Improved Multi-operator Differential Evolution Algorithm for Solving Unconstrained Problems

Karam M. Sallam *¶, Saber M. Elsayed†, Ripon K. Chakrabortty ‡, Michael J. Ryan§,
*§*Capability Systems Centre, School of Engineering & IT, University of New South Wales, Canberra, Australia,*
†‡*School of Engineering & IT, University of New South Wales, Canberra, Australia,*
¶*The faculty of Computers and Information, Zagazig University, Egypt*
*¶karam_sallam@zu.edu.eg, {†s.elsayed, ‡r.chakrabortty, §m.ryan}@unsw.edu.au

***Abstract*—In recent years, several multi-method and multi-operator- based algorithms have been proposed for solving optimization problems. Generally, their performance is better than other algorithms that based on a single operator and/or algorithm. However, they do not perform consistently well over all the problems tested in the literature. In this paper, we propose an improved optimization algorithm that uses the benefits of multiple differential evolution operators, with more emphasis placed on the best-performing operator. The performance of the proposed algorithm is tested by solving 10 problems with 5, 10, 15 and 20 dimensions taken from CEC2020 competition on single objective bound constrained optimization, with its results outperforming both single operator-based and different state-of-the-art algorithms.**

***Index Terms*—evolutionary algorithms, differential evolution, adaptive operator selection, unconstrained optimization.**

I. Introduction

Many real-world problems involve the process of determining the values of continuous decision variables to optimize one or more objective functions (either maximized or minimized) [1]. They can be categorized based on the types and number of decision variables, the number of their objective functions to be optimized, and the existence of constraints and many other factors [2], [3]. In this paper, our focus is on unconstrained problems.

In the literature, computational intelligence-based approaches, such as evolutionary algorithms (EAs), are successfully used to solve optimization problems and posses essential merits over traditional approaches [4], i.e., capability of self-adaptation, resilience to dynamic changes and do not require specific mathematical attributes to be satisfied [5]. However, their performance depends on the algorithm's design and its parameters and operators. These stochastic factors make it hard to guarantee achieving the optimal solutions.

Among EAs, differential evolution (DE) algorithm, which is considered efficient and simple, has attracted the attention of practitioners and researchers [6]–[10]. DE mainly has three basic search operators (mutation, crossover and selection) and three parameters (amplification factor, crossover rate and population size). However, no single DE operator (or parameter) performs best for all types of test problems [2], [6], [11], [12]. This motivated researchers to introduce frameworks that use the strengths of different operators. Although they commonly enhance the optimization process, they still do not ensure reliable results for a wide range of test problems [1] which reflects the need for better designs.

In light of the above, this paper introduces an improved multi-operator DE (IMODE) algorithm. It starts by dividing the initial population into several sub-populations, each of which is evolved using its own DE variant. The size of each sub-population is dynamically updated based on two indicators, the quality of solutions and the diversity of each sub-population. At the end of every generation, all sub-populations are gathered and then redivided based on the new sub-population sizes. In addition, a linear reduction of the population size is carried out. This framework is tested on 10 benchmark optimization problems with 5, 10, 15 and 20 dimensions taken from the CEC2020 [13]) competition on single-objective optimization. The results demonstrate that this algorithm statistically outperforms existing algorithms. Also, different components of it are analyzed to provide more insights into the proposed framework.

It is worth mentioning that, although IMODE shares the same structure as that of IUMOEAs-II [11], it has the following differences: 1) CMA-ES has not been used in this design, because of its poor performance [1]; 2) one adaptation mechanisms to determine the control parameters values have been used; and 3) randomly use one of two crossover (binomial and exponential).

The rest of this paper is organized as follows: a brief literature review on related research studies is presented in Section II; the proposed approach and details of its components in Section III; the experimental results and analyses in Section IV; and, finally, the conclusion in Section V.

II. Literature Review

A literature review of multi-operator DE-based algorithms is discussed in this section.

A. Multi-operator DE variants

As previously stated, the performance of DE algorithm is significantly depends on its parameters (i.e., population size (NP), crossover rate (Cr) and scaling factor (F)) and mutation strategies, so many researchers have tried to identify the most optimal one [14], [15]. Despite various DE variants existing in the literature, several studies indicated that a specific one may not perform well for all types of problems.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

Moreover, considering the shortcomings of single-operator DE variants (i.e., a single mutation strategy) for solving all types of optimization problems, research on multi-operator DE variants has received a great deal of attention recently [6], [11].

An auto-selection mechanism (ASM) was proposed by Fan et al. [16] to select a suitable DE variant during the evolutionary process for solving combinatorial optimization problems. Sallam et al. [6], proposed an adaptive operator selection (AOS) method, which uses the performance histories of DE mutation operators and information of the function's landscape to automatically choose the most suitable DE strategy from a pool of many. The superiority of this approach was demonstrated through it solving 45 unconstrained optimization problems taken from the CEC2014 and CEC2015 competitions. Also, in a fuzzy rule-based design of an EA proposed by Elsayed et al. [1], a fuzzy rule-based heuristic was applied to place more emphasis on the best-performing algorithm during the evolutionary process. Their framework was proven to be effective through it solving a number of unconstrained benchmark problems/data sets.

Tasgetiren et al. [17] proposed an ensemble of discrete DE algorithms for improving the performance of DE by selecting appropriate parameter sets. Elsayed et al. [18] developed an overall ensemble in a DE framework with allocations of 16 different combinations of mutation and crossover operators, and a constraint-handling mechanism. More recently, Wu et al. [19] proposed a multi-population-based framework for deriving potential DE variants based on their ensemble characteristics. They used a reward sub-population scheme to prioritize the best-performing DE variants and the suitability of their approach was proven after it solved a number of standard optimization problems.

To enhance the performance of DE, Zhang et al. [20] introduced a multi-layer competitive-cooperative (MLCC) framework. To improve the exploration and exploitation capabilities of DEs, an enhanced DE algorithm with three multi-mutation strategies and their associated self-adaptive control parameters was proposed in [21]. In it, to obtain a new solution, only one mutation strategy was automatically selected to generate the initial vector. Another novel mutation mechanism proposed by Yu et al. [22] could concurrently handle both feasible and infeasible solutions to accelerate convergence while maintaining diversity. They used a greedy constraint-handling method to guide infeasible solutions to feasible ones. A triangular mutation strategy was developed by Mohamed [23] to achieve a good balance between the exploration and exploitation capabilities of a DE algorithm. It obtained competitive results in comparison with other algorithms for solving standard benchmark problems. Recently, Chen et al. [24] developed a multi-operator DE algorithm with an interior-point method for optimizing the evolutionary process using efficient searches.

