect. 3.1, only the works that had as a main proposal a novel initialization technique were exposed, 19 published works related to the said argument. According to Bilal *et. al* [27], the initialization variants for the DE correspond to the 15% of published works, and the best authors' knowledge, said proportion has not changed for this work. In that sense, these initialization approaches will be discussed in the next paragraphs, trying to define their similarities and differences. Moreover, it will be defined the best and worst proposed initialization schemes according to specific effectiveness metrics.

Based on the above, the efficiency of the methods can be discussed according to the Beyer work *et. al* [139], which defines the efficiency of an EA based on three metrics: **1.**

Table 5 Summary of the selection variation approaches

Technique	Advantages	Limitations
ReDE (*Roulette wheel selection-elitist-differential evolution*) [121]	The ReDE reaches the best result in fewer iterations. Thus it requires less computational time; also, its selection stage allows the algorithm to accelerate its convergence rate	It has not been tested in competition benchmark functions or discrete problems
LSAOS-DE (*Landscape-based adaptive operator selection mechanism for differential evolution*) [123]	The proposed method has the well-known structure of a multi-method/multi-operator, this allows the reader to replicate the algorithm easily	The algorithm is very efficient for solving many optimization problems but at the expense of high computational cost
IDEI (*Improved differential evolution with individual-based selection*) [125]	The introduced IDEI effectively balances the exploration and exploitation phases	It is demonstrated that the CMA-ES [138] algorithm is the only DE variant capable of rivaling the IDEI in most of the experiments
STS (*Subset-to-subset*) [127]	The simplicity of the proposed selection technique allows to apply it easily to any DE variant	Its simplicity is its most important advantage and limitation since its convergence rates are irrelevant
NDE (*Noisy-objective-optimization-problem differential evolution*) [128]	This method demonstrates to be a reliable option for multiple real-application optimization problems	Since this scheme is relatively new, there is still many future works to demonstrate its efficiency in real applications
$pbest_{rr}$**JADE** (*pbestrr Joint approximate diagonalization of eigenmatrices*) [130]	In optimization terms, the $pbest_{rr}$ JADE shows an improvement to the classical JADE algorithm, which makes it a reliable option for future optimization problems	The results demonstrate that the proposed variant is competitive, but it has not quite improved over the rest of the algorithms
NSO (*New selection operator*) [131]	The method is easily adapted to other DE variants and engineering problems, which generates many future works	The utilized parameters and their correct configuration have been demonstrated to be a hard task to tune
EDE-DBFSP (*Ensemble differential evolution for a distributed blocking flowshop scheduling problem*) [132]	The obtained results showed that EDE-DBFSP is better than the rest of algorithms in terms of constructing initial solutions and computational costs	The aforementioned approach was developed to solve a specific engineering problem, which makes it difficult to adapt to other classical optimization problems
OLSHADE-CS (*DE orthogonal array-based initialization and new selection strategy*) [134]	The proposed selection scheme aims to select better trial vectors	This is a unique algorithm that cannot be hybridized easily
MODE-ESM (*Multi-objective differential evolution with ensemble selection method*) [135]	The proposed method demonstrated to improve the rest of algorithms in terms of cost and emission metrics, which makes the scheme a feasible technique for solving multi-objective optimization problems	The algorithm was specifically designed for multi-objective purposes, which makes it difficult to adapt to single optimization problems

Its convergence rate, **2. It can reach the global optima in different OP's**, and **3. Its computational cost**. In that sense, the best operator must allow the algorithm to have the perfect balance of these 3 aspects. When Table 2 is analyzed, it is easy to note that the majority of initialization approaches enhance their respective algorithms in terms of a speeder convergence rate, being the APTS and RCM the only techniques that do not reach this goal. This behavior has sense since it is well-known that the quality of the initial population directly affects the exploration phase. Therefore, an improvement in said stage benefits the early appearance of the exploitation phase. On the other hand, Table 2 also demonstrates that an improved initial population minimizes the computational resources the algorithm needs. However, the SSDE, ISDEMS, APTDE, CIDE, CBPI, DEc, sTDE-dR, and k-means DE are the schemes that do not benefit the algorithm in the aforementioned metric. Nonetheless, these works correspond to the 42% of the related techniques, probing that the computational cost is improved in the 52% of the published techniques. Finally, the third metric related to the diversity of OP's tested for each scheme is also suggested in Table 2. The algorithms probed with two or more OP's are the minority of the works, though the ISDEMS, Cumu-DE, CIDE, CBPI, sTDE-dR, k-means DE, and CODE are the only approaches with said feature. However, these last algorithms have demonstrated their capabilities through various experiments.

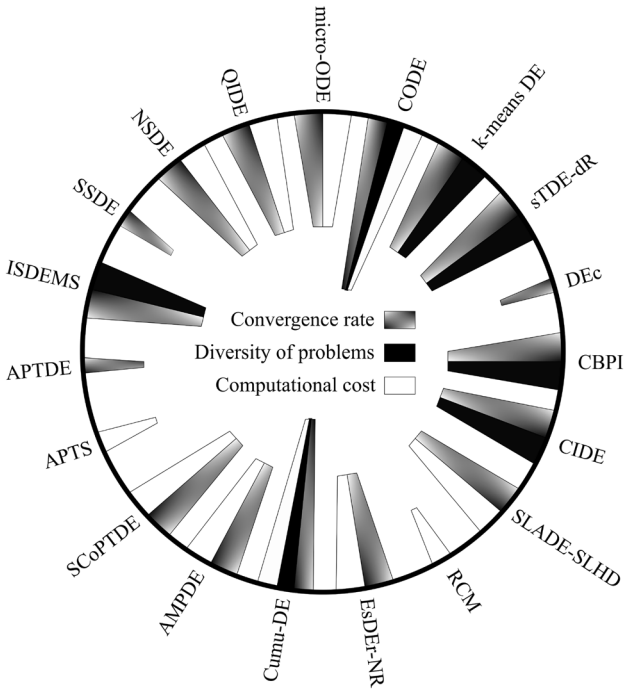

Fig. 7 Relation of the DE initialization variants and the metrics of Beyer

To visualize the reported in Table 2 related to the Beyer metrics, Fig. 7 is presented. In said Figure, the three Beyer's metrics are exposed as bars of different textures for better comprehension. Also, it can be noted that each algorithm is represented by bars of different sizes. More specifically, only one reached metric is represented by its respective bar with the smaller size; if the algorithm reaches two metrics, then it is represented with its two respective bars in a medium size; finally, if the scheme reaches the three metrics, then it is exposed with the three bars with the longest size. As can be seen, only two algorithms reached the three Beyer's metrics, the Cumu-DE and CODE, while the SSDE, APTDE, APTS, RCM, and DEc obtained only one metric. Based on Fig. 7, it can be deduced that Cumu-DE and CODE are generally the best-proposed initialization approaches for the DE in the state-of-the-art. Thus, such algorithms are the most promising schemes for future research without demeriting the rest of the techniques.