Similarly, Elsayed et al. [25], introduced a united multi-operator EA (UMOEA). They started with dividing initial large population into multiple equal sizes, which were then expected to be evolved by using different UMOEAs. Based on the success rate, the superior multi-operator EA were selected by adaptively differing population sizes. Meanwhile, the under-performed ones were also updated based on an information exchange procedure, which further facilitated effective searching. Because of their success in obtaining competitive results, Elsayed et al. [26] later developed an improved version of their earlier UMOEA, called UMOEAsII. Rather than limiting to single search operators, this work incorporated multiple search operators at each multi-operator EA run. In their improved version, populations were independently evolved by using both multi-operator algorithm and its search operators. For more information regarding multi-operator algorithms, readers are requested to read research articles of Das et al. [27] and Wu et al. [28], and also a recent survey paper by Wu et al. [29].

III. Proposed Algorithm (IMODE)

This section presents the details of the proposed algorithm (IMODE). The main steps are presented in Algorithm 1, where FES and MAX_{FES} are the number of fitness evaluations and maximum number of fitness evaluations, respectively.

IMODE has three DE mutation strategies (DE/ϕbest/1, DE/current-to-ϕbest/1/archive and DE/current-to-ϕbest/1/without archive). IMODE starts with an initialization phase of its parameters after which an initial population of size NP is randomly generated and evaluated. Then each DE search operator evolves the same number of the solutions from the initial population. After that, for every solution, a new corresponding individual is generated using its assigned mutation strategy. At the same time, the diversity for each DE operator and the quality of solutions, using equations 7 and 9, respectively, are calculated for each single operator, based on which the number of solutions evolved by each operator is updated as discussed at Section III-C. Furthermore, to boost IMODE's convergence, a local search is employed during the latter stages of the optimization process, as described in Subsection III-D, and the main steps in the algorithm are performed until a stopping condition is reached.

The following sub-sections discuss the components of IMODE in detail.

A. Population initialization and updating method

The initial population is randomly generated as

$$x_{i,j} = x_j^{min} + (x_j^{max} - x_j^{min}) \times rand(1, NP) \quad i \in NP \text{ and } j = 1,\ 2,\ ...,\ D \tag{1}$$

where $rand$ is a function used to produce random numbers in the range $[0, 1]$.

Also, a mechanism is used to linearly reduce population size dynamically NP during the search process [30] as

$$NP_{G+1} = round[(\frac{NP^{min} - NP^{init}}{MAX_{FES}}) \times FES + NP^{init}] \tag{2}$$

where NP^{min} is the minimum number of individuals the algorithm can use, FES the current number of function evaluations (FES), MAX_{FES} the largest number of FES.

Algorithm 1 IMODE algorithm

Define nop, $Prob_{ls} \leftarrow 0.1$, MAX_{FES}, $prob_1 \leftarrow 1$, $prob_2 \leftarrow 1$, NP, $G \leftarrow 1$ and $FES \leftarrow 0$;
Generate an initial random population (X) of size NP;
Evaluate $f(X)$, and update number of fitness evaluations $FES \leftarrow FES + NP$;
Each operator *op* is assigned the same number of solutions NP_{op};
while $FES \leq MAX_{FES}$ **do**
  $G \leftarrow G + 1$;
  Generate new population using the assigned DE operators, i.e., each operator *op* evolves its assigned number of individuals NP_{op};
  Calculate the the diversity obtained from each operator *op* (D_{op}) and the quality rate of solutions QR_{op} as described in Section III-C;
  Update the number of solutions (NP_{op}) each DE operator evolves by using Equation 11;
  Generate new population using the assigned DE operators;
  Evaluate $f(X)$, and update number of fitness evaluations $FES \leftarrow FES + NP$;
  Update NP 2;
  if $FES \geq 0.85 \times MAX_{FES}$ **then**
    Apply local search as in Section III-D;
    Update FES
  end if
end while

B. DE mutation strategies

In our algorithm, the following three DE mutation operators are used to evolve the entire population as they perform well in solving unconstrained optimization problems [6].

- DE/current-to-ϕbest with archive/1

$$v_{i,j} = x_{i,j} + F_i \times (x_{\phi,j} - x_{i,j} + x_{r_1,j} - x_{r_2,j}) \quad (3)$$

- DE/current-to-ϕbest without archive/1

$$v_{i,j} = x_{i,j} + F_i \times (x_{\phi,j} - x_{i,j} + x_{r_1,j} - x_{r_3,j}) \quad (4)$$

- DE weighted-rand-to-ϕbest [26]

$$v_{i,j} = F \times x_{r_1,j} + (x_{\phi,j} - x_{r_3,j}) \quad (5)$$

where $r_1 \neq r_2 \neq r_3 \neq i$ are random integer numbers, $\overrightarrow{x}_{r1}$ and $\overrightarrow{x}_{r3}$ randomly selected from the whole population, with $\overrightarrow{x}_{\phi,j}$ chosen from the best 10% of solutions in the whole populations and $\overrightarrow{x}_{r2}$ from the union of the whole population and archive. In the proposed algorithm, an archive is applied to preserve the population diversity, with new solutions worse than their offspring solutions inserted into the archive [31]. To make a room for newly generated solutions, if the size of the archive is larger than its predefined size, the worst individuals are removed from it.

After, mutation operation, a binomial or exponential crossover are randomly carried out to generate the new solutions, as

$$u_{i,j} = \begin{cases} \begin{cases} v_{i,j} & \text{if } (rand \leq Cr_i \text{ or } j = j_{rand}) \\ x_{i,j} & \text{otherwise} \end{cases} \\ \text{if } rnd \leq 0.3 \\ \begin{cases} v_{i,j} & \text{for } j = \langle l \rangle_D, \langle l+1 \rangle_D, ..., \langle l+L-1 \rangle_D \\ x_{i,j} & \text{for all other } j \in [1, D] \end{cases} \\ \text{otherwise} \end{cases} \quad (6)$$

Note that the proposed framework is flexible to adopt more operators.

C. Updating number of individuals evolved by operator op (NP_{op})

As previously mentioned, the sub-population's diversity and quality of solutions are used to update the number of individuals every DE search operator evolves.

The diversity obtained from each DE search operator is the mean deviation of each solution obtained from *op* from the best solution, i.e.,

$$D_{op} = \frac{1}{NP_{op}} \left(\sum_{i=1}^{NP_{op}} dis(\overrightarrow{x}_{op,i} - \overrightarrow{x}_{op}^{best}) \right), \forall op = 1, 2, 3 \quad (7)$$

where $dis(\overrightarrow{x}_{op,i} - \overrightarrow{x}_{op}^{best})$ is the Euclidean distance between the i^{th} solution and best one obtained by each operator *op*. Also the diversity rate is calculated by

$$DR_{op} = \frac{D_{op}}{\sum_{op=1}^{3} D_{op}}, \forall op = 1, 2, 3 \quad (8)$$

Similarly, for the solutions' quality, the best individual at the end of each generation in each sub-population is used, based on which the rate of quality is computed as

$$QR_{op} = \frac{fit_{G,op}^{best}}{\sum_{op=1}^{3} fit_{G,op}^{best}}, \forall op = 1, 2, 3 \quad (9)$$

where $fit_{G,op}^{best}$ is the best objective function value obtained by operator *op* at the end of the cycle.

Based on the above equations, the improvement rate value (IRV_{op}) is calculated as

$$IRV_{op} = (1 - QR_{op}) + DR_{op}, \forall op = 1, 2, 3 \quad (10)$$

Note: to satisfy the aim of maximizing the IRV_{op}, we subtracted QR_{op} from one.

Finally, the number of solutions that each DE operator (NP_{op}) evolves is calculated by

$$NP_{op} = max \left(0.1, min \left(0.9, \frac{IRV_{op}}{\sum_{op=1}^{2} IRV_{op}} \right) \right) \times NP, \quad \forall op = 1, 2, 3 \quad (11)$$

Note: the summation of NP_{op} must equal the whole population size. As a kind of information sharing, the individuals every operator op evolves, is randomly assigned at every generation [3].

D. Local Search

To accelerate the convergence of the proposed IMODE, sequential quadratic programming (SQP) is employed to the best solution in each generation during last 15% of the optimization process with a probability of $P_{ls} = 0.1$ and for up to CFE_{ls} objective function evaluations. The probability of applying this local search is dynamically updated, as described steps 5-9 in Algorithm 2.

Algorithm 2 Local search (SQP)

1: **Input:** the best solution from the whole population $\overrightarrow{x}_{best}$;
2: Generate random number $rand \in [0, 1]$;
3: **if** $rand \leq P_{ls}$ **then**
4:    Apply SQP to $\overrightarrow{x}_{best}$ for CFE_{ls} fitness evaluations;
5:    **if** $f(\overrightarrow{x}_{sqp}) < f(\overrightarrow{x}_{best})$ **then**
6:       Update $P_{ls} \leftarrow 0.1$;
7:       $\overrightarrow{x}_{best} \leftarrow \overrightarrow{x}_{sqp}$ and $f(\overrightarrow{x}_{best}) \leftarrow f(\overrightarrow{x}_{sqp})$;
8:    **else**
9:       Update $P_{ls} \leftarrow 0.0001$;
10:    **end if**
11:    Update number of fitness evaluations ($FES \leftarrow FES + CFE_{ls}$);
12: **end if**

IV. Experimental Results and analysis

To judge the performance of the proposed IMODE algorithm, several experiments were conducted on 10 optimization problems with 5, 10, 15 and 20 dimensions and a search space of $[-100, 100]^D$. The results obtained from IMODE were compared with those obtained from (1) EBOwithCMAR: Effective Butterfly Optimizer with Covariance Matrix Adapted Retreat Phase [32]; (2) HSES: Hybrid Sampling Evolution Strategy [33]; (3) LSHADE-cnEpSin: LSHADE with an Ensemble Sinsoidal Parameter Adaptation [34]; and (4) LSHADE-SPACMA: LSHADE with Semi-Parameter Adaptation and Covariance Matrix Adaptation [35].

The proposed IMODE was coded in MATLAB R2018b and run on a PC with 3.4 GHz Core I7 processor, 16 GB RAM and Windows 10. For all the other comparative algorithms, the values of their parameters were obtained from relevant articles and, to ensure fair comparisons, they all used the same seed. According to the benchmark rules, all the algorithms were run 30 times for 50,000, 1,000,000, 3,000,000 and 10,000,000 fitness function evaluations for problems with 5D, 10D, 15D and 20D, respectively, or $|f(\overrightarrow{x}_{best} - f(\overrightarrow{x^*})| \leq 1e-08$, where x^* is the global optimal solution and x_{best} the best solution obtained by the proposed algorithm, with the average and standard deviation results recorded. For every run, if the distance from the optimal solution was less than or equal to $1E-08$, it was set as zero.

For statistical comparisons of the algorithms, we conducted two non-parametric tests (the Wilcoxon signed-rank and Friedman ranking tests [36]). The proposed algorithm's performances were also graphically judged by plotting their performance profiles [37] which is a tool used to compare the performance of several methods (M) using several test functions (P) and a comparison goal (i.e., the average computational time and number of FES) to attain certain level of the performance criteria (i.e., optimal fitness function value). For a method (m), the performance profile ρ_m is calculated as

$$\rho_m(\tau) = \frac{1}{n_p} \times |p \in P : r_{p,m} \leq \tau| \tag{12}$$

where $\rho_m(\tau)$ is the percentage of $m \in M$ that the performance ratio $r_{p,m}$ is within a factor $\tau \in R$ for the best possible probability and ρ_m a function that returns the cumulative distribution for the $r_{p,m}$.

A. Parameter Setting and Analysis

In terms of the algorithms' parameters: for MODE NP^{init} was set to a value of $6*D^2$ solutions, NP^{min} to 4, the archive rate (A) to 2.6 and memory size (H) to $20 * D$; and, for the local search, FES_{LS} was set to $0.85 \times FES_{max}$. The values of F and Cr are managed by the same technique proposed in [30].

B. Detailed results obtained from IMODE

The detailed results obtained from the proposed IMODE and the rival algorithms are presented at this section.

1) 5D results: the best and average fitness errors $(|f(\overrightarrow{x}_{best}) - f(\overrightarrow{x^*}))$ and standard deviations obtained from the proposed IMODE and the rival algorithms are provided in Table I.

Based on the obtained results, IMODE performed very well for the uni-modal problems (F01), obtaining the optimal best and mean values. For the multi-modal ones (F02-F04), it achieved optimal solutions for F02, F04 and very close to optimal ones for F03. For the hybrid functions (F05-F07), the proposed algorithm was able to achieve the optimal value for all of them for both best and average fitness errors. Finally, for the composition function (F08-F10), IMODE obtained the optimal fitness function value for F08 and F09 and optimal for most of the runs for F10.