4.2 Mutation

One of the main reasons for the good performance of any optimization algorithm is the ability to balance two opposing aspects, global exploration, and local exploitation, or, as seen from another perspective, find the appropriate trade-off between population diversity and a high convergence rate. To achieve this, during the past two decades, many researchers have designed different strategies to enhance the selection of the vectors involved in generating a mutant vector. Some consider the individual's fitness or ranking in the population, whereas others define the selection criterion in design space. One of the first deep and serious studies of an intelligent selection was realized by Kaelo and Ali [140], who used the tournament selection method to obtain the base vector from three randomly chosen vectors.

Even though many of the mutation operators recently proposed have been designed to balance the aforementioned aspects by making an intelligent selection, they are usually accompanied by other complementary modifications and techniques such as parameter adaptation mechanism, the use of an external archive or a different topology of the algorithm [30, 141]. Therefore, the performance of a mutation strategy may vary depending on the approach. One example of this was presented by Bujok and Tvrdík [142], where they found that including the DE/current-to-pbest/1 mutation in their proposal did not bring sufficient enhancement of the performance on the CEC2013 benchmark. However, in the LSHADE algorithm [143], this same strategy outperformed the results in the CEC2014 problems of the JADE algorithm where it was originally introduced.

The mutation strategies described above can be classified into two main categories: random and guided. In the

first group are all the operators, where the individuals of the population do not have a defined search pattern and move freely in the search space. The best representatives are DE/rand/*i* and DE/current-to-rand/*i*. Nonetheless, they should be utilized depending on the needs of the search process since DE/rand/*i* perturbs any individual in any direction, which is beneficial at early generations to discover regions that may not have been reached at initialization or to help to escape from local optima in multi-modal problems.

With DE/rand/*i*, some individuals may be disturbed a greater number of times than others in the same population. Whereas, with DE/current-to-rand/1, the contribution of each particle in the search process is more equitable since each individual is perturbed once every generation, giving each individual the same opportunity to explore their neighborhood.

The second category contains all those operators that influence the search process toward certain positions, usually to the best one of the population or the best of a sub-group. This guided search prevents the algorithm from wandering in the search space, wasting computational resources and time, since in some applications, one function access could represent more than a simple evaluation of a variable in an equation. Nonetheless, one issue that should be highlighted is that the quality of the guide vectors is a determining factor in the performance of these strategies because, in multimodal problems, individuals that may seem like promising solutions at the beginning could attract other individuals of the population to regions that are not close to the best objective solution the problem causing the algorithm to converge prematurely and even get stuck in a local optimum. In this category, we can find examples like DE/best/*i*, DE/current-to-best/*i*, DE/rand-to-best/*i*, Historical population-based mutation, and Hemostasis-based mutation.

All these are easy to classify because the vector X_{best} or X_{pbest} is present in their respective equation, but what happens with strategies where there is no evident best vector, and even all of them are randomly selected as is the case of the Reflection based mutation and the Trigonometric mutation? We might consider them as random strategies, but since the information's contribution of each vector is determined by its fitness, the resultant mutant vector has a certain tendency towards the best of the selected ones. Therefore, it becomes a partially guided mutation.

Before including any mutation strategy in a new differential evolution-based algorithm, it is recommended that the needs of the proposal be analyzed to identify which one is better suited. A common approach in the literature, considered a research hotspot [83], is the combination of several mutation operators to take advantage of the capabilities of each one. The combination of strategies can be done in different ways; here are two common:

The first one employs a candidate pool of mutation schemes in which the operators are used depending on some selection criteria. For example, in the SaDE algorithm [144], a mutation is picked from the candidate pool according to the probability learned from its success rate in generating improved solutions in previous generations. The selected strategy is subsequently applied to the corresponding base vector to generate a mutant vector. More specifically, at each generation, the probabilities of choosing each strategy in the candidate pool are summed to 1. Another example is found in the EPSEDE algorithm [145], where, at the initialization phase, each member of the initial population is randomly assigned to a mutation strategy and associated with parameter values taken from the respective pools. If the trial vector produced is better than the base vector, the mutation strategy and parameter values are maintained for the next generation. Otherwise, the individual is randomly reinitialized with a new mutation strategy and associated parameter values from the respective pools.

The second common method found in the literature is dividing the population into subgroups to evolve, each with different mutation operators. Chatterjee and Zhou [146] generate three subpopulations based on the fitness values of the individuals, where the operator DE/best/1 is applied to the subpopulation with the higher fitness values, DE/rand/1 to the lower subgroup and DE/current-to-best/1 to the subset with average fitness value. Likewise, Zhan et al. [147] employ three subpopulations (each one with a different mutation strategy) called the exploration population, exploitation population, and balance population, which co-evolve by using the master–slave multi-population framework.

4.3 Crossover

This section will discuss the crossover operator stage and the various variants. As can be seen, the Crossover Operator has undergone many variations and adaptations since the Genetic Algorithm and Differential Evolution algorithms were introduced. Due to the large number of proposals that have arisen, several kinds of research have been done to discuss the differences and make a detailed analysis of the crossover operator [26, 95–97]. One of the most significant comparative works on binomial and exponential crossover is the one presented by [96]. Then in [97], it is possible to demonstrate a good performance of the exponential crossover in high-dimensional problems. Outside of these works, we can mention the different versions that have emerged of the crossover operator. It is not necessary to start from a specific order. Still, it can be observed that in the crossover rate repair technique, we see that this technique is applied directly to the adaptive DE (JADE), emphasizing in an analysis where they noticed that three algorithms update their parameters based on their previous successful experience