2) 10D results: The detailed computational results obtained from the proposed IMODE for 10D test problems are presented at Table II. From these results, IMODE provided the optimal solution for the unimodal function (F01) for both best and average fitness error. Considering the multi-modal functions, it was able to obtain the optimal for F04 and near optimal for both F02 and F03. For the hybrid functions, the proposed algorithm was able to achieve very close results to the optimal solution. Regarding the composition function, IMODE was able to obtain the optimal solution for F08 and F09 and a good result for F10.

3) 15D results: Table III presents the detailed results obtained from the proposed IMODE and the rival algorithms for test problems with 15 decision variables. From this table, IMODE obtained the optimal solution for the unimodal function. For the multi-modal test problems, the proposed algorithm was able to reach the optimal solution for F04, near optimal for F02 and far from optimal for F03. In case of hybrid functions, IMODE obtained close results to the

TABLE I: Results for 5D

Algorithms	IMODE			EBOwithCMAR			HSES			LSHADE-cnEpSin			LSHADE-SPACMA		
Problem	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**
F01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F02	**0.000E+00**	**8.332E-02**	**8.885E-02**	**0.000E+00**	1.540E-01	1.021E-01	**0.000E+00**	4.764E+01	5.908E+01	1.252E-01	2.980E+00	3.273E+00	6.898E-03	4.422E-01	1.238E+00
F03	5.148E+00	5.148E+00	**0.000E+00**	5.148E+00	5.185E+00	8.349E-02	5.148E+00	5.412E+00	1.968E-01	**1.033E-01**	**1.304E+00**	6.962E-01	5.148E+00	5.248E+00	1.428E-01
F04	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	6.072E-02	2.540E-02	1.090E-01	2.571E-01	1.168E-01	3.529E-07	6.880E-02	3.905E-02	0.000E+00	9.814E-06	5.374E-05
F05	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	1.411E-01	3.892E-01	9.950E-01	3.318E+00	2.116E+00	**0.000E+00**	1.404E-01	3.660E-01	0.000E+00	2.080E-02	1.139E-01
F06	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	2.503E-01	3.111E-01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F07	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	4.156E+00	2.159E+01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F08	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	1.207E+01	3.083E+01	0.000E+00	4.761E+01	3.843E+01	**0.000E+00**	3.142E-01	1.719E+00	**0.000E+00**	**0.000E+00**	**0.000E+00**
F09	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	9.667E+01	1.826E+01	1.000E+02	1.000E+02	2.523E-11	**0.000E+00**	3.333E+00	1.826E+01	0.000E+00	9.667E+01	1.826E+01
F10	**0.000E+00**	**2.437E+02**	**1.363E+02**	1.001E+02	3.375E+02	4.568E+01	3.474E+02	3.474E+02	1.174E-02	3.000E+02	3.016E+02	8.648E+00	3.474E+02	3.474E+02	5.226E-04

TABLE II: Results for 10D

Algorithms	IMODE			EBOwithCMAR			HSES			LSHADE-cnEpSin			LSHADE-SPACMA		
Problem	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**
F01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F02	1.249E-01	4.195E+00	**3.701E+00**	2.4982E-01	4.6310E+00	3.8043E+00	3.540E+00	8.503E+00	4.081E+00	**1.249E-01**	**3.514E+00**	3.238E+00	1.887E-01	5.205E+00	9.391E+00
F03	1.068E+01	1.212E+01	7.825E-01	**1.0367E+01**	**1.0399E+01**	**5.2151E-02**	1.044E+01	1.141E+01	6.567E-01	1.060E+01	1.157E+01	4.627E-01	1.063E+01	1.292E+01	1.868E+00
F04	**0.000E+00**	**0.000E+00**	**0.000E+00**	9.8647E-02	1.3331E-01	2.5029E-02	5.040E-01	9.651E-01	2.587E-01	9.865E-02	1.457E-01	2.621E-02	0.000E+00	1.444E-01	1.700E-01
F05	**4.026E-06**	**3.882E-01**	**3.833E-01**	2.0814E-01	1.0112E+01	1.1011E+01	2.087E-01	1.031E+02	1.987E+02	2.081E-01	2.783E+01	4.472E+01	0.000E+00	2.246E+00	5.216E+00
F06	**2.672E-02**	**9.145E-02**	**5.082E-02**	4.6613E-03	1.2924E-01	9.1148E-02	2.826E-01	8.633E+00	3.001E+01	1.152E-02	3.318E-01	2.295E-01	2.300E-02	3.796E-01	1.942E-01
F07	**1.407E-05**	**8.537E-04**	**1.096E-03**	5.2457E-05	1.5504E-01	1.9904E-01	7.512E-04	3.264E+01	8.241E+01	2.140E-06	4.278E-01	2.856E-01	2.269E-06	2.784E-01	3.033E-01
F08	**0.000E+00**	**2.723E+00**	**7.458E+00**	1.0000E+02	1.0000E+02	0.0000E+00	1.000E+02	1.000E+02	0.000E+00	1.000E+02	1.000E+02	1.154E-13	1.000E+02	1.000E+02	0.000E+00
F09	**0.000E+00**	**4.105E+01**	**4.463E+01**	1.0000E+02	1.6010E+02	9.7970E+01	3.272E+02	3.286E+02	1.095E+00	1.000E+02	3.051E+02	6.957E+01	1.000E+02	2.873E+02	9.529E+01
F10	**3.977E+02**	**3.977E+02**	**2.891E-13**	3.9774E+02	4.1015E+02	2.0357E+01	4.440E+02	4.464E+02	1.233E+00	3.977E+02	4.223E+02	2.319E+01	3.977E+02	4.102E+02	2.055E+01

optimal for F05, F06 and F07. Considering the composition test problems, the proposed algorithm was able to achieve the optimal solution for F08, very close result to optimal for F09 and stuck in a local solution for F10.

4) 20D results: Table IV presents the detailed results obtained from IMODE and other algorithms for test functions with 20 decision variables. Considering the uni-modal test function, IMODE achieved optimal value for both best and mean fitness error. For the multi-modal functions, it obtained optimal solutions for F04, very close to optimal ones for F02 but its performance deteriorated when solving F3. Regarding the hybrid functions, although IMODE obtained results very close to 0 for F06 and F07, its best and average ones for F05 were slightly worse. However, when solving the composite functions, similar to the 15D problems, it was stuck in local solutions for F10, but obtained the optimal solutions for F08 and F09.

5) Complexity of the proposed algorithm: The computational complexity of the proposed IMODE was calculated based on 5D, 10D and 15D based on the guidelines of the competition (). A summary of the obtained results is presented at Table V, from which, the computational time is very small and linearly increases as the problem dimension increase.

C. comparison with other rival algorithms

To judge the effectiveness of the proposed IMODE is judged by comparing its performance with four state-of-the-art algorithms, EBOwithCMAR, HSES, LSHADE-cnEpSin and LSHADE-SPACMA. As previously mentioned, these algorithms were executed using the same parameters suggested by their authors in their papers and other conditions are the same as the competition guides.

1) 5D results: considering the quality of solutions presented in Table VI and in regards to best results obtained, the proposed IMODE was better than EBOwithCMAR, HSES, LSHADE-cnEpSin and LSHADE-SPACMA in 1, 4, 3 and 2 test problems, respectively, equal to them in 9, 6, 6 and 8 test problems, respectively, and was inferior to them for 0, 0, 1 and 0 test functions respectively. Regarding the average results obtained, IMODE was superior to EBOwithCMAR, HSES, LSHADE-cnEpSin and LSHADE-SPACMA in 7, 9, 6 and 6 test problems, respectively, was similar to them in 3, 1, 3 and 4 test functions, respectively, but was inferior to them in 0, 0, 1 and 0 test problems, respectively.

Considering the Wilcoxon test, IMODE was significantly better than EBOwithCMAR, HSES, and LSHADE-SPACMA in regards to average results obtained, while there is no significant difference in the other cases. Further analysis was conducted using the Friedman test to rank all the algorithms based on their average results presented in Table VII, with IMODE ranked first.

For further analysis, a graph of the performance profiles for the test problems plotted to compare all the algorithms is presented in Figure 1 a. It indicates that consistent results were obtained from the Friedman and Wilcoxon tests as IMODE attained a ratio of 1.0 first at $\tau = 4$.

2) 10D results: The summary of the results obtained from IMODE and the other state-of-the-art algorithms are presented at Table VI. Regarding the quality of the results and based on the best results obtained, IMODE was better than EBOwithCMAR, HSES, and LSHADE-SPACMA in 5, 8, 5 and 4 test functions, respectively, was worse than them in 2, 1, 4 and 4 test problems respectively, was equal to them in 3, 1, 1 and 2 test problems respectively. Considering the average obtained results, the proposed algorithm was superior to EBOwithCMAR, HSES, and LSHADE-SPACMA in 8, 8, 7 and 9 test problems, respectively, was inferior in 1, 1, 2 and 0 test functions, respectively.

The Wilcoxon test was done to check the statistical differences between the algorithms, with the results presented at Table VI. From this table, the proposed IMODE was statistically better than EBOwithCMAR, LSHADE-cnEpSin and LSHADE-SPACMA for average results obtained and HSES for both best and average results. The Friedman test was also conducted to rank all rival algorithms with the results presented at Table VII showed that the proposed IMODE was ranked first for both best and average results. This conclusion was also confirmed from the result obtained from the performance profiles depicted in Figure 1 b, in which the proposed algorithm attained a value of 1 first $\tau \approx 1.2$.

TABLE III: Results for 15D

Algorithms	IMODE			EBOwithCMAR			HSES			LSHADE-cnEpSin			LSHADE-SPACMA		
Problem	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**
F01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F02	**1.249E-01**	**3.137E+00**	**3.219E+00**	2.498E-01	7.969E+00	7.775E+00	1.188E+02	1.949E+02	1.262E+02	2.416E+00	5.258E+01	5.016E+01	1.638E-01	3.732E+00	3.385E+00
F03	1.557E+01	1.608E+01	3.118E-01	1.557E+01	1.557E+01	1.448E-02	1.612E+01	1.794E+01	1.829E+00	**4.890E+00**	**1.000E+01**	**2.438E+00**	1.557E+01	1.612E+01	8.304E-01
F04	**0.000E+00**	**0.000E+00**	**0.000E+00**	1.480E-01	1.997E-01	2.262E-02	1.059E+00	1.448E+00	3.067E-01	1.787E-01	2.733E-01	4.690E-02	1.781E-01	2.810E-01	2.228E-01
F05	**1.151E+00**	**7.789E+00**	**3.657E+00**	3.122E-01	2.792E+01	4.774E+01	8.956E+00	4.217E+01	4.489E+01	1.154E+01	2.406E+01	7.739E+00	1.561E-01	7.014E+00	2.166E+01
F06	2.809E-01	6.921E-01	2.516E-01	**4.053E-02**	**2.071E-01**	**1.195E-01**	4.539E-01	2.656E+01	5.697E+01	5.379E-01	1.131E+00	5.397E-01	7.773E-02	6.242E-01	2.756E-01
F07	**1.283E-01**	**5.299E-01**	**2.232E-01**	2.907E-01	4.605E+00	2.160E+01	5.058E-01	1.972E+02	1.564E+02	2.769E-01	6.968E-01	2.037E-01	4.765E-01	4.742E+00	2.189E+01
F08	**0.000E+00**	**4.178E+00**	**9.608E+00**	1.000E+02	1.000E+02	0.000E+00	1.000E+02	1.000E+02	1.573E-13	2.152E-03	9.110E+01	2.419E+01	1.000E+02	1.000E+02	0.000E+00
F09	**0.000E+00**	**9.333E+01**	**2.537E+01**	1.000E+02	2.999E+02	1.234E+02	3.851E+02	3.879E+02	1.927E+00	1.000E+02	3.432E+02	1.106E+02	3.897E+02	3.905E+02	1.391E+00
F10	**4.000E+02**	**4.000E+02**	**0.000E+00**	**4.000E+02**	**4.000E+02**	**0.000E+00**	**4.000E+02**	**4.000E+02**	1.661E-10	**4.000E+02**	**4.000E+02**	2.891E-13	**4.000E+02**	**4.000E+02**	**0.000E+00**