SaDE, JADE, and MDE_pBX. Still, only the implementation in JADE is used; it would be worth investigating more what would happen if this improvement were implemented in the other algorithms of this style using the proposed methodology. In the case of EIDE, applying the linear increasing crossover strategy has only been tested with two different versions of the DE that handle adaptive parameters. With the original DE, it would be interesting to check the scope of this crossover when applied in a way that seeks to improve the existing DE. Its application and modifying the scaling factor showed competitive results in this comparison. However, in this case, it does not analyze why only these two crossovers are used to select which one to choose, nor does it investigate what would happen if only the mBLX crossover were left alone. The RCO crossover strategy is one of the most complex to implement in the JADE algorithm. Unfortunately, implementing this strategy in any other algorithm would require an extensive study of the methodology, which would complicate adapting this method to other DE variants. In the implementation of Superior Inferior Crossover, it can be noticed that the diversity of the population is used as a selection method for the implementation of SI or SS. Interestingly, this methodology is rarely used and would be worth considering for future implementations in different DE schemes. In the case of the DEPCX method, the PCX implementation is treated as a mutant vector and applied to each candidate solution. This is interesting because a crossover is manipulated and used as a mutant vector generator. However, even though the implementation starts from a crossover version of PCX, it is not applied to the crossover in the final proposal. The philosophy of epistasis is used, which in genetics refers to the effect of one gene on the expression of another gene. This means that the so-called epistatic gene influences the characteristics of the so-called hypostatic genes. In this work, the effect of epistatic genes is studied, where each epistatic gene of the offspring depends on the corresponding hypostatic genes of its parents using an epistatic arithmetic crossover. Although in this proposal, the eXEDE algorithm has an implementation of a version of a crossover (epistatic arithmetic crossover), it is noted that the implementation is applied to the mutation and not in any way, specifically in the crossover operator. The purpose of this comparison is not to prove which of all the crossover proposals is better but to make a compilation of all the proposals that have been made based on the crossover. Another purpose of this work is to provide relevant information from previous research based on each of the stages of the DE algorithm, as well as mention the different research on how to parameterize these stages. According to [26], the research suggests that for the parameter *CR*, most researchers use the value of 0.5 to implement DE variants. However, one of the most significant tables (Table 9 of the article) is the one in [26], summarizing all the parameters used (including the *CR* for the crossover) in all the DE versions analyzed. At this point, it is important to mention that the crossover phase determines how much information is exchanged between the mutant vector and the target vector to create a trial solution. The choice of crossover strategy depends on the desired balance between exploration (finding new solutions) and exploitation (refining existing ones). For example, binomial crossover randomly selects elements from either the mutant vector or the target vector to create the trial solution, and exponential crossover is similar to binomial but with a probability factor that controls the influence of the mutant vector.

4.4 Selection

When Table 5 is analyzed, it is easy to note two important aspects of the selection stage of DE: firstly, the number of proposed selection variations for the DE is quite scarce compared with the rest of the operators. Secondly, unlike the other operators, the selection stage does not allow the establishment of a homogeneous experimental analysis of the scheme itself. These two aspects will be discussed below, trying to explain their causes and define the best and worst exposed DE selection variations.

Bilal et. al [27] in 2020 exposed that the selection variations correspond only to the 2% of the proposed DE operator variations in the state-of-the-art, and to the best authors' knowledge, said statistical proportion has not changed. This phenomenon can be attributed to the fact that the most remarkable contribution of the DE was the mutation scheme [1], while its crossover scheme gained popularity over the years. On the other hand, the DE defines a simple greedy selection method, which corroborates the aforementioned. However, as Table 5 showed, the interest in improving the selection stage has increased in the last decade. When Table 5 is analyzed, it can be noticed that most proposals are tested and designed for general OP's. This is demonstrated with the results of the LSAOS-DE, IDEI, STS, NDE, $pbest_{rr}$ JADE, NSO, and OLSHADE-CS algorithms that were obtained through the evaluation of classical benchmark functions datasets. Besides, only a few proposed algorithms were designed for specific engineering applications, such as the ReDE for truss OP's and EDE for solving DBFS problems. Finally, the most different case is the MODE-ESM, which corresponds to the only DE selection approach designed for a multi-objective OP. This allows us to compare all these proposed algorithms at the same time.

Now, regarding the Beyer metrics explained in Sect. 4.1 [139], when the first metric is studied, it can be noted that ReDE, IDEI, and OLSHADE-CS are the best DE variants in terms of exploration and exploitation, which makes them the DE variants with the best convergence rates. However, ReDE is a very recent algorithm, so it has not been probed in other classical OP's. Also, the OLSHADE-CS has a complex

structure that makes it hard to implement in other OP's. Nonetheless, this algorithm is the best of the three EAs in terms of its convergence rate capabilities. Regarding the second aspect, an algorithm will be considered competitive if its obtained results involve being the best in two or more optimization cases. In that sense, LSAOS-DE, IDEI, STS, NDE, $pbest_{rr}$ JADE, and NSO are the best DE variations. Nevertheless, even though LSAOS-DE and STS perform well in many tasks, they are not computationally competitive. On the other hand, NDE is a relatively new algorithm, and it has not been tested in engineering problems, which makes the experimentation of the NDE a very bounded result. The rest of the aforementioned approaches are competitive since their simplified structure allows them to be tested in many tasks. Finally, the third metric is related to the computational requirements of the method. Therefore, a computationally competitive algorithm can be defined as one with the lowest working time. Under that scheme, ReDE, IDEI, EDE, and OLSHADE-CS are the best algorithms in terms of computational efficiency. As noted, this third aspect is only outstanding by four algorithms, which is the lowest number of competitors compared with the other metrics. This can be attributed to the fact that a proposed new selection method can be defined by a mathematical model but said model is just an algebraic representation of a tabular comparison process, and these types of processes are well-known for being computationally expensive.

To summarize the given information, Fig. 8 presents the relation between each DE variant according to its respective reached efficiency metric. As can be seen, the best DE selection proposal is the IDEI, which is the only competitive DE variant in the three aspects defined by Beyer. It is true that IDEI still needs much more experimentation to test its competitiveness with other OPs. However, compared with the rest of the DE selection variants, the IDEI probed to have a high competitive convergence rate through many benchmark functions from different datasets, and it performed said datasets with very efficient computational costs.

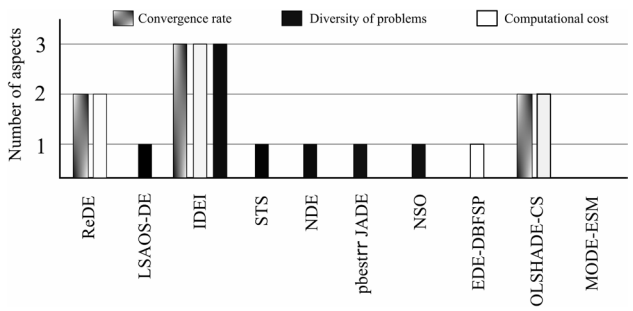

Fig. 8 Relation of the DE selection variants and the metrics of Beyer

4.5 General Discussion

The DE has three main phases: mutation, crossover, and selection. The variants propose new operators for each phase. Each modification directly affects the performance of the DE. To properly select an operator, it is necessary to understand each phase. Mutation strategies control how much variation is introduced into the population. Crossover strategies determine how much information the mutant and target vectors exchange to create a trial solution. Selection strategies determine which individual (target vector vs. trial solution) survives in the next generation. Depending on the complexity of the problem, the designer should determine which phase wants to modify to intensify its benefits. The DE often involves parameters like the scaling factor (F) and crossover probability (CR) influencing the mutation and crossover steps. These parameters can be fine-tuned for your specific problem. Some algorithms even employ adaptive strategies that adjust these values during optimization. Besides, it is important to mention that the scalability of the DE variants can be affected by the dimensionality and complexity of the problems. For example, if the dimensionality increases, it becomes harder to find the prominent solutions where the global optimal is located. Also, the control parameters need to be tuned depending on the problem features. Ultimately, in this situation, the DE can fail in suboptimal solutions.

4.6 Resume of Applied Experimentation

As presented throughout the article, each approach was tested on a different dataset, benefiting the respective technique. Each author decides the best dataset to test their proposal based on his knowledge, optimization focus, or the scientific context of the moment. However, this variety of datasets makes determining the best proposal among the specific DE stage complex. Furthermore, an experimental analysis of these techniques could be counterproductive for a single article, given the resulting extension of the work. Therefore, it has been decided to include Table 6 to summarize the applied dataset to each proposed algorithm. The distribution of Table 6 follows the same distribution of the rest of the article, including the proposals of initialization, mutation, crossover, and selection separately. In addition, the experimentation treated in each method is divided into two parts: the main and complementary datasets. This allows understanding of each dataset's relevance over the years and its use in optimization. Finally, it is worth mentioning that many works use classical benchmark functions, but these functions tend to be organized through different distributions or different numbers of functions. Hence, to simplify the explanation of these datasets, we have decided to name them simply Benchmark functions.

Table 6 Experimentation applied to each variation approach

Stage	Method	Cite	Experimentation		Year
			Main	Complement	
Initialize	Micro-ODE	[39]	hard-to-threshold images		2008
	QIDE	[41]	Benchmark functions		2009
	NSDE	[42]	Benchmark functions		2012
	SSDE	[44]	Benchmark functions		2012
	ISDEMS	[45]	Benchmark functions	Pressure vessel structure	2013
	APTDE	[46]	CEC-2005		2012
	APTS	[47]	CEC-2005		2013
	SCoPTDE	[48]	CEC-2005		2016
	AMPDE	[49]	Bi-objective problems	Three-objective problems	2016
	Cumu-DE	[50]	CEC-2005		2015
	EsDEr-NR	[51]	CEC-2014	CEC-2016	2018
	RCM	[54]	MPB benchmark		2012
	SLADE-SLHD	[56]	CEC-2005		2015
	CIDE	[57]	Benchmark functions		2010
	CBPI	[58]	CEC-2013	Lennard–Jones Potential minimization	2015
	DEc	[59]	Benchmark functions		2016
	sTDE-dR	[61]	CEC-2014		2016
	k-means DE	[62]	BBC dataset	Email dataset	2019
	CODE	[63]	CEC-2014		2022
Mutation	DE/rand/1	[68]	Benchmark functions		2006
	DE/rand/2	[70]	Benchmark functions		2018
	DE/best/1	[74]	IRIS data	Glass data	2020
	DE/best/2	[76]	Bar truss structures		2016
	DE/current-to-rand/1	[79]	CEC-2017		2021
	DE/current-to-best/1	[72]	Benchmark functions	CEC-2005	2009
	DE/rand-to-best/1	[72]	Benchmark functions		2008
	DE/current-to-*p*best/1	[83]	Benchmark functions	AFS problem	2020
	DE/current-to-*gr*best/1	[84]	CEC-2005		2011
	DE/rand-to-best &current/1	[85]	Test problems		2006
	Triangular mutation	[86]	Benchmark functions		2015
	Trigonometric mutation	[87]	Benchmark functions	Training NN	2003
	Hip-DE	[88]	Benchmark functions	CEC-2013	2021
	Reflection-based mutation	[89]	CEC-2013		2018
	DEHeO	[90]	BBOB-2015		2020
	UDE	[91]	CEC-2005		2017
	GBDE	[92]	Benchmark functions		2013
	MDE	[94]	CEC-2008	Benchmark functions	2011
Crossover	Rcr	[98]	CEC-2005		2014
	DEPCX	[99]	Benchmark functions		2008
	eXEDE	[100]	CEC-2014		2016
	DEPC	[101]	Benchmark functions	Schubert problem	2007
	CSA-SADE	[102]	CEC-2005	Reaction Kinetic	2015

Table 6 (continued)

Stage	Method	Cite	Experimentation		Year
			Main	Complement	
	OXDE	[103]	Benchmark functions		2012
	ADE-LbX	[104]	CEC-2009		2013
	EBC	[105]	CEC-2011	BBOB-2012	2015
	HLXDE	[112]	Benchmark functions		2015
	JADE-SI	[113]	Benchmark functions		2015
	CRm	[114]	Benchmark functions		2016
	RCO	[115]	CEC-2013		2017
	WDE_Success_mBLX	[116]	CEC-2013	CEC-2010	2017
	EIDE	[106]	Benchmark functions		2012
	Binomial Crossover	[1]	Benchmark functions		1997
	Exponential Crossover	[119]	Fuzzy rules parameters		1996
Selection	ReDE	[121]	Bar truss structures		2018
	LSAOS-DE	[123]	CEC-2014	CEC-2015	2017
	IDEI	[125]	CEC-2005	CEC-2014	2017
	STS	[127]	CEC-2014		2018
	NDE	[128]	CEC-2013	GECCO-2010	2020
	$pbest_{rr}$ *JADE*	[130]	CEC-2005	Four real-world applications	2016
	NSO	[131]	CEC-2013	CEC-2014	2021
	EDE-DBFSP	[132]	Taillard benchmark 1993		2020
	OLSHADE-CS	[134]	CEC-2017	CEC-2020	2022
	MODE-ESM	[135]	S-ZDT1	R-ZDT4	2019