TABLE IV: Results for 20D

Algorithms	IMODE			EBOwithCMAR			HSES			LSHADE-cnEpSin			LSHADE-SPACMA		
Problem	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**	**Best**	**Mean**	**Std.**
F01	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**	**0.000E+00**
F02	**3.123E-02**	**5.130E-01**	**7.125E-01**	9.369E-02	1.143E+00	1.273E+00	1.926E+00	2.358E+02	1.698E+02	1.771E+00	1.458E+01	2.923E+01	9.480E-02	1.602E+00	1.132E+00
F03	2.039E+01	2.051E+01	1.255E-01	**2.039E+01**	**2.039E+01**	**1.809E-13**	2.057E+01	2.375E+01	1.834E+00	5.421E+00	1.112E+01	4.229E+00	2.039E+01	2.055E+01	3.003E-01
F04	**0.000E+00**	**0.000E+00**	**0.000E+00**	1.973E-01	2.914E-01	3.478E-02	1.315E+00	1.936E+00	4.444E-01	2.565E-01	3.822E-01	4.809E-02	2.322E-01	2.989E-01	2.887E-02
F05	**2.615E+00**	**1.089E+01**	**4.325E+00**	2.094E+00	8.665E+01	7.875E+01	3.144E-01	1.803E+02	1.093E+02	4.396E+00	1.647E+01	8.459E+00	2.198E+00	4.304E+01	6.443E+01
F06	**1.757E-01**	**3.017E-01**	**8.167E-02**	4.369E-02	3.234E-01	2.475E-01	7.417E-01	3.697E+01	5.517E+01	2.662E-01	4.344E-01	8.987E-02	1.979E-01	4.054E-01	1.139E-01
F07	**2.383E-01**	**5.242E-01**	**1.640E-01**	3.569E-02	3.007E+00	5.930E+00	1.866E-01	1.518E+01	2.853E+01	1.459E-01	4.090E-01	1.678E-01	5.179E-01	4.697E+00	4.182E+00
F08	**3.055E+01**	**8.404E+01**	**1.887E+01**	1.000E+02	1.000E+02	2.120E-13	1.000E+02	1.000E+02	0.000E+00	1.000E+02	1.000E+02	0.000E+00	1.000E+02	1.000E+02	0.000E+00
F09	**1.342E-04**	**9.667E+01**	**1.826E+01**	1.000E+02	3.814E+02	6.427E+01	3.850E+02	3.951E+02	3.433E+00	4.113E+02	4.233E+02	5.287E+00	4.017E+02	4.097E+02	4.633E+00
F10	**3.993E+02**	**3.998E+02**	**6.181E-01**	4.105E+02	4.136E+02	5.741E-01	4.137E+02	4.248E+02	2.285E+01	3.990E+02	4.021E+02	3.592E+00	4.137E+02	4.137E+02	4.943E-04

TABLE V: Run time complexity of the IMODE algorithm

	T_0	T_1	$\hat{T_2}$	$(\hat{T_2} - T_1)/T_0$
$D = 5$	0.111789	0.223535	3.33E-01	0.978029144
$D = 10$		0.568714	1.41E+00	7.499718219
$D = 15$		0.727411	2.03E+00	11.62000734

TABLE VI: Summary of comparisons of performances of IMODE, EBOwithCMAR, HSES, LSHADE-cnEpSin and LSHADE-SPACMA

Dimensions	Algorithms	Criteria	Better	Similar	Worse	(p, Dec.)
5D	IMODE vs. EBOwithCMAR	Best	1	9	0	(0.317, ≈)
		Mean	7	3	0	(0.018, +)
	IMODE vs.HSES	Best	4	6	0	(0.068, ≈)
		Mean	9	1	0	(0.008, +)
	IMODE vs. LSHADE-cnEpSin	Best	3	6	1	(0.0465, ≈)
		Mean	6	3	1	(0.176, ≈)
	IMODE vs. LSHADE-SPACMA	Best	2	8	0	(0.18, ≈)
		Mean	6	4	0	(0.028, +)
10D	IMODE vs. EBOwithCMAR	Best	5	3	2	(0.161, ≈)
		Mean	8	1	1	(0.038, +)
	IMODE vs.HSES	Best	8	1	1	(0.028, +)
		Mean	8	1	1	(0.011, +)
	IMODE vs. LSHADE-cnEpSin	Best	5	1	4	(0.213, ≈)
		Mean	7	1	2	(0.048, +)
	IMODE vs. LSHADE-SPACMA	Best	4	2	4	(0.40, ≈)
		Mean	9	1	0	(0.008, +)
15D	IMODE vs. EBOwithCMAR	Best	5	3	2	(0.398, ≈)
		Mean	6	2	2	(0.068,≈)
	IMODE vs.HSES	Best	8	2	0	(0.012, +)
		Mean	8	2	0	(0.012, +)
	IMODE vs. LSHADE-cnEpSin	Best	7	2	1	(0.093, ≈)
		Mean	7	2	1	(0.048, +)
	IMODE vs. LSHADE-SPACMA	Best	5	3	2	(0.398, ≈)
		Mean	6	2	2	(0.036, +)
20D	IMODE vs. EBOwithCMAR	Best	5	2	3	(0.207, ≈)
		Mean	8	1	1	(0.015, +)
	IMODE vs.HSES	Best	7	1	2	(0.048, +)
		Mean	9	1	0	(0.008, +)
	IMODE vs. LSHADE-cnEpSin	Best	6	2	2	(0.123, ≈)
		Mean	7	1	2	(0.048, +)
	IMODE vs. LSHADE-SPACMA	Best	7	2	1	(0.017, +)
		Mean	9	1	0	(0.008, +)

3) 15D results: Table VI presents the summary of the results obtained by IMODE and the existing algorithms. Considering the solutions' quality and based on the best obtained results, the proposed IMODE algorithm was superior to EBOwithCMAR, LSHADE-cnEpSin and LSHADE-SPACMA in 5, 8, 7 and 5 test problems, respectively, was inferior in 2, 0, 1 and 2 test functions, respectively, and obtained equal results to them for 3, 2, 2 and 3 test problems, respectively. While based on the average results obtained, IMODE was better than EBOwithCMAR, LSHADE-cnEpSin and LSHADE-SPACMA in 6, 8, 7 and 6 test functions, respectively, was worse than them in 2, 0, 1 and 2 functions, respectively, was similar to them in 2, 2, 2 and 2 test problems, respectively.

Based on the Wilcoxon test, the proposed IMODE was significantly better than HSES for both the best and average results, LSHADE-cnEpSin and LSHADE-SPACMA for average results, while there was no significant difference in the other scenarios. The Friedman test was also conducted to rank all algorithms with the results presented at Table VII showed that the proposed IMODE was ranked first for both best and mean obtained results. The performance profiles graph was depicted in Figure 1 c, which confirmed that the performance of the proposed IMODE is better than all the rival algorithms as it was reached a value of 1 first $\tau \approx 3.5$.