As noted, the classical benchmark functions lead the majority of initialization variants, followed by the CEC-2005 and the CEC-2014 in recent years. This makes sense since most initialization methods tend to improve the exploration phase of the algorithm, which can only be tested on complex mathematical objectives. Also, it is remarkable that competition datasets like CEC datasets have tended to replace classical benchmark functions over the years. The mutation variants tend to be tested exclusively with benchmark functions, excluding some minimal exceptions. However, it is worth mentioning that the CEC-2005 is usually preferred over the rest of the competition datasets for testing the mutation proposals. The crossover stage has been usually tested with benchmark functions and some specific application datasets. Finally, the selection stage variants have been tested almost exclusively on competition datasets, while no selection approach has been tested on any classical benchmark function. This can be attributed to the fact that the selection stage is one of the main parts for exploitation of the method and to avoid getting trapped into local optima. Therefore, these methodologies must be tested on highly competitive optimization problems.

5 Conclusions

The DE algorithm has become one of the most popular evolutionary strategies and has been used since its creation more than twenty years ago. At the same time, its simple design has triggered an enormous quantity of variants and modifications in its operators. Due to this, several surveys and review works have been published to gather information on the DE variants and said works had broached the subject from different and valuable perspectives. However, no work has been collected on all the DE-modified operators until now. This article grouped the aforementioned publications according to their scheme (initialization, mutation, crossover, and selection). In the same way, each compilation group ends with a table that summarizes and adds remarkable information. Moreover, the gathered data is analyzed in detail in a respective discussion section that allows an understanding of the advantages

and disadvantages of each proposed approach compared against the rest of them. The collected information has been carried out through an exhaustive search process in the state-of-the-art, and it has covered 63 direct research articles and 50 directly related works, which assures the complete summary of the published DE-modified operators until the final writing of this work to the best author's knowledge. Additionally, it is worth mentioning that the descriptions made for each approach, such as the ranking carried out in some of the discussion subsections, are done impartially and without any academic or research interest. On the other hand, as future research lines, the authors suggest focusing the following related works on the improvement and novel proposals of initialization and selection schemes since said operators have demonstrated to improve the DE; nonetheless, their respective publications comprise only the 20% of the modified operator published articles. Also, extended analysis of the best mutation and crossover operators is feasible for future research. Finally, an experimental analysis of each DE stage comparing the respective approaches on a single dataset would benefit the state-of-the-art.

Author Contributions All authors contributed equally to the study conception and design.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability The data used to support the findings are cited within the article. Also, the datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest All the authors declare that there is no Conflict of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed Consent Informed consent was obtained from all individual participants included in the study.

References

1. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341
2. Price K, Storn RM, Lampinen JA (2006) Differential evolution: a practical approach to global optimization. Springer, Berlin
3. Cuevas E, Rodríguez A (2020) Metaheuristic computation with MATLAB®. CRC Press, Boca Raton
4. Yao X (1991) Optimization by genetic annealing. In: Proceedings of the Second Australian Conference on Neural Networks, 94–97
5. Gong W, Cai Z (2013) Differential evolution with ranking-based mutation operators. IEEE Trans Cybern 43(6):2066–2081
6. Das S, Konar A (2009) Automatic image pixel clustering with an improved differential evolution. Appl Soft Comput 9(1):226–236
7. Sarkar S, Das S (2013) Multilevel image thresholding based on 2d histogram and maximum Tsallis entropy- a differential evolution approach. IEEE Trans Image Proc 22(12):4788–4797
8. Md A, Kalam A (2012) A modified differential evolution based solution technique for economic dispatch problems. J Ind Manag Optim. https://doi.org/10.3934/jimo.2012.8.1017
9. Biswas PP, Suganthan PN, Wu G, Amaratunga GAJ (2019) Parameter estimation of solar cells using datasheet information with the application of an adaptive differential evolution algorithm. Renew Energy 132:425–438
10. Kaur M, Gianey HK, Singh D, Sabharwal M (2019) Multiobjective differential evolution based random forest for e-health applications. Modern Phys Lett B 33(5):1950022
11. Hamdi T, Ben Ali J, Di Costanzo V, Fnaiech F, Moreau E, Ginoux J-M (2018) Accurate prediction of continuous blood glucose based on support vector regression and differential evolution algorithm. Biocybern Biomed Eng 38(2):362–372
12. Feoktistov V (2006) Differ Evol. Springer
13. Opara KR, Arabas J (2019) Differential evolution: a survey of theoretical analyses. Swarm Evol Comput 44:546–558
14. Parouha RP, Verma P (2022) A systematic overview of developments in differential evolution and particle swarm optimization with their advanced suggestion. Appl Intel 52:10448–10492
15. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proc ICNN'95 - Int Conf Neural Netw 4:1942–1948
16. Mashwani WK (2014) Enhanced versions of differential evolution: state-of-the-art survey. Int J Comput Sci Math 5(2):107–126
17. Chakraborty S, Saha AK, Ezugwu AE, Agushaka JO, Zitar RA, Abualigah L (2022) Differential evolution and its applications in image processing problems: a comprehensive review. Arch Comput Methods Eng 30:985–1040
18. Qing A (2009) DIFFERENTIAL EVOLUTION - Fundamentals and aplications in electrical engineering. Wiley, Hoboken
19. Dragoi EN, Curteanu S (2016) The use of differential evolution algorithm for solving chemical engineering problems. Rev Chem Eng 32(2):149–180
20. Okagbue HI, Adamu MO, Anake TA (2019) Differential evolution in wireless communications: a review. Int J Online Biomed Eng 15(11):29–52
21. Piotrowski AP (2017) Review of differential evolution population size. Swarm Evol Comput 32:1–24
22. Xin B, Chen J, Zhang J, Fang H, Peng Z-H (2011) Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: a review and taxonomy. IEEE Trans Syst Man Cybern Part C 42(5):744–767
23. Tanabe R, Fukunaga A (2020) Reviewing and benchmarking parameter control methods in differential evolution. IEEE Trans Cybern 50(3):1170–1184
24. Ayaz M, Panwar A, Pant M (2018) A brief review on multiobjective differential evolution. Soft Comput: Theor Appl 1053:1027–1040
25. Sharma P, Kumar S, Bansal JC (2018) A review on scale factor strategies in differential evolution algorithm. Soft Comput Problem Solving 817:925–943
26. Ahmad MF, Isa NAM, Lim WH, Ang KM (2022) Differential evolution: a recent review based on state-of-the-art works. Alexandria Eng J 61(5):3831–3872

27. Millie Pant B, Zaheer H, Garcia-Hernandez L, Ajith A (2020) Differential evolution: a review of more than two decades of research. Eng Appl Artif Intel 90:103479
28. Maučec MS, Brest J (2019) A review of the recent use of differential evolution for large-scale global optimization: An analysis of selected algorithms on the cec 2013 lsgo benchmark suite. Swarm Evol Comput 50:100428
29. Piotrowski AP (2017) Review of differential evolution population size. Swarm Evol Comput 32:1–24
30. Das S, Suganthan PN (2010) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 15(1):4–31
31. Dragoi E-N, Dafinescu V (2016) Parameter control and hybridization techniques in differential evolution: a survey. Artif Intel Rev 45:447–470
32. Singh A, Kumar S (2016) Differential evolution: an overview. In: Proceedings of Fifth International Conference on Soft Computing for Problem Solving: SocProS 2015, Volume 1, pp 209–217. Springer
33. Chiang T-C, Chen C-N, Lin Y-C (2013) Parameter control mechanisms in differential evolution: a tutorial review and taxonomy. In: 2013 IEEE Symposium on Differential Evolution (SDE), pp 1–8. IEEE
34. Suganthan PN (2012) Differential evolution algorithm: recent advances. In: Theory and Practice of Natural Computing: First International Conference, TPNC 2012, Tarragona, Spain, October 2-4, 2012. Proceedings 1, pp 30–46. Springer
35. Neri F, Tirronen V (2010) Recent advances in differential evolution: a survey and experimental analysis. Artif Intel Rev 33:61–106
36. Das S, Suganthan PN (2010) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 15(1):4–31
37. Li Q, Bai Y, Gao W (2021) Improved initialization method for metaheuristic algorithms: a novel search space view. IEEE Xplore 9:121366–121384
38. Agushaka OJ, Ezugwu EA, Abualigah L, Alharbi SK, El-Wahed Khalifa HA (2023) Efficient initialization methods for population-based metaheuristic algorithms: a comparative study. Arch Comput Methods Eng 30:1727–1787
39. Rahnamayan S, Reza Tizhoosh H (2008) Image thresholding using micro opposition-based differential evolution (micro-ode). Congr Evol Comput (CEC-2008) 2008:1409–1416
40. Sezan MI (1990) A peak detection algorithm and its application to histogram-based image data reduction. Comput Vis Graph Image Proc 49(1):36–51
41. Pant M, Ali M, Singh VP (2009) Differential evolution using quadratic interpolation for initializing the population. Int Adv Comput Conf 2009:1–6
42. Ali M, Pant M, Abraham A (2012) Unconventional initialization methods for differential evolution. Appl Math Comput 12(9):1–21
43. Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7(4):308–313
44. De Melo VV, Botazzo Delbem AC (2012) Investigating smart sampling as a population initialization method for differential evolution in continuous problems. Inf Sci 193:36–53
45. Deng W, Yang X, Zou L, Wang M, Liu Y, Li Y (2013) An improved self-adaptive differential evolution algorithm and its application. Chemom Intel Lab Syst 128:66–76
46. Sun Y, Li Y, Liu G, Liu J (2012) A novel differential evolution algorithm with adaptive of population Topolog. Int Conf Inf Comput Appl 7473:531–538
47. Zhu W, Tang Y, Fang J-A, Zhang W (2013) Adaptive population tuning scheme for differential evolution. Inf Sci 223(20):164–191
48. Sun Y (2016) Symbiosis co-evolutionary population topology differential evolution. Int Conf Comput Intel Secur 12:530–533
49. Wang X, Tang L (2016) An adaptive multi-population differential evolution algorithm for continuous multi-objective optimization. Inf Sci 348(20):124–141
50. Aalto J, Lampinen J (2015) A population adaptation mechanism for differential evolution algorithm. Symposium Series on Computational Intelligence 1514–1521
51. Awad NH, Ali MZ, Suganthan PN (2018) Ensemble of parameters in a sinusoidal differential evolution with niching-based population reduction. Swarm Evol Comput 39:141–156
52. Tanabe R, Fukunaga AS (2014) Improving the search performance of shade using linear population size reduction. Congr Evol Comput. https://doi.org/10.1109/CEC.2014.6900380
53. Tanabe R, Fukunaga AS (2015) Tuning differential evolution for cheap, medium, and expensive computational budgets. Congr Evol Comput. https://doi.org/10.1109/CEC.2015.7257133
54. Du Plessis MC, Engelbrecht AP (2012) Using competitive population evaluation in a differential evolution algorithm for dynamic environments. European J Op Res 218(1):7–20
55. Mendes R, Mohais AS (2005) Dynde: a differential evolution for dynamic optimization problems. Congr Evol Comput 3:2808–2815
56. Zhao Z, Yang J, Hu Z, Che H (2015) A differential evolution algorithm with self-adaptive strategy and control parameters based on symmetric Latin hypercube design for unconstrained optimization problems. European J Op Res 250(1):1–16
57. Ozer AB (2010) Cide: chaotically initialized differential evolution. Exp Syst Appl 37:4632–4641
58. Poikolainen I, Neri F, Caraffini F (2015) Cluster-based population initialization for differential evolution frameworks. Inf Sci 297:216–235
59. Bajer D, Martinovic G, Brest J (2016) A population initialization method for evolutionary algorithms based on clustering and cauchy deviates. Exp Syst Appl 60:294–310
60. Dekking FM, Kraaikamp C, Lopuhaä HP, Meester LE (2005) A Modern Introduction to Probability and Statistics Understanding Why and How. Springer, Berlin
61. Ali ZM, Awad HN, Suganthan PN, Reynolds GR (2016) An adaptive multipopulation differential evolution with dynamic population reduction. IEEE Trans Cybern 47(9):2768–2779
62. Mustafi D, Sahoo G (2019) A hybrid approach using genetic algorithm and the differential evolution heuristic for enhanced initialization of the k-means algorithm with applications in text clustering. Soft Comput 23:6361–6378
63. Ahmad MF, Mat Isa NA, Lim WH, Ang KM (2022) Differential evolution with modified initialization scheme using chaotic oppositional based learning strategy. Alexandria Eng J 61:11835–11858
64. Mahdavi S, Rahnamayan S, Deb K (2018) Opposition based learning: a literature review. Swarm Evol Comput 39:1–23
65. Das S, Mullick SS, Suganthan PN (2016) Recent advances in differential evolution-an updated survey. Swarm Evol Comput 27:1–30
66. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
67. Mohamed AW, Suganthan PN (2018) Real-parameter unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation. Soft Comput 22:3215–3235
68. Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 10(6):646–657
69. Gämperle R, Müller SD, Koumoutsakos P (2002) A parameter study for differential evolution. Adv Intel Syst Fuzzy Syst Evol Comput 10(10):293–298