4) 20D results: Regarding the quality of solutions and based on the best results, as shown in Table **??**, IMODE was superior to EBOwithCMAR, LSHADE-cnEpSin and LSHADE-SPACMA for 5, 7, 6 and 7 test problems, respectively, similar for 2, 1, 2 and 2 test problems, respectively, and inferior for 3, 2, 2 and 1 test problems, respectively. Considering the average obtained results, IMODE was better than EBOwithCMAR, LSHADE-cnEpSin and LSHADE-SPACMA for 8, 9, 7 and 9 test functions, respectively, equal in 1, 1, 1 and 1 test problems, respectively, and worse for 1, 0, 2 and 0 test problems, respectively.

The Wilcoxon test was conducted to check whether IMODE was statistically better than the rival algorithms, with the obtained results shown in Table VI. It was clear that IMODE was statistically superior to all the other algorithms for the obtained average results and to HSES for the best results as well. Also, the Friedman test results depicted in Table VII demonstrate that IMODE was ranked first for both the best and average results.

A graph of the performance profiles for the test problems plotted to compare all the algorithms is presented in Figure 1. It showed that consistent results were obtained from the Friedman and Wilcoxon tests as IMODE attained a ratio of 1.0 first at $\tau \approx 1.9$.

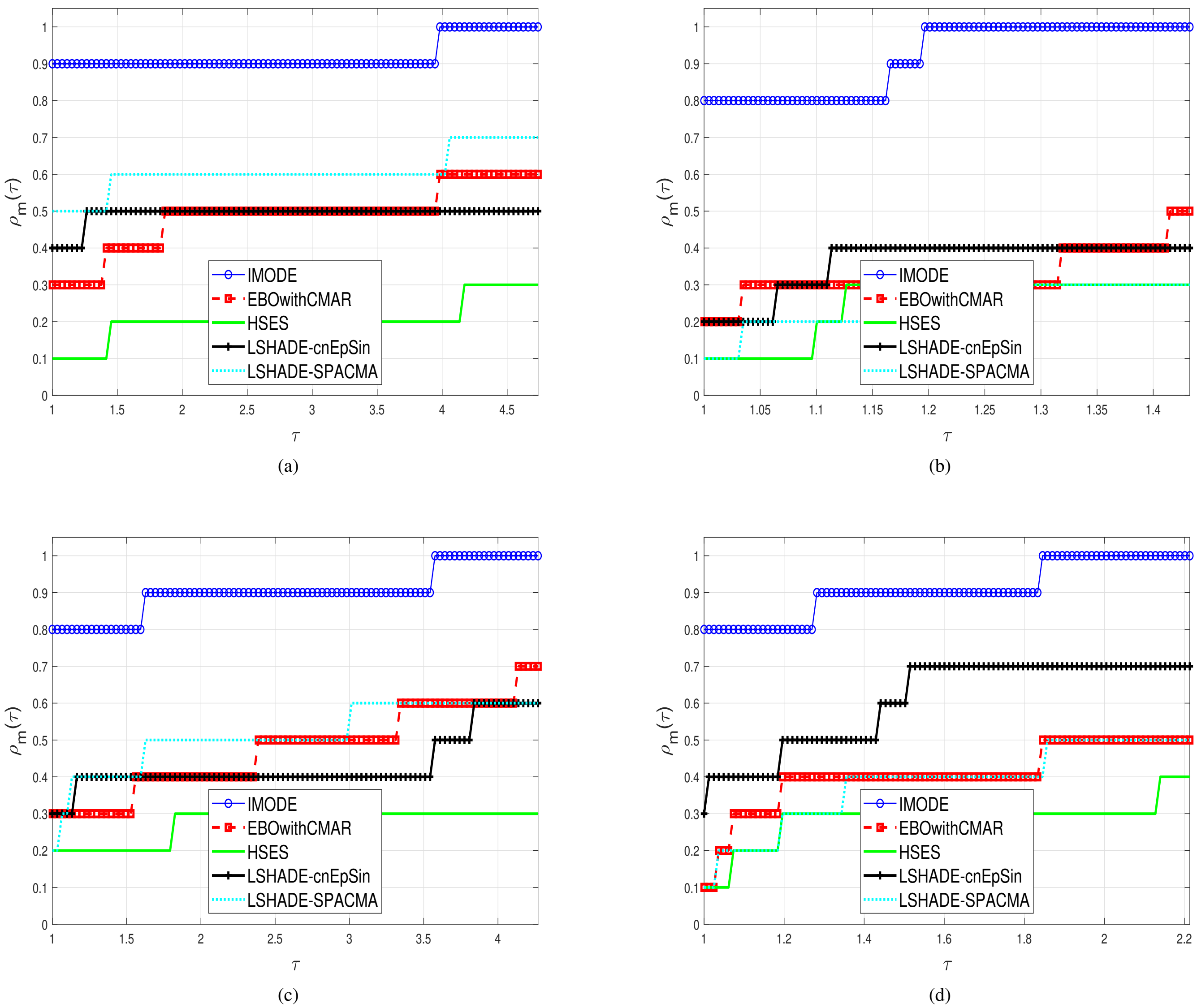

Fig. 1: Performance profiles graphs comparing the performance of IMODE, EBOwithCMAR, HSES,LSHADE-cnEpSin and LSHADE-SPACMA based on the mean results for (a) 5D; (b) 10D; (c) 15D; and (d) 20D

TABLE VII: Ranking of all algorithms for all dimensions obtained by Friedman test

Algorithms	Rank for 5D		Rank for 10D		Rank for 15D		Rank for 20D	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean
IMODE	**2.55**	**1.65**	**2.50**	**2.50**	**2.00**	**1.90**	**2.15**	**1.50**
EBOwithCMAR	2.65	3.05	2.75	2.75	2.65	2.70	2.25	2.65
HSES	3.70	4.80	4.35	4.35	4.20	4.40	3.80	4.45
LSHADE-cnEpSin	3.00	2.70	2.70	2.70	3.15	2.80	3.30	2.95
LSHADE-SPACMA	3.10	2.80	2.70	2.70	3.00	3.20	3.50	3.45

V. Conclusion

Although several EAs for solving optimization problems have been developed, the literature shows that no single search operator and/or algorithm is capable of successfully solving all types of optimization problems. Consequently, many multi-operator- and/or multi-method-based algorithms have been proposed. Their designs were based mainly on trial and error methods and, furthermore, their performances could be statistically outperformed by single-operator-based approaches. Therefore, in this paper, an improved multi-operator DE, with SQP used as a local search in the later stages of the evolutionary process was developed. In the multi-operator DE, the best operator was selected based on the quality of solutions and diversity of sub-populations. The proposed algorithm's performance was judged by using it to solve 10 unconstrained problems with different dimensions. The computational results showed that it was 100% statistically better than or similar to the rival algorithms for the 5D, 10D, 15D and 20D problems considered.