70. Wang S, Li Y, Yang H, Liu H (2018) Self-adaptive differential evolution algorithm with improved mutation strategy. Soft Comput 22:3433–3447
71. Mezura-Montes E, Velázquez-Reyes J, Coello Coello CA (2006) A comparative study of differential evolution variants for global optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp 485–492
72. Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417
73. Wang L, Xu Y, Li L (2011) Parameter identification of chaotic systems by hybrid nelder-mead simplex search and differential evolution algorithm. Exp Syst Appl 38(4):3238–3245
74. Patil S, Jayadharmarajan AR (2020) Clustering with modified mutation strategy in differential evolution. Pertanika J Sci Technol 28(1):141–162
75. Jeyakumar G, Velayutham CNS (2012) Differential evolution and dynamic differential evolution variants-an empirical comparative performance analysis. Int J Comput Appl 34(2):135–144
76. Ho-Huu V, Vo-Duy T, Luu-Van T, Le-Anh L, Nguyen-Thoi T (2016) Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme. Autom Constr 68:81–94
77. Price KV (1999). In: Corne D, Dorigo M, Glover F (eds) An Introduction to Differential Evolution. Mc Graw-Hill, New York, pp 79–108
78. Iorio AW, Li X (2005) Solving rotated multi-objective optimization problems using differential evolution. In: AI 2004: Advances in Artificial Intelligence: 17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia, December 4-6, 2004. Proceedings 17, pp 861–872. Springer
79. Tan Z, Li K, Wang Y (2021) Differential evolution with adaptive mutation strategy based on fitness landscape analysis. Inf Sci 549:142–163
80. Das S, Abraham A, Chakraborty UK, Konar A (2009) Differential evolution using a neighborhood-based mutation operator. IEEE Trans Evol Comput 13(3):526–553
81. Zhang J, Sanderson AC (2009) Jade: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958
82. Tanabe R, Fukunaga A (2013) Success-history based parameter adaptation for differential evolution. In: 2013 IEEE Congress on Evolutionary Computation, pp 71–78. IEEE
83. Li Y, Wang S, Yang B (2020) An improved differential evolution algorithm with dual mutation strategies collaboration. Exp Syst Appl 153:113451
84. Islam SM, Das S, Ghosh S, Roy S, Suganthan PN (2011) An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. IEEE Trans Syst Man Cybern Part B 42(2):482–500
85. Mezura-Montes E, Velázquez-Reyes J, Coello CC (2006) Modified differential evolution for constrained optimization. In: 2006 IEEE International Conference on Evolutionary Computation, pp 25–32. IEEE
86. Mohamed AW (2015) An improved differential evolution algorithm with triangular mutation for global numerical optimization. Comput Ind Eng 85:359–375
87. Fan H-Y, Lampinen J (2003) A trigonometric mutation operation to differential evolution. J Global Optim 27:105–129
88. Meng Z, Yang C (2021) Hip-de: historical population based mutation strategy in differential evolution with parameter adaptive mechanism. Inf Sci 562:44–77
89. Qian W, Chai J, Xu Z, Zhang Z (2018) Differential evolution algorithm with multiple mutation strategies based on roulette wheel selection. Appl Intel 48:3612–3629
90. Prabha S, Yadav R (2020) Differential evolution with biological-based mutation operator. Eng Sci Technol Int J 23(2):253–263
91. Sharifi-Noghabi H, Rajabi Mashhadi H, Shojaee K (2017) A novel mutation operator based on the union of fitness and design spaces information for differential evolution. Soft Comput 21:6555–6562
92. Wang H, Rahnamayan S, Sun H, Omran MG (2013) Gaussian bare-bones differential evolution. IEEE Trans Cybern 43(2):634–647
93. Stacey A, Jancic M, Grundy I (2003) Particle swarm optimization with mutation. In: The 2003 Congress on Evolutionary Computation, 2003. CEC'03., vol. 2, pp 1425–1430. IEEE
94. Ali M, Pant M (2011) Improving the performance of differential evolution algorithm using Cauchy mutation. Soft Comput 15:991–1007
95. Zaharie D (2007) A comparative analysis of crossover variants in differential evolution. In: Proceedings of IMCSIT, vol. 2007, pp 171–181
96. Zaharie D (2009) Influence of crossover on the behavior of differential evolution algorithms. Appl Soft Comput 9(3):1126–1138
97. Zhao S-Z, Suganthan PN (2013) Empirical investigations into the exponential crossover of differential evolutions. Swarm Evol Comput 9:27–36
98. Gong W, Cai Z, Wang Y (2014) Repairing the crossover rate in adaptive differential evolution. Appl Soft Comput 15:149–168
99. Pant M, Ali M, Singh VP (2008) Differential evolution with parent centric crossover. In: 2008 Second UKSIM European Symposium on Computer Modeling and Simulation, pp 141–146. IEEE
100. Fister I, Tepeh A, Fister I Jr (2016) Epistatic arithmetic crossover based on cartesian graph product in ensemble differential evolution. Appl Math Comput 283:181–194
101. Ali M (2007) Differential evolution with preferential crossover. European J Oper Res 181(3):1137–1147
102. Fan Q, Zhang Y (2016) Self-adaptive differential evolution algorithm with crossover strategies adaptation and its application in parameter estimation. Chemom Intel Lab Syst 151:164–171
103. Wang Y, Cai Z, Zhang Q (2012) Enhancing the search ability of differential evolution through orthogonal crossover. Inf Sci 185(1):153–177
104. Mukherjee R, Debchoudhury S, Kundu R, Das S, Suganthan PN (2013) Adaptive differential evolution with locality based crossover for dynamic optimization. In: 2013 IEEE Congress on Evolutionary Computation, pp 63–70. IEEE
105. Guo S-M, Yang C-C (2014) Enhancing differential evolution utilizing eigenvector-based crossover operator. IEEE Trans Evol Comput 19(1):31–49
106. Dexuan Z, Liqun G (2012) An efficient improved differential evolution algorithm. In: Proceedings of the 31st Chinese Control Conference, pp 2385–2390. IEEE
107. Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: 2005 IEEE Congress on Evolutionary Computation, vol. 2, pp 1785–1791. IEEE
108. Deb K, Anand A, Joshi D (2002) A computationally efficient evolutionary algorithm for real-parameter optimization. Evol Comput 10(4):371–395
109. Deb K (2005) A population-based algorithm-generator for real-parameter optimization. Soft Comput 9:236–253
110. Zhang Q, Leung Y-W (1999) An orthogonal genetic algorithm for multimedia multicast routing. IEEE Trans Evol Comput 3(1):53–62
111. Leung Y-W, Wang Y (2001) An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Trans Evol Comput 5(1):41–53
112. Cai Y, Wang J (2015) Differential evolution with hybrid linkage crossover. Inf Sci 320:244–287

113. Xu Y, Fang J-A, Zhu W, Wang X, Zhao L (2015) Differential evolution using a superior-inferior crossover scheme. Comput Optim Appl 61:243–274
114. Qiu X, Tan KC, Xu J-X (2016) Multiple exponential recombination for differential evolution. IEEE Trans Cybern 47(4):995–1006
115. Deng L-B, Wang S, Qiao L-Y, Zhang B-Q (2017) De-rco: rotating crossover operator with multiangle searching strategy for adaptive differential evolution. IEEE Access 6:2970–2983
116. Ghosh A, Das S, Mullick SS, Mallipeddi R, Das AK (2017) A switched parameter differential evolution with optional blending crossover for scalable numerical optimization. Appl Soft Comput 57:329–352
117. Picek S, Jakobovic D, Golub M (2013) On the recombination operator in the real-coded genetic algorithms. In: 2013 IEEE Congress on Evolutionary Computation, pp 3103–3110. IEEE
118. Picek S, Jakobovic D (2014) From fitness landscape to crossover operator choice. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, pp 815–822
119. Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of North American Fuzzy Information Processing, pp 519–523. Ieee
120. Faiz Ahmad M, Mat Isa NA, Hong Lim W, Meng Ang K (2022) Differential evolution: a recent review based on state-of-the-art works. Alexandria Eng J 61:3831–3872
121. Ho-Huu V, Nguyen-Thoi T, Truong-Khac T, Le-Anh L, Vo-Duy T (2018) An improved differential evolution based on roulette wheel selection for shape and size optimization of truss structures with frequency constraints. Neural Comput Appl 29:167–185
122. Padhye N, Bhardawaj P, Deb K (2013) Improving differential evolution through a unified approach. J Global Optim 55:771–799
123. Sallam M, Elsayed KM, Sarker SA, Essam RLD (2017) Landscape-based adaptive operator selection mechanism for differential evolution. Inf Sci 418–419:383–404
124. Malan KM, Engelbrecht AP (2014) Characterising the searchability of continuous optimisation problems for PSO. Swarm Intel 8:275–302
125. Tian M, Gao X, Dai C (2017) Differential evolution with improved individual-based parameter setting and selection strategy. Appl Soft Comput 56:286–297
126. Tang L, Dong Y, Liu J (2015) Differential evolution with an individual-dependent mechanism. IEEE Trans Evol Comput 19(4):560–574
127. Guo J, Li Z, Yang S (2018) Accelerating differential evolution based on a subset-to-subset survivor selection operator. Soft Comput 23:4113–4130
128. Rakshit P (2020) Improved differential evolution for noisy optimization. Swarm Evol Comput 52:100628
129. Li X (2010) Niching without niching parameters: particle swarm optimization using a ring topology. IEEE Trans Evol Comput 14(1):150–169
130. Yi W, Zhou Y, Gao L, Li X, Mou J (2016) An improved adaptive differential evolution algorithm for continuous optimization. Exp Syst Appl 44:1–12
131. Zeng Z, Zhang M, Chen T, Hong Z (2021) A new selection operator for differential evolution algorithm. Knowl-Based Syst 226:107150
132. Zhao F, Zhao L, Wang L, Song H (2020) An ensemble discrete differential evolution for the distributed blocking flowshop scheduling with minimizing makespan criterion. Exp Syst Appl 160:113678
133. Behnamian J, Ghomi SMTF (2016) A survey of multi-factory scheduling. J Intel Manuf 27:231–249
134. Kumar A, Biswas P, P, N Suganthan P, (2022) Differential evolution with orthogonal array-based initialization and a novel selection strategy. Swarm Evol Comput 68:101010
135. Qu BY, Liang JJ, Zhu YS, Suganthan PN (2019) Solving dynamic economic emission dispatch problem considering wind power by multi-objective differential evolution with ensemble of selection method. Nat Comput 18:695–703
136. Xia X, Zhang J, Elaiw A (2011) An application of model predictive control to the dynamic economic dispatch of power generation. Control Eng Pract 19(6):638–648
137. Qu BY, Suganthan PN (2010) Multi-objective evolutionary algorithms based on the summation of normalized objectives and diversified selection. Inf Sci 180(17):3170–3181
138. Hansen N, Andreas O (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195
139. Beyer H-G, Schwefel H-P, Wegener I (2002) How to analyse evolutionary algorithms. Theor Comput Sci 287(1):101–130
140. Kaelo P, Ali M (2006) A numerical study of some modified differential evolution algorithms. European J Oper Res 169(3):1176–1184
141. Lynn N, Ali MZ, Suganthan PN (2018) Population topologies for particle swarm optimization and differential evolution. Swarm and evolutionary computation 39:24–35
142. Bujok P, Tvrdík J (2015) New variants of adaptive differential evolution algorithm with competing strategies. Acta Electrotech Info 15(2):49–56
143. Tanabe R, Fukunaga AS (2014) Improving the search performance of shade using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp 1658–1665. IEEE
144. Qin AK, Huang VL, Suganthan PN (2008) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417
145. Mallipeddi R, Suganthan PN, Pan Q-K, Tasgetiren MF (2011) Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl Soft Comput 11(2):1679–1696
146. Chatterjee I, Zhou M (2017) Differential evolution algorithms under multi-population strategy. In: 2017 26th Wireless and Optical Communication Conference (WOCC), pp 1–7. IEEE
147. Zhan Z-H, Wang Z-J, Jin H, Zhang J (2019) Adaptive distributed differential evolution. IEEE Trans Cybern 50(11):4633–4647