A possible future research direction could be to adapt the machine-learning based techniques to select the best-performing algorithm among several for a single-algorithm framework. Also, adapting the algorithm to solve other types of test problems, i.e., constrained and/or integer, may be worthy of investigation.

REFERENCES

[1] S. Elsayed, R. Sarker, and C. A. C. Coello, "Fuzzy rule-based design of evolutionary algorithm for optimization," *IEEE transactions on cybernetics*, vol. 49, no. 1, pp. 301–314, 2017.

[2] S. M. Elsayed, "Evolutionary approach for constrained optimization," PhD dissertation, University of New South Wales, Canberra, Australia, 2012.

[3] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Landscape-assisted multi-operator differential evolution for solving constrained optimization problems," *Expert Systems with Applications*, p. 113033, 2019.

[4] K. Deb, *Optimization for engineering design: Algorithms and examples*. PHI Learning Pvt. Ltd., 2012.

[5] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through simulated evolution," 1966.

[6] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Landscape-based adaptive operator selection mechanism for differential evolution," *Information Sciences*, vol. 418, pp. 383–404, 2017.

[7] S. M. Elsayed, R. A. Sarker, and T. Ray, "Differential evolution with automatic parameter configuration for solving the cec2013 competition on real-parameter optimization," in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 1932–1937.

[8] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2010.

[9] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Two-phase differential evolution framework for solving optimization problems," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2016, pp. 1–8.

[10] K. M. Sallam, R. A. Sarker, D. L. Essam, and S. M. Elsayed, "Neurodynamic differential evolution algorithm and solving cec2015 competition problems," in *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 1033–1040.

[11] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Improved united multi-operator algorithm for solving optimization problems," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–8.

[12] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Multi-method based orthogonal experimental design algorithm for solving cec2017 competition problems," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 1350–1357.

[13] J. Liang, B. Qu, P. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization," *Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, vol. 29, pp. 625–640, 2014.

[14] A. W. Mohamed, A. K. Mohamed, E. Z. Elfeky, and M. Saleh, "Enhanced directed differential evolution algorithm for solving constrained engineering optimization problems," *International Journal of Applied Metaheuristic Computing (IJAMC)*, vol. 10, no. 1, pp. 1–28, 2019.

[15] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Differential evolution with landscape-based operator selection for solving numerical optimization problems," in *Intelligent and evolutionary systems*. Springer, 2017, pp. 371–387.

[16] Q. Fan, X. Yan, and Y. Zhang, "Auto-selection mechanism of differential evolution algorithm variants and its application," *European Journal of Operational Research*, vol. 270, no. 2, pp. 636–653, 2018.

[17] M. F. Tasgetiren, P. N. Suganthan, and Q.-K. Pan, "An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem," *Applied Mathematics and Computation*, vol. 215, no. 9, pp. 3356–3368, 2010.

[20] S. X. Zhang, L. M. Zheng, K. S. Tang, S. Y. Zheng, and W. S. Chan, "Multi-layer competitive-cooperative framework for performance enhancement of differential evolution," *Information Sciences*, vol. 482, pp. 86–104, 2019.

[18] S. Elsayed, R. Sarker, C. C. Coello, and T. Ray, "Adaptation of operators and continuous control parameters in differential evolution for constrained optimization," *Soft Computing*, vol. 22, no. 19, pp. 6595–6616, 2018.

[19] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants," *Information Sciences*, vol. 423, pp. 172–186, 2018.

[21] M. Attia, M. Arafa, E. Sallam, and M. Fahmy, "An enhanced differential evolution algorithm with multi-mutation strategies and self-adapting control parameters," 2019.

[22] X. Yu, X. Yu, Y. Lu, G. G. Yen, and M. Cai, "Differential evolution mutation operators for constrained multi-objective optimization," *Applied Soft Computing*, vol. 67, pp. 452–466, 2018.

[23] A. W. Mohamed, "A novel differential evolution algorithm for solving constrained engineering optimization problems," *Journal of Intelligent Manufacturing*, vol. 29, no. 3, pp. 659–692, 2018.

[24] J. Chen, J. Chen, and H. Min, "A united framework with multi-operator evolutionary algorithms and interior point method for efficient single objective optimisation problem solving," *International Journal of High Performance Computing and Networking*, vol. 13, no. 3, pp. 340–353, 2019.

[25] S. M. Elsayed, R. A. Sarker, D. L. Essam, and N. M. Hamza, "Testing united multi-operator evolutionary algorithms on the cec2014 real-parameter numerical optimization," in *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 1650–1657.

[26] S. Elsayed, N. Hamza, and R. Sarker, "Testing united multi-operator evolutionary algorithms-ii on single objective optimization problems," in *2016 IEEE congress on evolutionary computation (CEC)*. IEEE, 2016, pp. 2966–2973.

[27] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution–an updated survey," *Swarm and Evolutionary Computation*, vol. 27, pp. 1–30, 2016.

[28] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Information Sciences*, vol. 329, pp. 329–345, 2016.

[29] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms–a survey," *Swarm and evolutionary computation*, vol. 44, pp. 695–711, 2019.

[30] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *2014 IEEE congress on evolutionary computation (CEC)*. IEEE, 2014, pp. 1658–1665.

[31] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *IEEE Transactions on evolutionary computation*, vol. 13, no. 5, pp. 945–958, 2009.

[32] A. Kumar, R. K. Misra, and D. Singh, "Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 1835–1842.

[33] G. Zhang and Y. Shi, "Hybrid sampling evolution strategy for solving single objective bound constrained problems," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–7.

[34] N. H. Awad, M. Z. Ali, and P. N. Suganthan, "Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving cec2017 benchmark problems," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 372–379.

[35] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi, "Lshade with semi-parameter adaptation hybrid with cma-es for solving cec 2017 benchmark problems," in *2017 IEEE Congress on evolutionary computation (CEC)*. IEEE, 2017, pp. 145–152.

[36] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[37] H. J. Barbosa, H. S. Bernardino, and A. M. Barreto, "Using performance profiles for the analysis and design of benchmark experiments," in *Advances in Metaheuristics*. Springer, 2013, pp. 21–36.